aws

Entwicklerhandbuch

Amazon Cognito

Amazon Cognito: Entwicklerhandbuch

Table of Contents

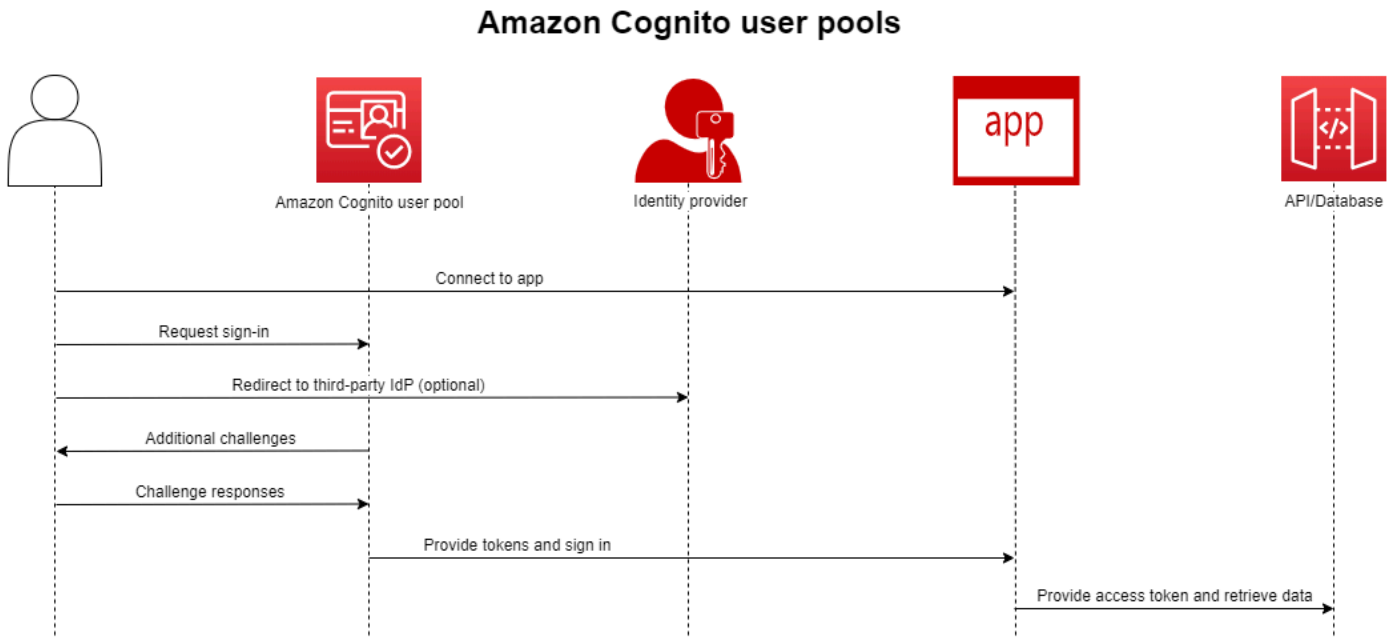
Was ist Amazon Cognito?

Amazon Cognito ist eine Identitätsplattform für Web- und mobile Apps. Es handelt sich um ein Benutzerverzeichnis, einen Authentifizierungsserver und einen Autorisierungsservice für OAuth-2.0-Zugriffs-Token und AWS -Anmeldeinformationen. Mit Amazon Cognito können Sie Benutzer über das integrierte Benutzerverzeichnis, über Ihr Unternehmensverzeichnis und über kommerzielle Identitätsanbieter wie Google und Facebook authentifizieren und autorisieren.

Themen

Amazon Cognito besteht aus den beiden folgenden Komponenten. Diese arbeiten unabhängig oder zusammen, je nach Ihren Zugriffsanforderungen für Ihre Benutzer.

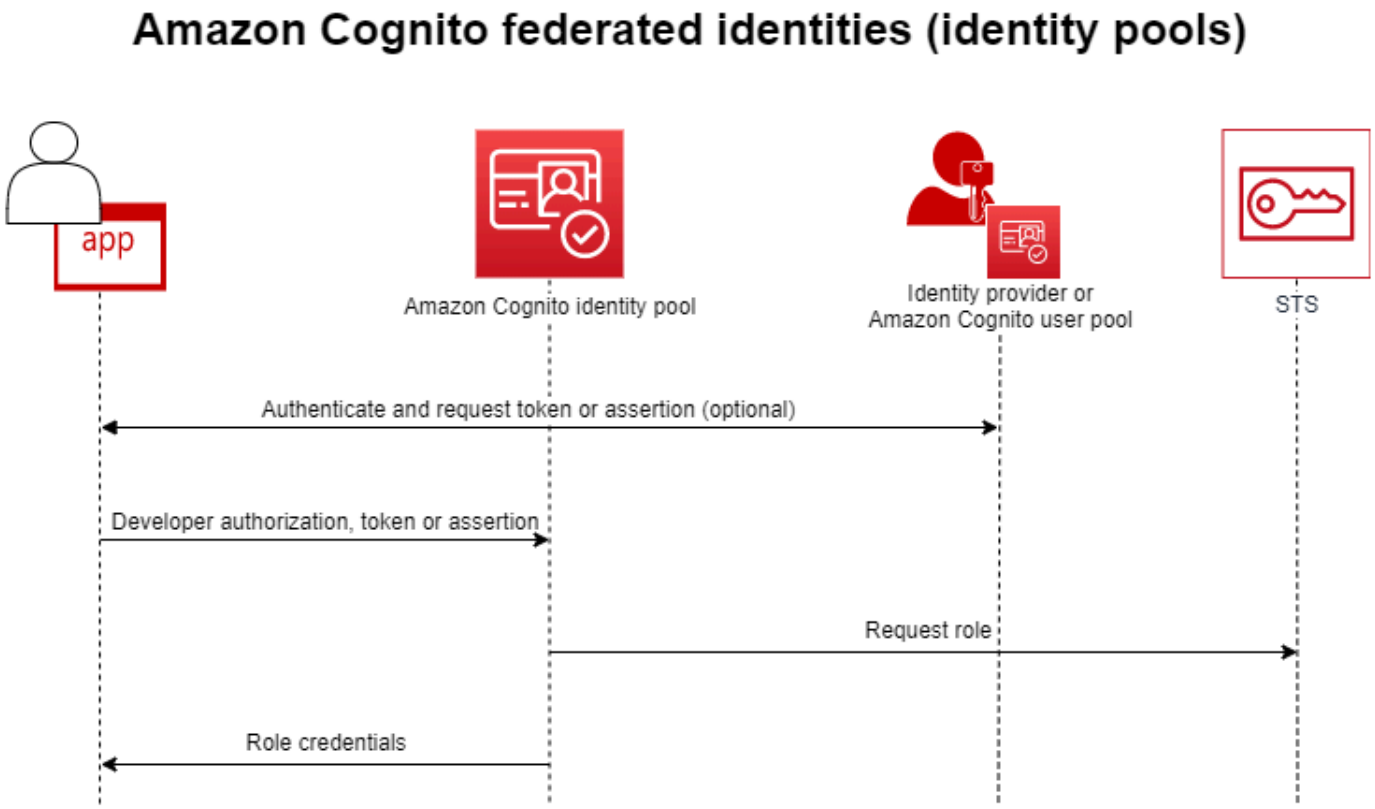
Benutzerpools



Erstellen Sie einen Benutzerpool, wenn Sie Benutzer für Ihre App oder API authentifizieren und autorisieren möchten. Benutzerpools sind Benutzerverzeichnisse mit Self-Service- und Administratorsteuerung zur Benutzererstellung, -verwaltung und -authentifizierung. Ihr Benutzerpool kann ein unabhängiges Verzeichnis und ein OIDC-Identitätsanbieter (IdP) sowie ein Zwischendienstanbieter (SP) für externe Personal- und Kundenidentitäten sein. Sie können in Ihrer App Single Sign-On (SSO) für die Mitarbeiteridentitäten Ihres Unternehmens in SAML 2.0 und IdPs OIDC mit Benutzerpools bereitstellen. Sie können SSO auch in Ihrer App für Kundenidentitäten Ihres Unternehmens in den öffentlichen OAuth-2.0-Identitätsspeichern Amazon, Google, Apple und Facebook bereitstellen. Weitere Informationen zu CIAM (Customer Identity and Access Management) finden Sie unter Was ist CIAM?.

Benutzerpools erfordern keine Integration in einen Identitätspool. Von einem Benutzerpool aus können Sie authentifizierte JSON-Web-Token (JWTs) direkt an eine App, einen Webserver oder eine API ausgeben.

Identitäten-Pools



Richten Sie einen Amazon Cognito Cognito-Identitätspool ein, wenn Sie authentifizierten oder anonymen Benutzern den Zugriff auf Ihre Ressourcen gewähren möchten. AWS Ein Identitätspool stellt AWS Anmeldeinformationen für Ihre App aus, um Benutzern Ressourcen bereitzustellen. Sie können Benutzer bei einem vertrauenswürdigen Identitätsanbieter authentifizieren, z. B. bei einem Benutzerpool oder einem SAML-2.0-Service. Es kann optional auch Anmeldeinformationen für Gastbenutzer ausgeben. Identitätspools verwenden sowohl rollen- als auch attributbasierte Zugriffskontrolle, um die Autorisierung Ihrer Benutzer für den Zugriff auf Ihre Ressourcen zu verwalten. AWS

Identitätspools erfordern keine Integration in einen Benutzerpool. Ein Identitätspool kann authentifizierte Anfragen sowohl von Mitarbeitern als auch von kommerziellen Identitätsanbietern direkt annehmen.

Ein Amazon-Cognito-Benutzerpool und ein Identitätspool, gemeinsam verwendet

In dem Diagramm am Anfang dieses Themas verwenden Sie Amazon Cognito, um Ihren Benutzer zu authentifizieren und ihm dann Zugriff auf einen AWS-Service zu gewähren.

1. Ihr App-Benutzer meldet sich über einen Benutzerpool an und erhält OAuth-2.0-Token.
2. Ihre App tauscht ein Benutzerpool-Token mit einem Identitätspool gegen temporäre AWS Anmeldeinformationen aus, die Sie mit AWS APIs und dem () verwenden können. AWS Command Line Interface AWS CLI
3. Ihre App weist Ihrem Benutzer die Sitzung mit den Anmeldeinformationen zu und gewährt autorisierten Zugriff auf Amazon AWS-Services S3 und Amazon DynamoDB.

Weitere Beispiele, die Identitäten- und Benutzerpools verwenden, finden Sie unter Häufige Amazon-Cognito-Szenarien.

In Amazon Cognito entspricht die Cloud-Sicherheit im Rahmen des Modells der geteilten Verantwortung den Anforderungen von SOC 1-3, PCI DSS, ISO 27001 und HIPAA-BAA. Sie können Ihre Cloud-Sicherheit in Amazon Cognito so gestalten, dass sie mit SOC1-3, ISO 27001 und HIPAA-BAA, jedoch nicht mit PCI DSS ist. Weitere Informationen finden Sie unter AWS -Services in Scope. Siehe auch Überlegungen zu regionenbezogenen Daten.

Funktionen von Amazon Cognito

Benutzerpools

Ein Amazon-Cognito-Benutzerpool ist ein Benutzerverzeichnis. Mit einem Benutzerpool können sich Ihre Benutzer über Amazon Cognito oder im Verbund durch einen Drittanbieter-Identitätsanbieter (IdP) bei Ihrer Web- oder mobilen App anmelden. Verbundene Benutzer und lokale Benutzer haben ein Benutzerprofil in Ihrem Benutzerpool.

Lokale Benutzer sind solche, die Sie erstellt haben oder die sich in Ihrem Benutzerpool angemeldet haben. Sie können diese Benutzerprofile im AWS Management Console, einem AWS SDK oder dem () verwalten und anpassen. AWS Command Line Interface AWS CLI

Amazon Cognito Cognito-Benutzerpools akzeptieren Token und Assertions von Drittanbietern IdPs und sammeln die Benutzerattribute in einem JWT, das an Ihre App ausgegeben wird. Sie können Ihre App auf einem Satz von JWTs standardisieren, während Amazon Cognito die Interaktionen mit IdPs diesen abwickelt und deren Ansprüche einem zentralen Token-Format zuordnet.

Ein Amazon-Cognito-Benutzerpool kann ein eigenständiger IdP sein. Amazon Cognito nutzt den OpenID Connect (OIDC)-Standard, um JWTs für die Authentifizierung und Autorisierung zu generieren. Wenn Sie lokale Benutzer anmelden, ist Ihr Benutzerpool für diese Benutzer maßgebend. Sie haben Zugriff auf die folgenden Funktionen, wenn Sie lokale Benutzer authentifizieren.

- Implementieren Sie Ihr eigenes Web-Frontend, das die Amazon-Cognito-Benutzerpool-API aufruft, um Ihre Benutzer zu authentifizieren, zu autorisieren und zu verwalten.
- Richten Sie Multi-Faktor-Authentifizierung (MFA) für Ihre Benutzer ein. Amazon Cognito unterstützt ein zeitgesteuertes Einmalpasswort (TOTP) und MFA für SMS-Nachrichten.
- Schützen Sie sich vor dem Zugriff von Benutzerkonten, die unter böswilliger Kontrolle stehen.
- Erstellen Sie Ihre eigenen benutzerdefinierten mehrstufigen Authentifizierungsabläufe.
- Suchen Sie nach Benutzern in einem anderen Verzeichnis und migrieren Sie sie zu Amazon Cognito.

Ein Amazon Cognito Cognito-Benutzerpool kann auch eine Doppelrolle als Service Provider (SP) für Ihre IdPs App und als IdP für Ihre App erfüllen. Amazon Cognito Cognito-Benutzerpools können eine Verbindung zu Verbrauchern IdPs wie Facebook und Google oder Mitarbeitern IdPs wie Okta und Active Directory Federation Services (ADFS) herstellen.

Mit den OAuth 2.0- und OpenID Connect (OIDC)-Tokens, die ein Amazon-Cognito-Benutzerpool ausgibt, können Sie:

- in Ihrer App ein ID-Token akzeptieren, das einen Benutzer authentifiziert und die Informationen bereitstellt, die Sie zum Einrichten des Benutzerprofils benötigen
- ein Zugriffs-Token in Ihrer API mit den OIDC-Bereichen abrufen, die die API-Aufrufe Ihrer Benutzer autorisieren.
- Rufen Sie AWS Anmeldeinformationen aus einem Amazon Cognito Cognito-Identitätspool ab.

Funktionen von Amazon-Cognito-Benutzerpools

Funktion	Beschreibung
OIDC-IdP	Geben Sie ID-Token aus, um Benutzer zu authentifizieren
Autorisierungsserver	Stellen Sie Zugriffstoken aus, um den Benutzerzugriff auf APIs zu autorisieren
SAML 2.0 SP	Verwandeln Sie SAML-Assertionen in ID- und Zugriffstoken

OIDC SP	Transformieren Sie OIDC-Token in ID- und Zugriffstoken
OAuth 2.0 SP	Verwandeln Sie ID-Token von Apple, Facebook, Amazon oder Google in Ihre eigenen ID- und Zugriffstoken
Frontend-Dienst zur Authentifizierung	Registrieren, verwalten und authentifizieren Sie Benutzer mit der gehosteten Benutzeroberfläche
API-Unterstützung für Ihre eigene Benutzeroberfläche	Erstellen, verwalten und authentifizieren Sie Benutzer über API-Anfragen in unterstützten AWS SDKs[1]
MFA	Verwenden Sie SMS-Nachrichten, TOTPs oder das Gerät Ihres Benutzers als zusätzlichen Authentifizierungsfaktor[1]
Sicherheitsüberwachung und Reaktion	Schützt vor böswilligen Aktivitäten und unsicheren Passworten[1]
Passen Sie die Authentifizierungsabläufe an	Erstellen Sie Ihren eigenen Authentifizierungsmechanismus oder fügen Sie benutzerdefinierte Schritte zu bestehenden Abläufen hinzu[1]
Gruppen	Erstellen Sie logische Gruppierungen von Benutzern und eine Hierarchie von IAM-Rollenansprüchen, wenn Sie Token an Identitätspools weitergeben
Passen Sie ID-Token an	Passen Sie Ihre ID-Token mit neuen, geänderten und unterdrückten Ansprüchen an
Passen Sie Benutzerattribute an	Weisen Sie Benutzerattributen Werte zu und fügen Sie Ihre eigenen benutzerdefinierten Attribute hinzu

[1] Die Funktion ist nur für lokale Benutzer verfügbar.

Weitere Informationen zu Benutzerpools finden Sie unter Erste Schritte mit Benutzerpools und in der API-Referenz der Amazon-Cognito-Benutzerpools.

Identitäten-Pools

Ein Identitätspool ist eine Sammlung von eindeutigen Kennungen oder Identitäten, die Sie Ihren Benutzern oder Gästen zuweisen und die Sie für den Empfang temporärer Anmeldeinformationen autorisieren. AWS Wenn Sie einem Identitätspool einen Authentifizierungsnachweis in Form vertrauenswürdiger Anforderungen eines SAML 2.0-, OpenID Connect (OIDC)- oder OAuth 2.0-Social-Identity-Anbieters (IdP) vorlegen, ordnen Sie Ihrem Benutzer eine Identität im Identitätspool zu. Das Token, das Ihr Identitätspool für die Identität erstellt, kann temporäre Sitzungsanmeldedaten von AWS Security Token Service () abrufen.AWS STS

Als Ergänzung zu authentifizierten Identitäten können Sie auch einen Identitätspool konfigurieren, um den AWS Zugriff ohne IdP-Authentifizierung zu autorisieren. Sie können Ihren eigenen benutzerdefinierten Authentifizierungsnachweis oder auch keine Authentifizierung anbieten. Sie können jedem App-Benutzer, der sie anfordert, temporäre AWS Anmeldeinformationen mit nicht authentifizierten Identitäten gewähren. Identitätspools akzeptieren auch Anforderungen und geben Anmeldeinformationen auf der Grundlage Ihres eigenen benutzerdefinierten Schemas mit vom Entwickler authentifizierten Identitäten aus.

Mit Amazon-Cognito-Identitätspools haben Sie zwei Möglichkeiten zur Integration in die IAM-Richtlinien in Ihrem AWS-Konto. Sie können diese beiden Funktionen zusammen oder einzeln verwenden.

Rollenbasierte Zugriffskontrolle

Wenn Ihr Benutzer Anforderungen an Ihren Identitätspool weitergibt, wählt Amazon Cognito die angeforderte IAM-Rolle aus. Um die Berechtigungen der Rolle an Ihre Bedürfnisse anzupassen, wenden Sie IAM-Richtlinien auf jede Rolle an. Wenn Ihr Benutzer beispielsweise nachweist, dass er in der Marketingabteilung tätig ist, erhält er Anmeldeinformationen für eine Rolle mit Richtlinien, die auf die Zugriffsanforderungen der Marketingabteilung zugeschnitten sind. Amazon Cognito kann eine Standardrolle anfordern; dies ist eine Rolle, die auf Regeln basiert, die die Anforderungen Ihres Benutzers abfragen, oder eine Rolle, die auf der Gruppenmitgliedschaft Ihres Benutzers in einem Benutzerpool basiert. Sie können die Rollen-Vertrauensrichtlinie auch so konfigurieren, dass IAM nur Ihrem Identitätspool vertraut, um temporäre Sitzungen zu generieren.

Attribute für Zugriffskontrolle

Ihr Identitätspool liest Attribute aus den Ansprüchen Ihres Benutzers und ordnet sie den Prinzipal-Tags in der temporären Sitzung Ihres Benutzers zu. Sie können dann Ihre ressourcenbasierten IAM-Richtlinien konfigurieren, um Zugriff auf Ressourcen zu gewähren oder zu verweigern, die auf IAM-Prinzipalen basieren, die die Sitzungs-Tags aus Ihrem Identitätspool enthalten. Wenn Ihr Benutzer beispielsweise nachweist, dass er in der Marketingabteilung tätig ist, AWS STS kennzeichnen Sie seine Sitzung. `Department: marketing` Ihr Amazon S3 S3-Bucket ermöglicht Lesevorgänge auf der Grundlage einer aws: PrincipalTag -Bedingung, die einen Wert von `marketing` für das `Department` Tag erfordert.

Eigenschaften von Amazon-Cognito-Identitätspools

Funktion	Beschreibung
Amazon Cognito Cognito-Benutzerpool SP	Tauschen Sie ein ID-Token aus Ihrem Benutzerpool gegen Web-Identitätsanmeldedaten von AWS STS
SAML 2.0 SP	Tauschen Sie SAML-Assertionen gegen Web-Identitätsanmeldedaten von AWS STS
OIDC SP	Tauschen Sie OIDC-Token gegen Web-Identitätsanmeldedaten von AWS STS
OAuth 2.0 SP	Tauschen Sie OAuth-Token von Amazon, Facebook, Google, Apple und Twitter gegen Web-Identitätsanmeldedaten von AWS STS
Benutzerdefinierter SP	Tauschen Sie mit AWS Anmeldeinformationen Ansprüche in einem beliebigen Format gegen Web-Identitätsanmeldedaten von AWS STS
Nicht authentifizierter Zugriff	Stellen Sie Web-Identitätsanmeldedaten mit eingeschränktem Zugriff ohne Authentifizierung aus AWS STS
Rollenbasierte Zugriffskontrolle	Wählen Sie eine IAM-Rolle für Ihren authentifizierten Benutzer auf der Grundlage seiner Ansprüche aus und konfigurieren Sie Ihre

	Rollen so, dass sie nur im Kontext Ihres Identitätspools übernommen werden
Attributbasierte Zugriffskontrolle	Wandeln Sie Ansprüche in Prinzipal-Tags für Ihre AWS STS temporäre Sitzung um und verwenden Sie IAM-Richtlinien, um den Ressourcenzugriff anhand von Prinzipal-Tags zu filtern

Für weitere Informationen zu Identitäten-Pools siehe Erste Schritte mit Amazon Cognito Cognito-Identitätspools und die API-Referenz für Amazon-Cognito-Identitäten-Pools.

Vergleich von Amazon-Cognito-Benutzerpools und -Identitätspools

Funktion	Beschreibung	Benutzerpools	Identitäten-Pools
OIDC-IdP	Stellen Sie OIDC-ID-Token aus, um App-Benutzer zu authentif izieren	✓	
API-Autorisierungs server	Stellen Sie Zugriffst oken aus, um den Benutzerzugriff auf APIs, Datenbank en und andere Ressourcen zu autorisieren, die OAuth 2.0-Autor isierungsbereiche akzeptieren	✓	
IAM-Web-Identitäts autorisierungsserver	Generieren Sie Token, die Sie gegen temporäre AWS Anmeldein		✓

	formationen AWS STS eintauschen können		
SAML 2.0 SP und OIDC IdP	Stellen Sie benutzerdefinierte OIDC-Token auf der Grundlage von Ansprüchen eines SAML 2.0-IdP aus	✓	
OIDC SP und OIDC IdP	Stellen Sie maßgeschneiderte OIDC-Token auf der Grundlage von Ansprüchen eines OIDC-IdP aus	✓	
OAuth 2.0 SP und OIDC IdP	Stellen Sie maßgeschneiderte OIDC-Token aus, die auf den Bereichen von sozialen OAuth 2.0-Anbietern wie Apple und Google basieren	✓	
Broker für SAML 2.0 SP und Anmeldeinformationen	Stellen Sie temporäre AWS Anmeldeinformationen auf der Grundlage von Ansprüchen eines SAML 2.0-IdP aus		✓

Broker für OIDC SP und Anmeldeinformationen	Stellen Sie temporäre AWS Anmeldeinformationen auf der Grundlage von Ansprüchen eines OIDC-IdP aus	✓
OAuth 2.0-Broker für SP und Anmeldeinformationen	Stellen Sie temporäre AWS Anmeldeinformationen aus, die auf den Bereichen von sozialen OAuth 2.0-Anbietern wie Apple und Google basieren	✓
Amazon Cognito Cognito-Benutzerpool SP und Broker für Anmeldeinformationen	Stellen Sie temporäre AWS Anmeldeinformationen auf der Grundlage von OIDC-Ansprüchen aus einem Amazon Cognito Cognito-Benutzerpool aus	✓
Benutzerdefinierter Broker für SP und Anmeldeinformationen	Stellen Sie temporäre AWS Anmeldeinformationen auf der Grundlage der IAM-Autorisierung des Entwicklers aus	✓

Frontend-Dienst zur Authentifizierung	Registrieren, verwalten und authentifizieren Sie Benutzer mit der gehosteten Benutzeroberfläche	✓
API-Unterstützung für Ihre eigene Authentifizierungsoberfläche	Erstellen, verwalten und authentifizieren Sie Benutzer über API-Anfragen in unterstützten AWS SDKs[1]	✓
MFA	Verwenden Sie SMS-Nachrichten, TOTPs oder das Gerät Ihres Benutzers als zusätzlichen Authentifizierungsfaktor[1]	✓
Sicherheitsüberwachung und Reaktion	Schützen Sie sich vor böswilligen Aktivitäten und unsicheren Passworten[1]	✓
Passen Sie die Authentifizierungsabläufe an	Erstellen Sie Ihren eigenen Authentifizierungsmechanismus oder fügen Sie benutzerdefinierte Schritte zu bestehenden Abläufen hinzu[1]	✓

Gruppen	Erstellen Sie logische Gruppierungen von Benutzern und eine Hierarchie von IAM-Rollenansprüchen, wenn Sie Token an Identitätspools weitergeben	✓	
Passen Sie ID-Token an	Passen Sie Ihre ID-Token mit neuen, geänderten und unterdrückten Ansprüchen an	✓	
AWS WAF Web-ACLs	Überwachen und kontrollieren Sie Anfragen an Ihre Authentifizierungsumgebung mit AWS WAF	✓	
Passen Sie Benutzerattribute an	Weisen Sie Benutzerattributen Werte zu und fügen Sie Ihre eigenen benutzerdefinierten Attribute hinzu	✓	
Nicht authentifizierter Zugriff	Geben Sie Anmeldeinformationen für Web-Identitäten mit eingeschränktem Zugriff ohne Authentifizierung aus AWS STS		✓

Rollenbasierte Zugriffskontrolle	Wählen Sie eine IAM-Rolle für Ihren authentifizierten Benutzer auf der Grundlage seiner Ansprüche aus und konfigurieren Sie Ihre Rollen so, dass sie nur im Kontext Ihres Identitätspools übernommen werden	✓
Attributbasierte Zugriffskontrolle	Verwandeln Sie Benutzeransprüche in Prinzipal-Tags für Ihre AWS STS temporäre Sitzung und filtern Sie den Ressourcenzugriff mithilfe von IAM-Richt linien anhand von Prinzipal-Tags	✓

[1] Die Funktion ist nur für lokale Benutzer verfügbar.

Erste Schritte mit Amazon Cognito

Beispiele für Benutzerpool-Anwendungen finden Sie unterErste Schritte mit Benutzerpools.

Eine Einführung in Identitätspools finden Sie unterErste Schritte mit Amazon Cognito Cognito-Identitätspools.

Links zu Anleitungen zur Einrichtung von Benutzerpools und Identitätspools finden Sie unterGeführte Einrichtungsoptionen für Amazon Cognito.

Videos, Artikel, Dokumentation und weitere Beispielanwendungen finden Sie unter Amazon Cognito Developer Resources.

Um Amazon Cognito zu verwenden, benötigen Sie ein AWS-Konto. Weitere Informationen finden Sie unter Erste Schritte mit AWS.

Regionale Verfügbarkeit

Amazon Cognito ist in mehreren AWS Regionen weltweit verfügbar. In jeder Region wird Amazon Cognito auf mehrere Availability Zones verteilt. Diese Availability Zones sind physisch voneinander isoliert, jedoch durch private, hochredundante Netzwerkverbindungen mit geringer Latenz und hohem Durchsatz miteinander verbunden. Diese Availability Zones AWS ermöglichen die Bereitstellung von Diensten, einschließlich Amazon Cognito, mit sehr hoher Verfügbarkeit und Redundanz bei gleichzeitiger Minimierung der Latenz.

Eine Liste aller Regionen, in denen Amazon Cognito derzeit verfügbar ist, finden Sie unter AWS - Regionen und -Endpunkte in der Allgemeine Amazon Web Services-Referenz. Weitere Informationen über die in jeder Region verfügbare Anzahl von Availability Zones finden Sie unter Globale AWS - Infrastruktur.

Preise für Amazon Cognito

Weitere Informationen zu den Amazon-Cognito-Preisen finden Sie unter Preise für Amazon Cognito.

So funktioniert die Authentifizierung mit Amazon Cognito Cognito-Benutzerpools und Identitätspools

Wenn sich Ihr Kunde bei einem Amazon Cognito Cognito-Benutzerpool anmeldet, erhält Ihre Anwendung JSON-Web-Tokens (JWTs).

Wenn sich Ihr Kunde bei einem Identitätspool anmeldet, entweder mit einem Benutzerpool-Token oder einem anderen Anbieter, erhält Ihre Anwendung temporäre Anmeldeinformationen. AWS

Mit der Benutzerpool-Anmeldung können Sie die Authentifizierung und Autorisierung vollständig mit einem AWS SDK implementieren. Wenn Sie keine eigenen Benutzeroberflächenkomponenten (UI) erstellen möchten, können Sie eine vorgefertigte Weboberfläche (die gehostete Benutzeroberfläche) oder die Anmeldeseite für Ihren externen Identitätsanbieter (IdP) aufrufen.

Dieses Thema bietet einen Überblick über einige der Möglichkeiten, wie Ihre Anwendung mit Amazon Cognito interagieren kann, um sich mit ID-Token zu authentifizieren, mit Zugriffstoken zu autorisieren und AWS-Services mit Identitätspool-Anmeldeinformationen zuzugreifen.

Themen

API-Authentifizierung und Autorisierung des Benutzerpools mit einem SDK AWS

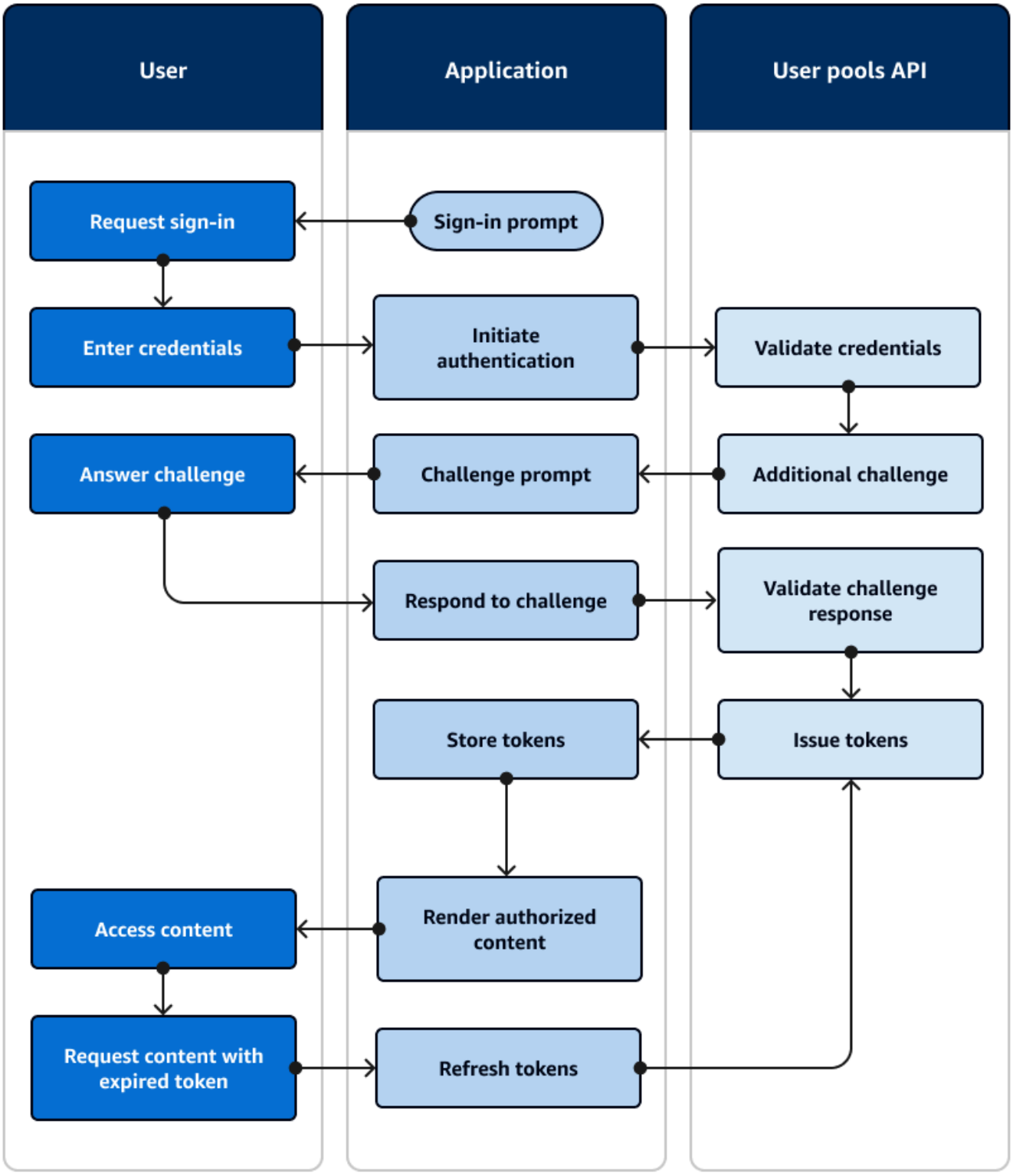
AWS hat Komponenten für Amazon Cognito Cognito-Benutzerpools oder Amazon Cognito Cognito-Identitätsanbieter in einer Vielzahl von Entwickler-Frameworks entwickelt. Die in diese SDKs integrierten Methoden rufen die Amazon Cognito Cognito-Benutzerpools-API auf. Derselbe API-Namespace für Benutzerpools enthält Operationen zur Konfiguration von Benutzerpools und zur Benutzerauthentifizierung. Eine ausführlichere Übersicht finden Sie unterVerwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte.

Die API-Authentifizierung eignet sich für das Modell, bei dem Ihre Anwendungen über vorhandene Benutzeroberflächenkomponenten verfügen und in erster Linie auf dem Benutzerpool als Benutzerverzeichnis basieren. Dieses Design fügt Amazon Cognito als Komponente innerhalb einer größeren Anwendung hinzu. Es erfordert programmatische Logik, um komplexe Herausforderungen und Reaktionen zu bewältigen.

Diese Anwendung muss keine vollständige OpenID Connect (OIDC) Relying Party Implementierung implementieren. Stattdessen ist sie in der Lage, JWTs zu dekodieren und zu verwenden. Wenn Sie Zugriff auf alle Benutzerpool-Funktionen für lokale Benutzer haben möchten, erstellen Sie Ihre Authentifizierung mit dem Amazon Cognito SDK in Ihrer Entwicklungsumgebung.

Die API-Authentifizierung mit benutzerdefinierten OAuth-Bereichen ist weniger auf die externe API-Autorisierung ausgerichtet. Um einem Zugriffstoken über die API-Authentifizierung benutzerdefinierte Bereiche hinzuzufügen, ändern Sie das Token zur Laufzeit mit einem. Lambda-Auslöser für die Vorab-Generierung von Token

Das folgende Diagramm zeigt eine typische Anmeldesitzung für die API-Authentifizierung.


User
Application
User pools API
Request sign-in
Sign-in prompt
Enter credentials
Initiate authentication
Validate credentials
Answer challenge
Challenge prompt
Additional challenge
Respond to challenge
Validate challenge response
Store tokens
Issue tokens
Access content
Render authorized content
Request content with expired token
Refresh tokens

Ablauf der API-Authentifizierung

1. Ein Benutzer greift auf Ihre Anwendung zu.
2. Er wählt einen Link „Anmelden“ aus.
3. Sie geben ihren Benutzernamen und ihr Passwort ein.
4. Die Anwendung ruft die Methode auf, die eine InitiateAuthAPI-Anfrage stellt. Die Anfrage leitet die Anmeldeinformationen des Benutzers an einen Benutzerpool weiter.
5. Der Benutzerpool validiert die Anmeldeinformationen des Benutzers und stellt fest, dass der Benutzer die Multi-Faktor-Authentifizierung (MFA) aktiviert hat.
6. Der Benutzerpool antwortet mit einer Aufforderung, die einen MFA-Code anfordert.
7. Die Anwendung generiert eine Aufforderung, die den MFA-Code vom Benutzer sammelt.
8. Die Anwendung ruft die Methode auf, die eine RespondToAuthChallengeAPI-Anfrage stellt. Die Anfrage übergibt den MFA-Code des Benutzers.
9. Der Benutzerpool validiert den MFA-Code des Benutzers.
10. Der Benutzerpool antwortet mit den JWTs des Benutzers.
11. Die Anwendung dekodiert, validiert und speichert oder speichert die JWTs des Benutzers im Cache.
12. Die Anwendung zeigt die angeforderte zugriffskontrollierte Komponente an.
13. Der Benutzer sieht sich seinen Inhalt an.
14. Später ist das Zugriffstoken des Benutzers abgelaufen und er möchte eine zugriffskontrollierte Komponente aufrufen.
15. Die Anwendung bestimmt, dass die Benutzersitzung bestehen bleiben soll. Sie ruft die InitiateAuthMethode erneut mit dem Aktualisierungstoken auf und ruft neue Token ab.

Varianten und Anpassung

Sie können diesen Ablauf durch zusätzliche Herausforderungen erweitern, z. B. durch Ihre eigenen benutzerdefinierten Authentifizierungsherausforderungen. Sie können den Zugriff automatisch für Benutzer einschränken, deren Passwörter kompromittiert wurden oder deren unerwartete Anmeldemerkmale auf einen böswilligen Anmeldeversuch hindeuten könnten. Dieser Ablauf sieht bei Vorgängen zum Registrieren, Aktualisieren von Benutzerattributen und Zurücksetzen von Kennwörtern sehr ähnlich aus. Die meisten dieser Flows haben doppelte öffentliche (clientseitige) und vertrauliche (serverseitige) API-Operationen.

Zugehörige Ressourcen

- Amazon Cognito Cognito-Benutzerpool-API
- Erste Schritte mit Benutzerpools
- Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps
- Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte

Benutzerpool-Authentifizierung mit der gehosteten Benutzeroberfläche

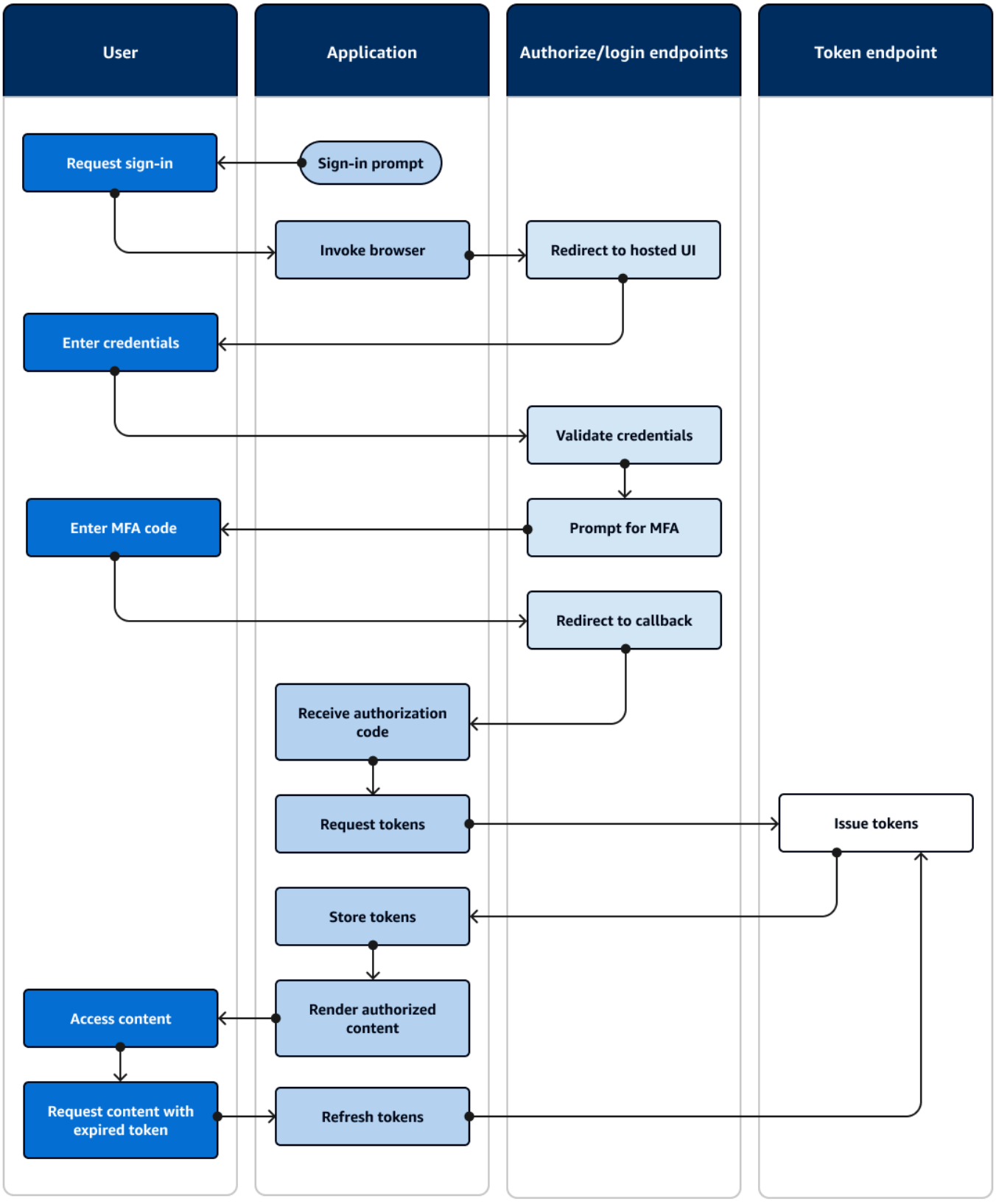
Die gehostete Benutzeroberfläche ist eine Website, die mit Ihrem Benutzerpool und App-Client verknüpft ist. Es kann Vorgänge zur Anmeldung, Registrierung und zum Zurücksetzen des Passworts für Ihre Benutzer ausführen. Die Implementierung einer Anwendung mit einer gehosteten UI-Komponente für die Authentifizierung kann weniger Aufwand durch Entwickler erfordern. Eine Anwendung kann UI-Komponenten für die Authentifizierung überspringen und die gehostete Benutzeroberfläche im Browser des Benutzers aufrufen.

Anwendungen erfassen die JWTs von Benutzern mit einer Web- oder App-Weiterleitungsadresse. Anwendungen, die die gehostete Benutzeroberfläche implementieren, können sich zur Authentifizierung mit Benutzerpools verbinden, als ob sie ein OpenID Connect (OIDC) IdP wären.

Die gehostete UI-Authentifizierung eignet sich für das Modell, bei dem Anwendungen einen Autorisierungsserver benötigen, aber keine Funktionen wie benutzerdefinierte Authentifizierung, Integration von Identitätspools oder Self-Service für Benutzerattribute benötigen. Wenn Sie einige dieser erweiterten Optionen verwenden möchten, können Sie sie mit einer Benutzerpools-Komponente für ein SDK implementieren.

Gehostete Benutzeroberflächen und IdP-Authentifizierungsmodelle von Drittanbietern, die sich hauptsächlich auf die OIDC-Implementierung verlassen, eignen sich am besten für erweiterte Autorisierungsmodelle mit OAuth 2.0-Bereichen.

Das folgende Diagramm zeigt eine typische Anmeldesitzung für die gehostete UI-Authentifizierung.


User
Application
Authorize/login endpoints
Token endpoint
Request sign-in
Sign-in prompt
Invoke browser
Redirect to hosted UI
Enter credentials
Validate credentials
Enter MFA code
Prompt for MFA
Redirect to callback
Receive authorization code
Request tokens
Issue tokens
Store tokens
Access content
Render authorized content
Request content with expired token
Refresh tokens

Ablauf der Authentifizierung über die gehostete Benutzeroberfläche

1. Ein Benutzer greift auf Ihre Anwendung zu.
2. Er wählt einen Link „Anmelden“ aus.
3. Die Anwendung leitet den Benutzer zu einer gehosteten UI-Anmeldeaufforderung weiter.
4. Sie geben ihren Benutzernamen und ihr Passwort ein.
5. Der Benutzerpool validiert die Anmeldeinformationen des Benutzers und stellt fest, dass der Benutzer die Multi-Faktor-Authentifizierung (MFA) aktiviert hat.
6. Die gehostete Benutzeroberfläche fordert den Benutzer auf, einen MFA-Code einzugeben.
7. Der Benutzer gibt seinen MFA-Code ein.
8. Die gehostete Benutzeroberfläche leitet den Benutzer zur Anwendung weiter.
9. Die Anwendung erfasst den Autorisierungscode aus dem URL-Anforderungsparameter, den die gehostete Benutzeroberfläche an die Callback-URL angehängt hat.
10. Die Anwendung fordert Token mit dem Autorisierungscode an.
11. Der Token-Endpunkt gibt JWTs an die Anwendung zurück.
12. Die Anwendung dekodiert, validiert und speichert oder speichert die JWTs des Benutzers im Cache.
13. Die Anwendung zeigt die angeforderte zugriffskontrollierte Komponente an.
14. Der Benutzer sieht sich seinen Inhalt an.
15. Später ist das Zugriffstoken des Benutzers abgelaufen und er möchte eine zugriffskontrollierte Komponente aufrufen.
16. Die Anwendung bestimmt, dass die Benutzersitzung bestehen bleiben soll. Sie fordert mit dem Aktualisierungstoken neue Token vom Token-Endpunkt an.

Varianten und Anpassung

Sie können das Erscheinungsbild der gehosteten Benutzeroberfläche mit CSS in jedem App-Client anpassen. Sie können App-Clients auch mit eigenen Identitätsanbietern, Bereichen, Zugriff auf Benutzerattribute und erweiterter Sicherheitskonfiguration konfigurieren.

Zugehörige Ressourcen

- Einrichtung und Verwendung der gehosteten Benutzeroberfläche und der Verbundendpunkte in Amazon Cognito

- Registrieren und Anmelden mit der gehosteten Benutzeroberfläche
- Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern
- Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI

Benutzerpool-Authentifizierung mit einem externen Identitätsanbieter

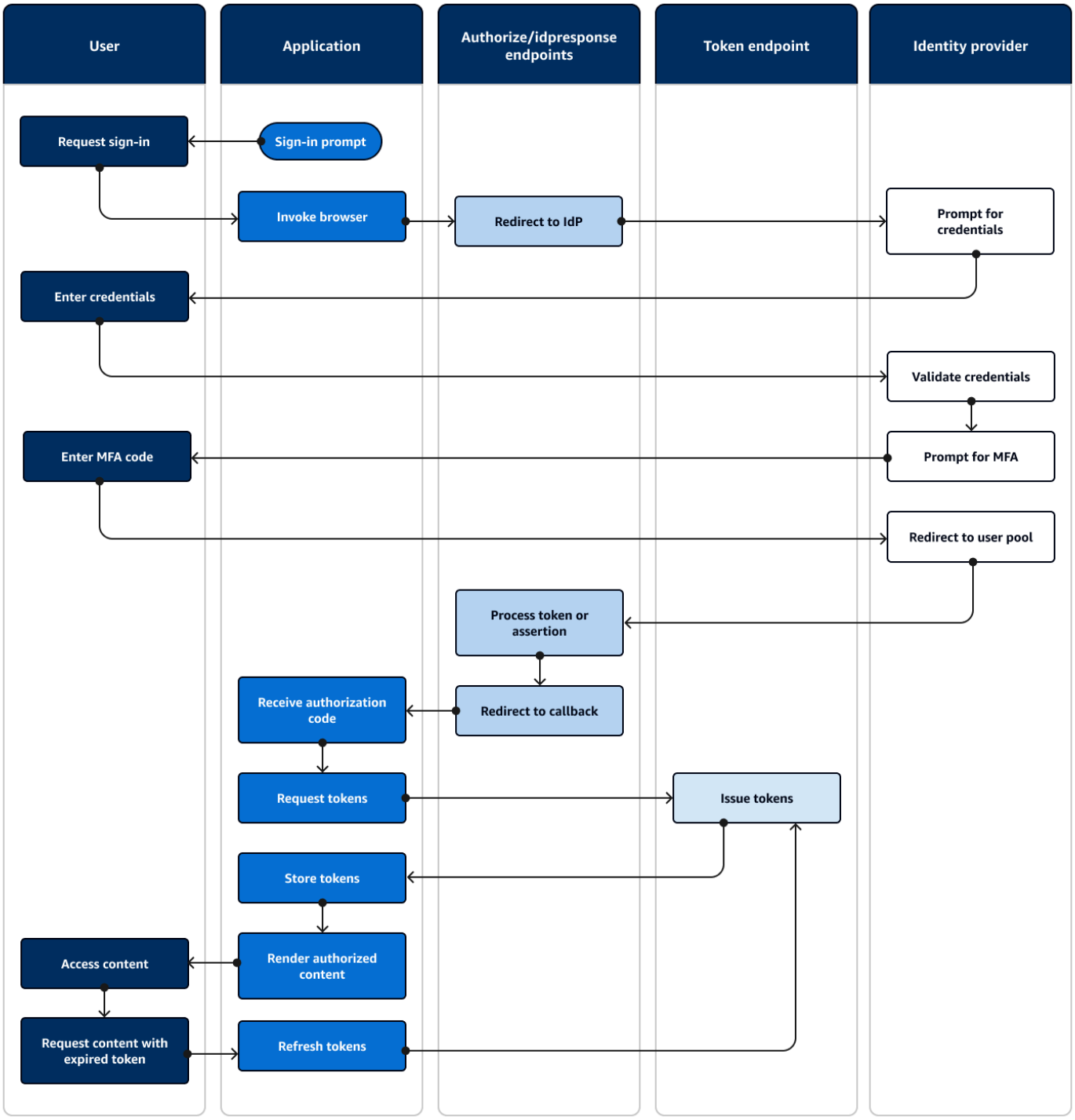
Die Anmeldung mit einem externen Identitätsanbieter (IdP) oder die Verbundauthentifizierung ist ein ähnliches Modell wie die gehostete Benutzeroberfläche. Ihre Anwendung ist eine OIDC-vertrauende Partei Ihres Benutzerpools, während Ihr Benutzerpool als Passthrough zu einem IdP dient. Der IdP kann ein Verbraucherbenutzerverzeichnis wie Facebook oder Google sein, oder es kann ein SAML 2.0- oder OIDC-Unternehmensverzeichnis wie Azure sein.

Anstatt der gehosteten Benutzeroberfläche im Browser des Benutzers ruft Ihre Anwendung einen Umleitungsendpunkt auf dem Autorisierungsserver für den Benutzerpool auf. Aus der Sicht des Benutzers wählt er die Anmeldeschaltfläche in Ihrer Anwendung aus. Dann fordert ihr IdP sie auf, sich anzumelden. Wie bei der gehosteten UI-Authentifizierung sammelt eine Anwendung JWTs an einer Weiterleitungsstelle in der App.

Die Authentifizierung mit einem Drittanbieter-IdP passt zu einem Modell, bei dem Benutzer möglicherweise kein neues Passwort eingeben möchten, wenn sie sich für Ihre Anwendung registrieren. Eine Drittanbieter-Authentifizierung kann mit geringem Aufwand zu einer Anwendung hinzugefügt werden, die die gehostete UI-Authentifizierung implementiert hat. Tatsächlich IdPs führen die gehostete Benutzeroberfläche und der Drittanbieter zu einem konsistenten Authentifizierungsergebnis, wenn geringfügige Abweichungen in den Browsern der Benutzer aufgerufen werden.

Wie die gehostete UI-Authentifizierung eignet sich die Verbundauthentifizierung am besten für erweiterte Autorisierungsmodelle mit OAuth 2.0-Bereichen.

Das folgende Diagramm zeigt eine typische Anmeldesitzung für die Verbundauthentifizierung.



Ablauf der föderierten Authentifizierung

1. Ein Benutzer greift auf Ihre Anwendung zu.
2. Er wählt einen Link „Anmelden“ aus.

3. Die Anwendung leitet den Benutzer zu einer Anmeldeaufforderung mit seinem IdP weiter.
4. Sie geben ihren Benutzernamen und ihr Passwort ein.
5. Der IdP validiert die Anmeldeinformationen des Benutzers und stellt fest, dass der Benutzer die Multi-Faktor-Authentifizierung (MFA) aktiviert hat.
6. Der IdP fordert den Benutzer auf, einen MFA-Code einzugeben.
7. Der Benutzer gibt seinen MFA-Code ein.
8. Der IdP leitet den Benutzer mit einer SAML-Antwort oder einem Autorisierungscode an den Benutzerpool weiter.
9. Wenn der Benutzer einen Autorisierungscode übergeben hat, tauscht der Benutzerpool den Code im Hintergrund gegen IdP-Token aus. Der Benutzerpool validiert die IdP-Token und leitet den Benutzer mit einem neuen Autorisierungscode zur Anwendung weiter.
10. Die Anwendung sammelt den Autorisierungscode aus dem URL-Anforderungsparameter, den der Benutzerpool an die Callback-URL angehängt hat.
11. Die Anwendung fordert Token mit dem Autorisierungscode an.
12. Der Token-Endpunkt gibt JWTs an die Anwendung zurück.
13. Die Anwendung dekodiert, validiert und speichert oder speichert die JWTs des Benutzers im Cache.
14. Die Anwendung zeigt die angeforderte zugriffskontrollierte Komponente an.
15. Der Benutzer sieht sich seinen Inhalt an.
16. Später ist das Zugriffstoken des Benutzers abgelaufen und er möchte eine zugriffskontrollierte Komponente aufrufen.
17. Die Anwendung bestimmt, dass die Benutzersitzung bestehen bleiben soll. Sie fordert mit dem Aktualisierungstoken neue Token vom Token-Endpunkt an.

Varianten und Anpassung

Sie können die Verbundauthentifizierung in der gehosteten Benutzeroberfläche initiieren, wo Benutzer aus einer Liste auswählen können IdPs , die Sie Ihrem App-Client zugewiesen haben. Die gehostete Benutzeroberfläche kann auch nach einer E-Mail-Adresse fragen und die Anfrage eines Benutzers automatisch an den entsprechenden SAML-IdP weiterleiten. Für die Authentifizierung mit einem externen Identitätsanbieter ist keine Benutzerinteraktion mit der gehosteten Benutzeroberfläche erforderlich. Ihre Anwendung kann der Autorisierungsserveranforderung eines Benutzers einen Anforderungsparameter hinzufügen und den Benutzer veranlassen, unbemerkt auf seine IdP-Anmeldeseite umzuleiten.

Zugehörige Ressourcen

- Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter
- Beispielszenario: Amazon Cognito-Apps in einem Unternehmens-Dashboard als Lesezeichen speichern
- Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern
- Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI

Authentifizierung des Identitätspools

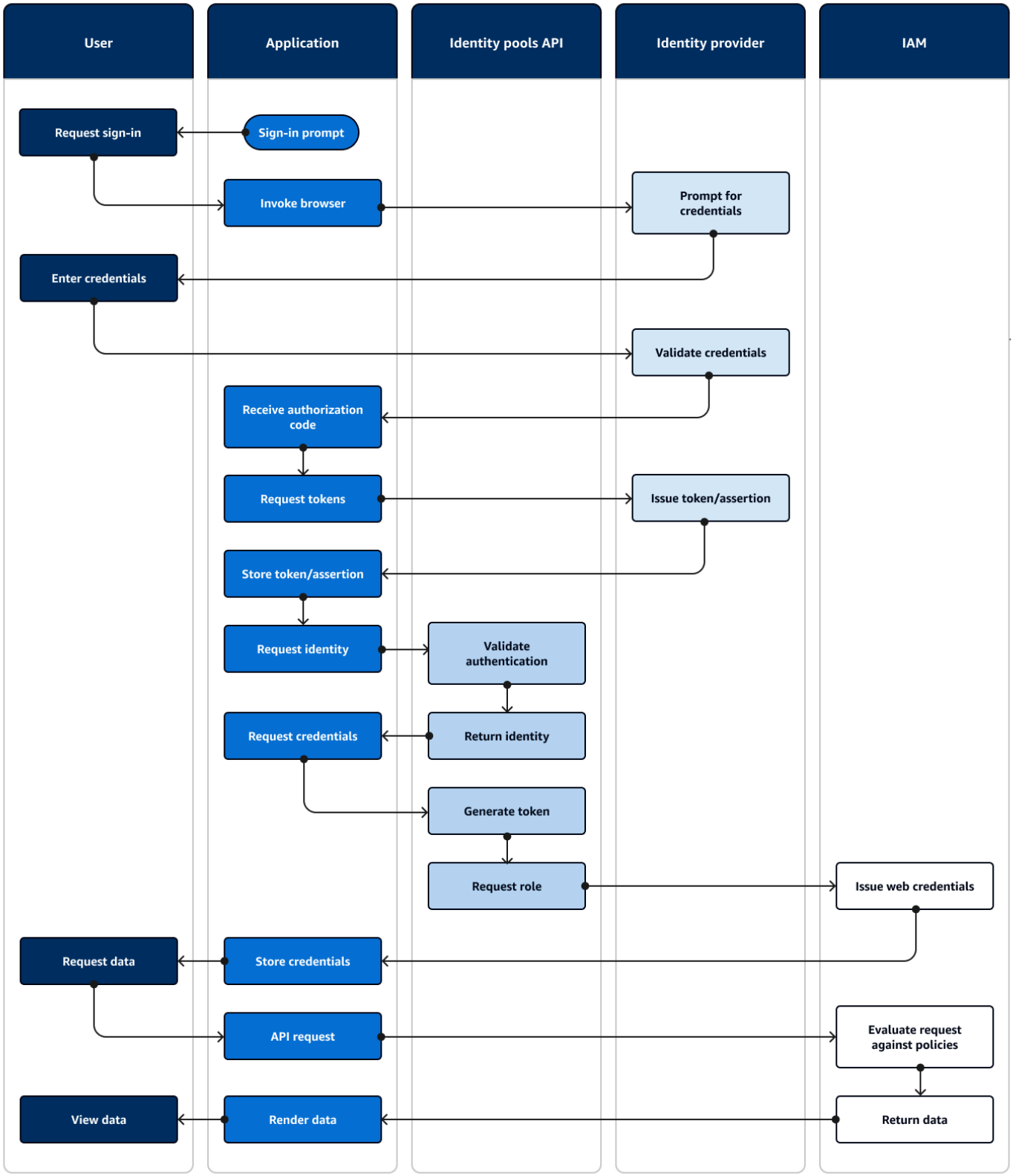
Ein Identitätspool ist eine Komponente für Ihre Anwendung, die sich in Funktion, API-Namespace und SDK-Modell von einem Benutzerpool unterscheidet. Während Benutzerpools tokenbasierte Authentifizierung und Autorisierung anbieten, bieten Identitätspools Autorisierung für AWS Identity and Access Management (IAM).

Sie können zwei Identitätspools zuweisen und Benutzer mit ihnen anmelden. IdPs Benutzerpools sind als Identitätspool eng integriert IdPs und bieten Identitätspools die meisten Optionen für die Zugriffskontrolle. Gleichzeitig gibt es eine große Auswahl an Authentifizierungsoptionen für Identitätspools. Benutzerpools verbinden SAML-, OIDC-, Social-, Entwickler- und Gast-Identitätsquellen als Routen zu temporären AWS Anmeldeinformationen aus Identitätspools.

Die Authentifizierung mit einem Identitätspool erfolgt extern — sie folgt einem der zuvor erläuterten Benutzerpool-Flows oder einem Ablauf, den Sie unabhängig von einem anderen IdP entwickeln. Nachdem Ihre Anwendung die erste Authentifizierung durchgeführt hat, leitet sie den Nachweis an einen Identitätspool weiter und erhält im Gegenzug eine temporäre Sitzung.

Die Authentifizierung mit einem Identitätspool passt zu einem Modell, bei dem Sie die Zugriffskontrolle für Anwendungsressourcen und Daten mithilfe der AWS-Services IAM-Autorisierung durchsetzen. Wie bei der API-Authentifizierung in Benutzerpools umfasst eine erfolgreiche Anwendung AWS SDKs für jeden der Dienste, auf die Sie zum Vorteil Ihrer Benutzer zugreifen möchten. AWS SDKs wenden die Anmeldeinformationen aus der Identitätspool-Authentifizierung als Signaturen auf API-Anfragen an.

Das folgende Diagramm zeigt eine typische Anmeldesitzung für die Identitätspoolauthentifizierung mit einem IdP.


User
Application
Identity pools API
Identity provider
IAM
Request sign-in
Sign-in prompt
Invoke browser
Prompt for credentials
Enter credentials
Validate credentials
Receive authorization code
Request tokens
Issue token/assertion
Store token/assertion
Request identity
Validate authentication
Request credentials
Return identity
Generate token
Request role
Issue web credentials
Request data
Store credentials
API request
Evaluate request against policies
View data
Render data
Return data

Ablauf der föderierten Authentifizierung

1. Ein Benutzer greift auf Ihre Anwendung zu.
2. Er wählt einen Link „Anmelden“ aus.
3. Die Anwendung leitet den Benutzer zu einer Anmeldeaufforderung mit seinem IdP weiter.
4. Sie geben ihren Benutzernamen und ihr Passwort ein.
5. Der IdP validiert die Anmeldeinformationen des Benutzers.
6. Der IdP leitet den Benutzer mit einer SAML-Antwort oder einem Autorisierungscode zur Anwendung weiter.
7. Wenn der Benutzer einen Autorisierungscode übergeben hat, tauscht die Anwendung den Code gegen IdP-Token aus.
8. Die Anwendung dekodiert, validiert und speichert oder zwischenspeichert die JWTs oder die Assertion des Benutzers.
9. Die Anwendung ruft die Methode auf, die eine API-Anfrage stellt. GetId Sie übergibt das Token oder die Assertion des Benutzers und fordert eine Identitäts-ID an.
10. Der Identitätspool validiert das Token oder die Assertion anhand konfigurierter Identitätsanbieter.
11. Der Identitätspool gibt eine Identitäts-ID zurück.
12. Die Anwendung ruft die Methode auf, die eine GetCredentialsForIdentityAPI-Anfrage stellt. Sie übergibt das Token oder die Assertionen des Benutzers und fordert eine IAM-Rolle an.
13. Der Identitätspool generiert ein neues JWT. Das neue JWT enthält Ansprüche, die eine IAM-Rolle anfordern. Der Identitätspool bestimmt die Rolle auf der Grundlage der Benutzeranfrage und der Rollenauswahlkriterien in der Identitätspoolkonfiguration für den IdP.
14. AWS Security Token Service (AWS STS) beantwortet die AssumeRoleWithWebIdentityAnfrage aus dem Identitätspool. Die Antwort enthält API-Anmeldeinformationen für eine temporäre Sitzung mit einer IAM-Rolle.
15. Die Anwendung speichert die Sitzungsanmeldedaten.
16. Der Benutzer führt eine Aktion in der App durch, für die zugriffsgeschützte Ressourcen erforderlich sind. AWS
17. Die Anwendung wendet die temporären Anmeldeinformationen als Signaturen auf API-Anfragen für die erforderlichen Daten an. AWS-Services
18. IAM bewertet die Richtlinien, die der Rolle in den Anmeldeinformationen zugewiesen sind. Es vergleicht sie mit der Anfrage.
19. Das AWS-Service gibt die angeforderten Daten zurück.

20 Die Anwendung rendert die Daten in der Benutzeroberfläche.

21 Der Benutzer sieht sich die Daten an.

Varianten und Anpassung

Um die Authentifizierung mit einem Benutzerpool zu visualisieren, fügen Sie nach dem Schritt Issue-Token/Assertion eine der vorherigen Benutzerpool-Übersichten ein. Bei der Entwicklerauthentifizierung werden alle Schritte vor „Identität anfordern“ durch eine mit den Anmeldeinformationen des Entwicklers signierte Anfrage ersetzt. Bei der Gastauthentifizierung wird außerdem direkt zu „Identität anfordern“ übergegangen, die Authentifizierung wird nicht validiert und es werden Anmeldeinformationen für eine IAM-Rolle mit eingeschränktem Zugriff zurückgegeben.

Zugehörige Ressourcen

- Amazon-Cognito-Identitätspools
- IAM-Rollen von Benutzern
- Identitäten-Pool-Konzepte
- Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf

Nutzungsbedingungen von Amazon Cognito

Amazon Cognito stellt Anmeldeinformationen für Web- und mobile Apps bereit. Es stützt sich auf Begriffe, die im Identitäts- und Zugriffsmanagement üblich sind, und baut auf ihnen auf. Es sind zahlreiche Leitfäden zu universellen Identitäts- und Zugangsbedingungen verfügbar. Einige Beispiele sind:

- Terminologie im IDPro Knowledge Body of Knowledge
- AWS Identitätsdienste
- Glossar von NIST CSRC

In den folgenden Listen werden Begriffe beschrieben, die nur für Amazon Cognito gelten oder einen bestimmten Kontext in Amazon Cognito haben.

Themen

- Allgemeines
- Benutzerpools

- Identitäten-Pools

Allgemeines

Die Begriffe in dieser Liste beziehen sich nicht auf Amazon Cognito und sind unter Fachleuten für Identitäts- und Zugriffsmanagement weithin anerkannt. Die folgende Liste ist keine vollständige Liste von Begriffen, sondern eine Anleitung zu ihrem spezifischen Amazon Cognito Cognito-Kontext in diesem Handbuch.

App

In der Regel eine mobile Anwendung. In diesem Handbuch ist App oft eine Abkürzung für eine Webanwendung oder mobile App, die eine Verbindung zu Amazon Cognito herstellt.

Attributbasierte Zugriffskontrolle (Attribute-Based Access Control, ABAC)

Ein Modell, bei dem eine App den Zugriff auf Ressourcen anhand der Eigenschaften eines Benutzers bestimmt, z. B. seiner Berufsbezeichnung oder Abteilung. Zu den Amazon Cognito Cognito-Tools zur Durchsetzung von ABAC gehören ID-Token in Benutzerpools und Prinzipal-Tags in Identitätspools.

Autorisierungsserver

Ein webbasiertes System, das JSON-Webtoken generiert. Die Verbundendpunkte der Amazon Cognito Cognito-Benutzerpools sind die Autorisierungsserver-Komponente der beiden Authentifizierungs- und Autorisierungsmethoden in Benutzerpools. Die andere Methode ist die Benutzerpools-API.

Vertrauliche App, serverseitige App

Eine Anwendung, mit der sich Benutzer remote verbinden, mit Code auf einem Anwendungsserver und Zugriff auf geheime Daten. Dies ist in der Regel eine Webanwendung.

Identity provider (IdP) (Identitätsanbieter (IdP))

Ein Dienst, der Benutzeridentitäten speichert und verifiziert. Amazon Cognito kann die Authentifizierung von externen Anbietern anfordern und als IdP für Apps fungieren.

JSON-Webtoken (JWT)

Ein Dokument im JSON-Format, das Behauptungen über einen authentifizierten Benutzer enthält. ID-Token authentifizieren Benutzer, Zugriffstoken autorisieren Benutzer und Aktualisierungstoken

aktualisieren Anmeldeinformationen. Amazon Cognito empfängt Token von externen Anbietern und gibt Token an Apps aus oder AWS STS.

Multi-Faktor-Authentifizierung (MFA)

Die Anforderung, dass Benutzer nach Eingabe ihres Benutzernamens und Kennworts eine zusätzliche Authentifizierung vornehmen müssen. Amazon Cognito Cognito-Benutzerpools verfügen über MFA-Funktionen für lokale Benutzer.

OAuth 2.0-Anbieter (sozial)

Ein IdP für einen Benutzerpool oder Identitätspool, der JWT-Zugriffs - und Aktualisierungstoken bereitstellt. Amazon Cognito Cognito-Benutzerpools automatisieren Interaktionen mit sozialen Anbietern, nachdem sich Benutzer authentifiziert haben.

OpenID Connect (OIDC) -Anbieter

Ein IdP für einen Benutzerpool oder Identitätspool, der die OAuth-Spezifikation um die Bereitstellung von ID-Token erweitert. Amazon Cognito Cognito-Benutzerpools automatisieren Interaktionen mit OIDC-Anbietern, nachdem sich Benutzer authentifiziert haben.

Öffentliche App

Eine eigenständige Anwendung auf einem Gerät mit lokal gespeichertem Code und ohne Zugriff auf Geheimnisse. Dies ist in der Regel eine mobile App.

Ressourcenserver

Eine API mit Zugriffskontrolle. Amazon Cognito Cognito-Benutzerpools verwenden auch den Ressourcenserver, um die Komponente zu beschreiben, die die Konfiguration für die Interaktion mit einer API definiert.

Rollenbasierte Zugriffskontrolle (RBAC)

Ein Modell, das den Zugriff auf der Grundlage der Funktionsbezeichnung eines Benutzers gewährt. Amazon Cognito Cognito-Identitätspools implementieren RBAC mit Differenzierung zwischen IAM-Rollen.

Dienstanbieter (SP), vertrauende Partei (RP)

Eine Anwendung, die sich auf einen IdP stützt, um zu bestätigen, dass Benutzer vertrauenswürdig sind. Amazon Cognito fungiert für externe IdPs SPs als SP und für App-basierte SPs als IdP.

SAML-Anbieter

Ein IdP für einen Benutzerpool oder Identitätspool, der digital signierte Assertion-Dokumente generiert, die Ihr Benutzer an Amazon Cognito weiterleitet.

Universally Unique Identifier (UUID)

Eine 128-Bit-Bezeichnung, die auf ein Objekt angewendet wird. Amazon Cognito Cognito-UUIDs sind pro Benutzerpool oder Identitätspool eindeutig.

Benutzerverzeichnis

Eine Sammlung von Benutzern und ihren Attributen, die diese Informationen anderen Systemen zur Verfügung stellt. Amazon Cognito Cognito-Benutzerpools sind Benutzerverzeichnisse und auch Tools zur Konsolidierung von Benutzern aus externen Benutzerverzeichnissen.

Benutzerpools

Wenn Sie die Begriffe in der folgenden Liste in diesem Handbuch sehen, beziehen sie sich auf eine bestimmte Funktion oder Konfiguration von Benutzerpools.

Amazon Cognito Cognito-Benutzerpool-API

Eine Reihe von API-Vorgängen für Authentifizierung und Autorisierung, die Sie Ihrer App mit einem AWS SDK hinzufügen können. Die API kann lokale Benutzer und verknüpfte Benutzer anmelden.

Adaptive Authentifizierung

Eine erweiterte Sicherheitsfunktion, die potenzielle böswillige Aktivitäten erkennt und Benutzerprofile zusätzlich schützt.

Erweiterte Sicherheitsfunktionen

Eine optionale Komponente, die Tools für die Benutzersicherheit hinzufügt.

App-Client

Eine Komponente, die die Einstellungen für einen Benutzerpool als IdP für eine App definiert.

Rückruf-URL, Umleitungs-URI

Eine Einstellung in einem App-Client und ein Parameter in Anfragen an Verbundendpunkte von Benutzerpools. Die Callback-URL ist das erste Ziel für authentifizierte Benutzer in Ihrer App.

Kompromittierte Anmeldeinformationen

Eine erweiterte Sicherheitsfunktion, die Benutzerkennwörter erkennt, die Angreifer möglicherweise kennen, und zusätzliche Sicherheit auf Benutzerprofile anwendet.

Bestätigung

Der Prozess, der feststellt, dass die Voraussetzungen erfüllt sind, damit sich ein neuer Benutzer anmelden kann. Die Bestätigung erfolgt in der Regel durch Bestätigung der E-Mail-Adresse oder Telefonnummer.

Benutzerdefinierte Authentifizierung

Eine Erweiterung der Authentifizierungsprozesse mit Lambda-Triggern, die zusätzliche Benutzerherausforderungen und -antworten definieren.

Geräteauthentifizierung

Ein Authentifizierungsprozess, der MFA durch eine Anmeldung ersetzt, die die ID eines vertrauenswürdigen Geräts verwendet.

Externer Anbieter, Drittanbieter

Ein IdP, der eine Vertrauensbeziehung zu einem Benutzerpool unterhält.

Verbundbenutzer

Ein Benutzer in einem Benutzerpool, der von einem externen Anbieter authentifiziert wurde.

Verbundendpunkte

Eine Reihe von Webseiten in Ihrer Benutzerpool-Domain, auf denen Dienste für die Interaktion mit IdPs und Apps gehostet werden.

Gehostete Benutzeroberfläche

Eine Reihe interaktiver Webseiten in Ihrer Benutzerpool-Domain, auf denen Dienste für die Benutzerauthentifizierung gehostet werden.

Lambda-Trigger

Eine Funktion AWS Lambda , bei der ein Benutzerpool automatisch an wichtigen Punkten von Benutzerauthentifizierungsprozessen aufgerufen werden kann. Sie können Lambda-Trigger verwenden, um Authentifizierungsergebnisse anzupassen.

Lokaler Benutzer

Ein Benutzerprofil im Benutzerpool-Benutzerverzeichnis, das nicht durch Authentifizierung bei einem externen Anbieter erstellt wurde.

Verlinkter Benutzer

Ein Benutzer von einem externen Anbieter, dessen Identität mit einem lokalen Benutzer zusammengeführt wird.

Anpassung von Tokens

Das Ergebnis eines Lambda-Triggers vor der Token-Generierung, der die ID oder das Zugriffstoken eines Benutzers zur Laufzeit ändert.

Benutzerpool, Amazon Cognito Cognito-Identitätsanbieter**`cognito-idp`**, Amazon Cognito Cognito-Benutzerpools

Eine AWS Ressource mit Authentifizierungs- und Autorisierungsdiensten für Anwendungen, die mit OIDC funktionieren. IdPs

Benutzerpool-Domäne

Ein Website-Name, den Sie einem Benutzerpool hinzufügen. Die Domain ist die Basis-URL für die gehostete Benutzeroberfläche und die Verbundendpunkte.

Verifizierung

Der Prozess der Bestätigung, dass ein Benutzer eine E-Mail-Adresse oder Telefonnummer besitzt. Ein Benutzerpool sendet einen Code an einen Benutzer, der eine neue E-Mail-Adresse oder Telefonnummer eingegeben hat. Wenn sie den Code an Amazon Cognito senden, verifizieren sie, dass sie Eigentümer des Nachrichtenziels sind und können zusätzliche Nachrichten aus dem Benutzerpool empfangen. Siehe auch Bestätigung.

Benutzerprofil, Benutzerkonto

Ein Eintrag für einen Benutzer im Benutzerverzeichnis. Alle Benutzer haben ein Profil in ihrem Benutzerpool.

Identitäten-Pools

Wenn Sie die Begriffe in der folgenden Liste in diesem Handbuch sehen, beziehen sie sich auf eine bestimmte Funktion oder Konfiguration von Identitätspools.

Attribute für Zugriffskontrolle

Eine Implementierung der attributbasierten Zugriffskontrolle in Identitätspools. Identitätspools wenden Benutzerattribute als Tags auf Benutzeranmeldedaten an.

Grundlegende (klassische) Authentifizierung

Ein Authentifizierungsprozess, bei dem Sie die Anforderung von Benutzeranmeldeinformationen anpassen können.

Entwicklerauthentifizierte Identitäten

Ein Authentifizierungsprozess, bei dem Benutzeranmeldedaten und Entwickleranmeldedaten für Identitätspools autorisiert werden.

Anmeldeinformationen für Entwickler

Die IAM-API-Schlüssel eines Identitätspool-Administrators.

Verbesserte Authentifizierung

Ein Authentifizierungsablauf, der eine IAM-Rolle auswählt und Prinzipal-Tags gemäß der Logik anwendet, die Sie in Ihrem Identitätspool definieren.

Identität

Eine UUID, die einen App-Benutzer und seine Benutzeranmeldeinformationen mit seinem Profil in einem externen Benutzerverzeichnis verknüpft, das eine Vertrauensstellung mit einem Identitätspool unterhält.

Identitätspool, Amazon Cognito Federated Identities, Amazon Cognito Identity, **`cognito-identity`**

Eine AWS Ressource mit Authentifizierungs- und Autorisierungsdiensten für Anwendungen, die temporäre Anmeldeinformationen verwenden. AWS

Nicht authentifizierte -Identität

Ein Benutzer, der sich nicht mit einem Identitätspool-IdP angemeldet hat. Sie können Benutzern erlauben, begrenzte Benutzeranmeldedaten für eine einzelne IAM-Rolle zu generieren, bevor sie sich authentifizieren.

Benutzeranmeldedaten

Temporäre AWS API-Schlüssel, die Benutzer nach der Identitätspool-Authentifizierung erhalten.

Verwenden Sie diesen Service mit einem SDK AWS

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

SDK-Dokumentation	Codebeispiele
AWS SDK for C++	AWS SDK for C++ Codebeispiele
AWS CLI	AWS CLI Code-Beispiele
AWS SDK for Go	AWS SDK for Go Code-Beispiele
AWS SDK for Java	AWS SDK for Java Code-Beispiele
AWS SDK for JavaScript	AWS SDK for JavaScript Code-Beispiele
AWS SDK for Kotlin	AWS SDK for Kotlin Code-Beispiele
AWS SDK for .NET	AWS SDK for .NET Code-Beispiele
AWS SDK for PHP	AWS SDK for PHP Code-Beispiele
AWS Tools for PowerShell	Tools für PowerShell Codebeispiele
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) Code-Beispiele
AWS SDK for Ruby	AWS SDK for Ruby Code-Beispiele
AWS SDK for Rust	AWS SDK for Rust Code-Beispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Code-Beispiele
AWS SDK for Swift	AWS SDK for Swift Code-Beispiele

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Feedback geben auswählen.

Erste Schritte mit AWS

Bevor Sie mit der Arbeit mit Amazon Cognito beginnen, sollten Sie sich mit einigen erforderlichen AWS Ressourcen vertraut machen.

Melden Sie sich an für ein AWS-Konto

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

Um sich für eine anzumelden AWS-Konto

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

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

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

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

Erstellen Sie einen Benutzer mit Administratorzugriff

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

Sichern Sie Ihre Root-Benutzer des AWS-Kontos

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

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

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

Erstellen Sie einen Benutzer mit Administratorzugriff

1. Aktivieren Sie das IAM Identity Center.

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

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

Melden Sie sich als Benutzer mit Administratorzugriff an

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

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

Weisen Sie weiteren Benutzern Zugriff zu

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

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

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

Erste Schritte mit Benutzerpools

Sie können die Anleitungen in diesem Abschnitt verwenden, um Ihre ersten Benutzerpool-Ressourcen zu erstellen. Für eine step-by-step exemplarische Vorgehensweise beginnen Sie mit einer einfachen Webanwendung in der JavaScript React-Entwicklerumgebung. Von dort aus können Sie weitere Funktionen wie die gehostete Benutzeroberfläche (gehostete Benutzeroberfläche) und die föderierte Anmeldung mit externen Social - oder SAML 2.0-Identitätsanbietern () hinzufügen. IdPs

Während Sie daran arbeiten, Ihren Funktionsumfang zu erweitern und weitere Komponenten von Amazon Cognito zu integrieren, lesen Sie das Kapitel Amazon Cognito Cognito-Benutzerpools. Dort finden Sie eine vollständige Beschreibung aller Möglichkeiten, die Sie mit Benutzerpools tun können.

Das Beispiel für einen Benutzerpool und eine Anwendung in diesem Abschnitt zeigt eine grundlegende Integration von Anwendungsressourcen mit Amazon Cognito Cognito-Benutzerpools. Später können Sie Ihren Benutzerpool anpassen, um mehr Optionen zu nutzen, die Ihnen zur Verfügung stehen. Anschließend können Sie Ihre Anwendung aktualisieren, um neue APIs zu übernehmen und mit der gehosteten Benutzeroberfläche zu interagieren und IdPs.

Das Tutorial in diesem Abschnitt erstellt eine Anwendung mit einer benutzerdefinierten Benutzeroberfläche und API-basierter Authentifizierung mit einem AWS SDK. Anwendungen, die Sie auf diese Weise erstellen, eignen sich ideal für die Authentifizierung lokaler Benutzer. Um mit einer Anwendung mit einer vordefinierten Benutzeroberfläche, automatischer Verarbeitung einiger Benutzerpoolfunktionen und Authentifizierung von Verbundbenutzern zu beginnen, fahren Sie fort mit. Fügen Sie einen App-Client mit der gehosteten Benutzeroberfläche hinzu

Themen

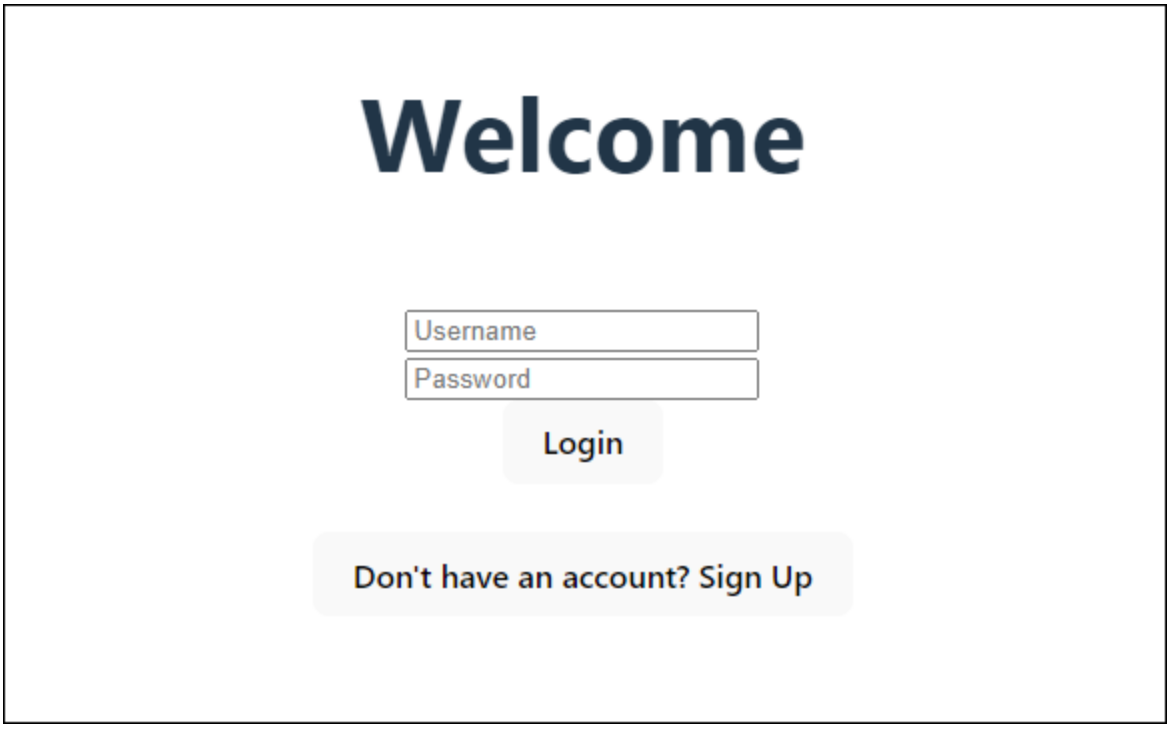
- Richten Sie eine einseitige React-Beispielanwendung ein
- Richten Sie eine Beispiel-Android-App mit Flutter ein
- Nächste Schritte

Richten Sie eine einseitige React-Beispielanwendung ein

In diesem Tutorial erstellen Sie eine einseitige React-Anwendung, in der Sie die Benutzeranmeldung, Bestätigung und Anmeldung testen können. React ist eine JavaScript basierte Bibliothek für Web- und mobile Apps, deren Schwerpunkt auf der Benutzeroberfläche (UI) liegt. Diese Beispielanwendung demonstriert einige grundlegende Funktionen von Amazon Cognito Cognito-

Benutzerpools. Wenn Sie bereits Erfahrung in der Entwicklung von Web-Apps mit React haben, laden Sie die Beispiel-App von GitHub herunter.

Der folgende Screenshot zeigt die erste Authentifizierungsseite in der Anwendung, die Sie erstellen werden.



Mit der Prozedur Benutzerpool erstellen richten Sie einen Benutzerpool ein, der mit der Beispielanwendung funktioniert. Sie können diesen Schritt überspringen, wenn Sie über einen Benutzerpool verfügen, der die folgenden Anforderungen erfüllt:

- Benutzer können sich mit ihrer E-Mail-Adresse anmelden. Anmeldeoptionen für den Cognito-Benutzerpool: E-Mail.
- Bei Benutzernamen wird nicht zwischen Groß- und Kleinschreibung unterschieden. Anforderungen an Benutzernamen: Bei Benutzernamen zwischen Groß- und Kleinschreibung unterscheiden ist nicht ausgewählt.
- Eine Multi-Faktor-Authentifizierung (MFA) ist nicht erforderlich. MFA-Durchsetzung: Optionales MFA.
- Ihr Benutzerpool überprüft die Attribute für die Bestätigung des Benutzerprofils mit einer E-Mail-Nachricht. Zu verifizierende Attribute: E-Mail-Nachricht senden, E-Mail-Adresse verifizieren.
- E-Mail ist das einzige erforderliche Attribut. Erforderliche Attribute: E-Mail.

- Benutzer können sich selbst in Ihrem Benutzerpool anmelden. Selbstregistrierung: Selbstregistrierung aktivieren ist ausgewählt.
- Ihr erster App-Client ist ein öffentlicher Client, der die Anmeldung mit Benutzername und Passwort ermöglicht. App-Typ: Öffentlicher Client, Authentifizierungsabläufe:`ALLOW_USER_PASSWORD_AUTH`.

Erstellen eines Benutzerpools

Erstellen Sie einen neuen Benutzerpool

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie die Schaltfläche Benutzerpool erstellen. Möglicherweise müssen Sie im linken Navigationsbereich Benutzerpools auswählen, um diese Option anzuzeigen.
3. Wählen Sie in der rechten oberen Ecke der Seite Create a user pool (Benutzerpool erstellen) aus, um den Assistenten zur Erstellung von Benutzerpools zu starten.
4. Unter Anmeldevorgang konfigurieren können Sie die Identitätsanbieter (IdPs) auswählen, die Sie mit diesem Benutzerpool verwenden möchten. Weitere Informationen finden Sie unter Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter.
   a. Stellen Sie unter Authentifizierungsanbieter für Anbietertypen sicher, dass nur der Cognito-Benutzerpool ausgewählt ist.
   b. Wählen Sie für die Anmeldeoptionen für den Cognito-Benutzerpool die Option Benutzername aus. Wählen Sie keine zusätzlichen Anforderungen für den Benutzernamen aus.
   c. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.
5. Unter Sicherheitsanforderungen konfigurieren können Sie Ihre Passwortrichtlinie, die Anforderungen für die Multi-Faktor-Authentifizierung (MFA) und die Wiederherstellungsoptionen für Benutzerkonten auswählen. Weitere Informationen finden Sie unter Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools.
   a. Stellen Sie für die Passwortrichtlinie sicher, dass der Passwortrichtlinienmodus auf die Cognito-Standardeinstellungen eingestellt ist.
   b. Wählen Sie unter Multi-Faktor-Authentifizierung für die MFA-Durchsetzung die Option Optionales MFA aus.

c. Wählen Sie für MFA-Methoden die Option Authenticator-Apps und SMS-Nachricht.

d. Vergewissern Sie sich, dass für die Wiederherstellung von Benutzerkonten die Option Self-Service-Kontowiederherstellung aktivieren ausgewählt ist und dass die Nachrichtenübermittlungsmethode für die Benutzerkontowiederherstellung auf Nur E-Mail eingestellt ist.

e. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

6. Unter Anmeldevorgang konfigurieren können Sie festlegen, wie neue Benutzer ihre Identität überprüfen, wenn sie sich als neuer Benutzer registrieren, und welche Attribute bei der Benutzerregistrierung erforderlich oder optional sein sollen. Weitere Informationen finden Sie unter Verwalten von Benutzern in Ihrem Benutzerpool.

   a. Vergewissern Sie sich, dass Selbstregistrierung aktivieren ausgewählt ist. Diese Einstellung öffnet Ihren Benutzerpool, sodass Sie sich von jedem Benutzer im Internet anmelden können. Dies ist für die Zwecke der Beispielanwendung vorgesehen. Wenden Sie diese Einstellung jedoch in Produktionsumgebungen mit Vorsicht an.

   b. Vergewissern Sie sich, dass unter Cognito-gestützte Überprüfung und Bestätigung das Kontrollkästchen Cognito das automatische Senden von Nachrichten zur Überprüfung und Bestätigung zulassen aktiviert ist.

   c. Vergewissern Sie sich, dass für Zu überprüfende Attribute die Option E-Mail-Nachricht senden, E-Mail-Adresse verifizieren ausgewählt ist.

   d. Vergewissern Sie sich, dass unter Attributänderungen überprüfen die Standardoptionen ausgewählt sind: Ursprünglichen Attributwert beibehalten, wenn ein Update aussteht, und Aktive Attributwerte, wenn ein Update aussteht auf E-Mail-Adresse gesetzt ist.

   e. Vergewissern Sie sich, dass unter Erforderliche Attribute, die auf vorherigen Auswahlen basieren, E-Mail angezeigt wird.

   > **Important**
   >
   > Für diese Beispielanwendung darf Ihr Benutzerpool phone_number nicht als erforderliches Attribut festlegen. Wenn phone_number als erforderliches Attribut angezeigt wird, überprüfen und aktualisieren Sie Ihre vorherigen Einstellungen:
   >
   > - Optionales MFA, nur E-Mail als Zustellungsmethode für Nachrichten zur Wiederherstellung von Benutzerkonten
   > - E-Mail-Nachricht senden, E-Mail-Adresse für zu verifizierende Attribute verifizieren

f. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

7. Unter Nachrichtenzustellung konfigurieren können Sie die Integration mit Amazon Simple Email Service und Amazon Simple Notification Service konfigurieren, um Ihren Benutzern E-Mail- und SMS-Nachrichten zur Registrierung, Kontobestätigung, MFA und Kontowiederherstellung zu senden. Weitere Informationen finden Sie unter E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools und Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools.

   a. Wählen Sie für E-Mail-Anbieter die Option E-Mail mit Cognito senden und verwenden Sie den von Amazon Cognito bereitgestellten Standard-E-Mail-Absender. Diese Einstellung für ein niedriges E-Mail-Volumen ist für Anwendungstests ausreichend. Sie können zurückkehren, nachdem Sie eine E-Mail-Adresse mit Amazon Simple Email Service (Amazon SES) bestätigt und E-Mail mit Amazon SES senden ausgewählt haben.

   b. Wählen Sie für SMS die Option Neue IAM-Rolle erstellen aus und geben Sie einen IAM-Rollennamen ein. Dadurch wird eine Rolle erstellt, die Amazon Cognito Berechtigungen zum Senden von SMS-Nachrichten gewährt.

   c. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

8. Unter Integrieren Sie Ihre App können Sie Ihrem Benutzerpool einen Namen geben, die gehostete Benutzeroberfläche konfigurieren und einen App-Client erstellen. Weitere Informationen finden Sie unter Fügen Sie einen App-Client mit der gehosteten Benutzeroberfläche hinzu. Die Beispielanwendungen verwenden die gehostete Benutzeroberfläche nicht.

   a. Geben Sie unter Benutzerpoolname einen Namen für den Benutzerpool ein.

   b. Wählen Sie nicht Die von Cognito gehostete Benutzeroberfläche verwenden aus.

   c. Vergewissern Sie sich unter Erster App-Client, dass der App-Typ auf Öffentlicher Client eingestellt ist.

   d. Vergewissern Sie sich, dass unter Geheimer Client die Option Keinen geheimen Clientschlüssel generieren ausgewählt ist.

   e. Geben Sie einen App-Client-Namen ein.

   f. Erweitern Sie Erweiterte App-Client-Einstellungen. `ALLOW_USER_PASSWORD_AUTH`Zur Liste der Authentifizierungsabläufe hinzufügen.

   g. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

9. Überprüfen Sie Ihre Auswahl im Bildschirm Überprüfen und erstellen und ändern Sie alle Auswahlen nach Bedarf. Wenn Sie mit Ihrer Benutzerpool-Konfiguration zufrieden sind, wählen Sie Benutzerpool erstellen, um fortzufahren.

10. Wählen Sie auf der Seite Benutzerpools Ihren neuen Benutzerpool aus.
11. Notieren Sie sich unter Benutzerpool-Übersicht Ihre Benutzerpool-ID. Sie geben diese Zeichenfolge an, wenn Sie Ihre Beispielanwendung erstellen.
12. Wählen Sie die Registerkarte App-Integration und suchen Sie den Abschnitt App-Clients und Analysen. Wählen Sie Ihren neuen App-Client aus. Notieren Sie sich Ihre Kunden-ID.

Zugehörige Ressourcen

- Amazon-Cognito-Benutzerpools
- Ablauf der Authentifizierung in Benutzerpools
- Verwenden von Token mit Benutzerpools

Erstellen einer Anwendung

Um diese Anwendung zu erstellen, müssen Sie eine Entwicklerumgebung einrichten. Die Anforderungen an die Entwicklerumgebung lauten wie folgt:

1. Node.js ist installiert und aktualisiert.
2. Der Node Package Manager (npm) ist installiert und auf mindestens Version 10.2.3 aktualisiert.
3. Auf die Umgebung kann über den TCP-Port 5173 in einem Webbrowser zugegriffen werden.

Um eine React-Beispiel-Webanwendung zu erstellen

1. Melden Sie sich in Ihrer Entwicklerumgebung an und navigieren Sie zum übergeordneten Verzeichnis für Ihre Anwendung.

   ```
   cd ~/path/to/project/folder/
   ```

2. Erstellen Sie einen neuen React-Dienst.

   ```
   npm create vite@latest frontend-client -- --template react-ts
   ```

3. Klonen Sie den `cognito-developer-guide-react-example` Projektordner aus dem AWS Codebeispiel-Repository auf GitHub.

   ```
   cd ~/some/other/path
   ```

```
git clone https://github.com/awsdocs/aws-doc-sdk-examples.git
```

```
cp -r ./aws-doc-sdk-examples/javascriptv3/example_code/cognito-identity-provider/
scenarios/cognito-developer-guide-react-example/frontend-client ~/path/to/project/
folder/frontend-client
```

4. Navigieren Sie zu dem `src` Verzeichnis in Ihrem Projekt.

```
cd ~/path/to/project/folder/frontend-client/src
```

5. Bearbeiten `config.ts` und ersetzen Sie die folgenden Werte:

   a. Durch `YOUR_AWS_REGION` einen AWS-Region Code ersetzen. Zum Beispiel: `us-east-1`.

   b. `YOUR_COGNITO_USER_POOL_ID`Ersetzen Sie es durch die ID des Benutzerpools, den Sie zum Testen bestimmt haben. Zum Beispiel: `us-east-1_EXAMPLE`. Der Benutzerpool muss sich in dem befinden AWS-Region , den Sie im vorherigen Schritt eingegeben haben.

   c. `YOUR_COGNITO_APP_CLIENT_ID`Ersetzen Sie es durch die ID des App-Clients, den Sie zum Testen bestimmt haben. Zum Beispiel: `1example23456789`. Der App-Client muss sich im Benutzerpool aus dem vorherigen Schritt befinden.

6. Wenn Sie von einer anderen IP als auf Ihre Beispielanwendung zugreifen möchten`localhost`, bearbeiten Sie die Zeile `package.json` und ändern `"dev": "vite",` Sie sie in`"dev": "vite --host 0.0.0.0",`.

7. Installieren Sie Ihre Anwendung.

```
npm install
```

8. Starten Sie die Anwendung.

```
npm run dev
```

9. Greifen Sie in einem Webbrowser unter `http://localhost:5173` oder auf die Anwendung zu`http://[IP address]:5173`.

10. Melden Sie einen neuen Benutzer mit einer gültigen E-Mail-Adresse an.

11. Rufen Sie den Bestätigungscode aus Ihrer E-Mail-Nachricht ab. Geben Sie den Bestätigungscode in die Anwendung ein.

12. Melden Sie sich mit Ihrem Benutzernamen und Passwort an.

Erstellen einer React-Entwicklungsumgebung mit Amazon Lightsail

Eine schnelle Möglichkeit, mit dieser Anwendung zu beginnen, besteht darin, einen virtuellen Cloud-Server mit Amazon Lightsail zu erstellen.

Mit Lightsail können Sie schnell eine kleine Serverinstanz erstellen, die mit den Voraussetzungen für diese Beispielanwendung vorkonfiguriert geliefert wird. Sie können mit einem browserbasierten Client eine SSH-Verbindung zu Ihrer Instanz herstellen und über eine öffentliche oder private IP-Adresse eine Verbindung zum Webserver herstellen.

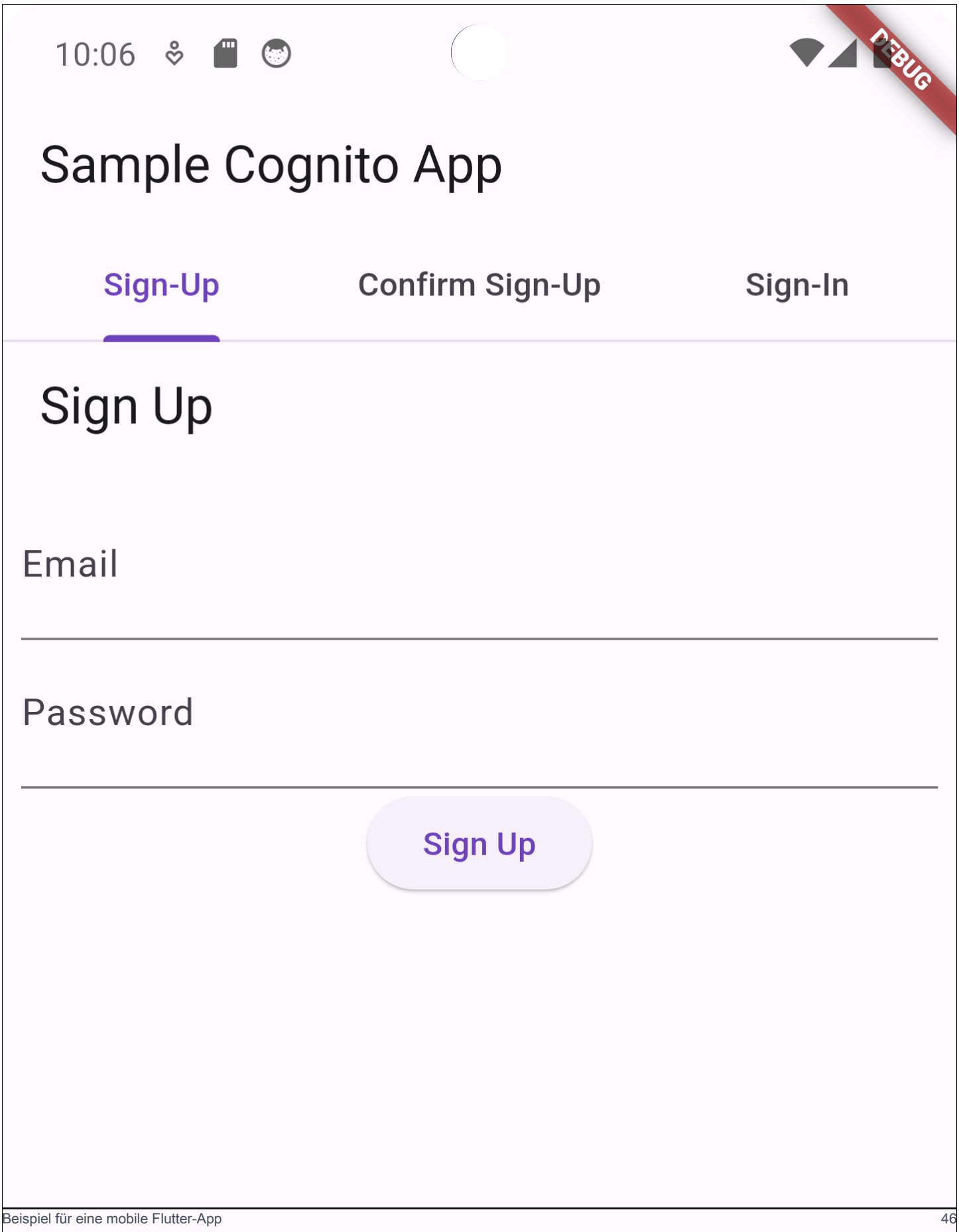
Um eine Lightsail-Instanz für diese Beispielanwendung zu erstellen

1. Gehen Sie zur Lightsail-Konsole. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie Create instance (Instance erstellen).
3. Wählen Sie für Wählen Sie eine Plattform die Option Linux/Unix aus.
4. Wählen Sie für Select a blueprint die Option Node.js aus.
5. Geben Sie Ihrer Entwicklungsumgebung unter Identifizieren Sie Ihre Instanz einen benutzerfreundlichen Namen.
6. Wählen Sie Create instance (Instance erstellen).
7. Nachdem Lightsail Ihre Instanz erstellt hat, wählen Sie sie aus und wählen Sie auf der Registerkarte Connect die Option Connect using SSH aus.
8. Eine SSH-Sitzung wird in einem Browserfenster geöffnet. Führen Sie das `npm -v` Programm aus `node -v` und bestätigen Sie, dass Ihre Instanz mit Node.js und der npm-Mindestversion 10.2.3 bereitgestellt wurde.
9. Fahren Sie mit der Konfiguration Ihrer React-Anwendung fort.

Richten Sie eine Beispiel-Android-App mit Flutter ein

In diesem Tutorial erstellen Sie eine mobile Anwendung in Android Studio, mit der Sie ein Gerät emulieren und die Benutzeranmeldung, Bestätigung und Anmeldung testen können. Diese Beispielanwendung erstellt einen einfachen mobilen Amazon Cognito Cognito-Benutzerpool-Client für Android in Flutter. Wenn Sie bereits Erfahrung mit der Entwicklung mobiler Apps mit Flutter haben, laden Sie die Beispiel-App von herunter. GitHub

Der folgende Screenshot zeigt, wie die App auf einem virtuellen Android-Gerät läuft.


10:06
DEBUG
Sample Cognito App
Sign-Up
Confirm Sign-Up
Sign-In
Sign Up
Email
Password
Sign Up

Mit dem Verfahren Benutzerpool erstellen richten Sie einen Benutzerpool ein, der mit der Beispielanwendung funktioniert. Sie können diesen Schritt überspringen, wenn Sie über einen Benutzerpool verfügen, der die folgenden Anforderungen erfüllt:

- Benutzer können sich mit ihrer E-Mail-Adresse anmelden. Anmeldeoptionen für den Cognito-Benutzerpool: E-Mail.
- Bei Benutzernamen wird nicht zwischen Groß- und Kleinschreibung unterschieden. Anforderungen an Benutzernamen: Bei Benutzernamen zwischen Groß- und Kleinschreibung unterscheiden ist nicht ausgewählt.
- Eine Multi-Faktor-Authentifizierung (MFA) ist nicht erforderlich. MFA-Durchsetzung: Optionales MFA.
- Ihr Benutzerpool überprüft die Attribute für die Bestätigung des Benutzerprofils mit einer E-Mail-Nachricht. Zu verifizierende Attribute: E-Mail-Nachricht senden, E-Mail-Adresse verifizieren.
- E-Mail ist das einzige erforderliche Attribut. Erforderliche Attribute: E-Mail.
- Benutzer können sich selbst in Ihrem Benutzerpool anmelden. Selbstregistrierung: Selbstregistrierung aktivieren ist ausgewählt.
- Ihr erster App-Client ist ein öffentlicher Client, der die Anmeldung mit Benutzername und Passwort ermöglicht. App-Typ: Öffentlicher Client, Authentifizierungsabläufe:`ALLOW_USER_PASSWORD_AUTH`.

Erstellen eines Benutzerpools

Erstellen Sie einen neuen Benutzerpool

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie die Schaltfläche Benutzerpool erstellen. Möglicherweise müssen Sie im linken Navigationsbereich Benutzerpools auswählen, um diese Option anzuzeigen.
3. Wählen Sie in der rechten oberen Ecke der Seite Create a user pool (Benutzerpool erstellen) aus, um den Assistenten zur Erstellung von Benutzerpools zu starten.
4. Unter Anmeldevorgang konfigurieren können Sie die Identitätsanbieter (IdPs) auswählen, die Sie mit diesem Benutzerpool verwenden möchten. Weitere Informationen finden Sie unter Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter.

a. Stellen Sie unter Authentifizierungsanbieter für Anbietertypen sicher, dass nur der Cognito-Benutzerpool ausgewählt ist.

b. Wählen Sie für die Anmeldeoptionen für den Cognito-Benutzerpool die Option Benutzername aus. Wählen Sie keine zusätzlichen Anforderungen für den Benutzernamen aus.

c. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

5. Unter Sicherheitsanforderungen konfigurieren können Sie Ihre Passwortrichtlinie, die Anforderungen für die Multi-Faktor-Authentifizierung (MFA) und die Wiederherstellungsoptionen für Benutzerkonten auswählen. Weitere Informationen finden Sie unter Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools.

   a. Stellen Sie für die Passwortrichtlinie sicher, dass der Passwortrichtlinienmodus auf die Cognito-Standardeinstellungen eingestellt ist.

   b. Wählen Sie unter Multi-Faktor-Authentifizierung für die MFA-Durchsetzung die Option Optionales MFA aus.

   c. Wählen Sie für MFA-Methoden die Option Authenticator-Apps und SMS-Nachricht.

   d. Vergewissern Sie sich, dass für die Wiederherstellung von Benutzerkonten die Option Self-Service-Kontowiederherstellung aktivieren ausgewählt ist und dass die Nachrichtenübermittlungsmethode für die Benutzerkontowiederherstellung auf Nur E-Mail eingestellt ist.

   e. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

6. Unter Anmeldevorgang konfigurieren können Sie festlegen, wie neue Benutzer ihre Identität überprüfen, wenn sie sich als neuer Benutzer registrieren, und welche Attribute bei der Benutzerregistrierung erforderlich oder optional sein sollen. Weitere Informationen finden Sie unter Verwalten von Benutzern in Ihrem Benutzerpool.

   a. Vergewissern Sie sich, dass Selbstregistrierung aktivieren ausgewählt ist. Diese Einstellung öffnet Ihren Benutzerpool, sodass Sie sich von jedem Benutzer im Internet anmelden können. Dies ist für die Zwecke der Beispielanwendung vorgesehen. Wenden Sie diese Einstellung jedoch in Produktionsumgebungen mit Vorsicht an.

   b. Vergewissern Sie sich, dass unter Cognito-gestützte Überprüfung und Bestätigung das Kontrollkästchen Cognito das automatische Senden von Nachrichten zur Überprüfung und Bestätigung zulassen aktiviert ist.

c. Vergewissern Sie sich, dass für Zu überprüfende Attribute die Option E-Mail-Nachricht senden, E-Mail-Adresse verifizieren ausgewählt ist.

d. Vergewissern Sie sich, dass unter Attributänderungen überprüfen die Standardoptionen ausgewählt sind: Ursprünglichen Attributwert beibehalten, wenn ein Update aussteht, und Aktive Attributwerte, wenn ein Update aussteht auf E-Mail-Adresse gesetzt ist.

e. Vergewissern Sie sich, dass unter Erforderliche Attribute, die auf vorherigen Auswahlen basieren, E-Mail angezeigt wird.

> **Important**
>
> Für diese Beispielanwendung darf Ihr Benutzerpool phone_number nicht als erforderliches Attribut festlegen. Wenn phone_number als erforderliches Attribut angezeigt wird, überprüfen und aktualisieren Sie Ihre vorherigen Einstellungen:
>
> - Optionales MFA, nur E-Mail als Zustellungsmethode für Nachrichten zur Wiederherstellung von Benutzerkonten
> - E-Mail-Nachricht senden, E-Mail-Adresse für zu verifizierende Attribute verifizieren

f. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

7. Unter Nachrichtenzustellung konfigurieren können Sie die Integration mit Amazon Simple Email Service und Amazon Simple Notification Service konfigurieren, um Ihren Benutzern E-Mail- und SMS-Nachrichten zur Registrierung, Kontobestätigung, MFA und Kontowiederherstellung zu senden. Weitere Informationen finden Sie unter E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools und Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools.

   a. Wählen Sie für E-Mail-Anbieter die Option E-Mail mit Cognito senden und verwenden Sie den von Amazon Cognito bereitgestellten Standard-E-Mail-Absender. Diese Einstellung für ein niedriges E-Mail-Volumen ist für Anwendungstests ausreichend. Sie können zurückkehren, nachdem Sie eine E-Mail-Adresse mit Amazon Simple Email Service (Amazon SES) bestätigt und E-Mail mit Amazon SES senden ausgewählt haben.

   b. Wählen Sie für SMS die Option Neue IAM-Rolle erstellen aus und geben Sie einen IAM-Rollennamen ein. Dadurch wird eine Rolle erstellt, die Amazon Cognito Berechtigungen zum Senden von SMS-Nachrichten gewährt.

   c. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

8. Unter Integrieren Sie Ihre App können Sie Ihrem Benutzerpool einen Namen geben, die gehostete Benutzeroberfläche konfigurieren und einen App-Client erstellen.

Weitere Informationen finden Sie unter Fügen Sie einen App-Client mit der gehosteten Benutzeroberfläche hinzu. Die Beispielanwendungen verwenden die gehostete Benutzeroberfläche nicht.

a. Geben Sie unter Benutzerpoolname einen Namen für den Benutzerpool ein.

b. Wählen Sie nicht Die von Cognito gehostete Benutzeroberfläche verwenden aus.

c. Vergewissern Sie sich unter Erster App-Client, dass der App-Typ auf Öffentlicher Client eingestellt ist.

d. Vergewissern Sie sich, dass unter Geheimer Client die Option Keinen geheimen Clientschlüssel generieren ausgewählt ist.

e. Geben Sie einen App-Client-Namen ein.

f. Erweitern Sie Erweiterte App-Client-Einstellungen. `ALLOW_USER_PASSWORD_AUTH`Zur Liste der Authentifizierungsabläufe hinzufügen.

g. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

9. Überprüfen Sie Ihre Auswahl im Bildschirm Überprüfen und erstellen und ändern Sie alle Auswahlen nach Bedarf. Wenn Sie mit Ihrer Benutzerpool-Konfiguration zufrieden sind, wählen Sie Benutzerpool erstellen, um fortzufahren.
10. Wählen Sie auf der Seite Benutzerpools Ihren neuen Benutzerpool aus.
11. Notieren Sie sich unter Benutzerpool-Übersicht Ihre Benutzerpool-ID. Sie geben diese Zeichenfolge an, wenn Sie Ihre Beispielanwendung erstellen.
12. Wählen Sie die Registerkarte App-Integration und suchen Sie den Abschnitt App-Clients und Analysen. Wählen Sie Ihren neuen App-Client aus. Notieren Sie sich Ihre Kunden-ID.

Zugehörige Ressourcen

- Amazon-Cognito-Benutzerpools
- Ablauf der Authentifizierung in Benutzerpools
- Verwenden von Token mit Benutzerpools

Erstellen einer Anwendung

Um eine Beispiel-App für Android zu erstellen

1. Installieren Sie Android Studio und die Befehlszeilentools.

2. Installieren Sie in Android Studio das Flutter-Plugin.
3. Erstellen Sie ein neues Android Studio-Projekt aus dem Inhalt des `cognito_flutter_mobile_app` Verzeichnisses in dieser Beispiel-App.
   - Bearbeiten `assets/config.json` und ersetzen Sie `<<YOUR USER POOL ID>>` und `<< YOUR CLIENT ID>>` durch die IDs des Benutzerpools und App-Clients, die Sie zuvor erstellt haben.
4. Installieren Sie Flutter.
   a. Fügen Sie Flutter zu Ihrer PATH-Variablen hinzu.
   b. Akzeptieren Sie Lizenzen mit dem folgenden Befehl.

      ```
      flutter doctor --android-licenses
      ```

   c. Überprüfen Sie Ihre Flutter-Umgebung und installieren Sie alle fehlenden Komponenten.

      ```
      flutter doctor
      ```

      - Wenn Komponenten fehlen, starten Sie, `flutter doctor -v` um zu erfahren, wie Sie das Problem beheben können.

   d. Wechseln Sie in das Verzeichnis Ihres neuen Flutter-Projekts und installieren Sie Abhängigkeiten.

      - Führen Sie `flutter pub add amazon_cognito_identity_dart_2`.

   e. Führen Sie `flutter pub add flutter_secure_storage`.
5. Erstellen Sie ein virtuelles Android-Gerät.
   1. Erstellen Sie in der Android Studio-GUI ein neues Gerät mit dem Gerätemanager.
   2. Führen Sie in der CLI den Befehl aus`flutter emulators --create --name android-device`.
6. Starten Sie Ihr virtuelles Android-Gerät.
   1. Wählen Sie in der Android Studio-GUI das

      Startsymbol neben Ihrem virtuellen Gerät aus.
   2. Führen Sie in der CLI den Befehl aus`flutter emulators --launch android-device`.
7. Starten Sie Ihre App auf Ihrem virtuellen Gerät.

1. Wählen Sie in der Android Studio-GUI das

   Bereitstellungssymbol aus.
2. Führen Sie in der CLI den Befehl aus`flutter run`.

8. Navigieren Sie in Android Studio zu Ihrem laufenden virtuellen Gerät.
9. Registriere einen neuen Benutzer mit einer gültigen E-Mail-Adresse.
10. Rufen Sie den Bestätigungscode aus Ihrer E-Mail-Nachricht ab. Geben Sie den Bestätigungscode in die Anwendung ein.
11. Melden Sie sich mit Ihrem Benutzernamen und Passwort an.

Nächste Schritte

Nachdem Sie die Anleitungen zur Fertigstellung der Beispielanwendungen befolgt haben, können Sie den Umfang Ihrer Benutzerpool-Implementierung erweitern. Sie können zusätzliche Benutzerpools erstellen, Benutzerpoolfunktionen für andere Anwendungen anpassen oder externe Identitätsanbieter hinzufügen. Bei der Planung Ihrer Umstellung auf Amazon Cognito Cognito-Benutzerpools in Produktionsanwendungen können Sie weitere Beispiele und Tutorials auswerten.

Im Folgenden sind einige zusätzliche Funktionen von Amazon Cognito Cognito-Benutzerpools aufgeführt:

- Anpassen der integrierten Registrierungs- und Anmeldungswebseiten
- Hinzufügen der MFA zu einem Benutzerpool
- Hinzufügen erweiterter Sicherheit zu einem Benutzerpool.
- Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern
- Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools

Einen Überblick über die Authentifizierungs- und Autorisierungsmodelle von Amazon Cognito finden Sie unterSo funktioniert die Authentifizierung mit Amazon Cognito Cognito-Benutzerpools und Identitätspools.

Informationen zum Zugriff auf andere AWS-Services nach einer erfolgreichen Benutzerpool-Authentifizierung finden Sie unterZugriff AWS-Services über einen Identitätspool nach der Anmeldung.

Sie können nicht nur die SDKs AWS Management Console und den Benutzerpool verwenden, sondern auch Ihre Benutzerpools mithilfe der AWS Command Line Interfaceverwalten.

Themen

Erstellen Sie einen neuen Benutzerpool

Mit einem Benutzerpool können sich Ihre Benutzer über Amazon Cognito bei Ihrer Web- oder mobilen Anwendung anmelden.

Erstellen Sie einen neuen Benutzerpool

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie die Schaltfläche Benutzerpool erstellen. Möglicherweise müssen Sie im linken Navigationsbereich Benutzerpools auswählen, um diese Option anzuzeigen.
3. Wählen Sie in der rechten oberen Ecke der Seite Create a user pool (Benutzerpool erstellen) aus, um den Assistenten zur Erstellung von Benutzerpools zu starten.
4. Unter Anmeldevorgang konfigurieren können Sie die Identitätsanbieter (IdPs) auswählen, die Sie mit diesem Benutzerpool verwenden möchten. Weitere Informationen finden Sie unter Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter.

    a. Stellen Sie unter Authentifizierungsanbieter für Anbietertypen sicher, dass nur der Cognito-Benutzerpool ausgewählt ist.

    b. Wählen Sie für die Anmeldeoptionen für den Cognito-Benutzerpool die Option Benutzername aus. Wählen Sie keine zusätzlichen Anforderungen für den Benutzernamen aus.

    c. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.
5. Unter Sicherheitsanforderungen konfigurieren können Sie Ihre Passwortrichtlinie, die Anforderungen für die Multi-Faktor-Authentifizierung (MFA) und die Wiederherstellungsoptionen für Benutzerkonten auswählen. Weitere Informationen finden Sie unter Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools.

a. Stellen Sie für die Passwortrichtlinie sicher, dass der Passwortrichtlinienmodus auf die Cognito-Standardeinstellungen eingestellt ist.

b. Wählen Sie unter Multi-Faktor-Authentifizierung für die MFA-Durchsetzung die Option Optionales MFA aus.

c. Wählen Sie für MFA-Methoden die Option Authenticator-Apps und SMS-Nachricht.

d. Vergewissern Sie sich, dass für die Wiederherstellung von Benutzerkonten die Option Self-Service-Kontowiederherstellung aktivieren ausgewählt ist und dass die Nachrichtenübermittlungsmethode für die Benutzerkontowiederherstellung auf Nur E-Mail eingestellt ist.

e. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

6. Unter Anmeldevorgang konfigurieren können Sie festlegen, wie neue Benutzer ihre Identität überprüfen, wenn sie sich als neuer Benutzer registrieren, und welche Attribute bei der Benutzerregistrierung erforderlich oder optional sein sollen. Weitere Informationen finden Sie unter Verwalten von Benutzern in Ihrem Benutzerpool.

   a. Vergewissern Sie sich, dass Selbstregistrierung aktivieren ausgewählt ist. Diese Einstellung öffnet Ihren Benutzerpool, sodass Sie sich von jedem Benutzer im Internet anmelden können. Dies ist für die Zwecke der Beispielanwendung vorgesehen. Wenden Sie diese Einstellung jedoch in Produktionsumgebungen mit Vorsicht an.

   b. Vergewissern Sie sich, dass unter Cognito-gestützte Überprüfung und Bestätigung das Kontrollkästchen Cognito das automatische Senden von Nachrichten zur Überprüfung und Bestätigung zulassen aktiviert ist.

   c. Vergewissern Sie sich, dass für Zu überprüfende Attribute die Option E-Mail-Nachricht senden, E-Mail-Adresse verifizieren ausgewählt ist.

   d. Vergewissern Sie sich, dass unter Attributänderungen überprüfen die Standardoptionen ausgewählt sind: Ursprünglichen Attributwert beibehalten, wenn ein Update aussteht, und Aktive Attributwerte, wenn ein Update aussteht auf E-Mail-Adresse gesetzt ist.

   e. Vergewissern Sie sich, dass unter Erforderliche Attribute, die auf vorherigen Auswahlen basieren, E-Mail angezeigt wird.

   > **Important**
   >
   > Für diese Beispielanwendung darf Ihr Benutzerpool phone_number nicht als erforderliches Attribut festlegen. Wenn phone_number als erforderliches Attribut angezeigt wird, überprüfen und aktualisieren Sie Ihre vorherigen Einstellungen:

- Optionales MFA, nur E-Mail als Zustellungsmethode für Nachrichten zur Wiederherstellung von Benutzerkonten
- E-Mail-Nachricht senden, E-Mail-Adresse für zu verifizierende Attribute verifizieren

f. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

7. Unter Nachrichtenzustellung konfigurieren können Sie die Integration mit Amazon Simple Email Service und Amazon Simple Notification Service konfigurieren, um Ihren Benutzern E-Mail- und SMS-Nachrichten zur Registrierung, Kontobestätigung, MFA und Kontowiederherstellung zu senden. Weitere Informationen finden Sie unter E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools und Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools.

   a. Wählen Sie für E-Mail-Anbieter die Option E-Mail mit Cognito senden und verwenden Sie den von Amazon Cognito bereitgestellten Standard-E-Mail-Absender. Diese Einstellung für ein niedriges E-Mail-Volumen ist für Anwendungstests ausreichend. Sie können zurückkehren, nachdem Sie eine E-Mail-Adresse mit Amazon Simple Email Service (Amazon SES) bestätigt und E-Mail mit Amazon SES senden ausgewählt haben.

   b. Wählen Sie für SMS die Option Neue IAM-Rolle erstellen aus und geben Sie einen IAM-Rollennamen ein. Dadurch wird eine Rolle erstellt, die Amazon Cognito Berechtigungen zum Senden von SMS-Nachrichten gewährt.

   c. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

8. Unter Integrieren Sie Ihre App können Sie Ihrem Benutzerpool einen Namen geben, die gehostete Benutzeroberfläche konfigurieren und einen App-Client erstellen. Weitere Informationen finden Sie unter Fügen Sie einen App-Client mit der gehosteten Benutzeroberfläche hinzu. Die Beispielanwendungen verwenden die gehostete Benutzeroberfläche nicht.

   a. Geben Sie unter Benutzerpoolname einen Namen für den Benutzerpool ein.

   b. Wählen Sie nicht Die von Cognito gehostete Benutzeroberfläche verwenden aus.

   c. Vergewissern Sie sich unter Erster App-Client, dass der App-Typ auf Öffentlicher Client eingestellt ist.

   d. Vergewissern Sie sich, dass unter Geheimer Client die Option Keinen geheimen Clientschlüssel generieren ausgewählt ist.

   e. Geben Sie einen App-Client-Namen ein.

   f. Erweitern Sie Erweiterte App-Client-Einstellungen. `ALLOW_USER_PASSWORD_AUTH`Zur Liste der Authentifizierungsabläufe hinzufügen.

g. Behalten Sie alle anderen Optionen als Standard bei und wählen Sie Weiter.

9. Überprüfen Sie Ihre Auswahl im Bildschirm Überprüfen und erstellen und ändern Sie alle Auswahlen nach Bedarf. Wenn Sie mit Ihrer Benutzerpool-Konfiguration zufrieden sind, wählen Sie Benutzerpool erstellen, um fortzufahren.
10. Wählen Sie auf der Seite Benutzerpools Ihren neuen Benutzerpool aus.
11. Notieren Sie sich unter Benutzerpool-Übersicht Ihre Benutzerpool-ID. Sie geben diese Zeichenfolge an, wenn Sie Ihre Beispielanwendung erstellen.
12. Wählen Sie die Registerkarte App-Integration und suchen Sie den Abschnitt App-Clients und Analysen. Wählen Sie Ihren neuen App-Client aus. Notieren Sie sich Ihre Kunden-ID.

So erstellen Sie einen Benutzerpool:

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie in der rechten oberen Ecke der Seite Create a user pool (Benutzerpool erstellen) aus, um den Assistenten zur Erstellung von Benutzerpools zu starten.
4. In Configure sign-in experience (Anmeldeerlebnis konfigurieren) wählen Sie die Verbundanbieter aus, die Sie mit diesem Benutzerpool verwenden möchten. Weitere Informationen finden Sie unter Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter.
5. Wählen Sie in Configure security requirements (Konfigurieren der Sicherheitsanforderungen) die Anforderungen an Ihre Passwortrichtlinie und Multi-Faktor-Authentifizierung (MFA) aus sowie die Wiederherstellungsoptionen für Benutzerkonten. Weitere Informationen finden Sie unter Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools.
6. Legen Sie unter Configure sign-up experience (Konfigurieren des Anmeldeerlebnisses) fest, wie neue Benutzer ihre Identitäten bei der Anmeldung überprüfen und welche Attribute während des Anmeldeflusses des Benutzers erforderlich oder optional sein sollten. Weitere Informationen finden Sie unter Verwalten von Benutzern in Ihrem Benutzerpool.

> **Important**
>
> Wenn Sie die Benutzerregistrierung in Ihrem Benutzerpool aktivieren, kann sich jeder im Internet für ein Konto registrieren und bei Ihren Apps anmelden. Aktivieren Sie die Selbstregistrierung in Ihrem Benutzerpool nur dann, wenn Sie die öffentliche Registrierung für Ihre App aktivieren möchten. Um diese Einstellung

zu ändern, aktualisieren Sie die Self-Service-Registrierung auf der Registerkarte Anmeldevorgang der Benutzerpool-Konsole oder aktualisieren Sie den Wert von AllowAdminCreateUserOnlyin einer CreateUserPooloder UpdateUserPoolAPI-Anfrage. Hinweise zu Sicherheitsfunktionen, die Sie in Ihren Benutzerpools einrichten können, finden Sie unter Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools.

7. Konfigurieren Sie unter Configure message delivery (Nachrichtenübermittlung konfigurieren) die Integration mit Amazon Simple Email Service und Amazon Simple Notification Service, um E-Mail- und SMS-Nachrichten an Ihre Benutzer zur Anmeldung, Kontobestätigung, MFA und Kontowiederherstellung zu senden. Weitere Informationen finden Sie unter E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools und Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools.
8. Benennen Sie unter Integrate your app (Anwendung integrieren) Ihren Benutzerpool, konfigurieren Sie die gehostete Benutzeroberfläche und erstellen Sie einen App-Client. Weitere Informationen finden Sie unter Fügen Sie einen App-Client mit der gehosteten Benutzeroberfläche hinzu.
9. Überprüfen Sie Ihre Auswahl im Bildschirm Überprüfen und erstellen und ändern Sie alle Auswahlen nach Bedarf. Wenn Sie mit Ihrer Benutzerpool-Konfiguration zufrieden sind, wählen Sie Benutzerpool erstellen aus, um fortzufahren.

Zugehörige Ressourcen

Weitere Informationen zu Benutzerpools finden Sie unter Amazon-Cognito-Benutzerpools.

Siehe auch: Ablauf der Authentifizierung in Benutzerpools undVerwenden von Token mit Benutzerpools.

Fügen Sie einen App-Client mit der gehosteten Benutzeroberfläche hinzu

Nachdem Sie einen Benutzerpool erstellt haben, können Sie einen App-Client für eine Anwendung erstellen, der die integrierten Webseiten der gehosteten Benutzeroberfläche aufruft. In der gehosteten Benutzeroberfläche können Benutzer:

- Melden Sie sich für ein Benutzerprofil an.
- Melden Sie sich bei einem externen Identitätsanbieter an.
- Melden Sie sich mit oder ohne Multi-Faktor-Authentifizierung an.

- Setze ihr Passwort zurück.

Um einen App-Client für die gehostete Benutzeroberfläche zu erstellen, melden Sie sich an

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool. Wenn Sie einen neuen Benutzerpool erstellen, werden Sie aufgefordert, einen App-Client einzurichten und die gehostete Benutzeroberfläche mit dem Assistenten zu konfigurieren.
4. Navigieren Sie zur Registerkarte App integration (Anwendungsintegration) für Ihren Benutzerpool.
5. Wählen Sie neben Domäne Aktionen aus und dann entweder Benutzerdefinierte Domäne erstellen oder Amazon-Cognito-Domäne erstellen. Wenn Sie bereits eine Benutzerpool-Domain konfiguriert haben, wählen Sie Amazon-Cognito-Domäne löschen oder Benutzerdefinierte Domäne löschen aus, bevor Sie Ihre neue benutzerdefinierte Domain erstellen.
6. Geben Sie ein verfügbares Domain-Präfix zur Verwendung mit einer Amazon-Cognito-Domäne ein. Weitere Informationen zum Einrichten einer Custom domain (Benutzerdefinierte Domäne) finden Sie unter Verwenden der eigenen Domäne für die gehostete Benutzeroberfläche.
7. Wählen Sie Create (Erstellen) aus.
8. Navigieren Sie für den gleichen Benutzerpool zurück zur Registerkarte App integration (Anwendungsintegration) und suchen Sie nach App-Clients. Wählen Sie Create an app client (App-Client erstellen) aus.
9. Wählen Sie einen Anwendungstyp aus. Einige empfohlene Einstellungen werden basierend auf Ihrer Auswahl bereitgestellt. Eine App, die die gehostete Benutzeroberfläche verwendet, ist einöffentlicher Client
10. Geben Sie einen App-Client-Namen ein.
11. Wählen Sie für diese Übung Don't generate client secret (Kein Clientgeheimnis generieren) aus. Das Clientgeheimnis wird von vertraulichen Apps verwendet, die Benutzer aus einer zentralisierten Anwendung authentifizieren. In dieser Übung präsentieren Sie Ihren Benutzern eine Anmeldeseite für die gehostete Benutzeroberfläche und benötigen kein Clientgeheimnis.
12. Wählen Sie die Authentifizierungsabläufe aus, die Sie mit Ihrer App zulassen möchten. Stellen Sie sicher, dass `USER_SRP_AUTH` ausgewählt wurde.

13. Passen Sie den Token-Ablauf, erweiterte Sicherheitskonfigurationen und Lese- und Schreibberechtigungen für Attribute nach Bedarf an. Weitere Informationen finden Sie unter Konfigurieren der App-Client-Einstellungen.
14. Fügen Sie eine Rückruf-URL für Ihren App-Client hinzu. Hier werden Sie zur gehosteten UI-Authentifizierung weitergeleitet. Sie müssen keine URL für die zulässige Abmeldung hinzufügen, bis Sie die Abmeldung in Ihrer App implementieren können.

    Bei einer iOS- oder Android-App, können Sie eine Rückruf-URL wie zum Beispiel verwenden `myapp://`.
15. Wählen Sie die Identitätsanbieter für den App-Client aus. Aktivieren Sie mindestens Amazon-Cognito-Benutzerpool als Anbieter.

    > **Note**
    >
    > Um sich bei externen Identitätsanbietern (IdPs) wie Facebook, Amazon, Google und Apple sowie über OpenID Connect (OIDC) oder SAML anzumelden IdPs, konfigurieren Sie diese zunächst wie unter Benutzerpool-Anmeldung über einen Drittanbieter hinzufügen beschrieben. Kehren Sie dann zur Seite mit den App-Client-Einstellungen zurück, um sie zu aktivieren.

16. Wählen Sie OAuth 2.0 Grant Types (OAuth 2.0 Erteilungstypen) aus. Wählen Sie Authorization code grant (Autorisierungscodegewährung), um einen Autorisierungscode auszugeben, der dann gegen die Benutzerpool-Tokens ausgetauscht wird. Da diese Tokens niemals einem Endbenutzer direkt gezeigt werden, sind sie weniger anfällig gegen Angriffe. Allerdings ist am Backend eine benutzerdefinierte Anwendung erforderlich, um den Autorisierungscode gegen Benutzerpool-Tokens austauschen zu können. Aus Sicherheitsgründen empfehlen wir Ihnen, den Ablauf für die Autorisierungscodegewährung in Verbindung mit Proof Key for Code Exchange (PKCE) für mobile Apps zu verwenden.

    Wählen Sie Implizite Gewährung, damit Sie von Amazon Cognito Benutzerpool-JSON-Web-Tokens (JWT) erhalten. Sie können diesen Ablauf verwenden, wenn kein Backend für den Austausch eines Autorisierungscodes gegen Tokens vorhanden ist. Er ist auch für das Debugging von Tokens nützlich.

> **Note**
>
> Sie können Authorization code grant (Autorisierungscodegewährung) und Implicit code grant (Implizite Codegewährung) aktivieren und dann beide Gewährungen nach Bedarf verwenden.
> Wählen Sie Client credentials (Client-Anmeldeinformationen) nur dann, wenn Ihre App Zugriffstoken für sich und nicht für einen Benutzer anfordern muss.

17. Sofern Sie nicht ausdrücklich etwas ausschließen möchten, wählen Sie alle OpenID-Connect-Bereiche aus.
18. Wählen Sie alle benutzerdefinierten Bereiche aus, die Sie konfiguriert haben. Benutzerdefinierte Bereiche werden normalerweise mit vertraulichen Clients verwendet.
19. Wählen Sie Create (Erstellen) aus.

So zeigen Sie Ihre Anmeldeseite an

Wählen Sie auf Ihrer App-Client-Seite die Option Gehostete Benutzeroberfläche anzeigen aus, um einen neuen Browser-Tab mit einer Anmeldeseite zu öffnen, auf der bereits die Parameter App-Client-ID, Bereich, Grant und Callback-URL aufgeführt sind.

Sie können die gehostete UI-Anmeldewebseite manuell mit der folgenden URL aufrufen. beachten Sie den `response_type`. In diesem Fall ist dies response_type=code für die Autorisierungscodegewährung.

```
https://your_domain/login?
response_type=code&client_id=your_app_client_id&redirect_uri=your_callback_url
```

Sie können die gehosteten UI-Anmelde-Webseite mit der folgenden URL für die implizite Codegewährung anzeigen, wenn response_type = Token. Nach einer erfolgreichen Anmeldung gibt Amazon Cognito Benutzerpool-Token in die Adresszeile Ihres Webbrowsers aus.

```
https://your_domain/login?
response_type=token&client_id=your_app_client_id&redirect_uri=your_callback_url
```

Sie finden die JSON Web Token (JWT)-Identitätstoken hinter dem Parameter `#idtoken=` in der Antwort.

Die folgende URL ist ein Beispiel für eine Antwort von einer impliziten Erteilungsanforderung. Ihre Identitätstoken-Zeichenfolge wird viel länger sein.

```
https://www.example.com/
#id_token=123456789tokens123456789&expires_in=3600&token_type=Bearer
```

Benutzerpool-Tokens von Amazon Cognito werden unter Verwendung eines RS256-Algorithmus signiert. Sie können Benutzerpool-Token mithilfe von dekodieren und verifizieren. AWS Lambda Weitere Informationen finden Sie auf der Website unter Amazon Cognito JWT-Token dekodieren und verifizieren. AWS GitHub

Ihre Domäne wird auf der Seite Domain name (Domänenname) angezeigt. Ihre App-Client-ID und die Callback-URL werden auf der Seite General settings (Allgemeine Einstellungen) angezeigt. Wenn die Änderungen, die Sie in der Konsole vorgenommen haben, nicht sofort angezeigt werden, warten Sie ein paar Minuten und aktualisieren Sie dann Ihren Browser.

Hinzufügen von Social Sign-in zu einem Benutzerpool (optional)

Sie können Ihren App-Benutzern ermöglichen, sich über einen Social Identity-Anbieter (IdP) wie beispielsweise Facebook, Google, Amazon oder Apple anzumelden. Unabhängig davon, ob Ihre Benutzer sich direkt oder über einen Drittanbieter anmelden, haben alle Benutzer ein Benutzerprofil im Benutzerpool. Überspringen Sie diesen Schritt, wenn keine Anmeldung über einen Social-Anmeldungsidentitätsanbieter hinzufügen möchten.

Registrieren mit einem Social-Identity-Anbieter

Bevor Sie einen soziale IdP mit Amazon Cognito anlegen, müssen Sie Ihre Anwendung bei dem sozialen IdP registrieren, um eine Kunden-ID und einen geheimen Client-Schlüssel zu erhalten.

Eine App bei Facebook registrieren

1. Erstellen Sie ein Entwickler-Konto bei Facebook.
2. Melden Sie sich mit Ihren Facebook-Anmeldeinformationen an.
3. Wählen Sie im Menü My Apps (Meine Apps) den Eintrag Create New App (Neue App erstellen).

Wenn du noch keine Facebook-App hast, wird dir eine andere Option angezeigt. Wählen Sie Create app (App erstellen).

4. Wählen Sie auf der Seite zum Erstellen einer App einen Anwendungsfall für Ihre App aus und klicken Sie dann auf Next (Weiter).
5. Geben Sie einen Namen für Ihre Facebook-App ein und wählen Sie dann Create App (App erstellen) aus.
6. Wählen Sie in der linken Navigationsleiste App Settings (App-Einstellungen) und klicken Sie dann auf Basic (Grundlegend).
7. Notieren Sie die App ID und das App Secret (Geheimer Schlüssel für die App). Sie brauchen diese Informationen im nächsten Abschnitt.
8. Wählen Sie unten auf der Seite + Add Platform (+ Plattform hinzufügen).
9. Wählen Sie auf dem Bildschirm „Plattform auswählen“ Ihre Plattformen aus und klicken Sie dann auf Weiter.
10. Wählen Sie Save Changes.
11. Geben Sie für App Domains (App-Domänen) Ihre Benutzerpool-Domäne ein.

    ```
    https://your_user_pool_domain
    ```

12. Wählen Sie Save Changes.
13. Wählen Sie in der Navigationsleiste „Produkte“ und anschließend „Über Facebook-Anmeldung konfigurieren“ aus.
14. Wählen Sie im Menü Facebook Login (Facebook-Anmeldung) unter Configure (Konfigurieren) die Option Settings (Einstellungen) aus.

    Geben Sie Ihre Umleitungs-URL in Valid OAuth Redirect URIs (Gültige Umleitungs-URIs für OAuth) ein. Die Weiterleitungs-URL besteht aus Ihrer Benutzerpool-Domain mit dem `/oauth2/idpresponse` Endpunkt.

    ```
    https://your_user_pool_domain/oauth2/idpresponse
    ```

15. Wählen Sie Save Changes.

Eine App bei Amazon registrieren

1. Erstellen Sie ein Entwickler-Konto bei Amazon.

2. Melden Sie sich mit Ihren Amazon-Anmeldeinformationen an.
3. Sie müssen ein Amazon Sicherheitsprofil erstellen, um die Amazon-Client-ID und den geheimen Client-Schlüssel zu erhalten.

   Wählen Sie in der Navigationsleiste oben auf der Seite Apps und Dienste aus und wählen Sie dann Login with Amazon aus.
4. Wählen Sie Create a Security Profile (Ein Sicherheitsprofil erstellen) aus.
5. Geben Sie einen Security Profile Name (Sicherheitsprofilnamen), eine Security Profile Description (Sicherheitsprofilbeschreibung) und eine Consent Privacy Notice URL (URL zur Zustimmung zum Datenschutzhinweis) ein.
6. Wählen Sie Save (Speichern) aus.
7. Wählen Sie Client ID (Client-ID) und Client Secret (Clientschlüssel), um die Client-ID und den Clientschlüssel anzuzeigen. Sie brauchen diese Informationen im nächsten Abschnitt.
8. Bewegen Sie den Mauszeiger über das Zahnrad, wählen Sie Web Settings (Web-Einstellungen) und dann Edit (Bearbeiten) aus.
9. Geben Sie Ihre Benutzerpool-Domäne in Allowed Origins (Autorisierte Quellen) ein.

   ```
   https://<your-user-pool-domain>
   ```
10. Geben Sie Ihre Benutzerpool-Domäne mit dem `/oauth2/idpresponse`-Endpunkt in Allowed Return URLs (Zulässige Rückgabe-URLs) ein.

    ```
    https://<your-user-pool-domain>/oauth2/idpresponse
    ```
11. Wählen Sie Save (Speichern) aus.

Eine App bei Google registrieren

Weitere Informationen über OAuth 2.0 auf der Google-Cloud-Plattform finden Sie unter Erfahren Sie mehr über Authentifizierung und Autorisierung in der Dokumentation zu Google Workspace für Entwickler.

1. Erstellen Sie ein Entwickler-Konto bei Google.
2. Melden Sie sich bei der Konsole für Google Cloud Platform an.

3. Klicken Sie in der oberen Navigationsleiste auf Select a project (Projekt auswählen). Wenn Sie bereits ein Projekt auf der Google-Plattform haben, zeigt dieses Menü stattdessen Ihr Standardprojekt an.
4. Wählen Sie NEW PROJECT (NEUES PROJEKT) aus.
5. Geben Sie einen Namen für Ihr Projekt ein und wählen Sie dann CREATE (ERSTELLEN) aus.
6. Wählen Sie in der linken Navigationsleiste APIs and Services und anschließend Oauth-Zustimmungsbildschirm aus.
7. Geben Sie die App-Informationen, eine App-Domain, autorisierte Domains und Kontaktinformationen für Entwickler ein. Ihre autorisierten Domains müssen `amazoncognito.com` den Stamm Ihrer benutzerdefinierten Domain enthalten. Zum Beispiel: `example.com`. Wählen Sie SAVE AND CONTINUE (SPEICHERN UND FORTFAHREN) aus.
8. 1. Wählen Sie unter Bereiche die Option Bereiche hinzufügen oder entfernen und wählen Sie dann mindestens die folgenden OAuth-Bereiche aus.

    1. `.../auth/userinfo.email`
    2. `.../auth/userinfo.profile`
    3. openid
9. Wählen Sie unter Test users (Testbenutzer) die Option Add users (Benutzer hinzufügen) aus. Geben Sie Ihre E-Mail-Adresse und alle anderen autorisierten Testbenutzer ein und wählen Sie dann SPEICHERN UND FORTFAHREN.
10. Erweitern Sie die linke Navigationsleiste erneut, wählen Sie APIs and Services und dann Credentials aus.
11. Wählen Sie CREATE CREDENTIALS und wählen Sie dann OAuth-Client-ID aus.
12. Wählen Sie einen Application type (Anwendungstyp) aus und geben Sie Ihrem Client im Feld Name (Name) einen Namen.
13. Wählen Sie unter Autorisierte JavaScript Ursprünge die Option URI HINZUFÜGEN aus. Geben Sie Ihre Benutzerpool-Domäne ein.

    ```
    https://<your-user-pool-domain>
    ```

14. Wählen Sie unter Authorized redirect URIs (Autorisierte Umleitungs-URIs) die Option ADD URI (URI HINZUFÜGEN) aus. Geben Sie den Pfad zum Endpunkt `/oauth2/idpresponse` Ihrer Benutzerpool-Domäne ein.

    ```
    https://<your-user-pool-domain>/oauth2/idpresponse
    ```

15. Wählen Sie CREATE (Erstellen) aus.
16. Bewahren Sie die Werte sicher auf, die Google unter Ihre Client-ID und Ihr Client-Schlüssel anzeigt. Stellen Sie diese Werte Amazon Cognito zur Verfügung, wenn Sie einen Google-IDP hinzufügen.

Eine App bei Apple registrieren

Weitere Informationen zur Einrichtung von „Mit Apple anmelden“ finden Sie unter Konfigurieren Ihrer Umgebung für „Mit Apple anmelden“ in der Apple-Dokumentation für Entwickler.

1. Erstellen Sie ein Entwickler-Konto bei Apple.
2. Melden Sie sich mit Ihren Apple-Anmeldeinformationen an.
3. Wählen Sie in der linken Navigationsleiste Certificates, Identifiers & Profiles (Zertifikate, IDs und Profile) aus.
4. Wählen Sie in der linken Navigationsleiste Kennungen aus.
5. Wählen Sie auf der Seite Kennungen das Symbol + aus.
6. Wählen Sie auf der Seite Neue Kennung registrieren die Option App-IDs und dann Weiter aus.
7. Wählen Sie auf der Seite Typ auswählen die Option App und dann Weiter aus.
8. Machen Sie auf der Seite Registrieren einer App-ID das Folgende:
    1. Geben Sie unter Description (Beschreibung) eine Beschreibung ein.
    2. Geben Sie unter App ID Prefix (App-ID-Präfix) eine Bundle ID (Bündel-ID) ein. Notieren Sie sich den Wert unter App ID Prefix (App-ID-Präfix). Sie benötigen diesen Wert, nachdem Sie Apple als Identitätsanbieter in Schritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool ausgewählt haben.
    3. Wählen Sie unter Funktionen die Option Mit Apple anmelden und dann Bearbeiten aus.
    4. Wählen Sie auf der Seite Sign in with Apple: App ID Configuration (Mit Apple anmelden: App-ID-Konfiguration) aus, ob Sie die App entweder als primär oder mit anderen App-IDs gruppiert einrichten möchten. Klicken Sie dann auf Save (Speichern).
    5. Klicken Sie auf Continue.
9. Wählen Sie auf der Seite App-ID bestätigen die Option Registrieren aus.
10. Wählen Sie auf der Seite Kennungen das Symbol + aus.
11. Wählen Sie auf der Seite Neue Kennung registrieren die Option Services-IDs und dann Weiter aus.

12. Machen Sie auf der Seite Registrieren einer Service-ID das Folgende:

    1. Geben Sie unter Description (Beschreibung) eine Beschreibung ein.
    2. Geben Sie unter Identifier (ID) eine ID ein. Notieren Sie sich diese Dienste-ID, da Sie diesen Wert benötigen, nachdem Sie Apple als Identitätsanbieter in ausgewählt habenSchritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool.
    3. Wählen Sie Weiter und dann Registrieren aus.

13. Wähle auf der Seite „Identifikatoren“ die Services-ID aus, die du gerade erstellt hast.

    1. Wählen Sie Mit Apple anmelden und dann Konfigurieren aus.
    2. Wählen Sie auf der Seite Web Authentication Configuration (Konfiguration der Web-Authentifizierung) die App-ID, die Sie zuvor erstellt haben, als Primary App ID (Primäre App-ID) aus.
    3. Wählen Sie neben Website URLs (Website-URLs) das Symbol + aus.
    4. Geben Sie unter Domains and subdomains (Domänen und Subdomänen) Ihre Benutzerpool-Domäne ohne das Präfix `https://` ein.

       ```
       <your-user-pool-domain>
       ```

    5. Geben Sie unter Return URLs (URLs zurückgeben) den Pfad zum Endpunkt `/oauth2/idpresponse` Ihrer Benutzerpool-Domäne ein.

       ```
       https://<your-user-pool-domain>/oauth2/idpresponse
       ```

    6. Wählen Sie Weiter und dann Fertig. Sie müssen die Domäne nicht verifizieren.
    7. Wählen Sie Continue (Weiter) und anschließend Save (Speichern) aus.

14. Wählen Sie in der linken Navigationsleiste die Option Schlüssel aus.
15. Klicken Sie auf der Seite Schlüssel auf das Symbol +.
16. Machen Sie auf der Seite Registrieren eines neuen Schlüssels das Folgende:

    1. Geben Sie unter Key Name (Schlüsselname) einen Schlüsselnamen ein.
    2. Wählen Sie Mit Apple anmelden und dann Konfigurieren aus.
    3. Wählen Sie auf der Seite „Schlüssel konfigurieren“ die App-ID, die Sie zuvor erstellt haben, als primäre App-ID aus. Wählen Sie Speichern.
    4. Wählen Sie Weiter und dann Registrieren aus.

17. Wählen Sie auf der Seite „Ihren Schlüssel herunterladen“ die Option Herunterladen aus, um den privaten Schlüssel herunterzuladen, notieren Sie sich die angezeigte Schlüssel-ID und wählen Sie dann Fertig aus. Sie benötigen diesen privaten Schlüssel und den auf dieser Seite angezeigten Wert für die Schlüssel-ID, nachdem Sie Apple als Identitätsanbieter in Schritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool ausgewählt haben.

Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool

In diesem Abschnitt konfigurieren Sie einen sozialen IdP unter Verwendung der Client-ID und des Client-Geheimnisses aus dem vorherigen Abschnitt im Benutzerpool.

Um einen Anbieter für soziale Identitäten für einen Benutzerpool zu konfigurieren, verwenden Sie AWS Management Console

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Möglicherweise werden Sie zur Eingabe Ihrer AWS Anmeldeinformationen aufgefordert.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Federated sign-in (Verbundanmeldung) und wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus.
5. Wählen Sie einen Social-Identity-Anbieter aus: Facebook, Google, Login with Amazon oder Mit Apple anmelden.
6. Wählen Sie basierend auf Ihrer Wahl des Social-Identity-Anbieters aus den folgenden Schritten:

   - Google und Login with Amazon — Geben Sie die App-Client-ID und das App-Client-Geheimnis ein, die im vorherigen Abschnitt generiert wurden.
   - Facebook — Geben Sie die App-Client-ID und das App-Client-Geheimnis ein, die im vorherigen Abschnitt generiert wurden, und wählen Sie dann eine API-Version aus (z. B. Version 2.12). Wir empfehlen, die neueste mögliche Version zu wählen — jede Facebook-API hat einen Lebenszyklus und ein Verfallsdatum. Facebook-Bereiche und Attribute können zwischen API-Versionen variieren. Wir empfehlen, Ihre Social-Identity-Anmeldung mit Facebook zu testen, um sicherzustellen, dass der Verband wie vorgesehen funktioniert.
   - Mit Apple anmelden — Gib die Service-ID, die Team-ID, die Schlüssel-ID und den privaten Schlüssel ein, die im vorherigen Abschnitt generiert wurden.

7. Geben Sie die Namen der autorisierten Bereiche ein, die Sie verwenden möchten. Bereiche definieren, auf welche Benutzerattribute (wie z. B. `name` und `email`) mit Ihrer App zugreifen möchten. Für Facebook müssen diese durch Kommata voneinander getrennt werden. Für Google und "Login with Amazon (Anmelden mit Amazon)" müssen die Werte mit Leerzeichen getrennt werden. Aktivieren Sie für "Sign in with Apple (Mit Apple anmelden)" die Kontrollkästchen der Bereiche, auf die Sie Zugriff benötigen.

Anbieter sozialer Identitäten	Beispiel-Bereiche
Facebook	`public_profile, email`
Google	`profile email openid`
Login with Amazon	`profile postal_code`
Mit Apple anmelden	`email name`

   Der App-Benutzer wird aufgefordert, der Bereitstellung dieser Attribute für die App zuzustimmen. Weitere Informationen zu den jeweiligen Bereichen enthält die Dokumentation von Google, Facebook, Login with Amazon oder Mit Apple anmelden.

   Bei Verwendung von „Mit Apple anmelden“ werden Bereiche in den folgenden Benutzerszenarien unter Umständen nicht zurückgegeben:

   - Ein Endbenutzer stößt nach dem Verlassen der Anmeldeseite von Apple auf Fehler (diese können auf interne Fehler in Amazon Cognito oder auf vom Entwickler geschriebene Daten zurückzuführen sein).
   - Die Service-ID-ID wird in allen Benutzerpools und/oder anderen Authentifizierungsdiensten verwendet.
   - Ein Entwickler fügt zusätzliche Bereiche hinzu, nachdem sich der Benutzer angemeldet hat. Benutzer rufen neue Informationen nur ab, wenn sie sich authentifizieren und ihre Token aktualisieren.
   - Ein Entwickler löscht den Benutzer und der Benutzer meldet sich dann erneut an, ohne die App aus seinem Apple-ID-Profil zu entfernen.

8. Ordnen Sie Ihrem Benutzerpool Attribute von Ihrem Identitätsanbieter zu. Weitere Informationen finden Sie unter Wissenswertes über Mappings.

9. Wählen Sie Create (Erstellen) aus.
10. Wählen Sie auf der Registerkarte App client integration (App-Client-Integration) einen der App-Clients aus der Liste aus und klicken Sie anschließend auf Edit hosted UI settings (Einstellungen für gehostete UI bearbeiten). Fügen Sie unter Identity providers (Identitätsanbieter) den neuen Social-Identity-Anbieter zum App-Client hinzu.
11. Wählen Sie Save Changes.

Testen der Konfiguration Ihres Social-Identity-Anbieters

Unter Verwendung der Elemente aus den vorherigen zwei Abschnitten können Sie eine Anmelde-URL erstellen. Verwenden Sie sie zum Testen der Konfiguration Ihres sozialen IdP.

```
https://mydomain.us-east-1.amazoncognito.com/login?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Sie finden Ihre Domäne auf der Konsolenseite Domain name (Domänenname) für den Benutzerpool. Die Client-ID befindet sich auf der Registerkarte App client settings (App-Client-Einstellungen). Verwenden Sie Ihre Callback-URL für den redirect_uri-Parameter. Dies ist die URL der Seite, auf die Ihre Benutzer nach einer erfolgreichen Authentifizierung umgeleitet werden.

> **Note**
>
> Amazon Cognito bricht Authentifizierungsanfragen ab, die nicht innerhalb von 5 Minuten abgeschlossen werden, und leitet den Benutzer an die gehostete Benutzeroberfläche um. Für die Seite wird eine `Something went wrong`-Fehlermeldung angezeigt.

Hinzufügen der Anmeldung mit einem SAML-Identitätsanbieter zu einem Benutzerpool (optional)

Sie können Ihren App-Benutzern ermöglichen, sich über einen SAML-Identitätsanbieter (IdP) anzumelden. Unabhängig davon, ob Ihre Benutzer sich direkt oder über einen Drittanbieter anmelden, haben alle Benutzer ein Benutzerprofil im Benutzerpool. Überspringen Sie diesen Schritt, wenn keine Anmeldung über einen SAML-Identitätsanbieter hinzufügen möchten.

Weitere Informationen finden Sie unter Verwenden von SAML-Identitätsanbietern mit einem Benutzerpool.

Sie müssen Ihren SAML-Identitätsanbieter aktualisieren und Ihren Benutzerpool konfigurieren. Informationen darüber, wie Sie Ihren Benutzerpool als vertrauende Partei oder Anwendung für Ihren SAML 2.0-Identitätsanbieter hinzufügen können, finden Sie in der Dokumentation zu Ihrem SAML-Identitätsanbieter.

Sie müssen Ihrem SAML-Identitätsanbieter auch einen Assertion Consumer Service (ACS) -Endpunkt zur Verfügung stellen. Konfigurieren Sie den folgenden Endpunkt in Ihrer Benutzerpool-Domäne für SAML-2.0-POST-Binding Ihres SAML-Identitätsanbieters. Weitere Informationen zu Benutzerpool-Domänen finden Sie unter. Konfigurieren einer Benutzerpool-Domäne

```
https://Your user pool domain/saml2/idpresponse
With an Amazon Cognito domain:
https://<yourDomainPrefix>.auth.<region>.amazoncognito.com/saml2/idpresponse
With a custom domain:
https://Your custom domain/saml2/idpresponse
```

Sie finden Ihr Domain-Präfix und den Regionswert für Ihren Benutzerpool auf der Registerkarte Domainname der Amazon Cognito Cognito-Konsole.

Bei einigen SAML-Identitätsanbietern müssen Sie auch den Service Provider (SP)`urn`, auch Zielgruppen-URI oder SP-Entitäts-ID genannt, im folgenden Format angeben:

```
urn:amazon:cognito:sp:<yourUserPoolID>
```

Sie finden Ihre Benutzerpool-ID auf der Registerkarte Allgemeine Einstellungen in der Amazon-Cognito-Konsole.

Sie müssen Ihren SAML-Identitätsanbieter so konfigurieren, dass dieser Attributwerte für alle Attribute, die in Ihrem Benutzerpool erforderlich sind, bereitstellt. In der Regel ist `email` ein erforderliches Attribut für Benutzer-Pools. In diesem Fall sollte der SAML-Identitätsanbieter einen `email` Wert (Claim) in der SAML-Assertion bereitstellen.

Amazon-Cognito-Benutzerpools unterstützen die SAML-2.0-Föderierung mit Post-Binding-Endpunkten. Dadurch muss Ihre App keine SAML-Assertion-Antworten abrufen oder analysieren, da der Benutzerpool die SAML-Antwort direkt von Ihrem Identitätsanbieter über einen Benutzeragenten erhält.

So konfigurieren Sie einen SAML-2.0-Identitätsanbieter in Ihrem Benutzerpool:

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre Anmeldeinformationen ein. AWS
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Federated sign-in (Verbundanmeldung) und wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus.
5. Wählen Sie einen SAML-Social-Identity-Anbieter aus.
6. Geben Sie IDs durch Kommas getrennt ein. Eine Kennung teilt Amazon Cognito mit, dass es die E-Mail-Adresse überprüfen soll, die ein Benutzer bei der Anmeldung eingibt. Anschließend werden sie an den Anbieter weitergeleitet, der ihrer Domain entspricht.
7. Wählen Sie Add sign-out flow (Abmeldeablauf hinzufügen) aus, wenn Amazon Cognito signierte Abmeldeanfragen an Ihren Anbieter senden soll, wenn sich ein Benutzer abmeldet. Sie müssen Ihren SAML-2.0-Identitätsanbieter so konfigurieren, dass er Abmeldeantworten an den `https://<your Amazon Cognito domain>/saml2/logout`-Endpunkt sendet, der erstellt wird, wenn Sie die gehostete Benutzeroberfläche konfigurieren. Der `saml2/logout` Endpunkt verwendet die POST-Bindung.

   > **Note**
   >
   > Wenn diese Option ausgewählt ist und Ihr SAML-Identitätsanbieter eine signierte Abmeldeanforderung erwartet, müssen Sie auch das Signaturzertifikat konfigurieren, das von Amazon Cognito mit Ihrem SAML-IdP bereitgestellt wird.
   > Der SAML-IdP verarbeitet die signierte Abmeldeanforderung und meldet Ihren Benutzer von der Amazon Cognito-Sitzung ab.

8. Wählen Sie eine Metadaten-Dokumentquelle aus. Wenn Ihr Identitätsanbieter SAML-Metadaten unter einer öffentlichen URL anbietet, können Sie Metadata document URL (URL für Metadatendokumente) auswählen und die öffentliche URL eingeben. Wählen Sie andernfalls Upload metadata document (Hochladen eines Metadatendokuments) und anschließend eine Metadatendatei aus, die Sie zuvor von Ihrem Anbieter heruntergeladen haben.

> **Note**
>
> Wir empfehlen Ihnen, eine URL für ein Metadaten-Dokument einzugeben, wenn Ihr Anbieter über einen öffentlichen Endpunkt verfügt, anstatt eine Datei hochzuladen. Dadurch kann Amazon Cognito die Metadaten automatisch aktualisieren. Normalerweise werden die Metadaten alle sechs Stunden oder bevor sie ablaufen aktualisiert, je nachdem, was zuerst eintritt.

9. Wählen Sie Map attributes between your SAML provider and your app (Zuordnen von Attributen zwischen Ihrem SAML-Anbieter und Ihrer Anwendung) aus, um SAML-Anbieterattribute dem Benutzerprofil in Ihrem Benutzerpool zuzuordnen. Fügen Sie die erforderlichen Attribute Ihres Benutzerpools in Ihre Attributzuordnung ein.

   Wenn Sie beispielsweise das Benutzerpool-Attribut `email`auswählen, geben Sie den SAML-Attributnamen so ein, wie dieser in der SAML-Assertion Ihres Identitätsanbieters angezeigt wird. Ihr Identitätsanbieter bietet möglicherweise SAML-Zusicherungen als Referenz an. Einige Identitätsanbieter verwenden einfache Namen wie z. B. `email`, während andere URL-formatierte Attributnamen verwenden, wie folgt:

   ```
   http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
   ```

10. Wählen Sie Create (Erstellen) aus.

Erste Schritte mit Amazon Cognito Cognito-Identitätspools

Mit Amazon-Cognito-Identitäten-Pools können Sie eindeutige Identitäten erstellen und Benutzern Berechtigungen zuweisen. Ein Identitätspool kann folgende Elemente enthalten:

- Benutzer in einem Amazon-Cognito-Benutzerpool
- Benutzer, die die Authentifizierung mit externen Identitätsanbietern (z. B. Facebook, Google, Apple, OIDC- oder SAML-basierte Identitätsanbieter) durchführen.
- Benutzer, die sich mit Ihrem eigenen vorhandenen Authentifizierungsablauf authentifizieren

Mit einem Identitätspool können Sie temporäre AWS Anmeldeinformationen mit von Ihnen definierten Berechtigungen für den direkten Zugriff auf andere AWS-Services oder für den Zugriff auf Ressourcen über Amazon API Gateway abrufen.

Themen

Erstellen eines Identitätspools in Amazon Cognito

Sie können einen Identitätspool über die Amazon-Cognito-Konsole erstellen oder die AWS Command Line Interface (CLI) oder die Amazon-Cognito-APIs verwenden.

So erstellen Sie einen neuen Identitäten-Pool in der Konsole

1. Melden Sie sich bei der Amazon-Cognito-Konsole an und wählen Sie Identitätspools aus.
2. Wählen Sie Identitätspool erstellen.
3. Wählen Sie unter Identitätspool-Vertrauen konfigurieren aus, ob Sie Ihren Identitätspool für authentifizierten Zugriff, Gastzugriff oder beides einrichten möchten.

    - Wenn Sie Authentifizierter Zugriff ausgewählt haben, wählen Sie einen oder mehrere Identitätstypen aus, die Sie als Quelle für authentifizierte Identitäten in Ihrem Identitätspool festlegen möchten. Wenn Sie einen benutzerdefinierten Entwickleranbieter konfigurieren,

können Sie diesen nicht ändern oder löschen, nachdem Sie Ihren Identitätspool erstellt haben.

4. Wählen Sie unter Berechtigungen konfigurieren eine Standard-IAM-Rolle für authentifizierte Benutzer oder Gastbenutzer in Ihrem Identitätspool aus.

   a. Wählen Sie Neue IAM-Rolle erstellen, wenn Sie möchten, dass Amazon Cognito für Sie eine neue Rolle mit grundlegenden Berechtigungen und einer Vertrauensbeziehung zu Ihrem Identitätspool erstellt. Geben Sie einen IAM-Rollen-Namen ein, um Ihre neue Rolle zu identifizieren, zum Beispiel `myidentitypool_authenticatedrole`. Wählen Sie Richtliniendokument anzeigen aus, um die Berechtigungen zu überprüfen, die Amazon Cognito Ihrer neuen IAM-Rolle zuweist.

   b. Sie können sich dafür entscheiden, eine bestehende IAM-Rolle zu verwenden, wenn Sie bereits eine Rolle in Ihrer haben AWS-Konto , die Sie verwenden möchten. Sie müssen Ihre IAM-Rollen-Vertrauensrichtlinie so konfigurieren, dass sie `cognito-identity.amazonaws.com` beinhaltet. Konfigurieren Sie Ihre Rollen-Vertrauensrichtlinie so, dass Amazon Cognito die Rolle nur übernehmen kann, wenn nachgewiesen wird, dass die Anforderung von einem authentifizierten Benutzer in Ihrem spezifischen Identitätspool stammt. Weitere Informationen finden Sie unter Vertrauensstellungen und Berechtigungen für Rollen.

5. Geben Sie in Connect Identity Providers die Details der Identitätsanbieter (IdPs) ein, die Sie unter Identitätspool-Trust konfigurieren ausgewählt haben. Möglicherweise werden Sie aufgefordert, OAuth-App-Client-Informationen anzugeben, einen Amazon-Cognito-Benutzerpool auszuwählen, einen IAM-IdP auszuwählen oder eine benutzerdefinierte ID für einen Entwickleranbieter einzugeben.

   a. Wählen Sie die Rolleneinstellungen für jeden IdP aus. Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen. Mit einem Amazon-Cognito-Benutzerpool-IdP können Sie auch eine Rolle mit preferred_role in Token auswählen. Weitere Informationen zur `cognito:preferred_role`-Anforderung finden Sie unter Zuweisen von Prioritätswerten zu Gruppen.

      i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die

Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

b. Sie können Attribute für die Zugriffskontrolle für jeden IdP separat konfigurieren. Attribute für die Zugriffskontrolle ordnen Benutzeranforderungen den Prinzipal-Tags zu, die Amazon Cognito auf die temporäre Sitzung anwendet. Sie können IAM-Richtlinien erstellen, um den Benutzerzugriff anhand der Tags zu filtern, die Sie auf die jeweilige Sitzung anwenden.

i. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

ii. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

iii. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

6. Geben Sie unter Eigenschaften konfigurieren unter Identitätspool-Name einen Namen ein.
7. Wählen Sie unter Standardauthentifizierung (klassische Authentifizierung) aus, ob Sie den Standardablauf aktivieren möchten. Wenn der Basisablauf aktiv ist, können Sie die Rollenauswahl, die Sie für Sie getroffen haben, umgehen IdPs und AssumeRoleWithWebIdentitydirekt anrufen. Weitere Informationen finden Sie unter Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf.
8. Wählen Sie unter Tags die Option Tag hinzufügen aus, wenn Sie Tags auf Ihren Identitätspool anwenden möchten.
9. Bestätigen Sie unter Überprüfen und erstellen die Auswahl, die Sie für Ihren neuen Identitätspool getroffen haben. Wählen Sie Bearbeiten, um zum Assistenten zurückzukehren und Einstellungen zu ändern. Wählen Sie danach Identitätspool erstellen aus.

Einrichten eines SDK

Um Amazon Cognito Cognito-Identitätspools zu verwenden, richten Sie AWS Amplify AWS SDK for Java, den oder den AWS SDK for .NET ein. Weitere Informationen finden Sie unter den folgenden Themen.

- Das SDK für einrichten JavaScript im AWS SDK for Java Entwicklerhandbuch
- Amplify-Dokumentation im Amplify Dev Center
- Anbieter von Amazon-Cognito-Anmeldeinformationen im AWS SDK for .NET -Entwicklerhandbuch

Integrieren der Identitätsanbieter

Amazon-Cognito-Identitätspools (Verbundidentitäten) unterstützen die Benutzerauthentifizierung über Amazon-Cognito-Benutzerpools, verbundene Identitätsanbieter wie Amazon-, Facebook-, Google-, Apple- und SAML-Identitätsanbieter, sowie nicht authentifizierte Identitäten. Diese Funktion unterstützt auch Entwicklerauthentifizierte Identitäten (Identitätspools), mit denen Sie Benutzer über Ihren eigenen Backend-Authentifizierungsablauf registrieren und authentifizieren können.

Weitere Informationen zur Verwendung eines Amazon-Cognito-Benutzerpools zum Einrichten eines eigenen Benutzerverzeichnisses finden Sie unter Amazon-Cognito-Benutzerpools und Zugriff AWS-Services über einen Identitätspool nach der Anmeldung.

Weitere Informationen zur Verwendung von externen Identitätsanbietern erhalten Sie unter Externe Identitätsanbieter von Identitäten-Pools.

Weitere Informationen zur Integration eines eigenen Backend-Authentifizierungsablaufs finden Sie unter Entwicklerauthentifizierte Identitäten (Identitätspools).

Abrufen von Anmeldeinformationen

Amazon Cognito Cognito-Identitätspools bieten temporäre AWS Anmeldeinformationen für Benutzer, die Gäste sind (nicht authentifiziert), und für Benutzer, die sich authentifiziert haben und ein Token erhalten haben. Mit diesen AWS Anmeldeinformationen kann Ihre App AWS über Amazon API Gateway sicher auf ein Backend innerhalb AWS oder außerhalb zugreifen. Siehe Abrufen von Anmeldeinformationen.

Geführte Einrichtungsoptionen für Amazon Cognito

Vielleicht möchten Sie die Funktionen von Amazon Cognito in einer strukturierten, geführten Erfahrung testen. Im Folgenden finden Sie einige externe Ressourcen, die maßgeschneiderte Erfahrungen mit Benutzerpools und Identitätspools bieten.

Schließe einen Workshop ab

AWS Workshop Studio veranstaltet einen Workshop, der Sie durch die Einrichtung der meisten Funktionen von Amazon Cognito führt. Zu diesen Funktionen gehören die Benutzerpools-API, die gehostete Benutzeroberfläche für Benutzerpools, Identitätspools und die Sicherheitskonfiguration.

Fügen Sie Anwendungscode aus Beispielen hinzu

Das Kapitel mit den Codebeispielen in diesem Handbuch enthält Anwendungscode, den Sie mit Benutzerpools und Identitätspools verwenden können. Der Abschnitt Benutzerpools des Kapitels mit den Codebeispielen enthält kurze Auszüge, die einzelne Operationen behandeln, und längere Beispiele für end-to-end Beispielanwendungen in einer Vielzahl von Programmiersprachen.

Erstellen Sie eine Full-Stack-Anwendung mit AWS Amplify

AWS Amplifyrichtet sich AWS-Service an Entwickler, die eine Anwendung und eine Benutzeroberfläche entwickeln und hosten möchten. Amazon Cognito ist die Authentifizierungskomponente von Amplify. Wenn Sie Ihrer Anwendung eine Authentifizierung hinzufügen, kann Amplify die Bereitstellung von Amazon Cognito Cognito-Benutzerpool- und Identitätspool-Ressourcen automatisieren. Weitere Informationen finden Sie auch in Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps.

Weitere Amazon Cognito Cognito-Anwendungsressourcen auf GitHub

- Beispiele für Authentifizierungsabläufe mit.NET für Amazon Cognito
- Passwortlose Amazon Cognito Cognito-Authentifizierung
- PetStoreBeispiel mit Amazon Verified Permissions
- Beispiel für eine React-App, die ABAC+-Identitätspools für den Zugriff auf Ressourcen AWS verwendet
- Maschine-zu-Maschine-Autorisierung auf Basis von Amazon Cognito und API Gateway mithilfe von CDK AWS
- Aufbau einer detaillierten Autorisierung mit Amazon Cognito, API Gateway und IAM

- CloudFrontAutorisierung @edge

Weitere Workshops

- Implementieren Sie die passwortlose Authentifizierung mit Amazon Cognito und WebAuthn
- Mehrinstanzenfähige SaaS-Identität mit Amazon Cognito Cognito-Benutzerpools
- Amazon Cognito JWT im Detail

Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps

Wenn Sie Ihre App in einen Amazon-Cognito-App-Client integrieren, können Sie API-Operationen zur Authentifizierung und Autorisierung Ihrer Benutzer aufrufen. Wir empfehlen Ihnen AWS Amplify, Amazon Cognito in Ihre Web- und mobilen Apps zu integrieren. AWS Amplify ist eine Komplettlösung, mit der Frontend-Web- und Mobilentwickler auf einfache Weise Full-Stack-Anwendungen erstellen, verbinden und hosten können. Dabei haben Sie die Flexibilität AWS, die Bandbreite der Anwendungen zu nutzen, wenn sich Ihre Anwendungsfälle AWS-Services weiterentwickeln. Amplify Auth verwendet hauptsächlich Amazon Cognito, um Authentifizierungsfunktionen zu erstellen.

Themen

Eine typische Implementierung von Amazon Cognito verwendet eine Mischung aus visuellen Tools und APIs. Die Amazon-Cognito-Konsole ist die visuelle Oberfläche für die Einrichtung und Verwaltung Ihrer Amazon-Cognito-Benutzerpools und -Identitätspools. Die gehostete Benutzeroberfläche ist eine ready-to-use webbasierte Anmeldeanwendung zum schnellen Testen und Bereitstellen von Amazon Cognito Cognito-Benutzerpools. Darüber hinaus müssen Sie in den meisten Amazon-Cognito-Bereitstellungen Code in Ihre Apps einfügen, um mit Ihren Benutzerpools und Identitätspools interagieren zu können. Ihre App könnte beispielsweise die gehostete Benutzeroberfläche für die Benutzeranmeldung aufrufen und dann den Token-Endpunkt von Ihrem App-Code aus aufrufen, um den Autorisierungscode Ihres Benutzers gegen Token auszutauschen. Dann muss Ihre App die Token des Benutzers interpretieren und speichern und sie im entsprechenden Kontext zur Authentifizierung und Autorisierung präsentieren. Amplify fügt Tools für die angeleitete Integration mit integrierten Funktionen für diese Verfahren hinzu.

Sie können Ihre Amazon-Cognito-Ressourcen auch vollständig in Code erstellen. Um mit Ihrem eigenen benutzerdefinierten App-Code zu beginnen, sehen Sie sich die Amazon-Cognito-Codebeispiele für AWS SDKs an. Verwenden Sie die OpenID-Connect-Entwicklertools für die Integration mit Amazon Cognito als OpenID-Connect-Identitätsanbieter.

Bevor Sie die Authentifizierung und Autorisierung von Amazon Cognito verwenden, wählen Sie eine App-Plattform und bereiten Sie Ihren Code für die Integration in den Service vor. Verfügbare Plattformen finden Sie unter Authentifizierung mit AWS SDKs. Das AWS CLI ist ein Befehlszeilen-SDK für Amazon Cognito und andere und ein wertvoller AWS-Services Ausgangspunkt, um sich mit der Amazon Cognito Cognito-API vertraut zu machen.

> **Note**
>
> Sie können bestimmte Komponenten von Amazon Cognito nur mit der API konfigurieren. Beispielsweise können Sie einen benutzerdefinierten Lambda-Trigger für einen Benutzerpool für SMS- oder E-Mail-Absender nur mit einer Anfrage einrichten, die die `LambdaConfig` Eigenschaft der UserPoolKlasse in einer CreateUserPooloder UpdateUserPoolAPI-Anfrage aktualisiert.

Die Benutzerpools-API in Amazon Cognito teilt ihren Namespace mit verschiedenen Klassen von API-Operationen. Eine Klasse konfiguriert Benutzerpools und ihre Prozesse, Identitätsanbieter und Benutzer. Eine andere Klasse beinhaltet nicht authentifizierte Operationen, mit denen sich Ihre Benutzer in einem öffentlichen Client an- und abmelden und ihre Profile verwalten können. Die letzte Klasse von API-Vorgängen führt Benutzeroperationen, die Sie mit Ihren eigenen AWS Anmeldeinformationen autorisieren, in einem vertraulichen serverseitigen Client aus. Sie müssen Ihre angestrebte App-Architektur kennen, bevor Sie mit der Implementierung von App-Code beginnen. Weitere Informationen finden Sie unter Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte.

Authentifizierung mit AWS Amplify

AWS Amplify ist eine Komplettlösung für die Erstellung von Web- und Mobilanwendungen. In Amplify können Sie mit den Amplify-Bibliotheken eine Verbindung mit vorhandenen Ressourcen herstellen oder mit der Amplify-Befehlszeilenschnittstelle (Command Line Interface, CLI) neue Ressourcen erstellen und konfigurieren. Darüber hinaus bietet Amplify verbundene Benutzeroberflächenkomponenten wie Authenticator zur Einrichtung und Anpassung der Anmelde- und Registrierungserfahrung in Ihrer App.

Informationen zur Verwendung der Amplify-Authentifizierungsfunktionen in Ihrer Frontend-App finden Sie in den folgenden plattformspezifischen Dokumentationen.

- Amplify Sie die Authentifizierung für JavaScript

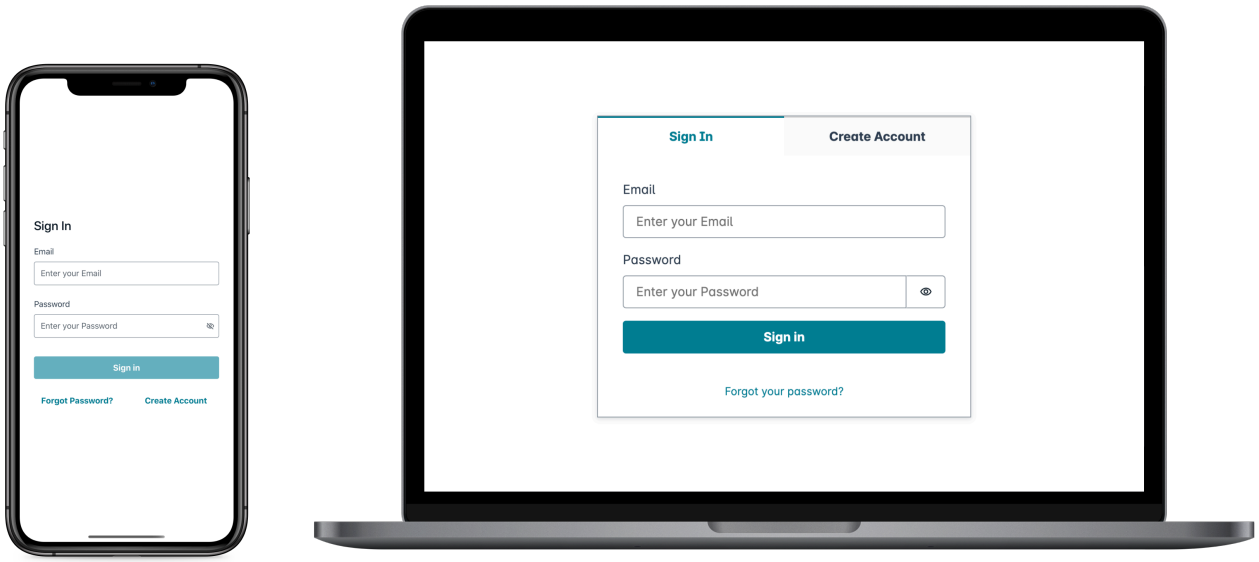
- Amplify-Authentifizierung für iOS
- Amplify-Authentifizierung für Android
- Amplify-Authentifizierung für Flutter

Die Amplify-Bibliotheken sind Open Source und unter GitHubverfügbar. Weitere Informationen dazu, wie Amplify Auth die Amazon-Cognito-Authentifizierung implementiert, finden Sie in den folgenden Bibliotheken.

- amplify-js
- amplify-swift
- amplify-flutter
- amplify-android

Erstellen einer Benutzeroberfläche (User Interface, UI) mit Amplify

Die gehostete UI von Amazon-Cognito-Benutzerpools kann die grundlegenden Anforderungen eines Authentifizierungs-Frontends für eine Web- oder mobile App erfüllen. Wenn Sie die Benutzeroberfläche (UI) über die Parameter hinaus anpassen möchten, die die gehostete UI bietet, erstellen Sie eine benutzerdefinierte App. Amplify UI ist eine anpassbare Sammlung von Frontend-Komponenten in verschiedenen Sprachen.



Informationen zu den ersten Schritten mit Ihrer benutzerdefinierten Authentifizierungskomponente finden Sie in den folgenden Dokumentationen für die Authenticator-Komponente.

- Authenticator für Android
- Authenticator für Angular
- Authenticator für Flutter
- Authenticator für React
- Authenticator für React Native
- Authenticator für Swift
- Authenticator für Vue

Authentifizierung mit AWS SDKs

Um ein sicheres Backend zu verwenden, um Ihren eigenen Identitäts-Microservice zu erstellen, der mit Amazon Cognito interagiert, stellen Sie mit einem AWS SDK in der Sprache Ihrer Wahl eine Verbindung zu den Amazon Cognito Cognito-Benutzerpools und der Amazon Cognito-Identitätspools-API her.

Weitere Informationen zu jedem API-Vorgang finden Sie in der API-Referenz der Amazon Cognito-Benutzerpools und in der API-Referenz zu Amazon Cognito. Diese Dokumente enthalten Siehe auch-Abschnitte mit Ressourcen zur Verwendung einer Vielzahl von SDKs in unterstützten Plattformen.

- AWS -Befehlszeilenschnittstelle
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java V2
- AWS SDK für JavaScript
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

Autorisierung mit Amazon Verified Permissions

Amazon Verified Permissions ist ein Autorisierungsservice für die von Ihnen erstellten Anwendungen. Wenn Sie einen Amazon-Cognito-Benutzerpool als Identitätsquelle hinzufügen, kann Ihre App Benutzerpoolzugriffs- oder Identitäts-Token (ID) an Verified Permissions weitergeben, um eine Entscheidung über Zulassen oder Ablehnen zu treffen. Verified Permissions berücksichtigt die Eigenschaften und den Anforderungskontext Ihres Benutzers auf der Grundlage von Richtlinien, die Sie in Cedar Policy Language verfasst haben. Der Anforderungskontext kann eine Kennung für das angeforderte Dokument, Bild oder eine andere Ressource sowie die Aktion enthalten, die Ihr Benutzer für die Ressource ausführen möchte.

Ihre App kann die Identität Ihres Benutzers oder Zugriffstoken für verifizierte Berechtigungen in IsAuthorizedWithTokenoder BatchIsAuthorizedWithTokenAPI-Anfragen bereitstellen. Diese API-Operationen akzeptieren Ihre Benutzer als Benutzer `Principal` und treffen Autorisierungsentscheidungen für die `Action` Person, auf `Resource` die sie zugreifen möchten. Zusätzliche benutzerdefinierte `Context` Einstellungen können zu einer detaillierten Zugriffsentscheidung beitragen.

Wenn Ihre App in einer `IsAuthorizedWithToken`-API-Anfrage ein Token präsentiert, führt Verified Permissions die folgenden Validierungen durch.

1. Ihr Benutzerpool ist eine konfigurierte Identitätsquelle für Verified Permissions für den angeforderten Richtlinienspeicher.
2. Der `client_id`- oder `aud`-Anspruch in Ihrem Zugriffs- bzw. Identitäts-Token entspricht einer Client-ID für eine Benutzerpool-App, die Sie für Verified Permissions angegeben haben. Um diesen Anspruch zu überprüfen, müssen Sie die Client-ID-Validierung in Ihrer Verified-Permissions-Identitätsquelle konfigurieren.
3. Ihr Token ist nicht abgelaufen.
4. Der Wert des `token_use` Anspruchs in Ihrem Token entspricht den Parametern, an die Sie übergeben habenIsAuthorizedWithToken. Der `token_use` Anspruch muss lauten, `access` wenn Sie ihn an den `accessToken` Parameter übergeben haben und `id` ob Sie ihn an den `identityToken` Parameter übergeben haben.
5. Die Signatur in Ihrem Token stammt aus den veröffentlichten JSON-Webschlüsseln (JWKs) Ihres Benutzerpools. Sie können Ihre JWKs unter `https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/.well-known/jwks.json` einsehen.

Widerrufene Token und gelöschte Benutzer

Verified Permissions validiert nur die Informationen, die es aus Ihrer Identitätsquelle und aus der Ablaufzeit des Tokens Ihres Benutzers kennt. Verified Permissions überprüft nicht, ob ein Token gesperrt wurde oder ob ein Benutzer existiert. Wenn Sie das Token Ihres Benutzers gesperrt oder sein Benutzerprofil aus Ihrem Benutzerpool gelöscht haben, betrachtet Verified Permissions das Token weiterhin als gültig, bis es abläuft.

Richtlinienevaluierung

Konfigurieren Sie Ihren Benutzerpool als Identitätsquelle für Ihren Richtlinienspeicher. Konfigurieren Sie Ihre App so, dass sie die Token Ihrer Benutzer in Anfragen an Verified Permissions übermittelt. Für jede Anfrage vergleicht Verified Permissions die Ansprüche im Token mit einer Richtlinie. Eine Richtlinie für Verified Permissions entspricht einer IAM-Richtlinie in AWS. Sie deklariert einen Prinzipal, eine Ressource und eine Aktion. Verified Permissions beantwortet Ihre Anfrage mit, `Allow` ob sie mit einer zulässigen Aktion übereinstimmt und nicht mit einer expliziten `Deny` Aktion; andernfalls wird mit geantwortet`Deny`. Weitere Informationen finden Sie in den Richtlinien von Amazon Verified Permissions im Benutzerhandbuch zu Amazon Verified Permissions.

Anpassen von Token

Um die Benutzeransprüche, die Sie Verified Permissions vorlegen möchten, zu ändern, hinzuzufügen oder zu entfernen, passen Sie den Inhalt Ihrer Zugriffs- und Identitätstoken mit einem anLambda-

Auslöser für die Vorab-Generierung von Token. Mit einem Auslöser für die Vorab-Generierung von Token können Sie Ansprüche in Ihren Token hinzufügen und ändern. Sie können beispielsweise eine Datenbank nach zusätzlichen Benutzerattributen abfragen und diese in Ihr ID-Token kodieren.

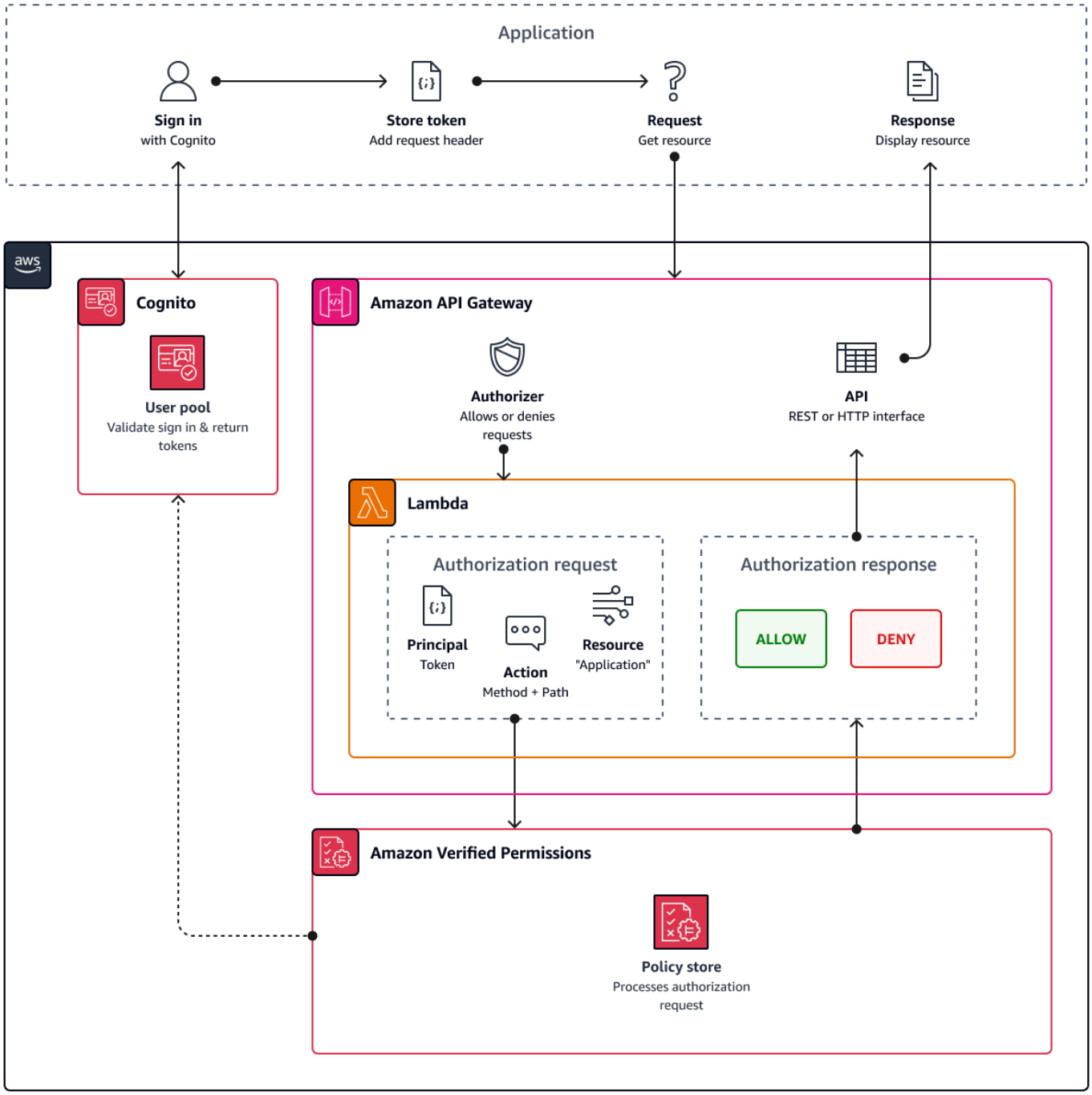
> **Note**
>
> Aufgrund der Art und Weise, wie Verified Permissions Ansprüche verarbeitet, sollten Sie Ihrer Funktion für die Vorab-Generierung von Token keine Ansprüche mit `cognito-`, `dev-` oder `custom`-Namen hinzufügen. Wenn Sie diese reservierten Anspruchspräfixe nicht im durch Doppelpunkte getrennten Format wie `cognito:username`, sondern als vollständige Anspruchsnamen angeben, schlagen Ihre Autorisierungsanfragen fehl.

Weitere Informationen darüber, wie Verified Permissions Ansprüche in Amazon-Cognito-Token zu Autorisierungsrichtlinien zuordnet, finden Sie unter Zuordnen von Amazon-Cognito-Token zum Verified-Permissions-Schema.

API-Autorisierung mit verifizierten Berechtigungen

Ihre ID oder Zugriffstoken können Anfragen an Back-End-REST-APIs von Amazon API Gateway mit verifizierten Berechtigungen autorisieren. Sie können einen Richtlinienspeicher mit direkten Links zu Ihrem Benutzerpool und Ihrer API erstellen. Mit der Startoption Mit Cognito und API Gateway einrichten fügt Verified Permissions dem Richtlinienspeicher eine Identitätsquelle für den Benutzerpool und der API einen Lambda-Authorizer hinzu. Wenn Ihre Anwendung ein Benutzerpool-Bearer-Token an die API weitergibt, ruft der Lambda-Authorizer Verified Permissions auf. Der Autorisierer übergibt das Token als Principal und den Anforderungspfad und die Methode als Aktion.

Das folgende Diagramm zeigt den Autorisierungsablauf für eine API-Gateway-API mit verifizierten Berechtigungen. Eine detaillierte Aufschlüsselung finden Sie unter API-verknüpfte Policy-Stores im Amazon Verified Permissions User Guide.



Verified Permissions strukturiert die API-Autorisierung anhand von Benutzerpoolgruppen. Da sowohl ID- als auch Zugriffstoken einen `cognito:groups` Anspruch enthalten, kann Ihr Richtlinienspeicher die rollenbasierte Zugriffskontrolle (RBAC) für Ihre APIs in einer Vielzahl von Anwendungskontexten verwalten.

Einstellungen für den Richtlinienspeicher auswählen

Wenn Sie eine Identitätsquelle in einem Richtlinienspeicher konfigurieren, müssen Sie auswählen, ob Sie Zugriffs- oder ID-Token verarbeiten möchten. Diese Entscheidung ist wichtig für die Funktionsweise Ihrer Policy-Engine. ID-Token enthalten Benutzerattribute. Zugriffstoken enthalten Informationen zur Benutzerzugriffskontrolle: OAuth-Bereiche. Obwohl beide Tokentypen Informationen zur Gruppenmitgliedschaft enthalten, empfehlen wir generell das Zugriffstoken für RBAC mit einem Richtlinienspeicher für verifizierte Berechtigungen. Das Zugriffstoken erweitert die Gruppenmitgliedschaft um Bereiche, die zur Autorisierungsentscheidung beitragen können. Die Ansprüche in einem Zugriffstoken werden in der Autorisierungsanfrage zum Kontext.

Sie müssen auch die Entitätstypen Benutzer und Gruppe konfigurieren, wenn Sie einen Benutzerpool als Identitätsquelle konfigurieren. Bei Entitätstypen handelt es sich um Prinzipal-, Aktions- und Ressourcen-IDs, auf die Sie in den Richtlinien für verifizierte Berechtigungen verweisen können. Entitäten in Richtlinienspeichern können eine Mitgliedschaftsbeziehung haben, bei der eine Entität Mitglied einer übergeordneten Entität sein kann. Mit der Mitgliedschaft können Sie auf Hauptgruppen, Aktionsgruppen und Ressourcengruppen verweisen. Bei Benutzerpoolgruppen muss der von Ihnen angegebene Benutzer-Entitätstyp Mitglied des Gruppen-Entitätstyps sein. Wenn Sie einen API-verknüpften Richtlinienspeicher einrichten oder der geführten Installation in der Konsole „Verifizierte Berechtigungen“ folgen, hat Ihr Richtlinienspeicher automatisch diese Beziehung zwischen übergeordnetem Mitglied.

Das ID-Token kann RBAC mit attributebasierter Zugriffskontrolle (ABAC) kombinieren. Nachdem Sie einen API-verknüpften Richtlinienspeicher erstellt haben, können Sie Ihre Richtlinien mit Benutzerattributen und Gruppenmitgliedschaften erweitern. Die Attributansprüche in einem ID-Token werden zu Hauptattributen in der Autorisierungsanfrage. Ihre Richtlinien können Autorisierungsentscheidungen auf der Grundlage von Hauptattributen treffen.

Sie können einen Richtlinienspeicher auch so konfigurieren, dass er Token akzeptiert, deren `client_id` Anspruch `aud` oder mit einer Liste akzeptabler App-Clients übereinstimmt, die Sie bereitstellen.

Beispielrichtlinie für die rollenbasierte API-Autorisierung

Die folgende Beispielrichtlinie wurde durch die Einrichtung eines Richtlinienspeichers für verifizierte Berechtigungen für eine PetStoreBeispiel-REST-API erstellt.

```
permit(
  principal in PetStore::UserGroup::"us-east-1_EXAMPLE|MyGroup",
```

```
  action in [ PetStore::Action::"get /pets", PetStore::Action::"get /pets/{petId}" ],
  resource
  );
```

Verified Permissions gibt in `Allow` folgenden Fällen eine Entscheidung über die Autorisierungsanfrage Ihrer Anwendung zurück:

1. Ihre Anwendung hat eine ID oder ein Zugriffstoken in einem `Authorization` Header als Trägertoken übergeben.
2. Ihre Anwendung hat ein Token mit einem `cognito:groups` Anspruch übergeben, der die Zeichenfolge `MyGroup` enthält.
3. In Ihrer Anwendung wurde beispielsweise eine `HTTP GET` Anfrage an `https://myapi.example.com/pets` oder gestellt`https://myapi.example.com/pets/scrappy`.

Beispielrichtlinie für einen Amazon-Cognito-Benutzer

Ihr Benutzerpool kann auch Autorisierungsanfragen für verifizierte Berechtigungen unter anderen Bedingungen als API-Anfragen generieren. Sie können alle Entscheidungen zur Zugriffskontrolle in Ihrer Anwendung an Ihren Richtlinienspeicher senden. Sie können beispielsweise die Sicherheit von Amazon DynamoDB oder Amazon S3 durch eine attributebasierte Zugriffskontrolle ergänzen, bevor Anfragen das Netzwerk übertragen, wodurch die Kontingentnutzung reduziert wird.

Im folgenden Beispiel wird die Cedar Policy Language verwendet, um Finance-Benutzern, die sich bei einem Benutzerpool-App-Client authentifizieren, das Lesen und Schreiben von `example_image.png` zu ermöglichen. John, ein Benutzer in Ihrer App, erhält ein ID-Token von Ihrem App-Client und leitet es in einer GET-Anfrage an eine URL weiter, für die eine Autorisierung erforderlich ist, `https://example.com/images/example_image.png`. Johns ID-Token hat einen `aud`-Anspruch auf die Client-ID Ihrer Benutzerpool-App `1234567890example`. Ihre Lambda-Funktion für die Vorab-Generierung von Token hat auch einen neuen Anspruch `costCenter` mit einem Wert für John von `Finance1234` eingefügt.

```
permit (
   principal,
   actions in [ExampleCorp::Action::"readFile", "writeFile"],
   resource == ExampleCorp::Photo::"example_image.png"
)
when {
   principal.aud == "1234567890example" &&
```

```
   principal.custom.costCenter like "Finance*"
};
```

Der folgende Anfragetext führt zu einer `Allow`-Antwort.

```
{
   "accesstoken": "[John's ID token]",
   "action": {
      "actionId": "readFile",
      "actionType": "Action"
   },
   "resource": {
      "entityId": "example_image.png",
      "entityType": "Photo"
   }
}
```

Wenn Sie in einer Richtlinie für Verified Permissions einen Prinzipal angeben möchten, verwenden Sie das folgende Format:

```
permit (
   principal == [Namespace]::[Entity]::"[user pool ID]"|"[user sub]",
   action,
   resource
);
```

Im Folgenden finden Sie ein Beispiel für einen Prinzipal für einen Benutzer in einem Benutzerpool mit einer ID `us-east-1_Example` mit Sub oder Benutzer-ID. `973db890-092c-49e4-a9d0-912a4c0a20c7`

```
principal == ExampleCorp::User::"us-east-1_Example|973db890-092c-49e4-
a9d0-912a4c0a20c7",
```

Wenn Sie eine Benutzergruppe in einer Richtlinie für verifizierte Berechtigungen angeben möchten, verwenden Sie das folgende Format:

```
permit (
   principal in [Namespace]::[Group Entity]::"[Group name]",
   action,
   resource
```

```
);
```

Das Folgende ist ein Beispiel

Attributbasierte Zugriffskontrolle

Die Autorisierung mit verifizierten Berechtigungen für Ihre Apps und die Funktion „Attribute für die Zugriffskontrolle“ der Amazon Cognito Cognito-Identitätspools für AWS Anmeldeinformationen sind beide Formen der attributebasierten Zugriffskontrolle (ABAC). Im Folgenden werden die Funktionen von Verified Permissions und Amazon Cognito ABAC miteinander verglichen. In ABAC untersucht ein System die Attribute einer Entität und trifft anhand von Bedingungen, die Sie definieren, eine Autorisierungsentscheidung.

Service	Prozess	Ergebnis
Amazon Verified Permissions	Gibt eine `Allow` `Deny` Oder-Entscheidung aus der Analyse eines Benutzerpools zurück (JWT).	Basierend auf der Bewertung der Cedar-Richtlinien ist der Zugriff auf Anwendungsressourcen erfolgreich oder schlägt fehl.
Amazon Cognito Cognito-Identitätspools (Attribute für die Zugriffskontrolle)	Weist Ihrem Benutzer Sitzungs-Tags auf der Grundlage seiner Attribute zu. In den IAM-Richtlinienbedingungen können Tags `Allow` oder der `Deny` Benutzerzugriff auf überprüft werden. AWS-Services	Eine mit Tags versehene Sitzung mit temporären AWS Anmeldeinformationen für eine IAM-Rolle.

Code-Beispiele für Amazon Cognito unter Verwendung von AWS-SDKs.

Die folgenden Code-Beispiele zeigen, wie man Amazon Cognito mit einem AWS-Software-Development-Kit (SDK) verwendet.

Eine vollständige Liste der AWS-SDK-Entwicklerhandbücher und Code-Beispiele finden Sie unter Verwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Code-Beispiele

Codebeispiele für Amazon Cognito Identity mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Amazon Cognito Identity mit einem AWS Software Development Kit (SDK) verwendet wird.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien und serviceübergreifenden Beispiele sehen.

Serviceübergreifende Beispiele sind Beispielanwendungen, die über mehrere AWS-Services hinweg arbeiten.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Aktionen für Amazon Cognito Identity mithilfe von AWS SDKs
  - Verwendung CreateIdentityPool mit einem AWS SDK oder CLI
  - Verwendung DeleteIdentityPool mit einem AWS SDK oder CLI
  - Verwendung DescribeIdentityPool mit einem AWS SDK oder CLI
  - Verwendung GetCredentialsForIdentity mit einem AWS SDK oder CLI
  - Verwendung GetIdentityPoolRoles mit einem AWS SDK oder CLI
  - Verwendung ListIdentityPools mit einem AWS SDK oder CLI
  - Verwendung SetIdentityPoolRoles mit einem AWS SDK oder CLI
  - Verwendung UpdateIdentityPool mit einem AWS SDK oder CLI
- Serviceübergreifende Beispiele für Amazon Cognito Identity mit SDKs AWS
  - Eine Amazon-Transcribe-App entwickeln
  - Erstellen Sie eine Amazon-Textract-Explorer-Anwendung

Aktionen für Amazon Cognito Identity mithilfe von AWS SDKs

Die folgenden Codebeispiele zeigen, wie einzelne Amazon Cognito Identity-Aktionen mit AWS SDKs durchgeführt werden. Diese Auszüge rufen die Amazon-Cognito–Identity-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der Amazon Cognito Identity-API-Referenz.

Beispiele

- Verwendung CreateIdentityPool mit einem AWS SDK oder CLI
- Verwendung DeleteIdentityPool mit einem AWS SDK oder CLI
- Verwendung DescribeIdentityPool mit einem AWS SDK oder CLI
- Verwendung GetCredentialsForIdentity mit einem AWS SDK oder CLI
- Verwendung GetIdentityPoolRoles mit einem AWS SDK oder CLI
- Verwendung ListIdentityPools mit einem AWS SDK oder CLI
- Verwendung SetIdentityPoolRoles mit einem AWS SDK oder CLI
- Verwendung UpdateIdentityPool mit einem AWS SDK oder CLI

Verwendung **`CreateIdentityPool`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateIdentityPool`.

CLI

AWS CLI

So erstellen Sie einen Identitätspool mit dem Cognito-Identitätspool-Anbieter

In diesem Beispiel wird ein Identitätspool mit dem Namen erstellt MyIdentityPool. Der Pool hat einen Cognito-Identitätspool-Anbieter. Nicht authentifizierte Identitäten sind nicht zulässig.

Befehl:

```
aws cognito-identity create-identity-pool --identity-pool-name
 MyIdentityPool --no-allow-unauthenticated-identities --cognito-
identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-
west-2_aaaaaaaaa",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

Ausgabe:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",
```

```
            "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
            "ServerSideTokenCheck": false
        }
    ]
}
```

- Einzelheiten zur API finden Sie CreateIdentityPoolin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.CreateIdentityPoolRequest;
import
 software.amazon.awssdk.services.cognitoidentity.model.CreateIdentityPoolResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateIdentityPool {
    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <identityPoolName>\s
```

```
                Where:
                    identityPoolName - The name to give your identity pool.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String identityPoolName = args[0];
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String identityPoolId = createIdPool(cognitoClient, identityPoolName);
        System.out.println("Unity pool ID " + identityPoolId);
        cognitoClient.close();
    }

    public static String createIdPool(CognitoIdentityClient cognitoClient, String
 identityPoolName) {
        try {
            CreateIdentityPoolRequest poolRequest =
 CreateIdentityPoolRequest.builder()
                    .allowUnauthenticatedIdentities(false)
                    .identityPoolName(identityPoolName)
                    .build();

            CreateIdentityPoolResponse response =
 cognitoClient.createIdentityPool(poolRequest);
            return response.identityPoolId();

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Einzelheiten zur API finden Sie CreateIdentityPoolin der AWS SDK for Java 2.x API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Erstellt einen neuen Identitätspool, der nicht authentifizierte Identitäten zulässt.

```
New-CGIIdentityPool -AllowUnauthenticatedIdentities $true -IdentityPoolName
 CommonTests13
```

Ausgabe:

```
LoggedAt                       : 8/12/2015 4:56:07 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3
IdentityPoolName               : CommonTests13
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
ContentLength                  : 136
HttpStatusCode                 : OK
```

- Einzelheiten zur API finden Sie unter CreateIdentityPoolCmdlet-Referenz.AWS Tools for PowerShell

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen eines neuen Identitätspools.

```
    /// Create a new identity pool and return its ID.
    ///
    /// - Parameters:
    ///     - name: The name to give the new identity pool.
    ///
    /// - Returns: A string containing the newly created pool's ID, or `nil`
    ///   if an error occurred.
    ///
    func createIdentityPool(name: String) async throws -> String? {
        let cognitoInputCall = CreateIdentityPoolInput(developerProviderName:
"com.exampleco.CognitoIdentityDemo",
                                                       identityPoolName: name)

        let result = try await cognitoIdentityClient.createIdentityPool(input:
cognitoInputCall)
        guard let poolId = result.identityPoolId else {
            return nil
        }

        return poolId
    }
```

- Weitere Informationen finden Sie im Entwicklerhandbuch zum AWS -SDK für Swift.
- Einzelheiten zur API finden Sie CreateIdentityPoolin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteIdentityPool`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteIdentityPool`.

CLI

AWS CLI

Löschen eines Identitätspools

Im folgenden `delete-identity-pool`-Beispiel wird der angegebene Identitätspool gelöscht.

Befehl:

```
aws cognito-identity delete-identity-pool \
    --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- Einzelheiten zur API finden Sie DeleteIdentityPool in der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.awscore.exception.AwsServiceException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.DeleteIdentityPoolRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteIdentityPool {

    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                    <identityPoolId>\s

                Where:
                    identityPoolId - The Id value of your identity pool.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String identityPoold = args[0];
        CognitoIdentityClient cognitoIdClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();

        deleteIdPool(cognitoIdClient, identityPoold);
        cognitoIdClient.close();
    }

    public static void deleteIdPool(CognitoIdentityClient cognitoIdClient, String
 identityPoold) {
        try {

            DeleteIdentityPoolRequest identityPoolRequest =
 DeleteIdentityPoolRequest.builder()
                    .identityPoolId(identityPoold)
                    .build();

            cognitoIdClient.deleteIdentityPool(identityPoolRequest);
            System.out.println("Done");

        } catch (AwsServiceException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie DeleteIdentityPoolin der AWS SDK for Java 2.x API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Löscht einen bestimmten Identitätspool.

```
Remove-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

- Einzelheiten zur API finden Sie unter DeleteIdentityPool AWS Tools for PowerShellCmdlet-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie den angegebenen Identitätspool.

```
    /// Delete the specified identity pool.
    ///
    /// - Parameters:
    ///   - id: The ID of the identity pool to delete.
    ///
```

```
    func deleteIdentityPool(id: String) async throws {
        let input = DeleteIdentityPoolInput(
            identityPoolId: id
        )

        _ = try await cognitoIdentityClient.deleteIdentityPool(input: input)
    }
```

- Weitere Informationen finden Sie im Entwicklerhandbuch zum AWS -SDK für Swift.
- Einzelheiten zur API finden Sie DeleteIdentityPoolin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DescribeIdentityPool`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DescribeIdentityPool`.

CLI

AWS CLI

Um einen Identitätspool zu beschreiben

Dieses Beispiel beschreibt einen Identitätspool.

Befehl:

```
aws cognito-identity describe-identity-pool --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111"
```

Ausgabe:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
```

```
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- Einzelheiten zur API finden Sie DescribeIdentityPoolin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Ruft Informationen über einen bestimmten Identitätspool anhand seiner ID ab.

```
Get-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

Ausgabe:

```
LoggedAt                       : 8/12/2015 4:29:40 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName               : CommonTests1
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
ContentLength                  : 142
HttpStatusCode                 : OK
```

- Einzelheiten zur API finden Sie unter DescribeIdentityPool AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetCredentialsForIdentity`** mit einem AWS SDK oder CLI

Das folgende Codebeispiel zeigt, wie es verwendet wird`GetCredentialsForIdentity`.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.GetCredentialsForIdentityRequest;
import
 software.amazon.awssdk.services.cognitoidentity.model.GetCredentialsForIdentityResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetIdentityCredentials {
    public static void main(String[] args) {

        final String usage = """

                Usage:
                    <identityId>\s

            Where:
```

```
                identityId - The Id of an existing identity in the format
REGION:GUID.
            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String identityId = args[0];
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .build();

        getCredsForIdentity(cognitoClient, identityId);
        cognitoClient.close();
    }

    public static void getCredsForIdentity(CognitoIdentityClient cognitoClient,
String identityId) {
        try {
            GetCredentialsForIdentityRequest getCredentialsForIdentityRequest =
GetCredentialsForIdentityRequest
                    .builder()
                    .identityId(identityId)
                    .build();

            GetCredentialsForIdentityResponse response = cognitoClient
                    .getCredentialsForIdentity(getCredentialsForIdentityRequest);
            System.out.println(
                    "Identity ID " + response.identityId() + ", Access key ID " +
response.credentials().accessKeyId());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie GetCredentialsForIdentityin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetIdentityPoolRoles`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`GetIdentityPoolRoles`.

CLI

AWS CLI

Um Identitätspool-Rollen abzurufen

In diesem Beispiel werden Identitätspool-Rollen abgerufen.

Befehl:

```
aws cognito-identity get-identity-pool-roles --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111"
```

Ausgabe:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "Roles": {
      "authenticated": "arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role",
      "unauthenticated": "arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolUnauth_Role"
  }
}
```

- Einzelheiten zur API finden Sie GetIdentityPoolRolesin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Ruft die Informationen zu Rollen für einen bestimmten Identitätspool ab.

```
Get-CGIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

Ausgabe:

```
LoggedAt         : 8/12/2015 4:33:51 PM
IdentityPoolId   : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
Roles            : {[unauthenticated, arn:aws:iam::123456789012:role/
CommonTests1Role]}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength    : 165
HttpStatusCode   : OK
```

- Einzelheiten zur API finden Sie unter GetIdentityPoolRoles AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListIdentityPools`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListIdentityPools`.

CLI

AWS CLI

Auflisten von Identitätspools

In diesem Beispiel werden Identitätspools aufgeführt. Es werden maximal 20 Identitäten aufgeführt.

Befehl:

```
aws cognito-identity list-identity-pools --max-results 20
```

Ausgabe:

```
{
```

```
  "IdentityPools": [
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "MyIdentityPool"
      },
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "AnotherIdentityPool"
      },
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "IdentityPoolRegionA"
      }
  ]
}
```

- Einzelheiten zur API finden Sie ListIdentityPoolsin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.ListIdentityPoolsRequest;
import
 software.amazon.awssdk.services.cognitoidentity.model.ListIdentityPoolsResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
```

```
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListIdentityPools {
    public static void main(String[] args) {
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listIdPools(cognitoClient);
        cognitoClient.close();
    }

    public static void listIdPools(CognitoIdentityClient cognitoClient) {
        try {
            ListIdentityPoolsRequest poolsRequest =
 ListIdentityPoolsRequest.builder()
                    .maxResults(15)
                    .build();

            ListIdentityPoolsResponse response =
 cognitoClient.listIdentityPools(poolsRequest);
            response.identityPools().forEach(pool -> {
                System.out.println("Pool ID: " + pool.identityPoolId());
                System.out.println("Pool name: " + pool.identityPoolName());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListIdentityPoolsin der AWS SDK for Java 2.x API-Referenz.

PowerShell

Tools für PowerShell

Beispiel 1: Ruft eine Liste vorhandener Identitätspools ab.

```
Get-CGIIdentityPoolList
```

Ausgabe:

```
IdentityPoolId
 IdentityPoolName
--------------
 ----------------
us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1                    CommonTests1
us-east-1:118d242d-204e-4b88-b803-EXAMPLEGUID2                    Tests2
us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3                    CommonTests13
```

- Einzelheiten zur API finden Sie unter ListIdentityPools AWS Tools for PowerShellCmdlet-Referenz.

Swift

SDK für Swift

> **Note**
>
> Diese ist die Vorabdokumentation für ein SDK in der Vorversion. Änderungen sind vorbehalten.

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Suchen Sie die ID eines Identitätspools anhand seines Namens.

```
    /// Return the ID of the identity pool with the specified name.
    ///
    /// - Parameters:
    ///   - name: The name of the identity pool whose ID should be returned.
    ///
    /// - Returns: A string containing the ID of the specified identity pool
    ///   or `nil` on error or if not found.
```

```
    ///
    func getIdentityPoolID(name: String) async throws -> String? {
        var token: String? = nil

        // Iterate over the identity pools until a match is found.

        repeat {
            /// `token` is a value returned by `ListIdentityPools()` if the
            /// returned list of identity pools is only a partial list. You
            /// use the `token` to tell Amazon Cognito that you want to
            /// continue where you left off previously. If you specify `nil`
            /// or you don't provide the token, Amazon Cognito will start at
            /// the beginning.

            let listPoolsInput = ListIdentityPoolsInput(maxResults: 25,
nextToken: token)

            /// Read pages of identity pools from Cognito until one is found
            /// whose name matches the one specified in the `name` parameter.
            /// Return the matching pool's ID. Each time we ask for the next
            /// page of identity pools, we pass in the token given by the
            /// previous page.

            let output = try await cognitoIdentityClient.listIdentityPools(input:
listPoolsInput)

            if let identityPools = output.identityPools {
                for pool in identityPools {
                    if pool.identityPoolName == name {
                        return pool.identityPoolId!
                    }
                }
            }

            token = output.nextToken
        } while token != nil

        return nil
    }
```

Rufen Sie ID eines vorhandenen Identitätspools ab oder erstellen Sie sie, wenn sie nicht bereits vorhanden ist.

```
    /// Return the ID of the identity pool with the specified name.
    ///
    /// - Parameters:
    ///   - name: The name of the identity pool whose ID should be returned
    ///
    /// - Returns: A string containing the ID of the specified identity pool.
    ///   Returns `nil` if there's an error or if the pool isn't found.
    ///
    public func getOrCreateIdentityPoolID(name: String) async throws -> String? {
        // See if the pool already exists. If it doesn't, create it.

        guard let poolId = try await self.getIdentityPoolID(name: name) else {
            return try await self.createIdentityPool(name: name)
        }

        return poolId
    }
```

- Weitere Informationen finden Sie im Entwicklerhandbuch zum AWS -SDK für Swift.
- Einzelheiten zur API finden Sie ListIdentityPoolsin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`SetIdentityPoolRoles`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`SetIdentityPoolRoles`.

CLI

AWS CLI

So legen Sie Identitätspool-Rollen fest

Im folgenden `set-identity-pool-roles` Beispiel wird eine Identitätspool-Rolle festgelegt.

```
aws cognito-identity set-identity-pool-roles \
    --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" \
```

```
    --roles authenticated="arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role"
```

- Einzelheiten zur API finden Sie SetIdentityPoolRolesin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell

Beispiel 1: Konfiguriert den spezifischen Identity Pool so, dass er eine nicht authentifizierte IAM-Rolle hat.

```
Set-CGIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1 -Role @{ "unauthenticated" = "arn:aws:iam::123456789012:role/
CommonTests1Role" }
```

- Einzelheiten zur API finden Sie unter SetIdentityPoolRolesCmdlet-Referenz.AWS Tools for PowerShell

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateIdentityPool`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateIdentityPool`.

CLI

AWS CLI

Um einen Identitätspool zu aktualisieren

In diesem Beispiel wird ein Identitätspool aktualisiert. Es setzt den Namen auf MyIdentityPool. Es fügt Cognito als Identitätsanbieter hinzu. Es verbietet nicht authentifizierte Identitäten.

Befehl:

```
aws cognito-identity update-identity-pool --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111" --identity-pool-name
 "MyIdentityPool" --no-allow-unauthenticated-identities --cognito-
```

```
identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

Ausgabe:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- Einzelheiten zur API finden Sie in der Befehlsreferenz UpdateIdentityPool.AWS CLI

PowerShell

Tools für PowerShell

Beispiel 1: Aktualisiert einige Eigenschaften des Identitätspools, in diesem Fall den Namen des Identitätspools.

```
Update-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1 -IdentityPoolName NewPoolName
```

Ausgabe:

```
LoggedAt                       : 8/12/2015 4:53:33 PM
AllowUnauthenticatedIdentities : False
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName               : NewPoolName
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
```

```
ContentLength                 : 135
HttpStatusCode                : OK
```

- Einzelheiten zur API finden Sie unter UpdateIdentityPool AWS Tools for PowerShellCmdlet-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Serviceübergreifende Beispiele für Amazon Cognito Identity mit SDKs AWS

Die folgenden Beispielanwendungen verwenden AWS SDKs, um Amazon Cognito Identity mit anderen zu kombinieren. AWS-Services Jedes Beispiel enthält einen Link zu GitHub, über den Sie Anweisungen zur Einrichtung und Ausführung der Anwendung finden.

Beispiele

- Eine Amazon-Transcribe-App entwickeln
- Erstellen Sie eine Amazon-Textract-Explorer-Anwendung

Eine Amazon-Transcribe-App entwickeln

Das folgende Codebeispiel zeigt, wie Amazon Transcribe verwendet wird, um Sprachaufnahmen im Browser zu transkribieren und anzuzeigen.

JavaScript

SDK für JavaScript (v3)

Erstellen Sie eine App, die Amazon Transcribe verwendet, um Sprachaufnahmen im Browser zu transkribieren und anzuzeigen. Die App verwendet zwei Amazon Simple Storage Service (Amazon S3)-Buckets, einen zum Hosten des Anwendungscodes und einen zum Speichern von Transkriptionen. Die App verwendet einen Amazon-Cognito-Benutzerpool zur Authentifizierung Ihrer Benutzer. Authentifizierte Benutzer verfügen über AWS Identity and Access Management (IAM-) Berechtigungen für den Zugriff auf die erforderlichen AWS Dienste.

Den vollständigen Quellcode und Anweisungen zur Einrichtung und Ausführung finden Sie im vollständigen Beispiel unter. GitHub

Dieses Beispiel ist auch verfügbar im AWS SDK for JavaScript Entwicklerhandbuch für v3.

In diesem Beispiel verwendete Dienste

- Amazon Cognito Identity
- Amazon S3
- Amazon Transcribe

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen Sie eine Amazon-Textract-Explorer-Anwendung

Die folgenden Code-Beispiele zeigen, wie man die Amazon-Textract-Ausgabe in einer interaktiven Anwendung untersuchen kann.

JavaScript

SDK für JavaScript (v3)

Zeigt, wie Sie mit AWS SDK for JavaScript dem eine React-Anwendung erstellen, die Amazon Textract verwendet, um Daten aus einem Dokumentbild zu extrahieren und auf einer interaktiven Webseite anzuzeigen. Dieses Beispiel wird in einem Webbrowser ausgeführt und erfordert eine authentifizierte Amazon-Cognito-Identität für Anmeldeinformationen. Es verwendet Amazon Simple Storage Service (Amazon S3) zur Speicherung und fragt für Benachrichtigungen eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange ab, die ein Amazon Simple Notification Service (Amazon SNS)-Thema abonniert hat.

Den vollständigen Quellcode und Anweisungen zur Einrichtung und Ausführung finden Sie im vollständigen Beispiel unter GitHub.

In diesem Beispiel verwendete Dienste

- Amazon Cognito Identity
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele für Amazon Cognito Identity Provider mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Amazon Cognito Identity Provider mit einem AWS Software Development Kit (SDK) verwendet wird.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien und serviceübergreifenden Beispiele sehen.

Szenarien sind Codebeispiele, die Ihnen zeigen, wie Sie eine bestimmte Aufgabe ausführen können, indem Sie mehrere Funktionen innerhalb desselben Services aufrufen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte

Hello Amazon Cognito

Die folgenden Codebeispiele veranschaulichen die ersten Schritte mit Amazon Cognito.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Code für die C MakeLists .txt-CMake-Datei.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)
```

```
# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS cognito-idp)

# Set this project's name.
project("hello_cognito")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
     # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

     # set(BIN_SUB_DIR "/Debug") # If you are building from the command line, you
 may need to uncomment this
                                     # and set the proper subdirectory to the
 executables' location.

     AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_cognito.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code für die Quelldatei hello_cognito.cpp.

```
#include <aws/core/Aws.h>
#include <aws/cognito-idp/CognitoIdentityProviderClient.h>
#include <aws/cognito-idp/model/ListUserPoolsRequest.h>
#include <iostream>

/*
 *  A "Hello Cognito" starter application which initializes an Amazon Cognito
 client and lists the Amazon Cognito
 *  user pools.
 *
 *  main function
 *
 *  Usage: 'hello_cognito'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
 cognitoClient(clientConfig);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::String> userPools;

        do {
            Aws::CognitoIdentityProvider::Model::ListUserPoolsRequest
 listUserPoolsRequest;
            if (!nextToken.empty()) {
                listUserPoolsRequest.SetNextToken(nextToken);
            }

            Aws::CognitoIdentityProvider::Model::ListUserPoolsOutcome
 listUserPoolsOutcome =
```

```
                    cognitoClient.ListUserPools(listUserPoolsRequest);

            if (listUserPoolsOutcome.IsSuccess()) {
                for (auto &userPool:
 listUserPoolsOutcome.GetResult().GetUserPools()) {

                    userPools.push_back(userPool.GetName());
                }

                nextToken = listUserPoolsOutcome.GetResult().GetNextToken();
            } else {
                std::cerr << "ListUserPools error: " <<
 listUserPoolsOutcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }


        } while (!nextToken.empty());
        std::cout << userPools.size() << " user pools found." << std::endl;
        for (auto &userPool: userPools) {
            std::cout << "   user pool: " << userPool << std::endl;
        }
    }

    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Einzelheiten zur API finden Sie unter ListUserPools AWS SDK for C++API-Referenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
package main

import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification
 Service
// (Amazon SNS) client and list the topics in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 cognitoClient := cognitoidentityprovider.NewFromConfig(sdkConfig)
 fmt.Println("Let's list the user pools for your account.")
 var pools []types.UserPoolDescriptionType
 paginator := cognitoidentityprovider.NewListUserPoolsPaginator(
  cognitoClient, &cognitoidentityprovider.ListUserPoolsInput{MaxResults:
 aws.Int32(10)})
 for paginator.HasMorePages() {
  output, err := paginator.NextPage(context.TODO())
  if err != nil {
   log.Printf("Couldn't get user pools. Here's why: %v\n", err)
  } else {
   pools = append(pools, output.UserPools...)
  }
 }
 if len(pools) == 0 {
  fmt.Println("You don't have any user pools!")
```

```
 } else {
  for _, pool := range pools {
   fmt.Printf("\t%v: %v\n", *pool.Name, *pool.Id)
  }
 }
}
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUserPools {
    public static void main(String[] args) {
```

```
        CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllUserPools(cognitoClient);
        cognitoClient.close();
    }

    public static void listAllUserPools(CognitoIdentityProviderClient
cognitoClient) {
        try {
            ListUserPoolsRequest request = ListUserPoolsRequest.builder()
                    .maxResults(10)
                    .build();

            ListUserPoolsResponse response =
cognitoClient.listUserPools(request);
            response.userPools().forEach(userpool -> {
                System.out.println("User pool " + userpool.name() + ", User ID "
+ userpool.id());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import {
  paginateListUserPools,
  CognitoIdentityProviderClient,
} from "@aws-sdk/client-cognito-identity-provider";

const client = new CognitoIdentityProviderClient({});

export const helloCognito = async () => {
  const paginator = paginateListUserPools({ client }, {});

  const userPoolNames = [];

  for await (const page of paginator) {
    const names = page.UserPools.map((pool) => pool.Name);
    userPoolNames.push(...names);
  }

  console.log("User pool names: ");
  console.log(userPoolNames.join("\n"));
  return userPoolNames;
};
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK for JavaScript API-Referenz.

Codebeispiele

- Aktionen für Amazon Cognito Identity Provider mithilfe von AWS SDKs
  - Verwendung AdminCreateUser mit einem AWS SDK oder CLI
  - Verwendung AdminGetUser mit einem AWS SDK oder CLI
  - Verwendung AdminInitiateAuth mit einem AWS SDK oder CLI
  - Verwendung AdminRespondToAuthChallenge mit einem AWS SDK oder CLI
  - Verwendung AdminSetUserPassword mit einem AWS SDK oder CLI
  - Verwendung AssociateSoftwareToken mit einem AWS SDK oder CLI
  - Verwendung ConfirmDevice mit einem AWS SDK oder CLI
  - Verwendung ConfirmForgotPassword mit einem AWS SDK oder CLI
  - Verwendung ConfirmSignUp mit einem AWS SDK oder CLI
  - Verwendung CreateUserPool mit einem AWS SDK oder CLI

- Verwendung CreateUserPoolClient mit einem AWS SDK oder CLI
- Verwendung DeleteUser mit einem AWS SDK oder CLI
- Verwendung ForgotPassword mit einem AWS SDK oder CLI
- Verwendung InitiateAuth mit einem AWS SDK oder CLI
- Verwendung ListUserPools mit einem AWS SDK oder CLI
- Verwendung ListUsers mit einem AWS SDK oder CLI
- Verwendung ResendConfirmationCode mit einem AWS SDK oder CLI
- Verwendung RespondToAuthChallenge mit einem AWS SDK oder CLI
- Verwendung SignUp mit einem AWS SDK oder CLI
- Verwendung UpdateUserPool mit einem AWS SDK oder CLI
- Verwendung VerifySoftwareToken mit einem AWS SDK oder CLI

- Szenarien für Amazon Cognito Identity Provider mit AWS SDKs
  - Bestätigen Sie bekannte Amazon Cognito Cognito-Benutzer automatisch mit einer Lambda-Funktion mithilfe eines SDK AWS
  - Automatisches Migrieren bekannter Amazon Cognito Cognito-Benutzer mit einer Lambda-Funktion mithilfe eines SDK AWS
  - Registrieren Sie einen Benutzer mit einem Amazon Cognito Cognito-Benutzerpool, für den MFA erforderlich ist, mithilfe eines SDK AWS
  - Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung mithilfe eines SDK AWS

Aktionen für Amazon Cognito Identity Provider mithilfe von AWS SDKs

Die folgenden Codebeispiele zeigen, wie einzelne Amazon Cognito Identity Provider-Aktionen mit AWS SDKs durchgeführt werden. Diese Auszüge rufen die Amazon-Cognito–Identity-Provider-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Die vollständige Liste finden Sie in der Amazon Cognito Identity Provider API-Referenz.

Beispiele

- Verwendung AdminCreateUser mit einem AWS SDK oder CLI

- Verwendung AdminGetUser mit einem AWS SDK oder CLI
- Verwendung AdminInitiateAuth mit einem AWS SDK oder CLI
- Verwendung AdminRespondToAuthChallenge mit einem AWS SDK oder CLI
- Verwendung AdminSetUserPassword mit einem AWS SDK oder CLI
- Verwendung AssociateSoftwareToken mit einem AWS SDK oder CLI
- Verwendung ConfirmDevice mit einem AWS SDK oder CLI
- Verwendung ConfirmForgotPassword mit einem AWS SDK oder CLI
- Verwendung ConfirmSignUp mit einem AWS SDK oder CLI
- Verwendung CreateUserPool mit einem AWS SDK oder CLI
- Verwendung CreateUserPoolClient mit einem AWS SDK oder CLI
- Verwendung DeleteUser mit einem AWS SDK oder CLI
- Verwendung ForgotPassword mit einem AWS SDK oder CLI
- Verwendung InitiateAuth mit einem AWS SDK oder CLI
- Verwendung ListUserPools mit einem AWS SDK oder CLI
- Verwendung ListUsers mit einem AWS SDK oder CLI
- Verwendung ResendConfirmationCode mit einem AWS SDK oder CLI
- Verwendung RespondToAuthChallenge mit einem AWS SDK oder CLI
- Verwendung SignUp mit einem AWS SDK oder CLI
- Verwendung UpdateUserPool mit einem AWS SDK oder CLI
- Verwendung VerifySoftwareToken mit einem AWS SDK oder CLI

Verwendung **`AdminCreateUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AdminCreateUser`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung

CLI

AWS CLI

Um einen Benutzer zu erstellen

Im folgenden `admin-create-user` Beispiel wird ein Benutzer mit den angegebenen Einstellungen E-Mail-Adresse und Telefonnummer erstellt.

```
aws cognito-idp admin-create-user \
    --user-pool-id us-west-2_aaaaaaaaa \
    --username diego \
    --user-attributes Name=email,Value=diego@example.com
 Name=phone_number,Value="+15555551212" \
    --message-action SUPPRESS
```

Ausgabe:

```
{
    "User": {
        "Username": "diego",
        "Attributes": [
            {
                "Name": "sub",
                "Value": "7325c1de-b05b-4f84-b321-9adc6e61f4a2"
            },
            {
                "Name": "phone_number",
                "Value": "+15555551212"
            },
            {
                "Name": "email",
                "Value": "diego@example.com"
            }
        ],
        "UserCreateDate": 1548099495.428,
        "UserLastModifiedDate": 1548099495.428,
        "Enabled": true,
        "UserStatus": "FORCE_CHANGE_PASSWORD"
    }
}
```

- Einzelheiten zur API finden Sie AdminCreateUserunter AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
 userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
 }
 return err
}
```

- Einzelheiten zur API finden Sie AdminCreateUserin der AWS SDK for Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AdminGetUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AdminGetUser`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   /// <summary>
   /// Get the specified user from an Amazon Cognito user pool with
administrator access.
   /// </summary>
   /// <param name="userName">The name of the user.</param>
   /// <param name="poolId">The Id of the Amazon Cognito user pool.</param>
   /// <returns>Async task.</returns>
   public async Task<UserStatusType> GetAdminUserAsync(string userName, string
poolId)
   {
       AdminGetUserRequest userRequest = new AdminGetUserRequest
       {
           Username = userName,
```

```
            UserPoolId = poolId,
        };

        var response = await _cognitoService.AdminGetUserAsync(userRequest);

        Console.WriteLine($"User status {response.UserStatus}");
        return response.UserStatus;
    }
```

- Einzelheiten zur API finden Sie AdminGetUserin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

    Aws::CognitoIdentityProvider::Model::AdminGetUserRequest request;
    request.SetUsername(userName);
    request.SetUserPoolId(userPoolID);

    Aws::CognitoIdentityProvider::Model::AdminGetUserOutcome outcome =
            client.AdminGetUser(request);

    if (outcome.IsSuccess()) {
        std::cout << "The status for " << userName << " is " <<

Aws::CognitoIdentityProvider::Model::UserStatusTypeMapper::GetNameForUserStatusType(
                          outcome.GetResult().GetUserStatus()) << std::endl;
```

```
        std::cout << "Enabled is " << outcome.GetResult().GetEnabled() <<
 std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminGetUser. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }
```

- Einzelheiten zur API finden Sie AdminGetUserin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Benutzer abrufen

In diesem Beispiel werden Informationen zum Benutzernamen jane@example.com abgerufen.

Befehl:

```
aws cognito-idp admin-get-user --user-pool-id us-west-2_aaaaaaaaa --username
 jane@example.com
```

Ausgabe:

```
{
  "Username": "4320de44-2322-4620-999b-5e2e1c8df013",
  "Enabled": true,
  "UserStatus": "FORCE_CHANGE_PASSWORD",
  "UserCreateDate": 1548108509.537,
  "UserAttributes": [
      {
          "Name": "sub",
          "Value": "4320de44-2322-4620-999b-5e2e1c8df013"
      },
      {
          "Name": "email_verified",
          "Value": "true"
      },
      {
```

```
            "Name": "phone_number_verified",
            "Value": "true"
        },
        {
            "Name": "phone_number",
            "Value": "+01115551212"
        },
        {
            "Name": "email",
            "Value": "jane@example.com"
        }
    ],
    "UserLastModifiedDate": 1548108509.537
}
```

- Einzelheiten zur API finden Sie AdminGetUserin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static void getAdminUser(CognitoIdentityProviderClient
identityProviderClient, String userName,
            String poolId) {
        try {
            AdminGetUserRequest userRequest = AdminGetUserRequest.builder()
                    .username(userName)
                    .userPoolId(poolId)
                    .build();

            AdminGetUserResponse response =
identityProviderClient.adminGetUser(userRequest);
            System.out.println("User status " + response.userStatusAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie AdminGetUserin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const adminGetUser = ({ userPoolId, username }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminGetUserCommand({
    UserPoolId: userPoolId,
    Username: username,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie AdminGetUserin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun getAdminUser(userNameVal: String?, poolIdVal: String?) {
    val userRequest = AdminGetUserRequest {
        username = userNameVal
        userPoolId = poolIdVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminGetUser(userRequest)
        println("User status ${response.userStatus}")
    }
}
```

- API-Details finden Sie AdminGetUserin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
```

```
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_up_user(self, user_name, password, user_email):
        """
        Signs up a new user with Amazon Cognito. This action prompts Amazon
Cognito
        to send an email to the specified email address. The email contains a
code that
        can be used to confirm the user.

        When the user already exists, the user status is checked to determine
whether
        the user has been confirmed.

        :param user_name: The user name that identifies the new user.
        :param password: The password for the new user.
        :param user_email: The email address for the new user.
        :return: True when the user is already confirmed with Amazon Cognito.
                 Otherwise, false.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "Password": password,
                "UserAttributes": [{"Name": "email", "Value": user_email}],
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.sign_up(**kwargs)
            confirmed = response["UserConfirmed"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "UsernameExistsException":
                response = self.cognito_idp_client.admin_get_user(
                    UserPoolId=self.user_pool_id, Username=user_name
                )
                logger.warning(
                    "User %s exists and is %s.", user_name,
response["UserStatus"]
                )
                confirmed = response["UserStatus"] == "CONFIRMED"
            else:
```

```
                logger.error(
                    "Couldn't sign up %s. Here's why: %s: %s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return confirmed
```

- Einzelheiten zur API finden Sie AdminGetUserin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AdminInitiateAuth`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AdminInitiateAuth`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Initiate an admin auth request.
```

```
    /// </summary>
    /// <param name="clientId">The client ID to use.</param>
    /// <param name="userPoolId">The ID of the user pool.</param>
    /// <param name="userName">The username to authenticate.</param>
    /// <param name="password">The user's password.</param>
    /// <returns>The session to use in challenge-response.</returns>
    public async Task<string> AdminInitiateAuthAsync(string clientId, string
userPoolId, string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var request = new AdminInitiateAuthRequest
        {
            ClientId = clientId,
            UserPoolId = userPoolId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.AdminInitiateAuthAsync(request);
        return response.Session;
    }
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";
```

```
   Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

   Aws::CognitoIdentityProvider::Model::AdminInitiateAuthRequest request;
   request.SetClientId(clientID);
   request.SetUserPoolId(userPoolID);
   request.AddAuthParameters("USERNAME", userName);
   request.AddAuthParameters("PASSWORD", password);
   request.SetAuthFlow(

Aws::CognitoIdentityProvider::Model::AuthFlowType::ADMIN_USER_PASSWORD_AUTH);


   Aws::CognitoIdentityProvider::Model::AdminInitiateAuthOutcome outcome =
           client.AdminInitiateAuth(request);

   if (outcome.IsSuccess()) {
       std::cout << "Call to AdminInitiateAuth was successful." << std::endl;
       sessionResult = outcome.GetResult().GetSession();
   }
   else {
       std::cerr << "Error with CognitoIdentityProvider::AdminInitiateAuth. "
                 << outcome.GetError().GetMessage()
                 << std::endl;
   }
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Authentifizierung initiieren

In diesem Beispiel wird die Authentifizierung mithilfe des ADMIN_NO_SRP_AUTH-Flows für den Benutzernamen jane@example.com initiiert

Auf dem Client muss die Anmelde-API für die serverbasierte Authentifizierung (ADMIN_NO_SRP_AUTH) aktiviert sein.

Verwenden Sie die Sitzungsinformationen im Rückgabewert, um admin-respond-to-auth -challenge aufzurufen.

Befehl:

```
aws cognito-idp admin-initiate-auth --user-pool-id us-west-2_aaaaaaaaa --client-
id 3n4b5urk1ft4fl3mg5e62d9ado --auth-flow ADMIN_NO_SRP_AUTH --auth-parameters
 USERNAME=jane@example.com,PASSWORD=password
```

Ausgabe:

```
{
  "ChallengeName": "NEW_PASSWORD_REQUIRED",
  "Session": "SESSION",
  "ChallengeParameters": {
      "USER_ID_FOR_SRP": "84514837-dcbc-4af1-abff-f3c109334894",
      "requiredAttributes": "[]",
      "userAttributes": "{\"email_verified\":\"true\",\"phone_number_verified\":
\"true\",\"phone_number\":\"+01xxx5550100\",\"email\":\"jane@example.com\"}"
  }
}
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static AdminInitiateAuthResponse
initiateAuth(CognitoIdentityProviderClient identityProviderClient,
           String clientId, String userName, String password, String userPoolId)
{
       try {
           Map<String, String> authParameters = new HashMap<>();
           authParameters.put("USERNAME", userName);
           authParameters.put("PASSWORD", password);
```

```
            AdminInitiateAuthRequest authRequest =
 AdminInitiateAuthRequest.builder()
                    .clientId(clientId)
                    .userPoolId(userPoolId)
                    .authParameters(authParameters)
                    .authFlow(AuthFlowType.ADMIN_USER_PASSWORD_AUTH)
                    .build();

            AdminInitiateAuthResponse response =
 identityProviderClient.adminInitiateAuth(authRequest);
            System.out.println("Result Challenge is : " +
 response.challengeName());
            return response;

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return null;
    }
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const adminInitiateAuth = ({ clientId, userPoolId, username, password }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminInitiateAuthCommand({
    ClientId: clientId,
```

```
    UserPoolId: userPoolId,
    AuthFlow: AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
    AuthParameters: { USERNAME: username, PASSWORD: password },
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun checkAuthMethod(clientIdVal: String, userNameVal: String,
 passwordVal: String, userPoolIdVal: String): AdminInitiateAuthResponse {
    val authParas = mutableMapOf<String, String>()
    authParas["USERNAME"] = userNameVal
    authParas["PASSWORD"] = passwordVal

    val authRequest = AdminInitiateAuthRequest {
        clientId = clientIdVal
        userPoolId = userPoolIdVal
        authParameters = authParas
        authFlow = AuthFlowType.AdminUserPasswordAuth
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminInitiateAuth(authRequest)
        println("Result Challenge is ${response.challengeName}")
        return response
    }
```

```
}
```

- API-Details finden Sie AdminInitiateAuthin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def start_sign_in(self, user_name, password):
        """
        Starts the sign-in process for a user by using administrator credentials.
        This method of signing in is appropriate for code running on a secure
 server.

        If the user pool is configured to require MFA and this is the first sign-
in
```

```
        for the user, Amazon Cognito returns a challenge response to set up an
        MFA application. When this occurs, this function gets an MFA secret from
        Amazon Cognito and returns it to the caller.

        :param user_name: The name of the user to sign in.
        :param password: The user's password.
        :return: The result of the sign-in attempt. When sign-in is successful,
this
                 returns an access token that can be used to get AWS credentials.
Otherwise,
                 Amazon Cognito returns a challenge to set up an MFA application,
                 or a challenge to enter an MFA code from a registered MFA
application.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
                "AuthParameters": {"USERNAME": user_name, "PASSWORD": password},
            }
            if self.client_secret is not None:
                kwargs["AuthParameters"]["SECRET_HASH"] =
self._secret_hash(user_name)
            response = self.cognito_idp_client.admin_initiate_auth(**kwargs)
            challenge_name = response.get("ChallengeName", None)
            if challenge_name == "MFA_SETUP":
                if (
                    "SOFTWARE_TOKEN_MFA"
                    in response["ChallengeParameters"]["MFAS_CAN_SETUP"]
                ):
                    response.update(self.get_mfa_secret(response["Session"]))
                else:
                    raise RuntimeError(
                        "The user pool requires MFA setup, but the user pool is
not "
                        "configured for TOTP MFA. This example requires TOTP
MFA."
                    )
        except ClientError as err:
            logger.error(
                "Couldn't start sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
```

```
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AdminRespondToAuthChallenge`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AdminRespondToAuthChallenge`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/// <summary>
/// Respond to an admin authentication challenge.
/// </summary>
/// <param name="userName">The name of the user.</param>
```

```
    /// <param name="clientId">The client ID.</param>
    /// <param name="mfaCode">The multi-factor authentication code.</param>
    /// <param name="session">The current application session.</param>
    /// <param name="clientId">The user pool ID.</param>
    /// <returns>The result of the authentication response.</returns>
    public async Task<AuthenticationResultType> AdminRespondToAuthChallengeAsync(
        string userName,
        string clientId,
        string mfaCode,
        string session,
        string userPoolId)
    {
        Console.WriteLine("SOFTWARE_TOKEN_MFA challenge is generated");

        var challengeResponses = new Dictionary<string, string>();
        challengeResponses.Add("USERNAME", userName);
        challengeResponses.Add("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        var respondToAuthChallengeRequest = new
AdminRespondToAuthChallengeRequest
        {
            ChallengeName = ChallengeNameType.SOFTWARE_TOKEN_MFA,
            ClientId = clientId,
            ChallengeResponses = challengeResponses,
            Session = session,
            UserPoolId = userPoolId,
        };

        var response = await
_cognitoService.AdminRespondToAuthChallengeAsync(respondToAuthChallengeRequest);
        Console.WriteLine($"Response to Authentication
{response.AuthenticationResult.TokenType}");
        return response.AuthenticationResult;
    }
```

- Einzelheiten zur API finden Sie AdminRespondToAuthChallengein der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region (overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeRequest
request;
       request.AddChallengeResponses("USERNAME", userName);
       request.AddChallengeResponses("SOFTWARE_TOKEN_MFA_CODE", mfaCode);
       request.SetChallengeName(

Aws::CognitoIdentityProvider::Model::ChallengeNameType::SOFTWARE_TOKEN_MFA);
       request.SetClientId(clientID);
       request.SetUserPoolId(userPoolID);
       request.SetSession(session);

       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeOutcome
outcome =
               client.AdminRespondToAuthChallenge(request);

       if (outcome.IsSuccess()) {
           std::cout << "Here is the response to the challenge.\n" <<

outcome.GetResult().GetAuthenticationResult().Jsonize().View().WriteReadable()
                     << std::endl;

           accessToken =
outcome.GetResult().GetAuthenticationResult().GetAccessToken();
       }
       else {
```

```
            std::cerr << "Error with
CognitoIdentityProvider::AdminRespondToAuthChallenge. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Einzelheiten zur API finden Sie AdminRespondToAuthChallengein der AWS SDK for C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    // Respond to an authentication challenge.
    public static void adminRespondToAuthChallenge(CognitoIdentityProviderClient
identityProviderClient,
            String userName, String clientId, String mfaCode, String session) {
        System.out.println("SOFTWARE_TOKEN_MFA challenge is generated");
        Map<String, String> challengeResponses = new HashMap<>();

        challengeResponses.put("USERNAME", userName);
        challengeResponses.put("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        AdminRespondToAuthChallengeRequest respondToAuthChallengeRequest =
AdminRespondToAuthChallengeRequest.builder()
                .challengeName(ChallengeNameType.SOFTWARE_TOKEN_MFA)
                .clientId(clientId)
                .challengeResponses(challengeResponses)
                .session(session)
                .build();

        AdminRespondToAuthChallengeResponse respondToAuthChallengeResult =
identityProviderClient
```

```
                .adminRespondToAuthChallenge(respondToAuthChallengeRequest);

 System.out.println("respondToAuthChallengeResult.getAuthenticationResult()"
                + respondToAuthChallengeResult.authenticationResult());
    }
```

- Einzelheiten zur API finden Sie AdminRespondToAuthChallengein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const adminRespondToAuthChallenge = ({
  userPoolId,
  clientId,
  username,
  totp,
  session,
}) => {
  const client = new CognitoIdentityProviderClient({});
  const command = new AdminRespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: totp,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie AdminRespondToAuthChallengein der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Respond to an authentication challenge.
suspend fun adminRespondToAuthChallenge(userName: String, clientIdVal: String?,
 mfaCode: String, sessionVal: String?) {
    println("SOFTWARE_TOKEN_MFA challenge is generated")
    val challengeResponsesOb = mutableMapOf<String, String>()
    challengeResponsesOb["USERNAME"] = userName
    challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

    val adminRespondToAuthChallengeRequest = AdminRespondToAuthChallengeRequest {
        challengeName = ChallengeNameType.SoftwareTokenMfa
        clientId = clientIdVal
        challengeResponses = challengeResponsesOb
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val respondToAuthChallengeResult =
 identityProviderClient.adminRespondToAuthChallenge(adminRespondToAuthChallengeRequest)
        println("respondToAuthChallengeResult.getAuthenticationResult()
 ${respondToAuthChallengeResult.authenticationResult}")
    }
}
```

- API-Details finden Sie AdminRespondToAuthChallengein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Reagieren Sie auf eine MFA-Herausforderung, indem Sie einen Code bereitstellen, der von einer zugehörigen MFA-Anwendung generiert wurde.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def respond_to_mfa_challenge(self, user_name, session, mfa_code):
        """
        Responds to a challenge for an MFA code. This completes the second step
 of
        a two-factor sign-in. When sign-in is successful, it returns an access
 token
```

```
        that can be used to get AWS credentials from Amazon Cognito.

        :param user_name: The name of the user who is signing in.
        :param session: Session information returned from a previous call to
initiate
                        authentication.
        :param mfa_code: A code generated by the associated MFA application.
        :return: The result of the authentication. When successful, this contains
an
                 access token for the user.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "ChallengeName": "SOFTWARE_TOKEN_MFA",
                "Session": session,
                "ChallengeResponses": {
                    "USERNAME": user_name,
                    "SOFTWARE_TOKEN_MFA_CODE": mfa_code,
                },
            }
            if self.client_secret is not None:
                kwargs["ChallengeResponses"]["SECRET_HASH"] = self._secret_hash(
                    user_name
                )
            response =
self.cognito_idp_client.admin_respond_to_auth_challenge(**kwargs)
            auth_result = response["AuthenticationResult"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "ExpiredCodeException":
                logger.warning(
                    "Your MFA code has expired or has been used already. You
might have "
                    "to wait a few seconds until your app shows you a new code."
                )
            else:
                logger.error(
                    "Couldn't respond to mfa challenge for %s. Here's why: %s:
%s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
```

```
                raise
        else:
            return auth_result
```

- Einzelheiten zur API finden Sie AdminRespondToAuthChallengein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AdminSetUserPassword`** mit einem AWS SDK oder CLI

Das folgende Codebeispiel zeigt, wie es verwendet wird`AdminSetUserPassword`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}
```

```
// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
  }
 }
 return err
}
```

- Einzelheiten zur API finden Sie AdminSetUserPasswordin der AWS SDK for Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`AssociateSoftwareToken`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`AssociateSoftwareToken`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Get an MFA token to authenticate the user with the authenticator.
    /// </summary>
    /// <param name="session">The session name.</param>
    /// <returns>The session name.</returns>
    public async Task<string> AssociateSoftwareTokenAsync(string session)
    {
        var softwareTokenRequest = new AssociateSoftwareTokenRequest
        {
            Session = session,
        };

        var tokenResponse = await
_cognitoService.AssociateSoftwareTokenAsync(softwareTokenRequest);
        var secretCode = tokenResponse.SecretCode;

        Console.WriteLine($"Use the following secret code to set up the
authenticator: {secretCode}");

        return tokenResponse.Session;
    }
```

- Einzelheiten zur API finden Sie AssociateSoftwareTokenin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
 client(clientConfig);

        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenRequest
 request;
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenOutcome
 outcome =
                client.AssociateSoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout
                    << "Enter this setup key into an authenticator app, for
 example Google Authenticator."
                    << std::endl;
            std::cout << "Setup key: " << outcome.GetResult().GetSecretCode()
                      << std::endl;
#ifdef USING_QR
            printAsterisksLine();
            std::cout << "\nOr scan the QR code in the file '" << QR_CODE_PATH <<
 "."
                      << std::endl;

            saveQRCode(std::string("otpauth://totp/") + userName + "?secret=" +
                       outcome.GetResult().GetSecretCode());
#endif // USING_QR
            session = outcome.GetResult().GetSession();
```

```
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::AssociateSoftwareToken. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Einzelheiten zur API finden Sie AssociateSoftwareTokenin der AWS SDK for C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static String getSecretForAppMFA(CognitoIdentityProviderClient
identityProviderClient, String session) {
        AssociateSoftwareTokenRequest softwareTokenRequest =
AssociateSoftwareTokenRequest.builder()
                .session(session)
                .build();

        AssociateSoftwareTokenResponse tokenResponse = identityProviderClient
                .associateSoftwareToken(softwareTokenRequest);
        String secretCode = tokenResponse.secretCode();
        System.out.println("Enter this token into Google Authenticator");
        System.out.println(secretCode);
        return tokenResponse.session();
   }
```

- Einzelheiten zur API finden Sie AssociateSoftwareTokenin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const associateSoftwareToken = (session) => {
  const client = new CognitoIdentityProviderClient({});
  const command = new AssociateSoftwareTokenCommand({
    Session: session,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie AssociateSoftwareTokenin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun getSecretForAppMFA(sessionVal: String?): String? {
    val softwareTokenRequest = AssociateSoftwareTokenRequest {
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
```

```
        val tokenResponse =
 identityProviderClient.associateSoftwareToken(softwareTokenRequest)
        val secretCode = tokenResponse.secretCode
        println("Enter this token into Google Authenticator")
        println(secretCode)
        return tokenResponse.session
    }
}
```

- API-Details finden Sie AssociateSoftwareTokenin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret
```

```
    def get_mfa_secret(self, session):
        """
        Gets a token that can be used to associate an MFA application with the
user.

        :param session: Session information returned from a previous call to
initiate
                        authentication.
        :return: An MFA token that can be used to set up an MFA application.
        """
        try:
            response =
self.cognito_idp_client.associate_software_token(Session=session)
        except ClientError as err:
            logger.error(
                "Couldn't get MFA secret. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response
```

- Einzelheiten zur API finden Sie AssociateSoftwareTokenin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ConfirmDevice`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ConfirmDevice`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Initiates and confirms tracking of the device.
    /// </summary>
    /// <param name="accessToken">The user's access token.</param>
    /// <param name="deviceKey">The key of the device from Amazon Cognito.</
param>
    /// <param name="deviceName">The device name.</param>
    /// <returns></returns>
    public async Task<bool> ConfirmDeviceAsync(string accessToken, string
 deviceKey, string deviceName)
    {
        var request = new ConfirmDeviceRequest
        {
            AccessToken = accessToken,
            DeviceKey = deviceKey,
            DeviceName = deviceName
        };

        var response = await _cognitoService.ConfirmDeviceAsync(request);
        return response.UserConfirmationNecessary;
    }
```

- Einzelheiten zur API finden Sie ConfirmDevicein der AWS SDK for .NET API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const confirmDevice = ({ deviceKey, accessToken, passwordVerifier, salt }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmDeviceCommand({
    DeviceKey: deviceKey,
    AccessToken: accessToken,
    DeviceSecretVerifierConfig: {
      PasswordVerifier: passwordVerifier,
      Salt: salt,
    },
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie ConfirmDevicein der AWS SDK for JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""
```

```
   def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
       """
       :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
       :param user_pool_id: The ID of an existing Amazon Cognito user pool.
       :param client_id: The ID of a client application registered with the user
pool.
       :param client_secret: The client secret, if the client has a secret.
       """
       self.cognito_idp_client = cognito_idp_client
       self.user_pool_id = user_pool_id
       self.client_id = client_id
       self.client_secret = client_secret


   def confirm_mfa_device(
       self,
       user_name,
       device_key,
       device_group_key,
       device_password,
       access_token,
       aws_srp,
   ):
       """
       Confirms an MFA device to be tracked by Amazon Cognito. When a device is
       tracked, its key and password can be used to sign in without requiring a
new
       MFA code from the MFA application.

       :param user_name: The user that is associated with the device.
       :param device_key: The key of the device, returned by Amazon Cognito.
       :param device_group_key: The group key of the device, returned by Amazon
Cognito.
       :param device_password: The password that is associated with the device.
       :param access_token: The user's access token.
       :param aws_srp: A class that helps with Secure Remote Password (SRP)
                       calculations. The scenario associated with this example
uses
                       the warrant package.
       :return: True when the user must confirm the device. Otherwise, False.
When
```

```
                False, the device is automatically confirmed and tracked.
        """
        srp_helper = aws_srp.AWSSRP(
            username=user_name,
            password=device_password,
            pool_id="_",
            client_id=self.client_id,
            client_secret=None,
            client=self.cognito_idp_client,
        )
        device_and_pw = f"{device_group_key}{device_key}:{device_password}"
        device_and_pw_hash = aws_srp.hash_sha256(device_and_pw.encode("utf-8"))
        salt = aws_srp.pad_hex(aws_srp.get_random(16))
        x_value = aws_srp.hex_to_long(aws_srp.hex_hash(salt +
device_and_pw_hash))
        verifier = aws_srp.pad_hex(pow(srp_helper.val_g, x_value,
srp_helper.big_n))
        device_secret_verifier_config = {
            "PasswordVerifier": base64.standard_b64encode(
                bytearray.fromhex(verifier)
            ).decode("utf-8"),
            "Salt":
base64.standard_b64encode(bytearray.fromhex(salt)).decode("utf-8"),
        }
        try:
            response = self.cognito_idp_client.confirm_device(
                AccessToken=access_token,
                DeviceKey=device_key,
                DeviceSecretVerifierConfig=device_secret_verifier_config,
            )
            user_confirm = response["UserConfirmationNecessary"]
        except ClientError as err:
            logger.error(
                "Couldn't confirm mfa device %s. Here's why: %s: %s",
                device_key,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return user_confirm
```

- Einzelheiten zur API finden Sie ConfirmDevicein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ConfirmForgotPassword`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ConfirmForgotPassword`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Automatisches Migrieren bekannter Benutzer mit einer Lambda-Funktion

CLI

AWS CLI

Um ein vergessenes Passwort zu bestätigen

Dieses Beispiel bestätigt ein vergessenes Passwort für den Benutzernamen diego@example.com.

Befehl:

```
aws cognito-idp confirm-forgot-password --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username=diego@example.com --password PASSWORD --confirmation-code CONF_CODE
```

- Einzelheiten zur API finden Sie ConfirmForgotPasswordin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
  Username:         aws.String(userName),
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}
```

- Einzelheiten zur API finden Sie ConfirmForgotPasswordin der AWS SDK for Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ConfirmSignUp`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ConfirmSignUp`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   /// <summary>
   /// Confirm that the user has signed up.
   /// </summary>
   /// <param name="clientId">The Id of this application.</param>
   /// <param name="code">The confirmation code sent to the user.</param>
   /// <param name="userName">The username.</param>
   /// <returns>True if successful.</returns>
   public async Task<bool> ConfirmSignupAsync(string clientId, string code,
string userName)
   {
       var signUpRequest = new ConfirmSignUpRequest
       {
           ClientId = clientId,
           ConfirmationCode = code,
```

```
            Username = userName,
        };

        var response = await _cognitoService.ConfirmSignUpAsync(signUpRequest);
        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            Console.WriteLine($"{userName} was confirmed");
            return true;
        }
        return false;
    }
```

- Einzelheiten zur API finden Sie ConfirmSignUpin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::ConfirmSignUpRequest request;
        request.SetClientId(clientID);
        request.SetConfirmationCode(confirmationCode);
        request.SetUsername(userName);

        Aws::CognitoIdentityProvider::Model::ConfirmSignUpOutcome outcome =
                client.ConfirmSignUp(request);

        if (outcome.IsSuccess()) {
```

```
            std::cout << "ConfirmSignup was Successful."
                      << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::ConfirmSignUp. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Einzelheiten zur API finden Sie ConfirmSignUpin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Registrierung bestätigen

In diesem Beispiel wird die Registrierung des Benutzernamens diego@example.com bestätigt.

Befehl:

```
aws cognito-idp confirm-sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username=diego@example.com --confirmation-code CONF_CODE
```

- Einzelheiten zur API finden Sie ConfirmSignUpin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static void confirmSignUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String code,
            String userName) {
```

```
        try {
            ConfirmSignUpRequest signUpRequest = ConfirmSignUpRequest.builder()
                    .clientId(clientId)
                    .confirmationCode(code)
                    .username(userName)
                    .build();

            identityProviderClient.confirmSignUp(signUpRequest);
            System.out.println(userName + " was confirmed");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie ConfirmSignUpin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const confirmSignUp = ({ clientId, username, code }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmSignUpCommand({
    ClientId: clientId,
    Username: username,
    ConfirmationCode: code,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie ConfirmSignUpin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun confirmSignUp(clientIdVal: String?, codeVal: String?, userNameVal:
 String?) {
    val signUpRequest = ConfirmSignUpRequest {
        clientId = clientIdVal
        confirmationCode = codeVal
        username = userNameVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.confirmSignUp(signUpRequest)
        println("$userNameVal  was confirmed")
    }
}
```

- API-Details finden Sie ConfirmSignUpin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def confirm_user_sign_up(self, user_name, confirmation_code):
        """
        Confirms a previously created user. A user must be confirmed before they
        can sign in to Amazon Cognito.

        :param user_name: The name of the user to confirm.
        :param confirmation_code: The confirmation code sent to the user's
 registered
                                  email address.
        :return: True when the confirmation succeeds.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "ConfirmationCode": confirmation_code,
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            self.cognito_idp_client.confirm_sign_up(**kwargs)
        except ClientError as err:
            logger.error(
                "Couldn't confirm sign up for %s. Here's why: %s: %s",
                user_name,
```

```
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return True
```

- Einzelheiten zur API finden Sie ConfirmSignUpin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateUserPool`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateUserPool`.

CLI

AWS CLI

So erstellen Sie einen minimal konfigurierten Benutzerpool

In diesem Beispiel wird ein Benutzerpool erstellt, der MyUserPool mit Standardwerten benannt wird. Es gibt keine erforderlichen Attribute und keine Anwendungs-Clients. MFA und erweiterte Sicherheit sind deaktiviert.

Befehl:

```
aws cognito-idp create-user-pool --pool-name MyUserPool
```

Ausgabe:

```
{
  "UserPool": {
      "SchemaAttributes": [
          {
              "Name": "sub",
              "StringAttributeConstraints": {
```

```
                "MinLength": "1",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": true,
            "AttributeDataType": "String",
            "Mutable": false
        },
        {
            "Name": "name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "given_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "family_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "middle_name",
            "StringAttributeConstraints": {
```

```
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "nickname",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "preferred_username",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "profile",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "picture",
            "StringAttributeConstraints": {
```

```
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "website",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "email",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "email_verified",
            "Mutable": true
        },
        {
            "Name": "gender",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
```

```
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "birthdate",
                "StringAttributeConstraints": {
                    "MinLength": "10",
                    "MaxLength": "10"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "zoneinfo",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "locale",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "phone_number",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
```

```
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "phone_number_verified",
            "Mutable": true
        },
        {
            "Name": "address",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "updated_at",
            "NumberAttributeConstraints": {
                "MinValue": "0"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "Number",
            "Mutable": true
        }
    ],
    "MfaConfiguration": "OFF",
    "Name": "MyUserPool",
    "LastModifiedDate": 1547833345.777,
    "AdminCreateUserConfig": {
        "UnusedAccountValidityDays": 7,
        "AllowAdminCreateUserOnly": false
    },
    "EmailConfiguration": {},
    "Policies": {
        "PasswordPolicy": {
            "RequireLowercase": true,
```

```
                "RequireSymbols": true,
                "RequireNumbers": true,
                "MinimumLength": 8,
                "RequireUppercase": true
            }
        },
        "CreationDate": 1547833345.777,
        "EstimatedNumberOfUsers": 0,
        "Id": "us-west-2_aaaaaaaaa",
        "LambdaConfig": {}
    }
}
```

So erstellen Sie einen Benutzerpool mit zwei erforderlichen Attributen

In diesem Beispiel wird ein Benutzerpool erstellt MyUserPool. Der Pool ist so konfiguriert, dass er E-Mail-Adressen als Benutzernamensattribut akzeptiert. Außerdem wird die E-Mail-Quelladresse mit Amazon Simple Email Service auf eine validierte Adresse gesetzt.

Befehl:

```
aws cognito-idp create-user-pool --pool-name MyUserPool --username-
attributes "email" --email-configuration=SourceArn="arn:aws:ses:us-
east-1:111111111111:identity/
jane@example.com",ReplyToEmailAddress="jane@example.com"
```

Ausgabe:

```
{
    "UserPool": {
        "SchemaAttributes": [
            {
                "Name": "sub",
                "StringAttributeConstraints": {
                    "MinLength": "1",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": true,
                "AttributeDataType": "String",
                "Mutable": false
            },
            {
```

```
                "Name": "name",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "given_name",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "family_name",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "middle_name",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
```

```
                "Name": "nickname",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "preferred_username",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "profile",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "picture",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
```

```
                "Name": "website",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "email",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "AttributeDataType": "Boolean",
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "Name": "email_verified",
                "Mutable": true
            },
            {
                "Name": "gender",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "birthdate",
                "StringAttributeConstraints": {
                    "MinLength": "10",
                    "MaxLength": "10"
```

```
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "zoneinfo",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "locale",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "phone_number",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "phone_number_verified",
```

```
                "Mutable": true
            },
            {
                "Name": "address",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "updated_at",
                "NumberAttributeConstraints": {
                    "MinValue": "0"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "Number",
                "Mutable": true
            }
        ],
        "MfaConfiguration": "OFF",
        "Name": "MyUserPool",
        "LastModifiedDate": 1547837788.189,
        "AdminCreateUserConfig": {
            "UnusedAccountValidityDays": 7,
            "AllowAdminCreateUserOnly": false
        },
        "EmailConfiguration": {
            "ReplyToEmailAddress": "jane@example.com",
            "SourceArn": "arn:aws:ses:us-east-1:111111111111:identity/
jane@example.com"
        },
        "Policies": {
            "PasswordPolicy": {
                "RequireLowercase": true,
                "RequireSymbols": true,
                "RequireNumbers": true,
                "MinimumLength": 8,
                "RequireUppercase": true
            }
```

```
      },
      "UsernameAttributes": [
          "email"
      ],
      "CreationDate": 1547837788.189,
      "EstimatedNumberOfUsers": 0,
      "Id": "us-west-2_aaaaaaaaa",
      "LambdaConfig": {}
  }
}
```

- Einzelheiten zur API finden Sie CreateUserPoolin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
```

```
public class CreateUserPool {
    public static void main(String[] args) {

        final String usage = """

                Usage:
                    <userPoolName>\s

                Where:
                    userPoolName - The name to give your user pool when it's
 created.
                """;

       if (args.length != 1) {
           System.out.println(usage);
           System.exit(1);
       }

       String userPoolName = args[0];
       CognitoIdentityProviderClient cognitoClient =
 CognitoIdentityProviderClient.builder()
               .region(Region.US_EAST_1)
               .build();

       String id = createPool(cognitoClient, userPoolName);
       System.out.println("User pool ID: " + id);
       cognitoClient.close();
   }

   public static String createPool(CognitoIdentityProviderClient cognitoClient,
 String userPoolName) {
       try {
           CreateUserPoolRequest request = CreateUserPoolRequest.builder()
                   .poolName(userPoolName)
                   .build();

           CreateUserPoolResponse response =
 cognitoClient.createUserPool(request);
           return response.userPool().id();

       } catch (CognitoIdentityProviderException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
```

```
        return "";
    }
}
```

- Einzelheiten zur API finden Sie CreateUserPoolin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateUserPoolClient`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`CreateUserPoolClient`.

CLI

AWS CLI

Um einen Benutzerpool-Client zu erstellen

In diesem Beispiel wird ein neuer Benutzerpool-Client mit zwei expliziten Autorisierungsabläufen erstellt: USER_PASSWORD_AUTH und ADMIN_NO_SRP_AUTH.

Befehl:

```
aws cognito-idp create-user-pool-client --user-pool-id us-west-2_aaaaaaaaa
  --client-name MyNewClient --no-generate-secret --explicit-auth-flows
 "USER_PASSWORD_AUTH" "ADMIN_NO_SRP_AUTH"
```

Ausgabe:

```
{
  "UserPoolClient": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "ClientName": "MyNewClient",
      "ClientId": "6p3bs000no6a4ue1idruvd05ad",
      "LastModifiedDate": 1548697449.497,
      "CreationDate": 1548697449.497,
      "RefreshTokenValidity": 30,
      "ExplicitAuthFlows": [
          "USER_PASSWORD_AUTH",
```

```
            "ADMIN_NO_SRP_AUTH"
        ],
        "AllowedOAuthFlowsUserPoolClient": false
    }
}
```

- Einzelheiten CreateUserPoolClientzur AWS CLI API finden Sie in der Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolClientReques
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolClientRespon

/**
 * A user pool client app is an application that authenticates with Amazon
 * Cognito user pools.
 * When you create a user pool, you can configure app clients that allow mobile
 * or web applications
 * to call API operations to authenticate users, manage user attributes and
 * profiles,
 * and implement sign-up and sign-in flows.
 *
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateUserPoolClient {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <clientName> <userPoolId>\s

                Where:
                    clientName - The name for the user pool client to create.
                    userPoolId - The ID for the user pool.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String clientName = args[0];
        String userPoolId = args[1];
        CognitoIdentityProviderClient cognitoClient =
 CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        createPoolClient(cognitoClient, clientName, userPoolId);
        cognitoClient.close();
    }

    public static void createPoolClient(CognitoIdentityProviderClient
 cognitoClient, String clientName,
            String userPoolId) {
        try {
            CreateUserPoolClientRequest request =
 CreateUserPoolClientRequest.builder()
                    .clientName(clientName)
                    .userPoolId(userPoolId)
                    .build();

            CreateUserPoolClientResponse response =
 cognitoClient.createUserPoolClient(request);
```

```
            System.out.println("User pool " +
 response.userPoolClient().clientName() + " created. ID: "
                    + response.userPoolClient().clientId());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie CreateUserPoolClientin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteUser`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteUser`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. In den folgenden Codebeispielen können Sie diese Aktion im Kontext sehen:

- Bestätigen Sie bekannte Benutzer automatisch mit einer Lambda-Funktion
- Automatisches Migrieren bekannter Benutzer mit einer Lambda-Funktion
- Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::DeleteUserRequest request;
        request.SetAccessToken(accessToken);

        Aws::CognitoIdentityProvider::Model::DeleteUserOutcome outcome =
                client.DeleteUser(request);

        if (outcome.IsSuccess()) {
            std::cout << "The user " << userName << " was deleted."
                      << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::DeleteUser. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
        }
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Benutzer löschen

In diesem Beispiel wird ein Benutzer gelöscht.

Befehl:

```
aws cognito-idp delete-user --access-token ACCESS_TOKEN
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK for Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ForgotPassword`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ForgotPassword`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Automatisches Migrieren bekannter Benutzer mit einer Lambda-Funktion

CLI

AWS CLI

Um eine Passwortänderung zu erzwingen

Im folgenden `forgot-password` Beispiel wird eine Nachricht an jane@example.com gesendet, um ihr Passwort zu ändern.

```
aws cognito-idp forgot-password --client-id 38fjsnc484p94kpqsnet7mpld0 --username
 jane@example.com
```

Ausgabe:

```
{
    "CodeDeliveryDetails": {
        "Destination": "j***@e***.com",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
    }
}
```

- Einzelheiten zur API finden Sie ForgotPasswordin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
}
```

- Einzelheiten zur API finden Sie ForgotPasswordin der AWS SDK for Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`InitiateAuth`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`InitiateAuth`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Bestätigen Sie bekannte Benutzer automatisch mit einer Lambda-Funktion
- Automatisches Migrieren bekannter Benutzer mit einer Lambda-Funktion
- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

- Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Initiate authorization.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The name of the user who is authenticating.</
param>
    /// <param name="password">The password for the user who is authenticating.</
param>
    /// <returns>The response from the initiate auth request.</returns>
    public async Task<InitiateAuthResponse> InitiateAuthAsync(string clientId,
 string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var authRequest = new InitiateAuthRequest

        {
            ClientId = clientId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.InitiateAuthAsync(authRequest);
        Console.WriteLine($"Result Challenge is : {response.ChallengeName}");

        return response;
```

```
    }
```

- Einzelheiten zur API finden Sie InitiateAuthin der AWS SDK for .NET API-Referenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
  if errors.As(err, &resetRequired) {
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
  }
 } else {
  authResult = output.AuthenticationResult
```

```
 }
 return authResult, err
}
```

- Einzelheiten zur API finden Sie InitiateAuthin der AWS SDK for Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const initiateAuth = ({ username, password, clientId }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new InitiateAuthCommand({
    AuthFlow: AuthFlowType.USER_PASSWORD_AUTH,
    AuthParameters: {
      USERNAME: username,
      PASSWORD: password,
    },
    ClientId: clientId,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie InitiateAuthin der AWS SDK for JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

In diesem Beispiel wird veranschaulicht, wie die Authentifizierung mit einem nachverfolgten Gerät gestartet wird. Um die Anmeldung abzuschließen, muss der Client korrekt auf SRP-Abfragen (Secure Remote Password) reagieren.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_in_with_tracked_device(
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
```

```
        """
        Signs in to Amazon Cognito as a user who has a tracked device. Signing in
        with a tracked device lets a user sign in without entering a new MFA
 code.

        Signing in with a tracked device requires that the client respond to the
 SRP
        protocol. The scenario associated with this example uses the warrant
 package
        to help with SRP calculations.

        For more information on SRP, see https://en.wikipedia.org/wiki/
Secure_Remote_Password_protocol.

        :param user_name: The user that is associated with the device.
        :param password: The user's password.
        :param device_key: The key of a tracked device.
        :param device_group_key: The group key of a tracked device.
        :param device_password: The password that is associated with the device.
        :param aws_srp: A class that helps with SRP calculations. The scenario
                        associated with this example uses the warrant package.
        :return: The result of the authentication. When successful, this contains
 an
                 access token for the user.
        """
        try:
            srp_helper = aws_srp.AWSSRP(
                username=user_name,
                password=device_password,
                pool_id="_",
                client_id=self.client_id,
                client_secret=None,
                client=self.cognito_idp_client,
            )

            response_init = self.cognito_idp_client.initiate_auth(
                ClientId=self.client_id,
                AuthFlow="USER_PASSWORD_AUTH",
                AuthParameters={
                    "USERNAME": user_name,
                    "PASSWORD": password,
                    "DEVICE_KEY": device_key,
                },
            )
```

```
            if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
                raise RuntimeError(
                    f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']}."
                )

            auth_params = srp_helper.get_auth_params()
            auth_params["DEVICE_KEY"] = device_key
            response_auth = self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_SRP_AUTH",
                ChallengeResponses=auth_params,
            )
            if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
                raise RuntimeError(
                    f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
                    f"{response_init['ChallengeName']}."
                )

            challenge_params = response_auth["ChallengeParameters"]
            challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
            cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
            cr["USERNAME"] = user_name
            cr["DEVICE_KEY"] = device_key
            response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_PASSWORD_VERIFIER",
                ChallengeResponses=cr,
            )
            auth_tokens = response_verifier["AuthenticationResult"]
        except ClientError as err:
            logger.error(
                "Couldn't start client sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return auth_tokens
```

- Einzelheiten zur API finden Sie InitiateAuthin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListUserPools`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListUserPools`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// List the Amazon Cognito user pools for an account.
    /// </summary>
    /// <returns>A list of UserPoolDescriptionType objects.</returns>
    public async Task<List<UserPoolDescriptionType>> ListUserPoolsAsync()
    {
        var userPools = new List<UserPoolDescriptionType>();

        var userPoolsPaginator = _cognitoService.Paginators.ListUserPools(new
ListUserPoolsRequest());

        await foreach (var response in userPoolsPaginator.Responses)
        {
            userPools.AddRange(response.UserPools);
        }

        return userPools;
    }
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

Benutzerpools auflisten

In diesem Beispiel werden bis zu 20 Benutzerpools aufgelistet.

Befehl:

```
aws cognito-idp list-user-pools --max-results 20
```

Ausgabe:

```
{
  "UserPools": [
      {
         "CreationDate": 1547763720.822,
         "LastModifiedDate": 1547763720.822,
         "LambdaConfig": {},
         "Id": "us-west-2_aaaaaaaaa",
         "Name": "MyUserPool"
      }
  ]
}
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
package main

import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification
 Service
// (Amazon SNS) client and list the topics in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 cognitoClient := cognitoidentityprovider.NewFromConfig(sdkConfig)
 fmt.Println("Let's list the user pools for your account.")
 var pools []types.UserPoolDescriptionType
 paginator := cognitoidentityprovider.NewListUserPoolsPaginator(
  cognitoClient, &cognitoidentityprovider.ListUserPoolsInput{MaxResults:
 aws.Int32(10)})
 for paginator.HasMorePages() {
  output, err := paginator.NextPage(context.TODO())
  if err != nil {
   log.Printf("Couldn't get user pools. Here's why: %v\n", err)
  } else {
   pools = append(pools, output.UserPools...)
  }
 }
 if len(pools) == 0 {
  fmt.Println("You don't have any user pools!")
```

```
 } else {
  for _, pool := range pools {
   fmt.Printf("\t%v: %v\n", *pool.Name, *pool.Id)
  }
 }
}
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUserPools {
    public static void main(String[] args) {
```

```
        CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllUserPools(cognitoClient);
        cognitoClient.close();
    }

    public static void listAllUserPools(CognitoIdentityProviderClient
cognitoClient) {
        try {
            ListUserPoolsRequest request = ListUserPoolsRequest.builder()
                    .maxResults(10)
                    .build();

            ListUserPoolsResponse response =
cognitoClient.listUserPools(request);
            response.userPools().forEach(userpool -> {
                System.out.println("User pool " + userpool.name() + ", User ID "
+ userpool.id());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK for Java 2.x API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn show_pools(client: &Client) -> Result<(), Error> {
    let response = client.list_user_pools().max_results(10).send().await?;
    let pools = response.user_pools();
    println!("User pools:");
    for pool in pools {
        println!("  ID:              {}", pool.id().unwrap_or_default());
        println!("  Name:            {}", pool.name().unwrap_or_default());
        println!("  Lambda Config:   {:?}", pool.lambda_config().unwrap());
        println!(
            "  Last modified:   {}",
            pool.last_modified_date().unwrap().to_chrono_utc()?
        );
        println!(
            "  Creation date:   {:?}",
            pool.creation_date().unwrap().to_chrono_utc()
        );
        println!();
    }
    println!("Next token: {}", response.next_token().unwrap_or_default());

    Ok(())
}
```

- Einzelheiten zur API finden Sie ListUserPoolsin der API-Referenz zum AWS SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListUsers`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListUsers`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/// <summary>
/// Get a list of users for the Amazon Cognito user pool.
/// </summary>
/// <param name="userPoolId">The user pool ID.</param>
/// <returns>A list of users.</returns>
public async Task<List<UserType>> ListUsersAsync(string userPoolId)
{
    var request = new ListUsersRequest
    {
        UserPoolId = userPoolId
    };

    var users = new List<UserType>();

    var usersPaginator = _cognitoService.Paginators.ListUsers(request);
    await foreach (var response in usersPaginator.Responses)
    {
        users.AddRange(response.Users);
    }

    return users;
}
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

Benutzer auflisten

In diesem Beispiel werden bis zu 20 Benutzer aufgelistet.

Befehl:

```
aws cognito-idp list-users --user-pool-id us-west-2_aaaaaaaaa --limit 20
```

Ausgabe:

```
{
  "Users": [
      {
          "Username": "22704aa3-fc10-479a-97eb-2af5806bd327",
          "Enabled": true,
          "UserStatus": "FORCE_CHANGE_PASSWORD",
          "UserCreateDate": 1548089817.683,
          "UserLastModifiedDate": 1548089817.683,
          "Attributes": [
              {
                  "Name": "sub",
                  "Value": "22704aa3-fc10-479a-97eb-2af5806bd327"
              },
              {
                  "Name": "email_verified",
                  "Value": "true"
              },
              {
                  "Name": "email",
                  "Value": "mary@example.com"
              }
          ]
      }
  ]
}
```

- Einzelheiten zur API finden Sie ListUsersin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUsersRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUsersResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {

        final String usage = """

                Usage:
                    <userPoolId>\s

                Where:
                    userPoolId - The ID given to your user pool when it's
 created.
                """;
```

```
        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userPoolId = args[0];
        CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllUsers(cognitoClient, userPoolId);
        listUsersFilter(cognitoClient, userPoolId);
        cognitoClient.close();
    }

    public static void listAllUsers(CognitoIdentityProviderClient cognitoClient,
String userPoolId) {
        try {
            ListUsersRequest usersRequest = ListUsersRequest.builder()
                    .userPoolId(userPoolId)
                    .build();

            ListUsersResponse response = cognitoClient.listUsers(usersRequest);
            response.users().forEach(user -> {
                System.out.println("User " + user.username() + " Status " +
user.userStatus() + " Created "
                        + user.userCreateDate());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Shows how to list users by using a filter.
    public static void listUsersFilter(CognitoIdentityProviderClient
cognitoClient, String userPoolId) {

        try {
            String filter = "email = \"tblue@noserver.com\"";
            ListUsersRequest usersRequest = ListUsersRequest.builder()
                    .userPoolId(userPoolId)
```

```
                    .filter(filter)
                    .build();

            ListUsersResponse response = cognitoClient.listUsers(usersRequest);
            response.users().forEach(user -> {
                System.out.println("User with filter applied " + user.username()
 + " Status " + user.userStatus()
                        + " Created " + user.userCreateDate());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const listUsers = ({ userPoolId }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ListUsersCommand({
    UserPoolId: userPoolId,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie ListUsersin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listAllUsers(userPoolId: String) {

    val request = ListUsersRequest {
        this.userPoolId = userPoolId
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use { cognitoClient ->
        val response = cognitoClient.listUsers(request)
        response.users?.forEach { user ->
            println("The user name is ${user.username}")
        }
    }
}
```

- API-Details finden Sie ListUsersin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def list_users(self):
        """
        Returns a list of the users in the current user pool.

        :return: The list of users.
        """
        try:
            response =
 self.cognito_idp_client.list_users(UserPoolId=self.user_pool_id)
            users = response["Users"]
        except ClientError as err:
            logger.error(
                "Couldn't list users for %s. Here's why: %s: %s",
                self.user_pool_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return users
```

- Einzelheiten zur API finden Sie ListUsersin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ResendConfirmationCode`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ResendConfirmationCode`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Send a new confirmation code to a user.
    /// </summary>
    /// <param name="clientId">The Id of the client application.</param>
    /// <param name="userName">The username of user who will receive the code.</
param>
    /// <returns>The delivery details.</returns>
    public async Task<CodeDeliveryDetailsType> ResendConfirmationCodeAsync(string
 clientId, string userName)
    {
        var codeRequest = new ResendConfirmationCodeRequest
        {
            ClientId = clientId,
            Username = userName,
        };

        var response = await
 _cognitoService.ResendConfirmationCodeAsync(codeRequest);
```

```
        Console.WriteLine($"Method of delivery is
{response.CodeDeliveryDetails.DeliveryMedium}");

        return response.CodeDeliveryDetails;
    }
```

- Einzelheiten zur API finden Sie ResendConfirmationCodein der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeRequest
request;
        request.SetUsername(userName);
        request.SetClientId(clientID);

        Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeOutcome
outcome =
                client.ResendConfirmationCode(request);

        if (outcome.IsSuccess()) {
            std::cout
```

```
                    << "CognitoIdentityProvider::ResendConfirmationCode was
 successful."
                    << std::endl;
        }
        else {
            std::cerr << "Error with
 CognitoIdentityProvider::ResendConfirmationCode. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Einzelheiten zur API finden Sie ResendConfirmationCodein der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Bestätigungscode erneut senden

Im folgenden `resend-confirmation-code`-Beispiel wird ein Bestätigungscode an den Benutzer `jane` gesendet.

```
aws cognito-idp resend-confirmation-code \
    --client-id 12a3b456c7de890f11g123hijk \
    --username jane
```

Ausgabe:

```
{
    "CodeDeliveryDetails": {
        "Destination": "j***@e***.com",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
    }
}
```

Weitere Informationen finden Sie unter Registrieren und Bestätigen von Benutzerkonten im Amazon-Cognito-Entwicklerhandbuch.

- Einzelheiten zur API finden Sie ResendConfirmationCodein der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static void resendConfirmationCode(CognitoIdentityProviderClient
identityProviderClient, String clientId,
           String userName) {
       try {
           ResendConfirmationCodeRequest codeRequest =
ResendConfirmationCodeRequest.builder()
                   .clientId(clientId)
                   .username(userName)
                   .build();

           ResendConfirmationCodeResponse response =
identityProviderClient.resendConfirmationCode(codeRequest);
           System.out.println("Method of delivery is " +
response.codeDeliveryDetails().deliveryMediumAsString());

       } catch (CognitoIdentityProviderException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie ResendConfirmationCodein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const resendConfirmationCode = ({ clientId, username }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ResendConfirmationCodeCommand({
    ClientId: clientId,
    Username: username,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie ResendConfirmationCodein der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun resendConfirmationCode(clientIdVal: String?, userNameVal: String?) {
    val codeRequest = ResendConfirmationCodeRequest {
        clientId = clientIdVal
        username = userNameVal
    }
```

```
    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.resendConfirmationCode(codeRequest)
        println("Method of delivery is " +
 (response.codeDeliveryDetails?.deliveryMedium))
    }
}
```

- API-Details finden Sie ResendConfirmationCodein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret
```

```
    def resend_confirmation(self, user_name):
        """
        Prompts Amazon Cognito to resend an email with a new confirmation code.

        :param user_name: The name of the user who will receive the email.
        :return: Delivery information about where the email is sent.
        """
        try:
            kwargs = {"ClientId": self.client_id, "Username": user_name}
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.resend_confirmation_code(**kwargs)
            delivery = response["CodeDeliveryDetails"]
        except ClientError as err:
            logger.error(
                "Couldn't resend confirmation to %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return delivery
```

- Einzelheiten zur API finden Sie ResendConfirmationCodein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`RespondToAuthChallenge`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`RespondToAuthChallenge`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

CLI

AWS CLI

Reaktion auf eine Amazon Cognito SRP-Authentifizierungs-Challenge

Dieses Beispiel veranschaulicht die Reaktion auf eine Authentifizierungs-Challenge, die mit „initiate-auth“ initiiert wurde. Es ist eine Antwort auf die Challenge „NEW_PASSWORD_REQUIRED“. Es wird ein Passwort für den Benutzer jane@example.com festgelegt.

Befehl:

```
aws cognito-idp respond-to-auth-challenge --client-id 3n4b5urk1ft4fl3mg5e62d9ado
 --challenge-name NEW_PASSWORD_REQUIRED --challenge-responses
 USERNAME=jane@example.com,NEW_PASSWORD="password" --session "SESSION_TOKEN"
```

Ausgabe:

```
{
  "ChallengeParameters": {},
  "AuthenticationResult": {
      "AccessToken": "ACCESS_TOKEN",
      "ExpiresIn": 3600,
      "TokenType": "Bearer",
      "RefreshToken": "REFRESH_TOKEN",
      "IdToken": "ID_TOKEN",
      "NewDeviceMetadata": {
          "DeviceKey": "us-west-2_fec070d2-fa88-424a-8ec8-b26d7198eb23",
          "DeviceGroupKey": "-wt2ha1Zd"
      }
  }
}
```

- Einzelheiten zur API finden Sie RespondToAuthChallengein der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const respondToAuthChallenge = ({
  clientId,
  username,
  session,
  userPoolId,
  code,
}) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new RespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: code,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie RespondToAuthChallengein der AWS SDK for JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Melden Sie sich mit einem nachverfolgten Gerät an. Um die Anmeldung abzuschließen, muss der Client korrekt auf SRP-Abfragen (Secure Remote Password) reagieren.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_in_with_tracked_device(
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
        """
```

```
        Signs in to Amazon Cognito as a user who has a tracked device. Signing in
        with a tracked device lets a user sign in without entering a new MFA
 code.

        Signing in with a tracked device requires that the client respond to the
 SRP
        protocol. The scenario associated with this example uses the warrant
 package
        to help with SRP calculations.

        For more information on SRP, see https://en.wikipedia.org/wiki/
Secure_Remote_Password_protocol.

        :param user_name: The user that is associated with the device.
        :param password: The user's password.
        :param device_key: The key of a tracked device.
        :param device_group_key: The group key of a tracked device.
        :param device_password: The password that is associated with the device.
        :param aws_srp: A class that helps with SRP calculations. The scenario
                        associated with this example uses the warrant package.
        :return: The result of the authentication. When successful, this contains
 an
                 access token for the user.
        """
        try:
            srp_helper = aws_srp.AWSSRP(
                username=user_name,
                password=device_password,
                pool_id="_",
                client_id=self.client_id,
                client_secret=None,
                client=self.cognito_idp_client,
            )

            response_init = self.cognito_idp_client.initiate_auth(
                ClientId=self.client_id,
                AuthFlow="USER_PASSWORD_AUTH",
                AuthParameters={
                    "USERNAME": user_name,
                    "PASSWORD": password,
                    "DEVICE_KEY": device_key,
                },
            )
            if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
```

```
                raise RuntimeError(
                    f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']}."
                )

            auth_params = srp_helper.get_auth_params()
            auth_params["DEVICE_KEY"] = device_key
            response_auth = self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_SRP_AUTH",
                ChallengeResponses=auth_params,
            )
            if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
                raise RuntimeError(
                    f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
                    f"{response_init['ChallengeName']}."
                )

            challenge_params = response_auth["ChallengeParameters"]
            challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
            cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
            cr["USERNAME"] = user_name
            cr["DEVICE_KEY"] = device_key
            response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_PASSWORD_VERIFIER",
                ChallengeResponses=cr,
            )
            auth_tokens = response_verifier["AuthenticationResult"]
        except ClientError as err:
            logger.error(
                "Couldn't start client sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return auth_tokens
```

- Einzelheiten zur API finden Sie RespondToAuthChallengein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`SignUp`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`SignUp`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Bestätigen Sie bekannte Benutzer automatisch mit einer Lambda-Funktion
- Automatisches Migrieren bekannter Benutzer mit einer Lambda-Funktion
- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Sign up a new user.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The username to use.</param>
    /// <param name="password">The user's password.</param>
    /// <param name="email">The email address of the user.</param>
    /// <returns>A Boolean value indicating whether the user was confirmed.</
returns>
    public async Task<bool> SignUpAsync(string clientId, string userName, string
 password, string email)
```

```
    {
        var userAttrs = new AttributeType
        {
            Name = "email",
            Value = email,
        };

        var userAttrsList = new List<AttributeType>();

        userAttrsList.Add(userAttrs);

        var signUpRequest = new SignUpRequest
        {
            UserAttributes = userAttrsList,
            Username = userName,
            ClientId = clientId,
            Password = password
        };

        var response = await _cognitoService.SignUpAsync(signUpRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie SignUpin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";
```

```
   Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

       Aws::CognitoIdentityProvider::Model::SignUpRequest request;
       request.AddUserAttributes(
               Aws::CognitoIdentityProvider::Model::AttributeType().WithName(
                       "email").WithValue(email));
       request.SetUsername(userName);
       request.SetPassword(password);
       request.SetClientId(clientID);
       Aws::CognitoIdentityProvider::Model::SignUpOutcome outcome =
               client.SignUp(request);

       if (outcome.IsSuccess()) {
           std::cout << "The signup request for " << userName << " was
successful."
                     << std::endl;
       }
       else if (outcome.GetError().GetErrorType() ==

Aws::CognitoIdentityProvider::CognitoIdentityProviderErrors::USERNAME_EXISTS) {
           std::cout
                   << "The username already exists. Please enter a different
username."
                   << std::endl;
           userExists = true;
       }
       else {
           std::cerr << "Error with CognitoIdentityProvider::SignUpRequest. "
                     << outcome.GetError().GetMessage()
                     << std::endl;
           return false;
       }
```

- Einzelheiten zur API finden Sie SignUpin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Benutzer registrieren

In diesem Beispiel wird jane@example.com registriert.

Befehl:

```
aws cognito-idp sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username jane@example.com --password PASSWORD --user-attributes
 Name="email",Value="jane@example.com" Name="name",Value="Jane"
```

Ausgabe:

```
{
  "UserConfirmed": false,
  "UserSub": "e04d60a6-45dc-441c-a40b-e25a787d4862"
}
```

- Einzelheiten zur API finden Sie SignUpin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &cognitoidentityprovider.SignUpInput{
```

```
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}
```

- Einzelheiten zur API finden Sie SignUpin der AWS SDK for Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static void signUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String userName,
           String password, String email) {
       AttributeType userAttrs = AttributeType.builder()
               .name("email")
               .value(email)
               .build();
```

```
        List<AttributeType> userAttrsList = new ArrayList<>();
        userAttrsList.add(userAttrs);
        try {
            SignUpRequest signUpRequest = SignUpRequest.builder()
                    .userAttributes(userAttrsList)
                    .username(userName)
                    .clientId(clientId)
                    .password(password)
                    .build();

            identityProviderClient.signUp(signUpRequest);
            System.out.println("User has been signed up ");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie SignUpin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const signUp = ({ clientId, username, password, email }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new SignUpCommand({
    ClientId: clientId,
    Username: username,
    Password: password,
    UserAttributes: [{ Name: "email", Value: email }],
  });
```

```
  return client.send(command);
};
```

- Einzelheiten zur API finden Sie SignUpin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun signUp(clientIdVal: String?, userNameVal: String?, passwordVal:
 String?, emailVal: String?) {
    val userAttrs = AttributeType {
        name = "email"
        value = emailVal
    }

    val userAttrsList = mutableListOf<AttributeType>()
    userAttrsList.add(userAttrs)
    val signUpRequest = SignUpRequest {
        userAttributes = userAttrsList
        username = userNameVal
        clientId = clientIdVal
        password = passwordVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.signUp(signUpRequest)
        println("User has been signed up")
    }
}
```

- API-Details finden Sie SignUpin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_up_user(self, user_name, password, user_email):
        """
        Signs up a new user with Amazon Cognito. This action prompts Amazon
 Cognito
        to send an email to the specified email address. The email contains a
 code that
        can be used to confirm the user.

        When the user already exists, the user status is checked to determine
 whether
        the user has been confirmed.

        :param user_name: The user name that identifies the new user.
```

```
        :param password: The password for the new user.
        :param user_email: The email address for the new user.
        :return: True when the user is already confirmed with Amazon Cognito.
                 Otherwise, false.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "Password": password,
                "UserAttributes": [{"Name": "email", "Value": user_email}],
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.sign_up(**kwargs)
            confirmed = response["UserConfirmed"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "UsernameExistsException":
                response = self.cognito_idp_client.admin_get_user(
                    UserPoolId=self.user_pool_id, Username=user_name
                )
                logger.warning(
                    "User %s exists and is %s.", user_name,
response["UserStatus"]
                )
                confirmed = response["UserStatus"] == "CONFIRMED"
            else:
                logger.error(
                    "Couldn't sign up %s. Here's why: %s: %s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return confirmed
```

- Einzelheiten zur API finden Sie SignUpin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UpdateUserPool`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`UpdateUserPool`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Bestätigen Sie bekannte Benutzer automatisch mit einer Lambda-Funktion
- Automatisches Migrieren bekannter Benutzer mit einer Lambda-Funktion
- Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung

CLI

AWS CLI

Um einen Benutzerpool zu aktualisieren

In diesem Beispiel werden einem Benutzerpool Tags hinzugefügt.

Befehl:

```
aws cognito-idp update-user-pool --user-pool-id us-west-2_aaaaaaaaa --user-pool-
tags Team=Blue,Area=West
```

- Einzelheiten zur API finden Sie UpdateUserPoolin der AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int

const (
 PreSignUp Trigger = iota
 UserMigration
 PostAuthentication
)

type TriggerInfo struct {
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
```

```
  case PostAuthentication:
   lambdaConfig.PostAuthentication = trigger.HandlerArn
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
 }
 return err
}
```

- Einzelheiten zur API finden Sie UpdateUserPoolin der AWS SDK for Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`VerifySoftwareToken`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`VerifySoftwareToken`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
    /// <summary>
    /// Verify the TOTP and register for MFA.
    /// </summary>
    /// <param name="session">The name of the session.</param>
    /// <param name="code">The MFA code.</param>
    /// <returns>The status of the software token.</returns>
    public async Task<VerifySoftwareTokenResponseType>
VerifySoftwareTokenAsync(string session, string code)
    {
        var tokenRequest = new VerifySoftwareTokenRequest
        {
            UserCode = code,
            Session = session,
        };

        var verifyResponse = await
_cognitoService.VerifySoftwareTokenAsync(tokenRequest);

        return verifyResponse.Status;
    }
```

- Einzelheiten zur API finden Sie VerifySoftwareTokenin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";
```

```
   Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

       Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenRequest request;
       request.SetUserCode(userCode);
       request.SetSession(session);

       Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenOutcome outcome =
               client.VerifySoftwareToken(request);

       if (outcome.IsSuccess()) {
           std::cout << "Verification of the code was successful."
                     << std::endl;
           session = outcome.GetResult().GetSession();
       }
       else {
           std::cerr << "Error with
CognitoIdentityProvider::VerifySoftwareToken. "
                     << outcome.GetError().GetMessage()
                     << std::endl;
           return false;
       }
```

- Einzelheiten zur API finden Sie VerifySoftwareTokenin der AWS SDK for C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   // Verify the TOTP and register for MFA.
   public static void verifyTOTP(CognitoIdentityProviderClient
identityProviderClient, String session, String code) {
       try {
           VerifySoftwareTokenRequest tokenRequest =
VerifySoftwareTokenRequest.builder()
```

```
                    .userCode(code)
                    .session(session)
                    .build();

            VerifySoftwareTokenResponse verifyResponse =
 identityProviderClient.verifySoftwareToken(tokenRequest);
            System.out.println("The status of the token is " +
 verifyResponse.statusAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie VerifySoftwareTokenin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
const verifySoftwareToken = (totp) => {
  const client = new CognitoIdentityProviderClient({});

  // The 'Session' is provided in the response to 'AssociateSoftwareToken'.
  const session = process.env.SESSION;

  if (!session) {
    throw new Error(
      "Missing a valid Session. Did you run 'admin-initiate-auth'?",
    );
  }
```

```
  const command = new VerifySoftwareTokenCommand({
    Session: session,
    UserCode: totp,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie VerifySoftwareTokenin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Verify the TOTP and register for MFA.
suspend fun verifyTOTP(sessionVal: String?, codeVal: String?) {
    val tokenRequest = VerifySoftwareTokenRequest {
        userCode = codeVal
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val verifyResponse =
 identityProviderClient.verifySoftwareToken(tokenRequest)
        println("The status of the token is ${verifyResponse.status}")
    }
}
```

- API-Details finden Sie VerifySoftwareTokenin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def verify_mfa(self, session, user_code):
        """
        Verify a new MFA application that is associated with a user.

        :param session: Session information returned from a previous call to
 initiate
                        authentication.
        :param user_code: A code generated by the associated MFA application.
        :return: Status that indicates whether the MFA application is verified.
        """
        try:
            response = self.cognito_idp_client.verify_software_token(
                Session=session, UserCode=user_code
```

```
            )
        except ClientError as err:
            logger.error(
                "Couldn't verify MFA. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response
```

- Einzelheiten zur API finden Sie VerifySoftwareTokenin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Service mit einem SDK AWS Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Szenarien für Amazon Cognito Identity Provider mit AWS SDKs

Die folgenden Codebeispiele zeigen Ihnen, wie Sie allgemeine Szenarien in Amazon Cognito Identity Provider mit AWS SDKs implementieren. Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben ausführen können, indem Sie mehrere Funktionen in Amazon Cognito Identity Provider aufrufen. Jedes Szenario enthält einen Link zu GitHub, wo Sie Anweisungen zur Einrichtung und Ausführung des Codes finden.

Beispiele

- Bestätigen Sie bekannte Amazon Cognito Cognito-Benutzer automatisch mit einer Lambda-Funktion mithilfe eines SDK AWS
- Automatisches Migrieren bekannter Amazon Cognito Cognito-Benutzer mit einer Lambda-Funktion mithilfe eines SDK AWS
- Registrieren Sie einen Benutzer mit einem Amazon Cognito Cognito-Benutzerpool, für den MFA erforderlich ist, mithilfe eines SDK AWS
- Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung mithilfe eines SDK AWS

Bestätigen Sie bekannte Amazon Cognito Cognito-Benutzer automatisch mit einer Lambda-Funktion mithilfe eines SDK AWS

Das folgende Codebeispiel zeigt, wie bekannte Amazon Cognito Cognito-Benutzer automatisch mit einer Lambda-Funktion bestätigt werden.

- Konfigurieren Sie einen Benutzerpool, um eine Lambda-Funktion für den `PreSignUp` Trigger aufzurufen.
- Melden Sie einen Benutzer bei Amazon Cognito an.
- Die Lambda-Funktion scannt eine DynamoDB-Tabelle und bestätigt automatisch bekannte Benutzer.
- Melden Sie sich als neuer Benutzer an und bereinigen Sie anschließend die Ressourcen.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
// AutoConfirm separates the steps of this scenario into individual functions so
 that
// they are simpler to read and understand.
type AutoConfirm struct {
 helper       IScenarioHelper
 questioner   demotools.IQuestioner
 resources    Resources
 cognitoActor *actions.CognitoActions
}

// NewAutoConfirm constructs a new auto confirm runner.
func NewAutoConfirm(sdkConfig aws.Config, questioner demotools.IQuestioner,
 helper IScenarioHelper) AutoConfirm {
 scenario := AutoConfirm{
```

```
  helper:       helper,
  questioner:   questioner,
  resources:    Resources{},
  cognitoActor: &actions.CognitoActions{CognitoClient:
 cognitoidentityprovider.NewFromConfig(sdkConfig)},
 }
 scenario.resources.init(scenario.cognitoActor, questioner)
 return scenario
}

// AddPreSignUpTrigger adds a Lambda handler as an invocation target for the
 PreSignUp trigger.
func (runner *AutoConfirm) AddPreSignUpTrigger(userPoolId string, functionArn
 string) {
 log.Printf("Let's add a Lambda function to handle the PreSignUp trigger from
 Cognito.\n" +
  "This trigger happens when a user signs up, and lets your function take action
 before the main Cognito\n" +
  "sign up processing occurs.\n")
 err := runner.cognitoActor.UpdateTriggers(
  userPoolId,
  actions.TriggerInfo{Trigger: actions.PreSignUp, HandlerArn:
 aws.String(functionArn)})
 if err != nil {
  panic(err)
 }
 log.Printf("Lambda function %v added to user pool %v to handle the PreSignUp
 trigger.\n",
  functionArn, userPoolId)
}

// SignUpUser signs up a user from the known user table with a password you
 specify.
func (runner *AutoConfirm) SignUpUser(clientId string, usersTable string)
 (string, string) {
 log.Println("Let's sign up a user to your Cognito user pool. When the user's
 email matches an email in the\n" +
  "DynamoDB known users table, it is automatically verified and the user is
 confirmed.")

 knownUsers, err := runner.helper.GetKnownUsers(usersTable)
 if err != nil {
  panic(err)
 }
```

```
 userChoice := runner.questioner.AskChoice("Which user do you want to use?\n",
 knownUsers.UserNameList())
 user := knownUsers.Users[userChoice]

 var signedUp bool
 var userConfirmed bool
 password := runner.questioner.AskPassword("Enter a password that has at least
 eight characters, uppercase, lowercase, numbers and symbols.\n"+
  "(the password will not display as you type):", 8)
 for !signedUp {
  log.Printf("Signing up user '%v' with email '%v' to Cognito.\n", user.UserName,
 user.UserEmail)
  userConfirmed, err = runner.cognitoActor.SignUp(clientId, user.UserName,
 password, user.UserEmail)
  if err != nil {
   var invalidPassword *types.InvalidPasswordException
   if errors.As(err, &invalidPassword) {
    password = runner.questioner.AskPassword("Enter another password:", 8)
   } else {
    panic(err)
   }
  } else {
   signedUp = true
  }
 }
 log.Printf("User %v signed up, confirmed = %v.\n", user.UserName, userConfirmed)

 log.Println(strings.Repeat("-", 88))

 return user.UserName, password
}

// SignInUser signs in a user.
func (runner *AutoConfirm) SignInUser(clientId string, userName string, password
 string) string {
 runner.questioner.Ask("Press Enter when you're ready to continue.")
 log.Printf("Let's sign in as %v...\n", userName)
 authResult, err := runner.cognitoActor.SignIn(clientId, userName, password)
 if err != nil {
  panic(err)
 }
 log.Printf("Successfully signed in. Your access token starts with: %v...\n",
 (*authResult.AccessToken)[:10])
 log.Println(strings.Repeat("-", 88))
```

```
 return *authResult.AccessToken
}

// Run runs the scenario.
func (runner *AutoConfirm) Run(stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup()
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(stackOutputs["TableName"])

 runner.AddPreSignUpTrigger(stackOutputs["UserPoolId"],
 stackOutputs["AutoConfirmFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers, actions.PreSignUp)
 userName, password := runner.SignUpUser(stackOutputs["UserPoolClientId"],
 stackOutputs["TableName"])
 runner.helper.ListRecentLogEvents(stackOutputs["AutoConfirmFunction"])
 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
  runner.SignInUser(stackOutputs["UserPoolClientId"], userName, password))

 runner.resources.Cleanup()

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Behandeln Sie den `PreSignUp` Trigger mit einer Lambda-Funktion.

```
const TABLE_NAME = "TABLE_NAME"

// UserInfo defines structured user data that can be marshalled to a DynamoDB
 format.
type UserInfo struct {
 UserName  string `dynamodbav:"UserName"`
 UserEmail string `dynamodbav:"UserEmail"`
}

// GetKey marshals the user email value to a DynamoDB key format.
func (user UserInfo) GetKey() map[string]dynamodbtypes.AttributeValue {
 userEmail, err := attributevalue.Marshal(user.UserEmail)
 if err != nil {
  panic(err)
 }
 return map[string]dynamodbtypes.AttributeValue{"UserEmail": userEmail}
}

type handler struct {
 dynamoClient *dynamodb.Client
}

// HandleRequest handles the PreSignUp event by looking up a user in an Amazon
 DynamoDB table and
// specifying whether they should be confirmed and verified.
func (h *handler) HandleRequest(ctx context.Context, event
 events.CognitoEventUserPoolsPreSignup) (events.CognitoEventUserPoolsPreSignup,
 error) {
 log.Printf("Received presignup from %v for user '%v'", event.TriggerSource,
 event.UserName)
 if event.TriggerSource != "PreSignUp_SignUp" {
  // Other trigger sources, such as PreSignUp_AdminInitiateAuth, ignore the
 response from this handler.
  return event, nil
 }
 tableName := os.Getenv(TABLE_NAME)
 user := UserInfo{
  UserEmail: event.Request.UserAttributes["email"],
 }
 log.Printf("Looking up email %v in table %v.\n", user.UserEmail, tableName)
 output, err := h.dynamoClient.GetItem(ctx, &dynamodb.GetItemInput{
  Key:       user.GetKey(),
```

```
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Error looking up email %v.\n", user.UserEmail)
  return event, err
 }
 if output.Item == nil {
  log.Printf("Email %v not found. Email verification is required.\n",
 user.UserEmail)
  return event, err
 }

 err = attributevalue.UnmarshalMap(output.Item, &user)
 if err != nil {
  log.Printf("Couldn't unmarshal DynamoDB item. Here's why: %v\n", err)
  return event, err
 }

 if user.UserName != event.UserName {
  log.Printf("UserEmail %v found, but stored UserName '%v' does not match
 supplied UserName '%v'. Verification is required.\n",
   user.UserEmail, user.UserName, event.UserName)
 } else {
  log.Printf("UserEmail %v found with matching UserName %v. User is confirmed.
\n", user.UserEmail, user.UserName)
  event.Response.AutoConfirmUser = true
  event.Response.AutoVerifyEmail = true
 }

 return event, err
}

func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Erstellen Sie eine Struktur, die allgemeine Aufgaben ausführt.

```
// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(stackName string) (actions.StackOutputs, error)
 PopulateUserTable(tableName string)
 GetKnownUsers(tableName string) (actions.UserList, error)
 AddKnownUser(tableName string, user actions.User)
 ListRecentLogEvents(functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
 example.
type ScenarioHelper struct {
 questioner  demotools.IQuestioner
 dynamoActor *actions.DynamoActions
 cfnActor    *actions.CloudFormationActions
 cwlActor    *actions.CloudWatchLogsActions
 isTestRun   bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
 ScenarioHelper {
 scenario := ScenarioHelper{
  questioner:  questioner,
  dynamoActor: &actions.DynamoActions{DynamoClient:
 dynamodb.NewFromConfig(sdkConfig)},
  cfnActor:    &actions.CloudFormationActions{CfnClient:
 cloudformation.NewFromConfig(sdkConfig)},
  cwlActor:    &actions.CloudWatchLogsActions{CwlClient:
 cloudwatchlogs.NewFromConfig(sdkConfig)},
 }
 return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
```

```
 if !helper.isTestRun {
  time.Sleep(time.Duration(secs) * time.Second)
 }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
 structured format.
func (helper ScenarioHelper) GetStackOutputs(stackName string)
 (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(tableName string) {
 log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
 this example.\n", tableName)
 err := helper.dynamoActor.PopulateTable(tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
 error) {
 knownUsers, err := helper.dynamoActor.Scan(tableName)
 if err != nil {
  log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
 tableName, err)
 }
 return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(tableName string, user actions.User) {
 log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
 table...\n",
  user.UserName, user.UserEmail)
 err := helper.dynamoActor.AddUser(tableName, user)
 if err != nil {
  panic(err)
 }
}
```

```
// ListRecentLogEvents gets the most recent log stream and events for the
 specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(functionName string) {
 log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
 helper.Pause(10)
 log.Println("Okay, let's check the logs to find what's happened recently with
 your Lambda function.")
 logStream, err := helper.cwlActor.GetLatestLogStream(functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(functionName,
 *logStream.LogStreamName, 10)
 if err != nil {
  panic(err)
 }
 for _, event := range events {
  log.Printf("\t%v", *event.Message)
 }
 log.Println(strings.Repeat("-", 88))
}
```

Erstellen Sie eine Struktur, die Amazon Cognito Cognito-Aktionen umschließt.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int

const (
 PreSignUp Trigger = iota
```

```
 UserMigration
 PostAuthentication
)

type TriggerInfo struct {
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
  case PostAuthentication:
   lambdaConfig.PostAuthentication = trigger.HandlerArn
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
 }
 return err
}
```

```
// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &cognitoidentityprovider.SignUpInput{
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}



// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
```

```
  if errors.As(err, &resetRequired) {
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
  }
 } else {
  authResult = output.AuthenticationResult
 }
 return authResult, err
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
  Username:         aws.String(userName),
 })
 if err != nil {
```

```
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
 userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
```

```
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
 }
 return err
}



// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
  }
 }
 return err
}
```

Erstellen Sie eine Struktur, die DynamoDB-Aktionen umschließt.

```
// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
 actions
// used in the examples.
type DynamoActions struct {
 DynamoClient *dynamodb.Client
```

```
}

// User defines structured user data.
type User struct {
 UserName  string
 UserEmail string
 LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
 UserPoolId string
 ClientId   string
 Time       string
}

// UserList defines a list of users.
type UserList struct {
 Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
 strings.
func (users *UserList) UserNameList() []string {
 names := make([]string, len(users.Users))
 for i := 0; i < len(users.Users); i++ {
  names[i] = users.Users[i].UserName
 }
 return names
}

// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(tableName string) error {
 var err error
 var item map[string]types.AttributeValue
 var writeReqs []types.WriteRequest
 for i := 1; i < 4; i++ {
  item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
  if err != nil {
   log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
 err)
   return err
  }
```

```
  writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
 &types.PutRequest{Item: item}})
 }
 _, err = actor.DynamoClient.BatchWriteItem(context.TODO(),
 &dynamodb.BatchWriteItemInput{
  RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
 })
 if err != nil {
  log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
 tableName, err)
 }
 return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(tableName string) (UserList, error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(context.TODO(), &dynamodb.ScanInput{
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
 err)
 } else {
  err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
  if err != nil {
   log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
  }
 }
 return userList, err
}

// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(tableName string, user User) error {
 userItem, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
 }
 _, err = actor.DynamoClient.PutItem(context.TODO(), &dynamodb.PutItemInput{
  Item:      userItem,
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
```

```
 }
 return err
}
```

Erstellen Sie eine Struktur, die Logs-Aktionen umschließt CloudWatch .

```
type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
 (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
 &cloudwatchlogs.DescribeLogStreamsInput{
  Descending:   aws.Bool(true),
  Limit:        aws.Int32(1),
  LogGroupName: aws.String(logGroupName),
  OrderBy:      types.OrderByLastEventTime,
 })
 if err != nil {
  log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
 logGroupName, err)
 } else {
  logStream = output.LogStreams[0]
 }
 return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
 stream.
func (actor CloudWatchLogsActions) GetLogEvents(functionName string,
 logStreamName string, eventCount int32) (
 []types.OutputLogEvent, error) {
 var events []types.OutputLogEvent
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.GetLogEvents(context.TODO(),
 &cloudwatchlogs.GetLogEventsInput{
```

```
  LogStreamName: aws.String(logStreamName),
  Limit:         aws.Int32(eventCount),
  LogGroupName:  aws.String(logGroupName),
 })
 if err != nil {
  log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
 logStreamName, err)
 } else {
  events = output.Events
 }
 return events, err
}
```

Erstellen Sie eine Struktur, die Aktionen umschließt. AWS CloudFormation

```
// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
 CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
 structured format.
func (actor CloudFormationActions) GetOutputs(stackName string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(context.TODO(),
 &cloudformation.DescribeStacksInput{
  StackName: aws.String(stackName),
 })
 if err != nil || len(output.Stacks) == 0 {
  log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
 stackName, err)
 }
 stackOutputs := StackOutputs{}
 for _, out := range output.Stacks[0].Outputs {
  stackOutputs[*out.OutputKey] = *out.OutputValue
 }
 return stackOutputs
}
```

Ressourcen bereinigen.

```
// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
 userPoolId       string
 userAccessTokens []string
 triggers         []actions.Trigger

 cognitoActor *actions.CognitoActions
 questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
 demotools.IQuestioner) {
 resources.userAccessTokens = []string{}
 resources.triggers = []actions.Trigger{}
 resources.cognitoActor = cognitoActor
 resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup() {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong during cleanup.\n%v\n", r)
   log.Println("Use the AWS Management Console to remove any remaining resources
 \n" +
    "that were created for this scenario.")
  }
 }()

 wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
 resources that were created "+
  "during this demo (y/n)?", "y")
 if wantDelete {
  for _, accessToken := range resources.userAccessTokens {
   err := resources.cognitoActor.DeleteUser(accessToken)
   if err != nil {
    log.Println("Couldn't delete user during cleanup.")
```

```
    panic(err)
   }
   log.Println("Deleted user.")
  }
  triggerList := make([]actions.TriggerInfo, len(resources.triggers))
  for i := 0; i < len(resources.triggers); i++ {
   triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
 HandlerArn: nil}
  }
  err := resources.cognitoActor.UpdateTriggers(resources.userPoolId,
 triggerList...)
  if err != nil {
   log.Println("Couldn't update Cognito triggers during cleanup.")
   panic(err)
  }
  log.Println("Removed Cognito triggers from user pool.")
 } else {
  log.Println("Be sure to remove resources when you're done with them to avoid
 unexpected charges!")
 }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Go -API-Referenz.
  - DeleteUser
  - InitiateAuth
  - SignUp
  - UpdateUserPool

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Automatisches Migrieren bekannter Amazon Cognito Cognito-Benutzer mit einer Lambda-Funktion mithilfe eines SDK AWS

Das folgende Codebeispiel zeigt, wie bekannte Amazon Cognito Cognito-Benutzer mit einer Lambda-Funktion automatisch migriert werden.

- Konfigurieren Sie einen Benutzerpool, um eine Lambda-Funktion für den `MigrateUser` Trigger aufzurufen.
- Melden Sie sich bei Amazon Cognito mit einem Benutzernamen und einer E-Mail-Adresse an, die sich nicht im Benutzerpool befinden.
- Die Lambda-Funktion scannt eine DynamoDB-Tabelle und migriert bekannte Benutzer automatisch in den Benutzerpool.
- Führen Sie den Vorgang „Passwort vergessen“ aus, um das Passwort für den migrierten Benutzer zurückzusetzen.
- Melden Sie sich als neuer Benutzer an und bereinigen Sie anschließend die Ressourcen.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
import (
 "errors"
 "fmt"
 "log"
 "strings"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// MigrateUser separates the steps of this scenario into individual functions so
 that
// they are simpler to read and understand.
```

```
type MigrateUser struct {
 helper       IScenarioHelper
 questioner   demotools.IQuestioner
 resources    Resources
 cognitoActor *actions.CognitoActions
}

// NewMigrateUser constructs a new migrate user runner.
func NewMigrateUser(sdkConfig aws.Config, questioner demotools.IQuestioner,
 helper IScenarioHelper) MigrateUser {
 scenario := MigrateUser{
  helper:       helper,
  questioner:   questioner,
  resources:    Resources{},
  cognitoActor: &actions.CognitoActions{CognitoClient:
 cognitoidentityprovider.NewFromConfig(sdkConfig)},
 }
 scenario.resources.init(scenario.cognitoActor, questioner)
 return scenario
}

// AddMigrateUserTrigger adds a Lambda handler as an invocation target for the
 MigrateUser trigger.
func (runner *MigrateUser) AddMigrateUserTrigger(userPoolId string, functionArn
 string) {
 log.Printf("Let's add a Lambda function to handle the MigrateUser trigger from
 Cognito.\n" +
  "This trigger happens when an unknown user signs in, and lets your function
 take action before Cognito\n" +
  "rejects the user.\n\n")
 err := runner.cognitoActor.UpdateTriggers(
  userPoolId,
  actions.TriggerInfo{Trigger: actions.UserMigration, HandlerArn:
 aws.String(functionArn)})
 if err != nil {
  panic(err)
 }
 log.Printf("Lambda function %v added to user pool %v to handle the MigrateUser
 trigger.\n",
  functionArn, userPoolId)

 log.Println(strings.Repeat("-", 88))
}
```

```
// SignInUser adds a new user to the known users table and signs that user in to
 Amazon Cognito.
func (runner *MigrateUser) SignInUser(usersTable string, clientId string) (bool,
 actions.User) {
 log.Println("Let's sign in a user to your Cognito user pool. When the username
 and email matches an entry in the\n" +
  "DynamoDB known users table, the email is automatically verified and the user
 is migrated to the Cognito user pool.")

 user := actions.User{}
 user.UserName = runner.questioner.Ask("\nEnter a username:")
 user.UserEmail = runner.questioner.Ask("\nEnter an email that you own. This
 email will be used to confirm user migration\n" +
  "during this example:")

 runner.helper.AddKnownUser(usersTable, user)

 var err error
 var resetRequired *types.PasswordResetRequiredException
 var authResult *types.AuthenticationResultType
 signedIn := false
 for !signedIn && resetRequired == nil {
  log.Printf("Signing in to Cognito as user '%v'. The expected result is a
 PasswordResetRequiredException.\n\n", user.UserName)
  authResult, err = runner.cognitoActor.SignIn(clientId, user.UserName, "_")
  if err != nil {
   if errors.As(err, &resetRequired) {
    log.Printf("\nUser '%v' is not in the Cognito user pool but was found in the
 DynamoDB known users table.\n"+
     "User migration is started and a password reset is required.",
 user.UserName)
   } else {
    panic(err)
   }
  } else {
   log.Printf("User '%v' successfully signed in. This is unexpected and probably
 means you have not\n"+
    "cleaned up a previous run of this scenario, so the user exist in the Cognito
 user pool.\n"+
    "You can continue this example and select to clean up resources, or manually
 remove\n"+
    "the user from your user pool and try again.", user.UserName)
   runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
 *authResult.AccessToken)
```

```
    signedIn = true
   }
  }

  log.Println(strings.Repeat("-", 88))
  return resetRequired != nil, user
}

// ResetPassword starts a password recovery flow.
func (runner *MigrateUser) ResetPassword(clientId string, user actions.User) {
 wantCode := runner.questioner.AskBool(fmt.Sprintf("In order to migrate the user
 to Cognito, you must be able to receive a confirmation\n"+
  "code by email at %v. Do you want to send a code (y/n)?", user.UserEmail), "y")
 if !wantCode {
  log.Println("To complete this example and successfully migrate a user to
 Cognito, you must enter an email\n" +
   "you own that can receive a confirmation code.")
  return
 }
 codeDelivery, err := runner.cognitoActor.ForgotPassword(clientId, user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("\nA confirmation code has been sent to %v.",
 *codeDelivery.Destination)
 code := runner.questioner.Ask("Check your email and enter it here:")

 confirmed := false
 password := runner.questioner.AskPassword("\nEnter a password that has at least
 eight characters, uppercase, lowercase, numbers and symbols.\n"+
  "(the password will not display as you type):", 8)
 for !confirmed {
  log.Printf("\nConfirming password reset for user '%v'.\n", user.UserName)
  err = runner.cognitoActor.ConfirmForgotPassword(clientId, code, user.UserName,
 password)
  if err != nil {
   var invalidPassword *types.InvalidPasswordException
   if errors.As(err, &invalidPassword) {
    password = runner.questioner.AskPassword("\nEnter another password:", 8)
   } else {
    panic(err)
   }
  } else {
   confirmed = true
```

```
  }
 }
 log.Printf("User '%v' successfully confirmed and migrated.\n", user.UserName)
 log.Println("Signing in with your username and password...")
 authResult, err := runner.cognitoActor.SignIn(clientId, user.UserName, password)
 if err != nil {
  panic(err)
 }
 log.Printf("Successfully signed in. Your access token starts with: %v...\n",
 (*authResult.AccessToken)[:10])
 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
 *authResult.AccessToken)

 log.Println(strings.Repeat("-", 88))
}

// Run runs the scenario.
func (runner *MigrateUser) Run(stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup()
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]

 runner.AddMigrateUserTrigger(stackOutputs["UserPoolId"],
 stackOutputs["MigrateUserFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers,
 actions.UserMigration)
 resetNeeded, user := runner.SignInUser(stackOutputs["TableName"],
 stackOutputs["UserPoolClientId"])
 if resetNeeded {
  runner.helper.ListRecentLogEvents(stackOutputs["MigrateUserFunction"])
```

```
  runner.ResetPassword(stackOutputs["UserPoolClientId"], user)
 }

 runner.resources.Cleanup()

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Behandeln Sie den `MigrateUser` Trigger mit einer Lambda-Funktion.

```
const TABLE_NAME = "TABLE_NAME"

// UserInfo defines structured user data that can be marshalled to a DynamoDB
 format.
type UserInfo struct {
 UserName  string `dynamodbav:"UserName"`
 UserEmail string `dynamodbav:"UserEmail"`
}

type handler struct {
 dynamoClient *dynamodb.Client
}

// HandleRequest handles the MigrateUser event by looking up a user in an Amazon
 DynamoDB table and
// specifying whether they should be migrated to the user pool.
func (h *handler) HandleRequest(ctx context.Context, event
 events.CognitoEventUserPoolsMigrateUser)
 (events.CognitoEventUserPoolsMigrateUser, error) {
 log.Printf("Received migrate trigger from %v for user '%v'",
 event.TriggerSource, event.UserName)
 if event.TriggerSource != "UserMigration_Authentication" {
  return event, nil
 }
 tableName := os.Getenv(TABLE_NAME)
 user := UserInfo{
  UserName: event.UserName,
 }
```

```
log.Printf("Looking up user '%v' in table %v.\n", user.UserName, tableName)
filterEx := expression.Name("UserName").Equal(expression.Value(user.UserName))
expr, err := expression.NewBuilder().WithFilter(filterEx).Build()
if err != nil {
 log.Printf("Error building expression to query for user '%v'.\n",
user.UserName)
 return event, err
}
output, err := h.dynamoClient.Scan(ctx, &dynamodb.ScanInput{
 TableName:                 aws.String(tableName),
 FilterExpression:          expr.Filter(),
 ExpressionAttributeNames:  expr.Names(),
 ExpressionAttributeValues: expr.Values(),
})
if err != nil {
 log.Printf("Error looking up user '%v'.\n", user.UserName)
 return event, err
}
if output.Items == nil || len(output.Items) == 0 {
 log.Printf("User '%v' not found, not migrating user.\n", user.UserName)
 return event, err
}

var users []UserInfo
err = attributevalue.UnmarshalListOfMaps(output.Items, &users)
if err != nil {
 log.Printf("Couldn't unmarshal DynamoDB items. Here's why: %v\n", err)
 return event, err
}

user = users[0]
log.Printf("UserName '%v' found with email %v. User is migrated and must reset
password.\n", user.UserName, user.UserEmail)
event.CognitoEventUserPoolsMigrateUserResponse.UserAttributes =
map[string]string{
 "email":          user.UserEmail,
 "email_verified": "true", // email_verified is required for the forgot password
flow.
}
event.CognitoEventUserPoolsMigrateUserResponse.FinalUserStatus =
"RESET_REQUIRED"
event.CognitoEventUserPoolsMigrateUserResponse.MessageAction = "SUPPRESS"

return event, err
```

```
}

func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Erstellen Sie eine Struktur, die allgemeine Aufgaben ausführt.

```
// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(stackName string) (actions.StackOutputs, error)
 PopulateUserTable(tableName string)
 GetKnownUsers(tableName string) (actions.UserList, error)
 AddKnownUser(tableName string, user actions.User)
 ListRecentLogEvents(functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
 example.
type ScenarioHelper struct {
 questioner  demotools.IQuestioner
 dynamoActor *actions.DynamoActions
 cfnActor    *actions.CloudFormationActions
 cwlActor    *actions.CloudWatchLogsActions
 isTestRun   bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
 ScenarioHelper {
 scenario := ScenarioHelper{
```

```
  questioner:  questioner,
  dynamoActor: &actions.DynamoActions{DynamoClient:
 dynamodb.NewFromConfig(sdkConfig)},
  cfnActor:    &actions.CloudFormationActions{CfnClient:
 cloudformation.NewFromConfig(sdkConfig)},
  cwlActor:    &actions.CloudWatchLogsActions{CwlClient:
 cloudwatchlogs.NewFromConfig(sdkConfig)},
 }
 return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 if !helper.isTestRun {
  time.Sleep(time.Duration(secs) * time.Second)
 }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
 structured format.
func (helper ScenarioHelper) GetStackOutputs(stackName string)
 (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(tableName string) {
 log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
 this example.\n", tableName)
 err := helper.dynamoActor.PopulateTable(tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
 error) {
 knownUsers, err := helper.dynamoActor.Scan(tableName)
 if err != nil {
  log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
 tableName, err)
 }
```

```
 return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(tableName string, user actions.User) {
 log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
 table...\n",
  user.UserName, user.UserEmail)
 err := helper.dynamoActor.AddUser(tableName, user)
 if err != nil {
  panic(err)
 }
}

// ListRecentLogEvents gets the most recent log stream and events for the
 specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(functionName string) {
 log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
 helper.Pause(10)
 log.Println("Okay, let's check the logs to find what's happened recently with
 your Lambda function.")
 logStream, err := helper.cwlActor.GetLatestLogStream(functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(functionName,
 *logStream.LogStreamName, 10)
 if err != nil {
  panic(err)
 }
 for _, event := range events {
  log.Printf("\t%v", *event.Message)
 }
 log.Println(strings.Repeat("-", 88))
}
```

Erstellen Sie eine Struktur, die Amazon Cognito Cognito-Aktionen umschließt.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int

const (
 PreSignUp Trigger = iota
 UserMigration
 PostAuthentication
)

type TriggerInfo struct {
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
  case PostAuthentication:
```

```
   lambdaConfig.PostAuthentication = trigger.HandlerArn
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
 }
 return err
}



// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &cognitoidentityprovider.SignUpInput{
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}
```

```
// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
  if errors.As(err, &resetRequired) {
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
  }
 } else {
  authResult = output.AuthenticationResult
 }
 return authResult, err
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
}
```

```
// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
  Username:         aws.String(userName),
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
 userEmail string) error {
```

```
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
 }
 return err
}



// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
  }
 }
 return err
```

```
}
```

Erstellen Sie eine Struktur, die DynamoDB-Aktionen umschließt.

```
// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
 actions
// used in the examples.
type DynamoActions struct {
 DynamoClient *dynamodb.Client
}

// User defines structured user data.
type User struct {
 UserName  string
 UserEmail string
 LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
 UserPoolId string
 ClientId   string
 Time       string
}

// UserList defines a list of users.
type UserList struct {
 Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
 strings.
func (users *UserList) UserNameList() []string {
 names := make([]string, len(users.Users))
 for i := 0; i < len(users.Users); i++ {
  names[i] = users.Users[i].UserName
 }
 return names
}
```

```
// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(tableName string) error {
 var err error
 var item map[string]types.AttributeValue
 var writeReqs []types.WriteRequest
 for i := 1; i < 4; i++ {
  item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
  if err != nil {
   log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
 err)
   return err
  }
  writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
 &types.PutRequest{Item: item}})
 }
 _, err = actor.DynamoClient.BatchWriteItem(context.TODO(),
 &dynamodb.BatchWriteItemInput{
  RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
 })
 if err != nil {
  log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
 tableName, err)
 }
 return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(tableName string) (UserList, error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(context.TODO(), &dynamodb.ScanInput{
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
 err)
 } else {
  err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
  if err != nil {
   log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
  }
 }
 return userList, err
}
```

```
// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(tableName string, user User) error {
 userItem, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
 }
 _, err = actor.DynamoClient.PutItem(context.TODO(), &dynamodb.PutItemInput{
  Item:      userItem,
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
 }
 return err
}
```

Erstellen Sie eine Struktur, die Logs-Aktionen umschließt CloudWatch .

```
type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
 (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
 &cloudwatchlogs.DescribeLogStreamsInput{
  Descending:   aws.Bool(true),
  Limit:        aws.Int32(1),
  LogGroupName: aws.String(logGroupName),
  OrderBy:      types.OrderByLastEventTime,
 })
 if err != nil {
  log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
 logGroupName, err)
 } else {
  logStream = output.LogStreams[0]
```

```
 }
 return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
 stream.
func (actor CloudWatchLogsActions) GetLogEvents(functionName string,
 logStreamName string, eventCount int32) (
 []types.OutputLogEvent, error) {
 var events []types.OutputLogEvent
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.GetLogEvents(context.TODO(),
 &cloudwatchlogs.GetLogEventsInput{
  LogStreamName: aws.String(logStreamName),
  Limit:         aws.Int32(eventCount),
  LogGroupName:  aws.String(logGroupName),
 })
 if err != nil {
  log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
 logStreamName, err)
 } else {
  events = output.Events
 }
 return events, err
}
```

Erstellen Sie eine Struktur, die Aktionen umschließt. AWS CloudFormation

```
// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
 CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
 structured format.
func (actor CloudFormationActions) GetOutputs(stackName string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(context.TODO(),
 &cloudformation.DescribeStacksInput{
```

```
  StackName: aws.String(stackName),
 })
 if err != nil || len(output.Stacks) == 0 {
  log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
 stackName, err)
 }
 stackOutputs := StackOutputs{}
 for _, out := range output.Stacks[0].Outputs {
  stackOutputs[*out.OutputKey] = *out.OutputValue
 }
 return stackOutputs
}
```

Ressourcen bereinigen.

```
// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
 userPoolId       string
 userAccessTokens []string
 triggers         []actions.Trigger

 cognitoActor *actions.CognitoActions
 questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
 demotools.IQuestioner) {
 resources.userAccessTokens = []string{}
 resources.triggers = []actions.Trigger{}
 resources.cognitoActor = cognitoActor
 resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup() {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong during cleanup.\n%v\n", r)
```

```
    log.Println("Use the AWS Management Console to remove any remaining resources
  \n" +
     "that were created for this scenario.")
   }
  }()

  wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
  resources that were created "+
   "during this demo (y/n)?", "y")
  if wantDelete {
   for _, accessToken := range resources.userAccessTokens {
    err := resources.cognitoActor.DeleteUser(accessToken)
    if err != nil {
     log.Println("Couldn't delete user during cleanup.")
     panic(err)
    }
    log.Println("Deleted user.")
   }
   triggerList := make([]actions.TriggerInfo, len(resources.triggers))
   for i := 0; i < len(resources.triggers); i++ {
    triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
  HandlerArn: nil}
   }
   err := resources.cognitoActor.UpdateTriggers(resources.userPoolId,
  triggerList...)
   if err != nil {
    log.Println("Couldn't update Cognito triggers during cleanup.")
    panic(err)
   }
   log.Println("Removed Cognito triggers from user pool.")
  } else {
   log.Println("Be sure to remove resources when you're done with them to avoid
  unexpected charges!")
  }
 }
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Go -API-Referenz.
  - ConfirmForgotPassword
  - DeleteUser
  - ForgotPassword

- InitiateAuth
- SignUp
- UpdateUserPool

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Registrieren Sie einen Benutzer mit einem Amazon Cognito Cognito-Benutzerpool, für den MFA erforderlich ist, mithilfe eines SDK AWS

Die folgenden Code-Beispiele veranschaulichen Folgendes:

- Registrieren Sie einen Benutzer mit einem Benutzernamen, einem Passwort und einer E-Mail-Adresse und bestätigen Sie ihn.
- Einrichten der Multi-Faktor-Authentifizierung durch Zuordnung einer MFA-Anwendung zu dem Benutzer.
- Anmelden unter Verwendung eines Passworts und eines MFA-Codes.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
namespace CognitoBasics;

public class CognitoBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon Cognito.
```

```
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonCognitoIdentityProvider>()
            .AddTransient<CognitoWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<CognitoBasics>();

        var configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var cognitoWrapper = host.Services.GetRequiredService<CognitoWrapper>();

        Console.WriteLine(new string('-', 80));
        UiMethods.DisplayOverview();
        Console.WriteLine(new string('-', 80));

        // clientId - The app client Id value that you get from the AWS CDK
script.
        var clientId = configuration["ClientId"]; // "*** REPLACE WITH CLIENT ID
VALUE FROM CDK SCRIPT";

        // poolId - The pool Id that you get from the AWS CDK script.
        var poolId = configuration["PoolId"]!; // "*** REPLACE WITH POOL ID VALUE
FROM CDK SCRIPT";
        var userName = configuration["UserName"];
        var password = configuration["Password"];
        var email = configuration["Email"];

        // If the username wasn't set in the configuration file,
        // get it from the user now.
        if (userName is null)
```

```
        {
            do
            {
                Console.Write("Username: ");
                userName = Console.ReadLine();
            }
            while (string.IsNullOrEmpty(userName));
        }
        Console.WriteLine($"\nUsername: {userName}");

        // If the password wasn't set in the configuration file,
        // get it from the user now.
        if (password is null)
        {
            do
            {
                Console.Write("Password: ");
                password = Console.ReadLine();
            }
            while (string.IsNullOrEmpty(password));
        }

        // If the email address wasn't set in the configuration file,
        // get it from the user now.
        if (email is null)
        {
            do
            {
                Console.Write("Email: ");
                email = Console.ReadLine();
            } while (string.IsNullOrEmpty(email));
        }

        // Now sign up the user.
        Console.WriteLine($"\nSigning up {userName} with email address:
{email}");
        await cognitoWrapper.SignUpAsync(clientId, userName, password, email);

        // Add the user to the user pool.
        Console.WriteLine($"Adding {userName} to the user pool");
        await cognitoWrapper.GetAdminUserAsync(userName, poolId);

        UiMethods.DisplayTitle("Get confirmation code");
        Console.WriteLine($"Conformation code sent to {userName}.");
```

```
        Console.Write("Would you like to send a new code? (Y/N) ");
        var answer = Console.ReadLine();

        if (answer!.ToLower() == "y")
        {
            await cognitoWrapper.ResendConfirmationCodeAsync(clientId, userName);
            Console.WriteLine("Sending a new confirmation code");
        }

        Console.Write("Enter confirmation code (from Email): ");
        var code = Console.ReadLine();

        await cognitoWrapper.ConfirmSignupAsync(clientId, code, userName);

        UiMethods.DisplayTitle("Checking status");
        Console.WriteLine($"Rechecking the status of {userName} in the user
pool");
        await cognitoWrapper.GetAdminUserAsync(userName, poolId);

        Console.WriteLine($"Setting up authenticator for {userName} in the user
pool");
        var setupResponse = await cognitoWrapper.InitiateAuthAsync(clientId,
userName, password);

        var setupSession = await
cognitoWrapper.AssociateSoftwareTokenAsync(setupResponse.Session);
        Console.Write("Enter the 6-digit code displayed in Google Authenticator:
");
        var setupCode = Console.ReadLine();

        var setupResult = await
cognitoWrapper.VerifySoftwareTokenAsync(setupSession, setupCode);
        Console.WriteLine($"Setup status: {setupResult}");

        Console.WriteLine($"Now logging in {userName} in the user pool");
        var authSession = await cognitoWrapper.AdminInitiateAuthAsync(clientId,
poolId, userName, password);

        Console.Write("Enter a new 6-digit code displayed in Google
Authenticator: ");
        var authCode = Console.ReadLine();
```

```
        var authResult = await
 cognitoWrapper.AdminRespondToAuthChallengeAsync(userName, clientId, authCode,
 authSession, poolId);
        Console.WriteLine($"Authenticated and received access token:
 {authResult.AccessToken}");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Cognito scenario is complete.");
        Console.WriteLine(new string('-', 80));
    }
}


using System.Net;

namespace CognitoActions;

/// <summary>
/// Methods to perform Amazon Cognito Identity Provider actions.
/// </summary>
public class CognitoWrapper
{
    private readonly IAmazonCognitoIdentityProvider _cognitoService;

    /// <summary>
    /// Constructor for the wrapper class containing Amazon Cognito actions.
    /// </summary>
    /// <param name="cognitoService">The Amazon Cognito client object.</param>
    public CognitoWrapper(IAmazonCognitoIdentityProvider cognitoService)
    {
        _cognitoService = cognitoService;
    }

    /// <summary>
    /// List the Amazon Cognito user pools for an account.
    /// </summary>
    /// <returns>A list of UserPoolDescriptionType objects.</returns>
    public async Task<List<UserPoolDescriptionType>> ListUserPoolsAsync()
    {
        var userPools = new List<UserPoolDescriptionType>();

        var userPoolsPaginator = _cognitoService.Paginators.ListUserPools(new
 ListUserPoolsRequest());
```

```
        await foreach (var response in userPoolsPaginator.Responses)
        {
            userPools.AddRange(response.UserPools);
        }

        return userPools;
    }


    /// <summary>
    /// Get a list of users for the Amazon Cognito user pool.
    /// </summary>
    /// <param name="userPoolId">The user pool ID.</param>
    /// <returns>A list of users.</returns>
    public async Task<List<UserType>> ListUsersAsync(string userPoolId)
    {
        var request = new ListUsersRequest
        {
            UserPoolId = userPoolId
        };

        var users = new List<UserType>();

        var usersPaginator = _cognitoService.Paginators.ListUsers(request);
        await foreach (var response in usersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Respond to an admin authentication challenge.
    /// </summary>
    /// <param name="userName">The name of the user.</param>
    /// <param name="clientId">The client ID.</param>
    /// <param name="mfaCode">The multi-factor authentication code.</param>
    /// <param name="session">The current application session.</param>
    /// <param name="clientId">The user pool ID.</param>
    /// <returns>The result of the authentication response.</returns>
    public async Task<AuthenticationResultType> AdminRespondToAuthChallengeAsync(
        string userName,
```

```
        string clientId,
        string mfaCode,
        string session,
        string userPoolId)
    {
        Console.WriteLine("SOFTWARE_TOKEN_MFA challenge is generated");

        var challengeResponses = new Dictionary<string, string>();
        challengeResponses.Add("USERNAME", userName);
        challengeResponses.Add("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        var respondToAuthChallengeRequest = new
AdminRespondToAuthChallengeRequest
        {
            ChallengeName = ChallengeNameType.SOFTWARE_TOKEN_MFA,
            ClientId = clientId,
            ChallengeResponses = challengeResponses,
            Session = session,
            UserPoolId = userPoolId,
        };

        var response = await
_cognitoService.AdminRespondToAuthChallengeAsync(respondToAuthChallengeRequest);
        Console.WriteLine($"Response to Authentication
{response.AuthenticationResult.TokenType}");
        return response.AuthenticationResult;
    }


    /// <summary>
    /// Verify the TOTP and register for MFA.
    /// </summary>
    /// <param name="session">The name of the session.</param>
    /// <param name="code">The MFA code.</param>
    /// <returns>The status of the software token.</returns>
    public async Task<VerifySoftwareTokenResponseType>
VerifySoftwareTokenAsync(string session, string code)
    {
        var tokenRequest = new VerifySoftwareTokenRequest
        {
            UserCode = code,
            Session = session,
        };
```

```
        var verifyResponse = await
_cognitoService.VerifySoftwareTokenAsync(tokenRequest);

        return verifyResponse.Status;
    }


    /// <summary>
    /// Get an MFA token to authenticate the user with the authenticator.
    /// </summary>
    /// <param name="session">The session name.</param>
    /// <returns>The session name.</returns>
    public async Task<string> AssociateSoftwareTokenAsync(string session)
    {
        var softwareTokenRequest = new AssociateSoftwareTokenRequest
        {
            Session = session,
        };

        var tokenResponse = await
_cognitoService.AssociateSoftwareTokenAsync(softwareTokenRequest);
        var secretCode = tokenResponse.SecretCode;

        Console.WriteLine($"Use the following secret code to set up the
authenticator: {secretCode}");

        return tokenResponse.Session;
    }


    /// <summary>
    /// Initiate an admin auth request.
    /// </summary>
    /// <param name="clientId">The client ID to use.</param>
    /// <param name="userPoolId">The ID of the user pool.</param>
    /// <param name="userName">The username to authenticate.</param>
    /// <param name="password">The user's password.</param>
    /// <returns>The session to use in challenge-response.</returns>
    public async Task<string> AdminInitiateAuthAsync(string clientId, string
userPoolId, string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);
```

```
        var request = new AdminInitiateAuthRequest
        {
            ClientId = clientId,
            UserPoolId = userPoolId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.AdminInitiateAuthAsync(request);
        return response.Session;
    }

    /// <summary>
    /// Initiate authorization.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The name of the user who is authenticating.</
param>
    /// <param name="password">The password for the user who is authenticating.</
param>
    /// <returns>The response from the initiate auth request.</returns>
    public async Task<InitiateAuthResponse> InitiateAuthAsync(string clientId,
 string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var authRequest = new InitiateAuthRequest

        {
            ClientId = clientId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.InitiateAuthAsync(authRequest);
        Console.WriteLine($"Result Challenge is : {response.ChallengeName}");

        return response;
    }

    /// <summary>
```

```
    /// Confirm that the user has signed up.
    /// </summary>
    /// <param name="clientId">The Id of this application.</param>
    /// <param name="code">The confirmation code sent to the user.</param>
    /// <param name="userName">The username.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ConfirmSignupAsync(string clientId, string code,
 string userName)
    {
        var signUpRequest = new ConfirmSignUpRequest
        {
            ClientId = clientId,
            ConfirmationCode = code,
            Username = userName,
        };

        var response = await _cognitoService.ConfirmSignUpAsync(signUpRequest);
        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            Console.WriteLine($"{userName} was confirmed");
            return true;
        }
        return false;
    }


    /// <summary>
    /// Initiates and confirms tracking of the device.
    /// </summary>
    /// <param name="accessToken">The user's access token.</param>
    /// <param name="deviceKey">The key of the device from Amazon Cognito.</
param>
    /// <param name="deviceName">The device name.</param>
    /// <returns></returns>
    public async Task<bool> ConfirmDeviceAsync(string accessToken, string
 deviceKey, string deviceName)
    {
        var request = new ConfirmDeviceRequest
        {
            AccessToken = accessToken,
            DeviceKey = deviceKey,
            DeviceName = deviceName
        };
```

```
        var response = await _cognitoService.ConfirmDeviceAsync(request);
        return response.UserConfirmationNecessary;
    }


    /// <summary>
    /// Send a new confirmation code to a user.
    /// </summary>
    /// <param name="clientId">The Id of the client application.</param>
    /// <param name="userName">The username of user who will receive the code.</
param>
    /// <returns>The delivery details.</returns>
    public async Task<CodeDeliveryDetailsType> ResendConfirmationCodeAsync(string
 clientId, string userName)
    {
        var codeRequest = new ResendConfirmationCodeRequest
        {
            ClientId = clientId,
            Username = userName,
        };

        var response = await
 _cognitoService.ResendConfirmationCodeAsync(codeRequest);

        Console.WriteLine($"Method of delivery is
 {response.CodeDeliveryDetails.DeliveryMedium}");

        return response.CodeDeliveryDetails;
    }


    /// <summary>
    /// Get the specified user from an Amazon Cognito user pool with
 administrator access.
    /// </summary>
    /// <param name="userName">The name of the user.</param>
    /// <param name="poolId">The Id of the Amazon Cognito user pool.</param>
    /// <returns>Async task.</returns>
    public async Task<UserStatusType> GetAdminUserAsync(string userName, string
 poolId)
    {
        AdminGetUserRequest userRequest = new AdminGetUserRequest
        {
            Username = userName,
```

```
            UserPoolId = poolId,
        };

        var response = await _cognitoService.AdminGetUserAsync(userRequest);

        Console.WriteLine($"User status {response.UserStatus}");
        return response.UserStatus;
    }


    /// <summary>
    /// Sign up a new user.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The username to use.</param>
    /// <param name="password">The user's password.</param>
    /// <param name="email">The email address of the user.</param>
    /// <returns>A Boolean value indicating whether the user was confirmed.</
returns>
    public async Task<bool> SignUpAsync(string clientId, string userName, string
 password, string email)
    {
        var userAttrs = new AttributeType
        {
            Name = "email",
            Value = email,
        };

        var userAttrsList = new List<AttributeType>();

        userAttrsList.Add(userAttrs);

        var signUpRequest = new SignUpRequest
        {
            UserAttributes = userAttrsList,
            Username = userName,
            ClientId = clientId,
            Password = password
        };

        var response = await _cognitoService.SignUpAsync(signUpRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp
  - VerifySoftwareToken

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

//! Scenario that adds a user to an Amazon Cognito user pool.
/*!
  \sa gettingStartedWithUserPools()
  \param clientID: Client ID associated with an Amazon Cognito user pool.
```

```
  \param userPoolID: An Amazon Cognito user pool ID.
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
 */
bool AwsDoc::Cognito::gettingStartedWithUserPools(const Aws::String &clientID,
                                                  const Aws::String &userPoolID,
                                                  const
 Aws::Client::ClientConfiguration &clientConfig) {
    printAsterisksLine();
    std::cout
            << "Welcome to the Amazon Cognito example scenario."
            << std::endl;
    printAsterisksLine();

    std::cout
            << "This scenario will add a user to an Amazon Cognito user pool."
            << std::endl;
    const Aws::String userName = askQuestion("Enter a new username: ");
    const Aws::String password = askQuestion("Enter a new password: ");
    const Aws::String email = askQuestion("Enter a valid email for the user: ");

    std::cout << "Signing up " << userName << std::endl;

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
 client(clientConfig);
    bool userExists = false;
    do {
        // 1. Add a user with a username, password, and email address.
        Aws::CognitoIdentityProvider::Model::SignUpRequest request;
        request.AddUserAttributes(
                Aws::CognitoIdentityProvider::Model::AttributeType().WithName(
                        "email").WithValue(email));
        request.SetUsername(userName);
        request.SetPassword(password);
        request.SetClientId(clientID);
        Aws::CognitoIdentityProvider::Model::SignUpOutcome outcome =
                client.SignUp(request);

        if (outcome.IsSuccess()) {
            std::cout << "The signup request for " << userName << " was
 successful."
                      << std::endl;
        }
        else if (outcome.GetError().GetErrorType() ==
```

```
Aws::CognitoIdentityProvider::CognitoIdentityProviderErrors::USERNAME_EXISTS) {
           std::cout
                   << "The username already exists. Please enter a different
username."
                   << std::endl;
           userExists = true;
       }
       else {
           std::cerr << "Error with CognitoIdentityProvider::SignUpRequest. "
                     << outcome.GetError().GetMessage()
                     << std::endl;
           return false;
       }
   } while (userExists);

   printAsterisksLine();
   std::cout << "Retrieving status of " << userName << " in the user pool."
             << std::endl;
   // 2. Confirm that the user was added to the user pool.
   if (!checkAdminUserStatus(userName, userPoolID, client)) {
       return false;
   }

   std::cout << "A confirmation code was sent to " << email << "." << std::endl;

   bool resend = askYesNoQuestion("Would you like to send a new code? (y/n) ");
   if (resend) {
       // Request a resend of the confirmation code to the email address.
(ResendConfirmationCode)
       Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeRequest
request;
       request.SetUsername(userName);
       request.SetClientId(clientID);

       Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeOutcome
outcome =
               client.ResendConfirmationCode(request);

       if (outcome.IsSuccess()) {
           std::cout
                   << "CognitoIdentityProvider::ResendConfirmationCode was
successful."
                   << std::endl;
```

```
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::ResendConfirmationCode. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    printAsterisksLine();

    {
        // 4. Send the confirmation code that's received in the email.
(ConfirmSignUp)
        const Aws::String confirmationCode = askQuestion(
                "Enter the confirmation code that was emailed: ");
        Aws::CognitoIdentityProvider::Model::ConfirmSignUpRequest request;
        request.SetClientId(clientID);
        request.SetConfirmationCode(confirmationCode);
        request.SetUsername(userName);

        Aws::CognitoIdentityProvider::Model::ConfirmSignUpOutcome outcome =
                client.ConfirmSignUp(request);

        if (outcome.IsSuccess()) {
            std::cout << "ConfirmSignup was Successful."
                      << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::ConfirmSignUp. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    std::cout << "Rechecking the status of " << userName << " in the user pool."
              << std::endl;
    if (!checkAdminUserStatus(userName, userPoolID, client)) {
        return false;
    }

    printAsterisksLine();
```

```
    std::cout << "Initiating authorization using the username and password."
              << std::endl;

    Aws::String session;
    // 5. Initiate authorization with username and password. (AdminInitiateAuth)
    if (!adminInitiateAuthorization(clientID, userPoolID,  userName, password,
 session, client)) {
        return false;
    }

    printAsterisksLine();

    std::cout
            << "Starting setup of time-based one-time password (TOTP) multi-
factor authentication (MFA)."
            << std::endl;

    {
        // 6. Request a setup key for one-time password (TOTP)
        //    multi-factor authentication (MFA). (AssociateSoftwareToken)
        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenRequest
 request;
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenOutcome
 outcome =
                client.AssociateSoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout
                    << "Enter this setup key into an authenticator app, for
 example Google Authenticator."
                    << std::endl;
            std::cout << "Setup key: " << outcome.GetResult().GetSecretCode()
                      << std::endl;
#ifdef USING_QR
            printAsterisksLine();
            std::cout << "\nOr scan the QR code in the file '" << QR_CODE_PATH <<
 "."
                      << std::endl;

            saveQRCode(std::string("otpauth://totp/") + userName + "?secret=" +
                       outcome.GetResult().GetSecretCode());
```

```
#endif // USING_QR
            session = outcome.GetResult().GetSession();
        }
        else {
            std::cerr << "Error with
 CognitoIdentityProvider::AssociateSoftwareToken. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }
    askQuestion("Type enter to continue...", alwaysTrueTest);

    printAsterisksLine();

    {
        Aws::String userCode = askQuestion(
                "Enter the 6 digit code displayed in the authenticator app: ");

        //  7. Send the MFA code copied from an authenticator app.
 (VerifySoftwareToken)
        Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenRequest request;
        request.SetUserCode(userCode);
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenOutcome outcome =
                client.VerifySoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout << "Verification of the code was successful."
                      << std::endl;
            session = outcome.GetResult().GetSession();
        }
        else {
            std::cerr << "Error with
 CognitoIdentityProvider::VerifySoftwareToken. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    printAsterisksLine();
    std::cout << "You have completed the MFA authentication setup." << std::endl;
```

```
   std::cout << "Now, sign in." << std::endl;

   // 8. Initiate authorization again with username and password.
(AdminInitiateAuth)
   if (!adminInitiateAuthorization(clientID, userPoolID, userName, password,
session, client)) {
       return false;
   }

   Aws::String accessToken;
   {
       Aws::String mfaCode = askQuestion(
               "Re-enter the 6 digit code displayed in the authenticator app:
");

       // 9. Send a new MFA code copied from an authenticator app.
(AdminRespondToAuthChallenge)
       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeRequest
request;
       request.AddChallengeResponses("USERNAME", userName);
       request.AddChallengeResponses("SOFTWARE_TOKEN_MFA_CODE", mfaCode);
       request.SetChallengeName(

Aws::CognitoIdentityProvider::Model::ChallengeNameType::SOFTWARE_TOKEN_MFA);
       request.SetClientId(clientID);
       request.SetUserPoolId(userPoolID);
       request.SetSession(session);

       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeOutcome
outcome =
               client.AdminRespondToAuthChallenge(request);

       if (outcome.IsSuccess()) {
           std::cout << "Here is the response to the challenge.\n" <<

outcome.GetResult().GetAuthenticationResult().Jsonize().View().WriteReadable()
                     << std::endl;

           accessToken =
outcome.GetResult().GetAuthenticationResult().GetAccessToken();
       }
       else {
           std::cerr << "Error with
CognitoIdentityProvider::AdminRespondToAuthChallenge. "
```

```
                    << outcome.GetError().GetMessage()
                    << std::endl;
            return false;
        }

        std::cout << "You have successfully added a user to Amazon Cognito."
                  << std::endl;
    }

    if (askYesNoQuestion("Would you like to delete the user that you just added?
 (y/n) ")) {
        // 10. Delete the user that you just added. (DeleteUser)
        Aws::CognitoIdentityProvider::Model::DeleteUserRequest request;
        request.SetAccessToken(accessToken);

        Aws::CognitoIdentityProvider::Model::DeleteUserOutcome outcome =
                client.DeleteUser(request);

        if (outcome.IsSuccess()) {
            std::cout << "The user " << userName << " was deleted."
                    << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::DeleteUser. "
                    << outcome.GetError().GetMessage()
                    << std::endl;
        }
    }

    return true;
}

//! Routine which checks the user status in an Amazon Cognito user pool.
/*!
 \sa checkAdminUserStatus()
 \param userName: A username.
 \param userPoolID: An Amazon Cognito user pool ID.
 \return bool: Successful completion.
 */
bool AwsDoc::Cognito::checkAdminUserStatus(const Aws::String &userName,
                                           const Aws::String &userPoolID,
                                           const
 Aws::CognitoIdentityProvider::CognitoIdentityProviderClient &client) {
    Aws::CognitoIdentityProvider::Model::AdminGetUserRequest request;
```

```
    request.SetUsername(userName);
    request.SetUserPoolId(userPoolID);

    Aws::CognitoIdentityProvider::Model::AdminGetUserOutcome outcome =
            client.AdminGetUser(request);

    if (outcome.IsSuccess()) {
        std::cout << "The status for " << userName << " is " <<

 Aws::CognitoIdentityProvider::Model::UserStatusTypeMapper::GetNameForUserStatusType(
                          outcome.GetResult().GetUserStatus()) << std::endl;
        std::cout << "Enabled is " << outcome.GetResult().GetEnabled() <<
 std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminGetUser. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which starts authorization of an Amazon Cognito user.
//! This routine requires administrator credentials.
/*!
 \sa adminInitiateAuthorization()
 \param clientID: Client ID of tracked device.
 \param userPoolID: An Amazon Cognito user pool ID.
 \param userName: A username.
 \param password: A password.
 \param sessionResult: String to receive a session token.
 \return bool: Successful completion.
 */
bool AwsDoc::Cognito::adminInitiateAuthorization(const Aws::String &clientID,
                                                 const Aws::String &userPoolID,
                                                 const Aws::String &userName,
                                                 const Aws::String &password,
                                                 Aws::String &sessionResult,
                                                 const
 Aws::CognitoIdentityProvider::CognitoIdentityProviderClient &client) {
    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthRequest request;
    request.SetClientId(clientID);
    request.SetUserPoolId(userPoolID);
```

```
    request.AddAuthParameters("USERNAME", userName);
    request.AddAuthParameters("PASSWORD", password);
    request.SetAuthFlow(

 Aws::CognitoIdentityProvider::Model::AuthFlowType::ADMIN_USER_PASSWORD_AUTH);


    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthOutcome outcome =
            client.AdminInitiateAuth(request);

    if (outcome.IsSuccess()) {
        std::cout << "Call to AdminInitiateAuth was successful." << std::endl;
        sessionResult = outcome.GetResult().GetSession();
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminInitiateAuth. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for C++ -API-Referenz.
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp
  - VerifySoftwareToken

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminGetUserRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminGetUserResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminInitiateAuthRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminInitiateAuthResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminRespondToAuthChalleng
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminRespondToAuthChalleng
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AssociateSoftwareTokenRequ
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AssociateSoftwareTokenResp
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AttributeType;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AuthFlowType;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ChallengeNameType;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ConfirmSignUpRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ResendConfirmationCodeRequ
```

```
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ResendConfirmationCodeResp
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.SignUpRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.VerifySoftwareTokenRequest
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.VerifySoftwareTokenRespons
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Scanner;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * TIP: To set up the required user pool, run the AWS Cloud Development Kit (AWS
 * CDK) script provided in this GitHub repo at
 * resources/cdk/cognito_scenario_user_pool_with_mfa.
 *
 * This code example performs the following operations:
 *
 * 1. Invokes the signUp method to sign up a user.
 * 2. Invokes the adminGetUser method to get the user's confirmation status.
 * 3. Invokes the ResendConfirmationCode method if the user requested another
 * code.
 * 4. Invokes the confirmSignUp method.
 * 5. Invokes the AdminInitiateAuth to sign in. This results in being prompted
 * to set up TOTP (time-based one-time password). (The response is
 * “ChallengeName”: “MFA_SETUP”).
 * 6. Invokes the AssociateSoftwareToken method to generate a TOTP MFA private
 * key. This can be used with Google Authenticator.
 * 7. Invokes the VerifySoftwareToken method to verify the TOTP and register for
 * MFA.
 * 8. Invokes the AdminInitiateAuth to sign in again. This results in being
```

```
 * prompted to submit a TOTP (Response: “ChallengeName”: “SOFTWARE_TOKEN_MFA”).
 * 9. Invokes the AdminRespondToAuthChallenge to get back a token.
 */

public class CognitoMVP {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws NoSuchAlgorithmException,
 InvalidKeyException {
        final String usage = """

                Usage:
                    <clientId> <poolId>

                Where:
                    clientId - The app client Id value that you can get from the
 AWS CDK script.
                    poolId - The pool Id that you can get from the AWS CDK
 script.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String clientId = args[0];
        String poolId = args[1];
        CognitoIdentityProviderClient identityProviderClient =
 CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon Cognito example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Enter your user name");
        Scanner in = new Scanner(System.in);
        String userName = in.nextLine();

        System.out.println("*** Enter your password");
```

```
        String password = in.nextLine();

        System.out.println("*** Enter your email");
        String email = in.nextLine();

        System.out.println("1. Signing up " + userName);
        signUp(identityProviderClient, clientId, userName, password, email);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Getting " + userName + " in the user pool");
        getAdminUser(identityProviderClient, userName, poolId);

        System.out
                .println("*** Conformation code sent to " + userName + ". Would
you like to send a new code? (Yes/No)");
        System.out.println(DASHES);

        System.out.println(DASHES);
        String ans = in.nextLine();

        if (ans.compareTo("Yes") == 0) {
            resendConfirmationCode(identityProviderClient, clientId, userName);
            System.out.println("3. Sending a new confirmation code");
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Enter confirmation code that was emailed");
        String code = in.nextLine();
        confirmSignUp(identityProviderClient, clientId, code, userName);
        System.out.println("Rechecking the status of " + userName + " in the user
pool");
        getAdminUser(identityProviderClient, userName, poolId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Invokes the initiateAuth to sign in");
        AdminInitiateAuthResponse authResponse =
initiateAuth(identityProviderClient, clientId, userName, password,
                poolId);
        String mySession = authResponse.session();
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("6. Invokes the AssociateSoftwareToken method to
generate a TOTP key");
        String newSession = getSecretForAppMFA(identityProviderClient,
mySession);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Enter the 6-digit code displayed in Google
Authenticator");
        String myCode = in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Verify the TOTP and register for MFA");
        verifyTOTP(identityProviderClient, newSession, myCode);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Re-enter a 6-digit code displayed in Google
Authenticator");
        String mfaCode = in.nextLine();
        AdminInitiateAuthResponse authResponse1 =
initiateAuth(identityProviderClient, clientId, userName, password,
                poolId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9.  Invokes the AdminRespondToAuthChallenge");
        String session2 = authResponse1.session();
        adminRespondToAuthChallenge(identityProviderClient, userName, clientId,
mfaCode, session2);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("All Amazon Cognito operations were successfully
performed");
        System.out.println(DASHES);
    }

    // Respond to an authentication challenge.
    public static void adminRespondToAuthChallenge(CognitoIdentityProviderClient
identityProviderClient,
            String userName, String clientId, String mfaCode, String session) {
```

```
        System.out.println("SOFTWARE_TOKEN_MFA challenge is generated");
        Map<String, String> challengeResponses = new HashMap<>();

        challengeResponses.put("USERNAME", userName);
        challengeResponses.put("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        AdminRespondToAuthChallengeRequest respondToAuthChallengeRequest =
AdminRespondToAuthChallengeRequest.builder()
                .challengeName(ChallengeNameType.SOFTWARE_TOKEN_MFA)
                .clientId(clientId)
                .challengeResponses(challengeResponses)
                .session(session)
                .build();

        AdminRespondToAuthChallengeResponse respondToAuthChallengeResult =
identityProviderClient
                .adminRespondToAuthChallenge(respondToAuthChallengeRequest);

System.out.println("respondToAuthChallengeResult.getAuthenticationResult()"
                + respondToAuthChallengeResult.authenticationResult());
    }

    // Verify the TOTP and register for MFA.
    public static void verifyTOTP(CognitoIdentityProviderClient
identityProviderClient, String session, String code) {
        try {
            VerifySoftwareTokenRequest tokenRequest =
VerifySoftwareTokenRequest.builder()
                    .userCode(code)
                    .session(session)
                    .build();

            VerifySoftwareTokenResponse verifyResponse =
identityProviderClient.verifySoftwareToken(tokenRequest);
            System.out.println("The status of the token is " +
verifyResponse.statusAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

```
   public static AdminInitiateAuthResponse
initiateAuth(CognitoIdentityProviderClient identityProviderClient,
           String clientId, String userName, String password, String userPoolId)
{
       try {
           Map<String, String> authParameters = new HashMap<>();
           authParameters.put("USERNAME", userName);
           authParameters.put("PASSWORD", password);

           AdminInitiateAuthRequest authRequest =
AdminInitiateAuthRequest.builder()
                   .clientId(clientId)
                   .userPoolId(userPoolId)
                   .authParameters(authParameters)
                   .authFlow(AuthFlowType.ADMIN_USER_PASSWORD_AUTH)
                   .build();

           AdminInitiateAuthResponse response =
identityProviderClient.adminInitiateAuth(authRequest);
           System.out.println("Result Challenge is : " +
response.challengeName());
           return response;

       } catch (CognitoIdentityProviderException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }

       return null;
   }

   public static String getSecretForAppMFA(CognitoIdentityProviderClient
identityProviderClient, String session) {
       AssociateSoftwareTokenRequest softwareTokenRequest =
AssociateSoftwareTokenRequest.builder()
               .session(session)
               .build();

       AssociateSoftwareTokenResponse tokenResponse = identityProviderClient
               .associateSoftwareToken(softwareTokenRequest);
       String secretCode = tokenResponse.secretCode();
       System.out.println("Enter this token into Google Authenticator");
       System.out.println(secretCode);
       return tokenResponse.session();
```

```
    }

    public static void confirmSignUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String code,
            String userName) {
        try {
            ConfirmSignUpRequest signUpRequest = ConfirmSignUpRequest.builder()
                    .clientId(clientId)
                    .confirmationCode(code)
                    .username(userName)
                    .build();

            identityProviderClient.confirmSignUp(signUpRequest);
            System.out.println(userName + " was confirmed");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void resendConfirmationCode(CognitoIdentityProviderClient
identityProviderClient, String clientId,
            String userName) {
        try {
            ResendConfirmationCodeRequest codeRequest =
ResendConfirmationCodeRequest.builder()
                    .clientId(clientId)
                    .username(userName)
                    .build();

            ResendConfirmationCodeResponse response =
identityProviderClient.resendConfirmationCode(codeRequest);
            System.out.println("Method of delivery is " +
response.codeDeliveryDetails().deliveryMediumAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void signUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String userName,
```

```
            String password, String email) {
        AttributeType userAttrs = AttributeType.builder()
                .name("email")
                .value(email)
                .build();

        List<AttributeType> userAttrsList = new ArrayList<>();
        userAttrsList.add(userAttrs);
        try {
            SignUpRequest signUpRequest = SignUpRequest.builder()
                    .userAttributes(userAttrsList)
                    .username(userName)
                    .clientId(clientId)
                    .password(password)
                    .build();

            identityProviderClient.signUp(signUpRequest);
            System.out.println("User has been signed up ");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getAdminUser(CognitoIdentityProviderClient
 identityProviderClient, String userName,
            String poolId) {
        try {
            AdminGetUserRequest userRequest = AdminGetUserRequest.builder()
                    .username(userName)
                    .userPoolId(poolId)
                    .build();

            AdminGetUserResponse response =
 identityProviderClient.adminGetUser(userRequest);
            System.out.println("User status " + response.userStatusAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp
  - VerifySoftwareToken

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Um die beste Erfahrung zu erzielen, klonen Sie das GitHub Repository und führen Sie dieses Beispiel aus. Der folgende Code ist ein Teil der vollständigen Beispielanwendung.

```
import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { signUp } from "../../../actions/sign-up.js";
import { FILE_USER_POOLS } from "./constants.js";
import { getSecondValuesFromEntries } from "@aws-doc-sdk-examples/lib/utils/util-
csv.js";

const validateClient = (clientId) => {
```

```
  if (!clientId) {
    throw new Error(
      `App client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateUser = (username, password, email) => {
  if (!(username && password && email)) {
    throw new Error(
      `Username, password, and email must be provided as arguments to the 'sign-
up' command.`,
    );
  }
};

const signUpHandler = async (commands) => {
  const [_, username, password, email] = commands;

  try {
    validateUser(username, password, email);
    /**
     * @type {string[]}
     */
    const values = getSecondValuesFromEntries(FILE_USER_POOLS);
    const clientId = values[0];
    validateClient(clientId);
    log(`Signing up.`);
    await signUp({ clientId, username, password, email });
    log(`Signed up. A confirmation email has been sent to: ${email}.`);
    log(`Run 'confirm-sign-up ${username} <code>' to confirm your account.`);
  } catch (err) {
    log(err);
  }
};

export { signUpHandler };

const signUp = ({ clientId, username, password, email }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new SignUpCommand({
    ClientId: clientId,
    Username: username,
```

```
    Password: password,
    UserAttributes: [{ Name: "email", Value: email }],
  });

  return client.send(command);
};

import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { confirmSignUp } from "../../../actions/confirm-sign-up.js";
import { FILE_USER_POOLS } from "./constants.js";
import { getSecondValuesFromEntries } from "@aws-doc-sdk-examples/lib/utils/util-
csv.js";

const validateClient = (clientId) => {
  if (!clientId) {
    throw new Error(
      `App client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateUser = (username) => {
  if (!username) {
    throw new Error(
      `Username name is missing. It must be provided as an argument to the
 'confirm-sign-up' command.`,
    );
  }
};

const validateCode = (code) => {
  if (!code) {
    throw new Error(
      `Verification code is missing. It must be provided as an argument to the
 'confirm-sign-up' command.`,
    );
  }
};

const confirmSignUpHandler = async (commands) => {
  const [_, username, code] = commands;

  try {
    validateUser(username);
```

```
    validateCode(code);
    /**
     * @type {string[]}
     */
    const values = getSecondValuesFromEntries(FILE_USER_POOLS);
    const clientId = values[0];
    validateClient(clientId);
    log(`Confirming user.`);
    await confirmSignUp({ clientId, username, code });
    log(
      `User confirmed. Run 'admin-initiate-auth ${username} <password>' to sign
 in.`,
    );
  } catch (err) {
    log(err);
  }
};

export { confirmSignUpHandler };

const confirmSignUp = ({ clientId, username, code }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmSignUpCommand({
    ClientId: clientId,
    Username: username,
    ConfirmationCode: code,
  });

  return client.send(command);
};

import qrcode from "qrcode-terminal";
import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { adminInitiateAuth } from "../../../actions/admin-initiate-auth.js";
import { associateSoftwareToken } from "../../../actions/associate-software-
token.js";
import { FILE_USER_POOLS } from "./constants.js";
import { getFirstEntry } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";

const handleMfaSetup = async (session, username) => {
  const { SecretCode, Session } = await associateSoftwareToken(session);

  // Store the Session for use with 'VerifySoftwareToken'.
```

```
  process.env.SESSION = Session;

  console.log(
    "Scan this code in your preferred authenticator app, then run 'verify-
software-token' to finish the setup.",
  );
  qrcode.generate(
    `otpauth://totp/${username}?secret=${SecretCode}`,
    { small: true },
    console.log,
  );
};

const handleSoftwareTokenMfa = (session) => {
  // Store the Session for use with 'AdminRespondToAuthChallenge'.
  process.env.SESSION = session;
};

const validateClient = (id) => {
  if (!id) {
    throw new Error(
      `User pool client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateId = (id) => {
  if (!id) {
    throw new Error(`User pool id is missing. Did you run 'create-user-pool'?`);
  }
};

const validateUser = (username, password) => {
  if (!(username && password)) {
    throw new Error(
      `Username and password must be provided as arguments to the 'admin-
initiate-auth' command.`,
    );
  }
};

const adminInitiateAuthHandler = async (commands) => {
  const [_, username, password] = commands;
```

```
  try {
    validateUser(username, password);

    const [userPoolId, clientId] = getFirstEntry(FILE_USER_POOLS);
    validateId(userPoolId);
    validateClient(clientId);

    log("Signing in.");
    const { ChallengeName, Session } = await adminInitiateAuth({
      clientId,
      userPoolId,
      username,
      password,
    });

    if (ChallengeName === "MFA_SETUP") {
      log("MFA setup is required.");
      return handleMfaSetup(Session, username);
    }

    if (ChallengeName === "SOFTWARE_TOKEN_MFA") {
      handleSoftwareTokenMfa(Session);
      log(`Run 'admin-respond-to-auth-challenge ${username} <totp>'`);
    }
  } catch (err) {
    log(err);
  }
};

export { adminInitiateAuthHandler };

const adminInitiateAuth = ({ clientId, userPoolId, username, password }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminInitiateAuthCommand({
    ClientId: clientId,
    UserPoolId: userPoolId,
    AuthFlow: AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
    AuthParameters: { USERNAME: username, PASSWORD: password },
  });

  return client.send(command);
};
```

```
import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { adminRespondToAuthChallenge } from "../../../actions/admin-respond-to-
auth-challenge.js";
import { getFirstEntry } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";
import { FILE_USER_POOLS } from "./constants.js";

const verifyUsername = (username) => {
  if (!username) {
    throw new Error(
      `Username is missing. It must be provided as an argument to the 'admin-
respond-to-auth-challenge' command.`,
    );
  }
};

const verifyTotp = (totp) => {
  if (!totp) {
    throw new Error(
      `Time-based one-time password (TOTP) is missing. It must be provided as an
 argument to the 'admin-respond-to-auth-challenge' command.`,
    );
  }
};

const storeAccessToken = (token) => {
  process.env.AccessToken = token;
};

const adminRespondToAuthChallengeHandler = async (commands) => {
  const [_, username, totp] = commands;

  try {
    verifyUsername(username);
    verifyTotp(totp);

    const [userPoolId, clientId] = getFirstEntry(FILE_USER_POOLS);
    const session = process.env.SESSION;

    const { AuthenticationResult } = await adminRespondToAuthChallenge({
      clientId,
      userPoolId,
      username,
      totp,
      session,
```

```
    });

    storeAccessToken(AuthenticationResult.AccessToken);

    log("Successfully authenticated.");
  } catch (err) {
    log(err);
  }
};

export { adminRespondToAuthChallengeHandler };

const respondToAuthChallenge = ({
  clientId,
  username,
  session,
  userPoolId,
  code,
}) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new RespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: code,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};

import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { verifySoftwareToken } from "../../../actions/verify-software-token.js";

const validateTotp = (totp) => {
  if (!totp) {
    throw new Error(
      `Time-based one-time password (TOTP) must be provided to the 'validate-
software-token' command.`,
    );
```

```
  }
};
const verifySoftwareTokenHandler = async (commands) => {
  const [_, totp] = commands;

  try {
    validateTotp(totp);

    log("Verifying TOTP.");
    await verifySoftwareToken(totp);
    log("TOTP Verified. Run 'admin-initiate-auth' again to sign-in.");
  } catch (err) {
    console.log(err);
  }
};

export { verifySoftwareTokenHandler };

const verifySoftwareToken = (totp) => {
  const client = new CognitoIdentityProviderClient({});

  // The 'Session' is provided in the response to 'AssociateSoftwareToken'.
  const session = process.env.SESSION;

  if (!session) {
    throw new Error(
      "Missing a valid Session. Did you run 'admin-initiate-auth'?",
    );
  }

  const command = new VerifySoftwareTokenCommand({
    Session: session,
    UserCode: totp,
  });

  return client.send(command);
};
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for JavaScript -API-Referenz.
  - AdminGetUser

- AdminInitiateAuth
- AdminRespondToAuthChallenge
- AssociateSoftwareToken
- ConfirmDevice
- ConfirmSignUp
- InitiateAuth
- ListUsers
- ResendConfirmationCode
- RespondToAuthChallenge
- SignUp
- VerifySoftwareToken

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/**
 Before running this Kotlin code example, set up your development environment,
 including your credentials.

 For more information, see the following documentation:
 https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html

 TIP: To set up the required user pool, run the AWS Cloud Development
 Kit (AWS CDK) script provided in this GitHub repo at resources/cdk/
cognito_scenario_user_pool_with_mfa.

 This code example performs the following operations:

 1. Invokes the signUp method to sign up a user.
 2. Invokes the adminGetUser method to get the user's confirmation status.
```

```
 3. Invokes the ResendConfirmationCode method if the user requested another code.
 4. Invokes the confirmSignUp method.
 5. Invokes the initiateAuth to sign in. This results in being prompted to
 set up TOTP (time-based one-time password). (The response is “ChallengeName”:
 “MFA_SETUP”).
 6. Invokes the AssociateSoftwareToken method to generate a TOTP MFA private key.
 This can be used with Google Authenticator.
 7. Invokes the VerifySoftwareToken method to verify the TOTP and register for
 MFA.
 8. Invokes the AdminInitiateAuth to sign in again. This results in being
 prompted to submit a TOTP (Response: “ChallengeName”: “SOFTWARE_TOKEN_MFA”).
 9. Invokes the AdminRespondToAuthChallenge to get back a token.
 */

suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <clientId> <poolId>
        Where:
            clientId - The app client Id value that you can get from the AWS CDK
 script.
            poolId - The pool Id that you can get from the AWS CDK script.
    """

    if (args.size != 2) {
        println(usage)
        exitProcess(1)
    }

    val clientId = args[0]
    val poolId = args[1]

    // Use the console to get data from the user.
    println("*** Enter your use name")
    val inOb = Scanner(System.`in`)
    val userName = inOb.nextLine()
    println(userName)

    println("*** Enter your password")
    val password: String = inOb.nextLine()

    println("*** Enter your email")
    val email = inOb.nextLine()
```

```
    println("*** Signing up $userName")
    signUp(clientId, userName, password, email)

    println("*** Getting $userName in the user pool")
    getAdminUser(userName, poolId)

    println("*** Conformation code sent to $userName. Would you like to send a
 new code? (Yes/No)")
    val ans = inOb.nextLine()

    if (ans.compareTo("Yes") == 0) {
        println("*** Sending a new confirmation code")
        resendConfirmationCode(clientId, userName)
    }
    println("*** Enter the confirmation code that was emailed")
    val code = inOb.nextLine()
    confirmSignUp(clientId, code, userName)

    println("*** Rechecking the status of $userName in the user pool")
    getAdminUser(userName, poolId)

    val authResponse = checkAuthMethod(clientId, userName, password, poolId)
    val mySession = authResponse.session
    val newSession = getSecretForAppMFA(mySession)
    println("*** Enter the 6-digit code displayed in Google Authenticator")
    val myCode = inOb.nextLine()

    // Verify the TOTP and register for MFA.
    verifyTOTP(newSession, myCode)
    println("*** Re-enter a 6-digit code displayed in Google Authenticator")
    val mfaCode: String = inOb.nextLine()
    val authResponse1 = checkAuthMethod(clientId, userName, password, poolId)
    val session2 = authResponse1.session
    adminRespondToAuthChallenge(userName, clientId, mfaCode, session2)
}

suspend fun checkAuthMethod(clientIdVal: String, userNameVal: String,
 passwordVal: String, userPoolIdVal: String): AdminInitiateAuthResponse {
    val authParas = mutableMapOf<String, String>()
    authParas["USERNAME"] = userNameVal
    authParas["PASSWORD"] = passwordVal

    val authRequest = AdminInitiateAuthRequest {
        clientId = clientIdVal
```

```
        userPoolId = userPoolIdVal
        authParameters = authParas
        authFlow = AuthFlowType.AdminUserPasswordAuth
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminInitiateAuth(authRequest)
        println("Result Challenge is ${response.challengeName}")
        return response
    }
}

suspend fun resendConfirmationCode(clientIdVal: String?, userNameVal: String?) {
    val codeRequest = ResendConfirmationCodeRequest {
        clientId = clientIdVal
        username = userNameVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.resendConfirmationCode(codeRequest)
        println("Method of delivery is " +
 (response.codeDeliveryDetails?.deliveryMedium))
    }
}

// Respond to an authentication challenge.
suspend fun adminRespondToAuthChallenge(userName: String, clientIdVal: String?,
 mfaCode: String, sessionVal: String?) {
    println("SOFTWARE_TOKEN_MFA challenge is generated")
    val challengeResponsesOb = mutableMapOf<String, String>()
    challengeResponsesOb["USERNAME"] = userName
    challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

    val adminRespondToAuthChallengeRequest = AdminRespondToAuthChallengeRequest {
        challengeName = ChallengeNameType.SoftwareTokenMfa
        clientId = clientIdVal
        challengeResponses = challengeResponsesOb
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
```

```
        val respondToAuthChallengeResult =
 identityProviderClient.adminRespondToAuthChallenge(adminRespondToAuthChallengeRequest)
        println("respondToAuthChallengeResult.getAuthenticationResult()
 ${respondToAuthChallengeResult.authenticationResult}")
    }
}

// Verify the TOTP and register for MFA.
suspend fun verifyTOTP(sessionVal: String?, codeVal: String?) {
    val tokenRequest = VerifySoftwareTokenRequest {
        userCode = codeVal
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val verifyResponse =
 identityProviderClient.verifySoftwareToken(tokenRequest)
        println("The status of the token is ${verifyResponse.status}")
    }
}

suspend fun getSecretForAppMFA(sessionVal: String?): String? {
    val softwareTokenRequest = AssociateSoftwareTokenRequest {
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val tokenResponse =
 identityProviderClient.associateSoftwareToken(softwareTokenRequest)
        val secretCode = tokenResponse.secretCode
        println("Enter this token into Google Authenticator")
        println(secretCode)
        return tokenResponse.session
    }
}

suspend fun confirmSignUp(clientIdVal: String?, codeVal: String?, userNameVal:
 String?) {
    val signUpRequest = ConfirmSignUpRequest {
        clientId = clientIdVal
        confirmationCode = codeVal
        username = userNameVal
```

```
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.confirmSignUp(signUpRequest)
        println("$userNameVal  was confirmed")
    }
}

suspend fun getAdminUser(userNameVal: String?, poolIdVal: String?) {
    val userRequest = AdminGetUserRequest {
        username = userNameVal
        userPoolId = poolIdVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminGetUser(userRequest)
        println("User status ${response.userStatus}")
    }
}

suspend fun signUp(clientIdVal: String?, userNameVal: String?, passwordVal:
 String?, emailVal: String?) {
    val userAttrs = AttributeType {
        name = "email"
        value = emailVal
    }

    val userAttrsList = mutableListOf<AttributeType>()
    userAttrsList.add(userAttrs)
    val signUpRequest = SignUpRequest {
        userAttributes = userAttrsList
        username = userNameVal
        clientId = clientIdVal
        password = passwordVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.signUp(signUpRequest)
        println("User has been signed up")
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Kotlin.
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp
  - VerifySoftwareToken

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie eine Klasse, die Amazon-Cognito-Funktionen einschließt, die im Szenario verwendet werden.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
```

```
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def _secret_hash(self, user_name):
        """
        Calculates a secret hash from a user name and a client secret.

        :param user_name: The user name to use when calculating the hash.
        :return: The secret hash.
        """
        key = self.client_secret.encode()
        msg = bytes(user_name + self.client_id, "utf-8")
        secret_hash = base64.b64encode(
            hmac.new(key, msg, digestmod=hashlib.sha256).digest()
        ).decode()
        logger.info("Made secret hash for %s: %s.", user_name, secret_hash)
        return secret_hash

    def sign_up_user(self, user_name, password, user_email):
        """
        Signs up a new user with Amazon Cognito. This action prompts Amazon
Cognito
        to send an email to the specified email address. The email contains a
code that
        can be used to confirm the user.

        When the user already exists, the user status is checked to determine
whether
        the user has been confirmed.

        :param user_name: The user name that identifies the new user.
        :param password: The password for the new user.
        :param user_email: The email address for the new user.
        :return: True when the user is already confirmed with Amazon Cognito.
```

```
                 Otherwise, false.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "Password": password,
                "UserAttributes": [{"Name": "email", "Value": user_email}],
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.sign_up(**kwargs)
            confirmed = response["UserConfirmed"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "UsernameExistsException":
                response = self.cognito_idp_client.admin_get_user(
                    UserPoolId=self.user_pool_id, Username=user_name
                )
                logger.warning(
                    "User %s exists and is %s.", user_name,
response["UserStatus"]
                )
                confirmed = response["UserStatus"] == "CONFIRMED"
            else:
                logger.error(
                    "Couldn't sign up %s. Here's why: %s: %s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return confirmed


    def resend_confirmation(self, user_name):
        """
        Prompts Amazon Cognito to resend an email with a new confirmation code.

        :param user_name: The name of the user who will receive the email.
        :return: Delivery information about where the email is sent.
        """
        try:
            kwargs = {"ClientId": self.client_id, "Username": user_name}
            if self.client_secret is not None:
```

```
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.resend_confirmation_code(**kwargs)
            delivery = response["CodeDeliveryDetails"]
        except ClientError as err:
            logger.error(
                "Couldn't resend confirmation to %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return delivery


    def confirm_user_sign_up(self, user_name, confirmation_code):
        """
        Confirms a previously created user. A user must be confirmed before they
        can sign in to Amazon Cognito.

        :param user_name: The name of the user to confirm.
        :param confirmation_code: The confirmation code sent to the user's
registered
                                  email address.
        :return: True when the confirmation succeeds.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "ConfirmationCode": confirmation_code,
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            self.cognito_idp_client.confirm_sign_up(**kwargs)
        except ClientError as err:
            logger.error(
                "Couldn't confirm sign up for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
```

```
            return True


    def list_users(self):
        """
        Returns a list of the users in the current user pool.

        :return: The list of users.
        """
        try:
            response =
 self.cognito_idp_client.list_users(UserPoolId=self.user_pool_id)
            users = response["Users"]
        except ClientError as err:
            logger.error(
                "Couldn't list users for %s. Here's why: %s: %s",
                self.user_pool_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return users


    def start_sign_in(self, user_name, password):
        """
        Starts the sign-in process for a user by using administrator credentials.
        This method of signing in is appropriate for code running on a secure
 server.

        If the user pool is configured to require MFA and this is the first sign-
in
        for the user, Amazon Cognito returns a challenge response to set up an
        MFA application. When this occurs, this function gets an MFA secret from
        Amazon Cognito and returns it to the caller.

        :param user_name: The name of the user to sign in.
        :param password: The user's password.
        :return: The result of the sign-in attempt. When sign-in is successful,
 this
                 returns an access token that can be used to get AWS credentials.
 Otherwise,
                 Amazon Cognito returns a challenge to set up an MFA application,
```

```
                or a challenge to enter an MFA code from a registered MFA
application.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
                "AuthParameters": {"USERNAME": user_name, "PASSWORD": password},
            }
            if self.client_secret is not None:
                kwargs["AuthParameters"]["SECRET_HASH"] =
self._secret_hash(user_name)
            response = self.cognito_idp_client.admin_initiate_auth(**kwargs)
            challenge_name = response.get("ChallengeName", None)
            if challenge_name == "MFA_SETUP":
                if (
                    "SOFTWARE_TOKEN_MFA"
                    in response["ChallengeParameters"]["MFAS_CAN_SETUP"]
                ):
                    response.update(self.get_mfa_secret(response["Session"]))
                else:
                    raise RuntimeError(
                        "The user pool requires MFA setup, but the user pool is
not "
                        "configured for TOTP MFA. This example requires TOTP
MFA."
                    )
        except ClientError as err:
            logger.error(
                "Couldn't start sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response


    def get_mfa_secret(self, session):
        """
```

```
        Gets a token that can be used to associate an MFA application with the
user.

        :param session: Session information returned from a previous call to
initiate
                        authentication.
        :return: An MFA token that can be used to set up an MFA application.
        """
        try:
            response =
self.cognito_idp_client.associate_software_token(Session=session)
        except ClientError as err:
            logger.error(
                "Couldn't get MFA secret. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response


    def verify_mfa(self, session, user_code):
        """
        Verify a new MFA application that is associated with a user.

        :param session: Session information returned from a previous call to
initiate
                        authentication.
        :param user_code: A code generated by the associated MFA application.
        :return: Status that indicates whether the MFA application is verified.
        """
        try:
            response = self.cognito_idp_client.verify_software_token(
                Session=session, UserCode=user_code
            )
        except ClientError as err:
            logger.error(
                "Couldn't verify MFA. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
        else:
            response.pop("ResponseMetadata", None)
            return response


    def respond_to_mfa_challenge(self, user_name, session, mfa_code):
        """
        Responds to a challenge for an MFA code. This completes the second step
of
        a two-factor sign-in. When sign-in is successful, it returns an access
token
        that can be used to get AWS credentials from Amazon Cognito.

        :param user_name: The name of the user who is signing in.
        :param session: Session information returned from a previous call to
initiate
                        authentication.
        :param mfa_code: A code generated by the associated MFA application.
        :return: The result of the authentication. When successful, this contains
an
                 access token for the user.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "ChallengeName": "SOFTWARE_TOKEN_MFA",
                "Session": session,
                "ChallengeResponses": {
                    "USERNAME": user_name,
                    "SOFTWARE_TOKEN_MFA_CODE": mfa_code,
                },
            }
            if self.client_secret is not None:
                kwargs["ChallengeResponses"]["SECRET_HASH"] = self._secret_hash(
                    user_name
                )
            response =
self.cognito_idp_client.admin_respond_to_auth_challenge(**kwargs)
            auth_result = response["AuthenticationResult"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "ExpiredCodeException":
                logger.warning(
```

```
                    "Your MFA code has expired or has been used already. You
might have "
                    "to wait a few seconds until your app shows you a new code."
                )
            else:
                logger.error(
                    "Couldn't respond to mfa challenge for %s. Here's why: %s:
%s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return auth_result


    def confirm_mfa_device(
        self,
        user_name,
        device_key,
        device_group_key,
        device_password,
        access_token,
        aws_srp,
    ):
        """
        Confirms an MFA device to be tracked by Amazon Cognito. When a device is
        tracked, its key and password can be used to sign in without requiring a
new
        MFA code from the MFA application.

        :param user_name: The user that is associated with the device.
        :param device_key: The key of the device, returned by Amazon Cognito.
        :param device_group_key: The group key of the device, returned by Amazon
Cognito.
        :param device_password: The password that is associated with the device.
        :param access_token: The user's access token.
        :param aws_srp: A class that helps with Secure Remote Password (SRP)
                        calculations. The scenario associated with this example
uses
                        the warrant package.
        :return: True when the user must confirm the device. Otherwise, False.
When
```

```
                False, the device is automatically confirmed and tracked.
        """
        srp_helper = aws_srp.AWSSRP(
            username=user_name,
            password=device_password,
            pool_id="_",
            client_id=self.client_id,
            client_secret=None,
            client=self.cognito_idp_client,
        )
        device_and_pw = f"{device_group_key}{device_key}:{device_password}"
        device_and_pw_hash = aws_srp.hash_sha256(device_and_pw.encode("utf-8"))
        salt = aws_srp.pad_hex(aws_srp.get_random(16))
        x_value = aws_srp.hex_to_long(aws_srp.hex_hash(salt +
device_and_pw_hash))
        verifier = aws_srp.pad_hex(pow(srp_helper.val_g, x_value,
srp_helper.big_n))
        device_secret_verifier_config = {
            "PasswordVerifier": base64.standard_b64encode(
                bytearray.fromhex(verifier)
            ).decode("utf-8"),
            "Salt":
base64.standard_b64encode(bytearray.fromhex(salt)).decode("utf-8"),
        }
        try:
            response = self.cognito_idp_client.confirm_device(
                AccessToken=access_token,
                DeviceKey=device_key,
                DeviceSecretVerifierConfig=device_secret_verifier_config,
            )
            user_confirm = response["UserConfirmationNecessary"]
        except ClientError as err:
            logger.error(
                "Couldn't confirm mfa device %s. Here's why: %s: %s",
                device_key,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return user_confirm


    def sign_in_with_tracked_device(
```

```
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
        """
        Signs in to Amazon Cognito as a user who has a tracked device. Signing in
        with a tracked device lets a user sign in without entering a new MFA
 code.

        Signing in with a tracked device requires that the client respond to the
 SRP
        protocol. The scenario associated with this example uses the warrant
 package
        to help with SRP calculations.

        For more information on SRP, see https://en.wikipedia.org/wiki/
Secure_Remote_Password_protocol.

        :param user_name: The user that is associated with the device.
        :param password: The user's password.
        :param device_key: The key of a tracked device.
        :param device_group_key: The group key of a tracked device.
        :param device_password: The password that is associated with the device.
        :param aws_srp: A class that helps with SRP calculations. The scenario
                        associated with this example uses the warrant package.
        :return: The result of the authentication. When successful, this contains
 an
                 access token for the user.
        """
        try:
            srp_helper = aws_srp.AWSSRP(
                username=user_name,
                password=device_password,
                pool_id="_",
                client_id=self.client_id,
                client_secret=None,
                client=self.cognito_idp_client,
            )

            response_init = self.cognito_idp_client.initiate_auth(
```

```
                ClientId=self.client_id,
                AuthFlow="USER_PASSWORD_AUTH",
                AuthParameters={
                    "USERNAME": user_name,
                    "PASSWORD": password,
                    "DEVICE_KEY": device_key,
                },
            )
            if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
                raise RuntimeError(
                    f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']}."
                )

            auth_params = srp_helper.get_auth_params()
            auth_params["DEVICE_KEY"] = device_key
            response_auth = self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_SRP_AUTH",
                ChallengeResponses=auth_params,
            )
            if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
                raise RuntimeError(
                    f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
                    f"{response_init['ChallengeName']}."
                )

            challenge_params = response_auth["ChallengeParameters"]
            challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
            cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
            cr["USERNAME"] = user_name
            cr["DEVICE_KEY"] = device_key
            response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_PASSWORD_VERIFIER",
                ChallengeResponses=cr,
            )
            auth_tokens = response_verifier["AuthenticationResult"]
        except ClientError as err:
            logger.error(
                "Couldn't start client sign in for %s. Here's why: %s: %s",
                user_name,
```

```
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return auth_tokens
```

Erstellen Sie eine Klasse, die das Szenario ausführt. Dieses Beispiel registriert auch ein MFA-Gerät für die Nachverfolgung durch Amazon Cognito und zeigt Ihnen, wie Sie sich mithilfe eines Passworts und der Informationen des nachverfolgten Geräts anmelden. Dadurch wird die Eingabe eines neuen MFA-Codes unnötig.

```
def run_scenario(cognito_idp_client, user_pool_id, client_id):
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Welcome to the Amazon Cognito user signup with MFA demo.")
    print("-" * 88)

    cog_wrapper = CognitoIdentityProviderWrapper(
        cognito_idp_client, user_pool_id, client_id
    )

    user_name = q.ask("Let's sign up a new user. Enter a user name: ",
 q.non_empty)
    password = q.ask("Enter a password for the user: ", q.non_empty)
    email = q.ask("Enter a valid email address that you own: ", q.non_empty)
    confirmed = cog_wrapper.sign_up_user(user_name, password, email)
    while not confirmed:
        print(
            f"User {user_name} requires confirmation. Check {email} for "
            f"a verification code."
        )
        confirmation_code = q.ask("Enter the confirmation code from the email: ")
        if not confirmation_code:
            if q.ask("Do you need another confirmation code (y/n)? ",
 q.is_yesno):
                delivery = cog_wrapper.resend_confirmation(user_name)
                print(
```

```
                    f"Confirmation code sent by {delivery['DeliveryMedium']} "
                    f"to {delivery['Destination']}."
                )
        else:
            confirmed = cog_wrapper.confirm_user_sign_up(user_name,
confirmation_code)
    print(f"User {user_name} is confirmed and ready to use.")
    print("-" * 88)

    print("Let's get a list of users in the user pool.")
    q.ask("Press Enter when you're ready.")
    users = cog_wrapper.list_users()
    if users:
        print(f"Found {len(users)} users:")
        pp(users)
    else:
        print("No users found.")
    print("-" * 88)

    print("Let's sign in and get an access token.")
    auth_tokens = None
    challenge = "ADMIN_USER_PASSWORD_AUTH"
    response = {}
    while challenge is not None:
        if challenge == "ADMIN_USER_PASSWORD_AUTH":
            response = cog_wrapper.start_sign_in(user_name, password)
            challenge = response["ChallengeName"]
        elif response["ChallengeName"] == "MFA_SETUP":
            print("First, we need to set up an MFA application.")
            qr_img = qrcode.make(
                f"otpauth://totp/{user_name}?secret={response['SecretCode']}"
            )
            qr_img.save("qr.png")
            q.ask(
                "Press Enter to see a QR code on your screen. Scan it into an MFA
"
                "application, such as Google Authenticator."
            )
            webbrowser.open("qr.png")
            mfa_code = q.ask(
                "Enter the verification code from your MFA application: ",
q.non_empty
            )
            response = cog_wrapper.verify_mfa(response["Session"], mfa_code)
```

```
            print(f"MFA device setup {response['Status']}")
            print("Now that an MFA application is set up, let's sign in again.")
            print(
                "You might have to wait a few seconds for a new MFA code to
appear in "
                "your MFA application."
            )
            challenge = "ADMIN_USER_PASSWORD_AUTH"
        elif response["ChallengeName"] == "SOFTWARE_TOKEN_MFA":
            auth_tokens = None
            while auth_tokens is None:
                mfa_code = q.ask(
                    "Enter a verification code from your MFA application: ",
q.non_empty
                )
                auth_tokens = cog_wrapper.respond_to_mfa_challenge(
                    user_name, response["Session"], mfa_code
                )
            print(f"You're signed in as {user_name}.")
            print("Here's your access token:")
            pp(auth_tokens["AccessToken"])
            print("And your device information:")
            pp(auth_tokens["NewDeviceMetadata"])
            challenge = None
        else:
            raise Exception(f"Got unexpected challenge
{response['ChallengeName']}")
    print("-" * 88)

    device_group_key = auth_tokens["NewDeviceMetadata"]["DeviceGroupKey"]
    device_key = auth_tokens["NewDeviceMetadata"]["DeviceKey"]
    device_password = base64.standard_b64encode(os.urandom(40)).decode("utf-8")

    print("Let's confirm your MFA device so you don't have re-enter MFA tokens
for it.")
    q.ask("Press Enter when you're ready.")
    cog_wrapper.confirm_mfa_device(
        user_name,
        device_key,
        device_group_key,
        device_password,
        auth_tokens["AccessToken"],
        aws_srp,
    )
```

```
    print(f"Your device {device_key} is confirmed.")
    print("-" * 88)

    print(
        f"Now let's sign in as {user_name} from your confirmed device
 {device_key}.\n"
        f"Because this device is tracked by Amazon Cognito, you won't have to re-
enter an MFA code."
    )
    q.ask("Press Enter when ready.")
    auth_tokens = cog_wrapper.sign_in_with_tracked_device(
        user_name, password, device_key, device_group_key, device_password,
 aws_srp
    )
    print("You're signed in. Your access token is:")
    pp(auth_tokens["AccessToken"])
    print("-" * 88)

    print("Don't forget to delete your user pool when you're done with this
 example.")
    print("\nThanks for watching!")
    print("-" * 88)


def main():
    parser = argparse.ArgumentParser(
        description="Shows how to sign up a new user with Amazon Cognito and
 associate "
        "the user with an MFA application for multi-factor authentication."
    )
    parser.add_argument(
        "user_pool_id", help="The ID of the user pool to use for the example."
    )
    parser.add_argument(
        "client_id", help="The ID of the client application to use for the
 example."
    )
    args = parser.parse_args()
    try:
        run_scenario(boto3.client("cognito-idp"), args.user_pool_id,
 args.client_id)
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

```
if __name__ == "__main__":
    main()
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp
  - VerifySoftwareToken

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Schreiben Sie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung mithilfe eines SDK AWS

Das folgende Codebeispiel zeigt, wie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito Cognito-Benutzerauthentifizierung geschrieben werden.

- Verwenden Sie Administratorfunktionen, um einen Benutzer zu einem Benutzerpool hinzuzufügen.
- Konfigurieren Sie einen Benutzerpool, um eine Lambda-Funktion für den `PostAuthentication` Trigger aufzurufen.
- Melden Sie den neuen Benutzer bei Amazon Cognito an.

- Die Lambda-Funktion schreibt benutzerdefinierte Informationen in CloudWatch Logs und in eine DynamoDB-Tabelle.
- Rufen Sie benutzerdefinierte Daten aus der DynamoDB-Tabelle ab, zeigen Sie sie an und bereinigen Sie anschließend die Ressourcen.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
// ActivityLog separates the steps of this scenario into individual functions so
 that
// they are simpler to read and understand.
type ActivityLog struct {
 helper       IScenarioHelper
 questioner   demotools.IQuestioner
 resources    Resources
 cognitoActor *actions.CognitoActions
}

// NewActivityLog constructs a new activity log runner.
func NewActivityLog(sdkConfig aws.Config, questioner demotools.IQuestioner,
 helper IScenarioHelper) ActivityLog {
 scenario := ActivityLog{
  helper:       helper,
  questioner:   questioner,
  resources:    Resources{},
  cognitoActor: &actions.CognitoActions{CognitoClient:
 cognitoidentityprovider.NewFromConfig(sdkConfig)},
 }
 scenario.resources.init(scenario.cognitoActor, questioner)
 return scenario
}
```

```
// AddUserToPool selects a user from the known users table and uses administrator
 credentials to add the user to the user pool.
func (runner *ActivityLog) AddUserToPool(userPoolId string, tableName string)
 (string, string) {
 log.Println("To facilitate this example, let's add a user to the user pool using
 administrator privileges.")
 users, err := runner.helper.GetKnownUsers(tableName)
 if err != nil {
  panic(err)
 }
 user := users.Users[0]
 log.Printf("Adding known user %v to the user pool.\n", user.UserName)
 err = runner.cognitoActor.AdminCreateUser(userPoolId, user.UserName,
 user.UserEmail)
 if err != nil {
  panic(err)
 }
 pwSet := false
 password := runner.questioner.AskPassword("\nEnter a password that has at least
 eight characters, uppercase, lowercase, numbers and symbols.\n"+
  "(the password will not display as you type):", 8)
 for !pwSet {
  log.Printf("\nSetting password for user '%v'.\n", user.UserName)
  err = runner.cognitoActor.AdminSetUserPassword(userPoolId, user.UserName,
 password)
  if err != nil {
   var invalidPassword *types.InvalidPasswordException
   if errors.As(err, &invalidPassword) {
    password = runner.questioner.AskPassword("\nEnter another password:", 8)
   } else {
    panic(err)
   }
  } else {
   pwSet = true
  }
 }

 log.Println(strings.Repeat("-", 88))

 return user.UserName, password
}
```

```
// AddActivityLogTrigger adds a Lambda handler as an invocation target for the
 PostAuthentication trigger.
func (runner *ActivityLog) AddActivityLogTrigger(userPoolId string,
 activityLogArn string) {
 log.Println("Let's add a Lambda function to handle the PostAuthentication
 trigger from Cognito.\n" +
  "This trigger happens after a user is authenticated, and lets your function
 take action, such as logging\n" +
  "the outcome.")
 err := runner.cognitoActor.UpdateTriggers(
  userPoolId,
  actions.TriggerInfo{Trigger: actions.PostAuthentication, HandlerArn:
 aws.String(activityLogArn)})
 if err != nil {
  panic(err)
 }
 runner.resources.triggers = append(runner.resources.triggers,
 actions.PostAuthentication)
 log.Printf("Lambda function %v added to user pool %v to handle
 PostAuthentication Cognito trigger.\n",
  activityLogArn, userPoolId)

 log.Println(strings.Repeat("-", 88))
}

// SignInUser signs in as the specified user.
func (runner *ActivityLog) SignInUser(clientId string, userName string, password
 string) {
 log.Printf("Now we'll sign in user %v and check the results in the logs and the
 DynamoDB table.", userName)
 runner.questioner.Ask("Press Enter when you're ready.")
 authResult, err := runner.cognitoActor.SignIn(clientId, userName, password)
 if err != nil {
  panic(err)
 }
 log.Println("Sign in successful.",
  "The PostAuthentication Lambda handler writes custom information to CloudWatch
 Logs.")

 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
 *authResult.AccessToken)
}
```

```
// GetKnownUserLastLogin gets the login info for a user from the Amazon DynamoDB
 table and displays it.
func (runner *ActivityLog) GetKnownUserLastLogin(tableName string, userName
 string) {
 log.Println("The PostAuthentication handler also writes login data to the
 DynamoDB table.")
 runner.questioner.Ask("Press Enter when you're ready to continue.")
 users, err := runner.helper.GetKnownUsers(tableName)
 if err != nil {
  panic(err)
 }
 for _, user := range users.Users {
  if user.UserName == userName {
   log.Println("The last login info for the user in the known users table is:")
   log.Printf("\t%+v", *user.LastLogin)
  }
 }
 log.Println(strings.Repeat("-", 88))
}

// Run runs the scenario.
func (runner *ActivityLog) Run(stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup()
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(stackOutputs["TableName"])
 userName, password := runner.AddUserToPool(stackOutputs["UserPoolId"],
 stackOutputs["TableName"])
```

```
 runner.AddActivityLogTrigger(stackOutputs["UserPoolId"],
 stackOutputs["ActivityLogFunctionArn"])
 runner.SignInUser(stackOutputs["UserPoolClientId"], userName, password)
 runner.helper.ListRecentLogEvents(stackOutputs["ActivityLogFunction"])
 runner.GetKnownUserLastLogin(stackOutputs["TableName"], userName)

 runner.resources.Cleanup()

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Behandeln Sie den `PostAuthentication` Trigger mit einer Lambda-Funktion.

```
const TABLE_NAME = "TABLE_NAME"

// LoginInfo defines structured login data that can be marshalled to a DynamoDB
 format.
type LoginInfo struct {
 UserPoolId string `dynamodbav:"UserPoolId"`
 ClientId   string `dynamodbav:"ClientId"`
 Time       string `dynamodbav:"Time"`
}

// UserInfo defines structured user data that can be marshalled to a DynamoDB
 format.
type UserInfo struct {
 UserName  string    `dynamodbav:"UserName"`
 UserEmail string    `dynamodbav:"UserEmail"`
 LastLogin LoginInfo `dynamodbav:"LastLogin"`
}

// GetKey marshals the user email value to a DynamoDB key format.
func (user UserInfo) GetKey() map[string]dynamodbtypes.AttributeValue {
 userEmail, err := attributevalue.Marshal(user.UserEmail)
 if err != nil {
  panic(err)
 }
 return map[string]dynamodbtypes.AttributeValue{"UserEmail": userEmail}
```

```
}

type handler struct {
 dynamoClient *dynamodb.Client
}

// HandleRequest handles the PostAuthentication event by writing custom data to
 the logs and
// to an Amazon DynamoDB table.
func (h *handler) HandleRequest(ctx context.Context,
 event events.CognitoEventUserPoolsPostAuthentication)
 (events.CognitoEventUserPoolsPostAuthentication, error) {
 log.Printf("Received post authentication trigger from %v for user '%v'",
 event.TriggerSource, event.UserName)
 tableName := os.Getenv(TABLE_NAME)
 user := UserInfo{
  UserName:  event.UserName,
  UserEmail: event.Request.UserAttributes["email"],
  LastLogin: LoginInfo{
   UserPoolId: event.UserPoolID,
   ClientId:   event.CallerContext.ClientID,
   Time:       time.Now().Format(time.UnixDate),
  },
 }
 // Write to CloudWatch Logs.
 fmt.Printf("%#v", user)

 // Also write to an external system. This examples uses DynamoDB to demonstrate.
 userMap, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshal to DynamoDB map. Here's why: %v\n", err)
 } else if len(userMap) == 0 {
  log.Printf("User info marshaled to an empty map.")
 } else {
  _, err := h.dynamoClient.PutItem(ctx, &dynamodb.PutItemInput{
   Item:      userMap,
   TableName: aws.String(tableName),
  })
  if err != nil {
   log.Printf("Couldn't write to DynamoDB. Here's why: %v\n", err)
  } else {
   log.Printf("Wrote user info to DynamoDB table %v.\n", tableName)
  }
 }
```

```
 return event, nil
}

func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Erstellen Sie eine Struktur, die allgemeine Aufgaben ausführt.

```
// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(stackName string) (actions.StackOutputs, error)
 PopulateUserTable(tableName string)
 GetKnownUsers(tableName string) (actions.UserList, error)
 AddKnownUser(tableName string, user actions.User)
 ListRecentLogEvents(functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
 example.
type ScenarioHelper struct {
 questioner  demotools.IQuestioner
 dynamoActor *actions.DynamoActions
 cfnActor    *actions.CloudFormationActions
 cwlActor    *actions.CloudWatchLogsActions
 isTestRun   bool
}

// NewScenarioHelper constructs a new scenario helper.
```

```
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
 ScenarioHelper {
 scenario := ScenarioHelper{
  questioner:  questioner,
  dynamoActor: &actions.DynamoActions{DynamoClient:
 dynamodb.NewFromConfig(sdkConfig)},
  cfnActor:    &actions.CloudFormationActions{CfnClient:
 cloudformation.NewFromConfig(sdkConfig)},
  cwlActor:    &actions.CloudWatchLogsActions{CwlClient:
 cloudwatchlogs.NewFromConfig(sdkConfig)},
 }
 return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 if !helper.isTestRun {
  time.Sleep(time.Duration(secs) * time.Second)
 }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
 structured format.
func (helper ScenarioHelper) GetStackOutputs(stackName string)
 (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(tableName string) {
 log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
 this example.\n", tableName)
 err := helper.dynamoActor.PopulateTable(tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
 error) {
 knownUsers, err := helper.dynamoActor.Scan(tableName)
 if err != nil {
```

```
  log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
 tableName, err)
 }
 return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(tableName string, user actions.User) {
 log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
 table...\n",
  user.UserName, user.UserEmail)
 err := helper.dynamoActor.AddUser(tableName, user)
 if err != nil {
  panic(err)
 }
}

// ListRecentLogEvents gets the most recent log stream and events for the
 specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(functionName string) {
 log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
 helper.Pause(10)
 log.Println("Okay, let's check the logs to find what's happened recently with
 your Lambda function.")
 logStream, err := helper.cwlActor.GetLatestLogStream(functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(functionName,
 *logStream.LogStreamName, 10)
 if err != nil {
  panic(err)
 }
 for _, event := range events {
  log.Printf("\t%v", *event.Message)
 }
 log.Println(strings.Repeat("-", 88))
}
```

Erstellen Sie eine Struktur, die Amazon Cognito Cognito-Aktionen umschließt.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int

const (
 PreSignUp Trigger = iota
 UserMigration
 PostAuthentication
)

type TriggerInfo struct {
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
```

```
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
  case PostAuthentication:
   lambdaConfig.PostAuthentication = trigger.HandlerArn
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
 }
 return err
}



// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &cognitoidentityprovider.SignUpInput{
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}
```

```
// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
  if errors.As(err, &resetRequired) {
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
  }
 } else {
  authResult = output.AuthenticationResult
 }
 return authResult, err
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
```

```
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
  Username:         aws.String(userName),
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
```

```
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
 userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
 }
 return err
}



// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
```

```
  }
 }
 return err
}
```

Erstellen Sie eine Struktur, die DynamoDB-Aktionen umschließt.

```
// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
 actions
// used in the examples.
type DynamoActions struct {
 DynamoClient *dynamodb.Client
}

// User defines structured user data.
type User struct {
 UserName  string
 UserEmail string
 LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
 UserPoolId string
 ClientId   string
 Time       string
}

// UserList defines a list of users.
type UserList struct {
 Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
 strings.
func (users *UserList) UserNameList() []string {
 names := make([]string, len(users.Users))
 for i := 0; i < len(users.Users); i++ {
  names[i] = users.Users[i].UserName
 }
```

```
 return names
}

// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(tableName string) error {
 var err error
 var item map[string]types.AttributeValue
 var writeReqs []types.WriteRequest
 for i := 1; i < 4; i++ {
  item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
  if err != nil {
   log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
 err)
   return err
  }
  writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
 &types.PutRequest{Item: item}})
 }
 _, err = actor.DynamoClient.BatchWriteItem(context.TODO(),
 &dynamodb.BatchWriteItemInput{
  RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
 })
 if err != nil {
  log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
 tableName, err)
 }
 return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(tableName string) (UserList, error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(context.TODO(), &dynamodb.ScanInput{
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
 err)
 } else {
  err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
  if err != nil {
   log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
  }
```

```
 }
 return userList, err
}

// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(tableName string, user User) error {
 userItem, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
 }
 _, err = actor.DynamoClient.PutItem(context.TODO(), &dynamodb.PutItemInput{
  Item:      userItem,
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
 }
 return err
}
```

Erstellen Sie eine Struktur, die Logs-Aktionen umschließt CloudWatch .

```
type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
 (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
 &cloudwatchlogs.DescribeLogStreamsInput{
  Descending:   aws.Bool(true),
  Limit:        aws.Int32(1),
  LogGroupName: aws.String(logGroupName),
  OrderBy:      types.OrderByLastEventTime,
 })
 if err != nil {
```

```
  log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
 logGroupName, err)
 } else {
  logStream = output.LogStreams[0]
 }
 return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
 stream.
func (actor CloudWatchLogsActions) GetLogEvents(functionName string,
 logStreamName string, eventCount int32) (
 []types.OutputLogEvent, error) {
 var events []types.OutputLogEvent
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.GetLogEvents(context.TODO(),
 &cloudwatchlogs.GetLogEventsInput{
  LogStreamName: aws.String(logStreamName),
  Limit:         aws.Int32(eventCount),
  LogGroupName:  aws.String(logGroupName),
 })
 if err != nil {
  log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
 logStreamName, err)
 } else {
  events = output.Events
 }
 return events, err
}
```

Erstellen Sie eine Struktur, die Aktionen umschließt. AWS CloudFormation

```
// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
 CfnClient *cloudformation.Client
}
```

```
// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
 structured format.
func (actor CloudFormationActions) GetOutputs(stackName string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(context.TODO(),
 &cloudformation.DescribeStacksInput{
  StackName: aws.String(stackName),
 })
 if err != nil || len(output.Stacks) == 0 {
  log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
 stackName, err)
 }
 stackOutputs := StackOutputs{}
 for _, out := range output.Stacks[0].Outputs {
  stackOutputs[*out.OutputKey] = *out.OutputValue
 }
 return stackOutputs
}
```

Ressourcen bereinigen.

```

// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
 userPoolId       string
 userAccessTokens []string
 triggers         []actions.Trigger

 cognitoActor *actions.CognitoActions
 questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
 demotools.IQuestioner) {
 resources.userAccessTokens = []string{}
 resources.triggers = []actions.Trigger{}
 resources.cognitoActor = cognitoActor
 resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
```

```
func (resources *Resources) Cleanup() {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong during cleanup.\n%v\n", r)
   log.Println("Use the AWS Management Console to remove any remaining resources
 \n" +
    "that were created for this scenario.")
  }
 }()

 wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
 resources that were created "+
  "during this demo (y/n)?", "y")
 if wantDelete {
  for _, accessToken := range resources.userAccessTokens {
   err := resources.cognitoActor.DeleteUser(accessToken)
   if err != nil {
    log.Println("Couldn't delete user during cleanup.")
    panic(err)
   }
   log.Println("Deleted user.")
  }
  triggerList := make([]actions.TriggerInfo, len(resources.triggers))
  for i := 0; i < len(resources.triggers); i++ {
   triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
 HandlerArn: nil}
  }
  err := resources.cognitoActor.UpdateTriggers(resources.userPoolId,
 triggerList...)
  if err != nil {
   log.Println("Couldn't update Cognito triggers during cleanup.")
   panic(err)
  }
  log.Println("Removed Cognito triggers from user pool.")
 } else {
  log.Println("Be sure to remove resources when you're done with them to avoid
 unexpected charges!")
 }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Go -API-Referenz.

- AdminCreateUser
- AdminSetUserPassword
- DeleteUser
- InitiateAuth
- UpdateUserPool

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele für Amazon Cognito Sync mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Amazon Cognito Sync mit einem AWS Software Development Kit (SDK) verwendet wird.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien und serviceübergreifenden Beispiele sehen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Aktionen für Amazon Cognito Sync mithilfe von AWS SDKs
  - Verwendung ListIdentityPoolUsage mit einem AWS SDK oder CLI

Aktionen für Amazon Cognito Sync mithilfe von AWS SDKs

Die folgenden Codebeispiele zeigen, wie einzelne Amazon Cognito Sync-Aktionen mit AWS SDKs ausgeführt werden. Diese Auszüge rufen die Amazon-Cognito–Sync-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der Amazon Cognito Sync-API-Referenz.

Beispiele

- Verwendung ListIdentityPoolUsage mit einem AWS SDK oder CLI

Verwendung **`ListIdentityPoolUsage`** mit einem AWS SDK oder CLI

Das folgende Codebeispiel zeigt, wie es verwendet wird`ListIdentityPoolUsage`.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
async fn show_pools(client: &Client) -> Result<(), Error> {
    let response = client
        .list_identity_pool_usage()
        .max_results(10)
        .send()
        .await?;

    let pools = response.identity_pool_usages();
    println!("Identity pools:");

    for pool in pools {
        println!(
            "  Identity pool ID:    {}",
            pool.identity_pool_id().unwrap_or_default()
        );
        println!(
            "  Data storage:        {}",
            pool.data_storage().unwrap_or_default()
        );
        println!(
            "  Sync sessions count: {}",
            pool.sync_sessions_count().unwrap_or_default()
        );
        println!(
```

```
            "  Last modified:       {}",
            pool.last_modified_date().unwrap().to_chrono_utc()?
        );
        println!();
    }

    println!("Next token: {}", response.next_token().unwrap_or_default());

    Ok(())
}
```

- Einzelheiten zur API finden Sie ListIdentityPoolUsagein der API-Referenz zum AWS SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Bewährte Methoden für Anwendungen für mehrere Mandanten

Amazon Cognito Cognito-Benutzerpools arbeiten mit Mehrmandantenanwendungen, die eine Menge von Anfragen generieren, die innerhalb der Amazon Cognito Cognito-Kontingente bleiben müssen. Um diese Kapazität zu erhöhen, wenn Ihr Kundenstamm wächst, können Sie zusätzliche Kontingentkapazität erwerben.

> **Note**
>
> Amazon Cognito Cognito-Kontingente werden pro AWS-Konto und AWS-Region angewendet. Diese Kontingente werden für alle Mandanten in Ihrer Anwendung freigegeben. Überprüfen Sie die Amazon Cognito-Servicekontingente und stellen Sie sicher, dass das Kontingent dem erwarteten Volumen und der erwarteten Anzahl von Mandanten in Ihrer Anwendung entspricht.

In diesem Abschnitt werden Methoden beschrieben, die Sie implementieren können, um Mandanten zwischen Amazon Cognito Cognito-Ressourcen innerhalb derselben Region und AWS-Konto zu trennen. Sie können Ihre Mandanten auch auf mehrere Mandanten AWS-Konto oder Regionen aufteilen und jedem von ihnen ein eigenes Kontingent zuweisen. Zu den weiteren Vorteilen der Mehrmandantenfähigkeit in mehreren Regionen gehören der höchstmögliche Isolationsgrad, die kürzeste Netzwerkübertragungszeit für global verteilte Benutzer und die Einhaltung der bestehenden Vertriebsmodelle in Ihrem Unternehmen.

Die Mehrmandantenfähigkeit in einer Region kann auch Vorteile für Ihre Kunden und Administratoren haben.

In der folgenden Liste werden einige der Vorteile von Mehrmandantenfähigkeit mit gemeinsam genutzten Ressourcen beschrieben.

Vorteile von Mehrmandantenverhältnissen

Gemeinsames Benutzerverzeichnis

Multi-Tenancy unterstützt Modelle, bei denen Kunden Konten in mehr als einer Anwendung haben. Sie können Identitäten von Drittanbietern zu einem einzigen konsistenten

Benutzerpoolprofil verknüpfen. In Fällen, in denen Benutzerprofile nur für ihren Mandanten gelten, verfügt jede Mehrmandantenstrategie mit einem einzigen Benutzerpool über einen einzigen Zugangspunkt zur Benutzerverwaltung.

Allgemeine Sicherheit

In einem gemeinsam genutzten Benutzerpool können Sie einen einzigen Sicherheitsstandard erstellen und dieselben erweiterten Sicherheits -, Multi-Faktor-Authentifizierung (MFA) und dieselben AWS WAFStandards auf alle Mandanten anwenden. Da sich eine AWS WAF Web-ACL in derselben AWS-Region Ressource befinden muss, der Sie sie zuordnen, bietet die Mehrmandantenfähigkeit gemeinsamen Zugriff auf eine komplexe Ressource. Wenn Sie eine konsistente Sicherheitskonfiguration in Amazon Cognito Cognito-Anwendungen mit mehreren Regionen aufrechterhalten möchten, müssen Sie Betriebsstandards anwenden, die Ihre Konfiguration zwischen Ressourcen replizieren.

Allgemeine Anpassung

Sie können Benutzerpools und Identitätspools mit anpassen AWS Lambda. Die Konfiguration von Lambda-Triggern in Benutzerpools und Amazon Cognito Cognito-Ereignissen in Identitätspools kann komplex werden. Lambda-Funktionen müssen sich im selben Bereich AWS-Region wie Ihr Benutzerpool oder Identitätspool befinden. Gemeinsam genutzte Lambda-Funktionen können Standards für benutzerdefinierte Authentifizierungsabläufe, Benutzermigration, Token-Generierung und andere Funktionen innerhalb einer Region durchsetzen.

Allgemeine Nachrichtenübermittlung

Amazon Simple Notification Service (Amazon SNS) erfordert eine zusätzliche Konfiguration in einer Region, bevor Sie SMS-Nachrichten an Ihre Benutzer senden können. Sie können E-Mail-Nachrichten mit von Amazon Simple Email Service (Amazon SES) verifizierten Identitäten und Domains versenden, die in einer Region enthalten sind.

Bei Mehrmandantenfähigkeit können Sie diese Konfiguration und den Wartungsaufwand auf alle Ihre Mandanten verteilen. Da Amazon SNS und Amazon SES nicht in allen verfügbar sind AWS-Regionen, müssen Sie bei der Aufteilung Ihrer Ressourcen zwischen Regionen zusätzliche Überlegungen anstellen.

Wenn Sie benutzerdefinierte Messaging-Anbieter verwenden, erhalten Sie die allgemeine Anpassung einer einzigen Lambda-Funktion zur Verwaltung Ihrer Nachrichtenzustellung.

Die gehostete Benutzeroberfläche setzt ein Sitzungscookie im Browser, sodass ein Benutzer erkannt wird, der sich bereits authentifiziert hat. Wenn Sie lokale Benutzer in einem Benutzerpool

authentifizieren, authentifiziert ihr Sitzungscookie sie für alle App-Clients im selben Benutzerpool. Ein lokaler Benutzer existiert ausschließlich in Ihrem Benutzerpool-Verzeichnis ohne Verbund über einen externen IdP. Der Sitzungscookie ist eine Stunde lang gültig. Sie können die Dauer des Sitzungscookies nicht ändern.

Es gibt zwei Möglichkeiten, die Anmeldung mehrerer App-Clients mit einem gehosteten UI-Sitzungscookie zu verhindern.

- Teilen Sie Ihre Benutzer in Benutzerpools pro Mandant auf.
- Ersetzen Sie die Anmeldung über die gehostete Benutzeroberfläche durch die API-Anmeldung für Amazon Cognito Cognito-Benutzerpools.

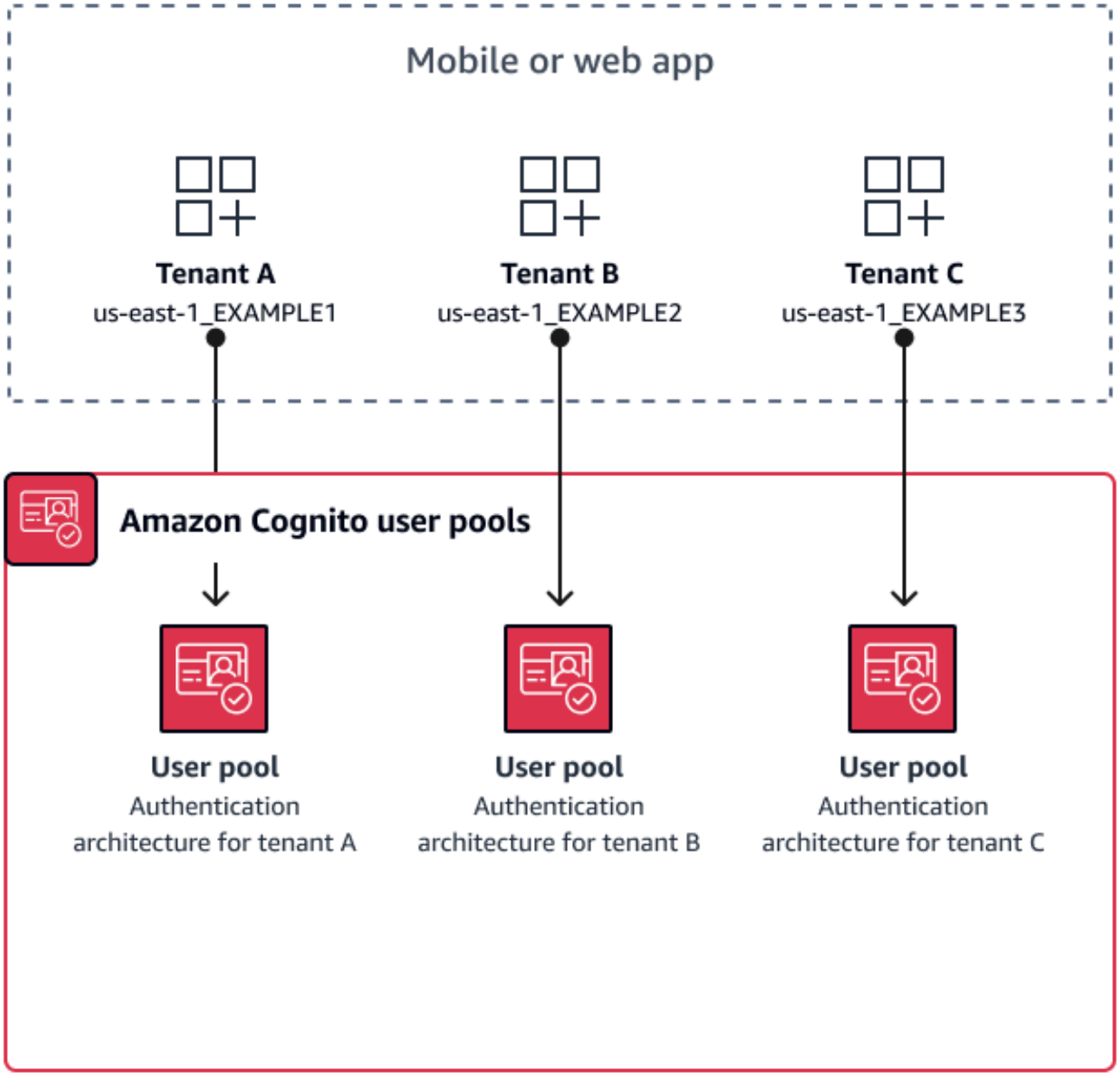
Themen

Bewährte Methoden für Benutzerpools und mehrere Mandanten

Erstellen Sie einen Benutzerpool für jeden Mandanten in Ihrer App. Dieser Ansatz bietet eine umfangreiche Isolation für jeden Mandanten. Sie können verschiedene Konfigurationen für jeden Mandanten implementieren. Die Mandantenisolierung nach Benutzerpool bietet Ihnen Flexibilität bei der user-to-tenant Zuordnung. Sie können mehrere Profile für denselben Benutzer erstellen. Jeder Benutzer muss sich jedoch individuell für jeden Mandanten registrieren, auf den er Zugriff hat.

Mit diesem Ansatz können Sie für jeden Mandanten unabhängig eine gehostete Benutzeroberfläche einrichten und Benutzer zu ihrer mandantenspezifischen Instanz Ihrer Anwendung weiterleiten. Sie können diesen Ansatz auch für die Integration mit Backend-Services wie Amazon API Gateway verwenden.

Das folgende Diagramm zeigt jeden Mandanten mit einem eigenen Benutzerpool.



Wann sollte ein Benutzerpool mit mehreren Mandanten implementiert werden

Wenn Isolierung und Anpassung Ihre Hauptanliegen sind. Die Beziehung zwischen Benutzern und Mandanten kann in einer Architektur mit mehreren Benutzerpools komplex sein. Stellen Sie sich ein Beispiel vor, in dem Sie zwei Mieter im Bildungswesen haben. Derselbe Benutzer könnte in einer App ein Schüler mit eingeschränktem Zugriff und in einer anderen ein Lehrer mit umfangreichen Berechtigungen sein. Möglicherweise benötigen Sie MFA in einer App, aber nicht

in einer anderen, oder Sie haben eine andere Passwortrichtlinie. Da sich lokale Benutzer mit der gehosteten Benutzeroberfläche bei mehreren App-Clients in Benutzerpools anmelden können, ist die Mehrmandantenfähigkeit eines Benutzerpools auch ideal, wenn Sie möchten, dass sich mehr als einer Ihrer Mandanten mit der gehosteten Benutzeroberfläche anmeldet.

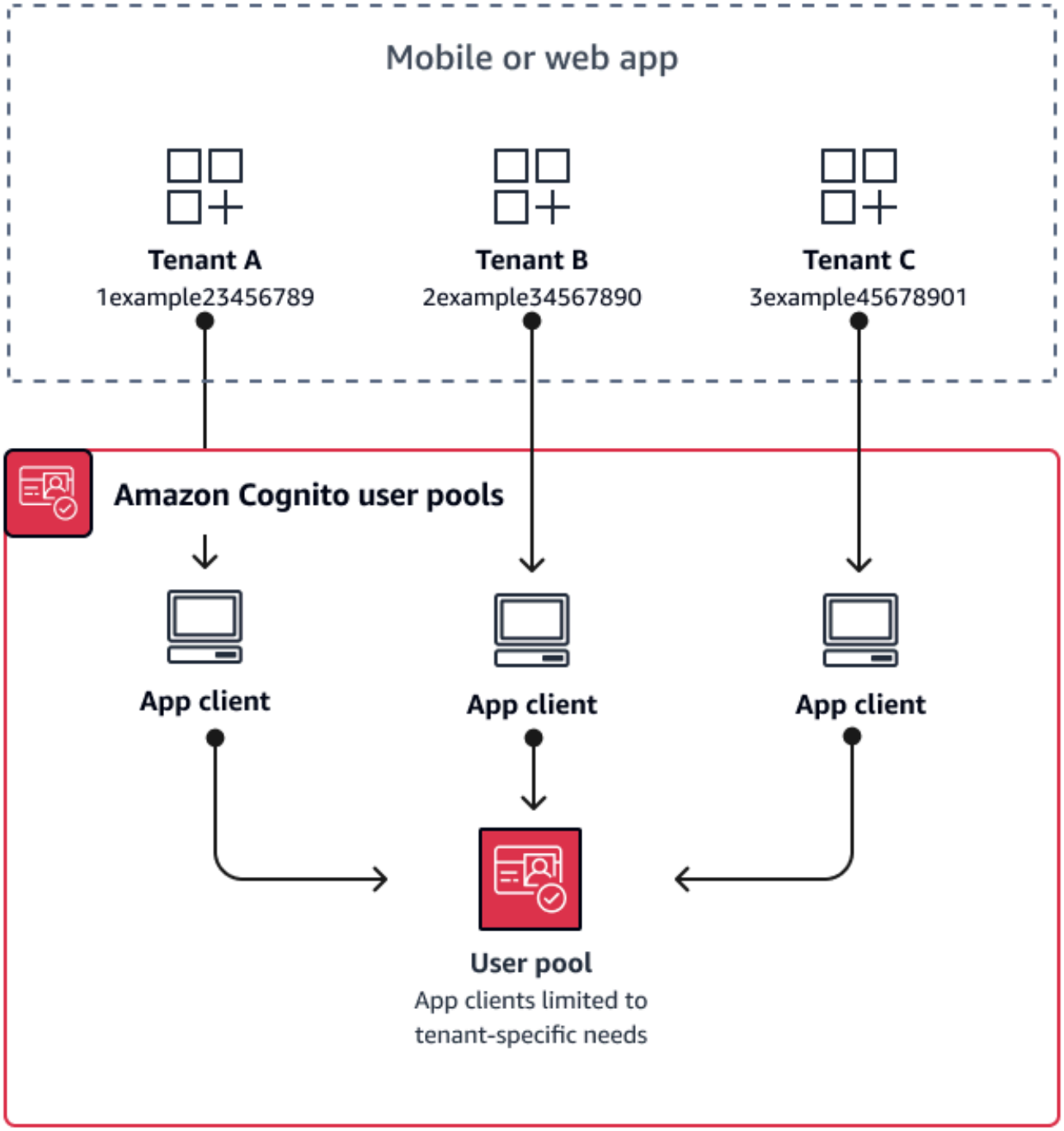
Grad des Aufwands

Der Entwicklungs- und Betriebsaufwand für diesen Ansatz ist hoch. Um konsistente und vorhersehbare Ergebnisse für Ihre App-Familie zu gewährleisten, müssen Sie Amazon Cognito Cognito-Ressourcen in Ihre Automatisierungstools integrieren und Ihre Basislinien beibehalten, wenn Ihre Authentifizierungsarchitektur immer komplexer wird. Wenn Sie einen zentralen Ausgangspunkt für Ihre Apps schaffen möchten, müssen Sie die Elemente der Benutzeroberfläche (UI) erstellen, um die ursprüngliche Entscheidung zu erfassen, mit der Benutzer zur richtigen Ressource weitergeleitet werden.

Bewährte Methoden für die Nutzung mehrerer Mandanten zwischen Apps und Clients

Erstellen Sie einen App-Client für jeden Mandanten in Ihrer App. Mit App-Client-Multi-Tenancy können Sie jeden Benutzer App-Clients zuweisen, die mit einem Mandanten verknüpft sind, und ein einziges Benutzerprofil beibehalten. Da Sie einem App-Client einen oder alle Identitätsanbieter (IdPs) in Ihrem Benutzerpool zuweisen können, kann ein Mandanten-App-Client die Anmeldung mit einem mandantenspezifischen IdP zulassen. Wenn Benutzer in mehreren Mandanten vorhanden sind, können Sie ihre Profile mit mehreren verknüpfen, um ein einheitliches Benutzererlebnis zu IdPs erzielen.

Das folgende Diagramm zeigt jeden Mandanten mit einem dedizierten App-Client in einem gemeinsam genutzten Benutzerpool.



Wann sollte App-Client-Mehrmandantenfähigkeit implementiert werden

Wenn Sie eine universelle Konfiguration für Einstellungen auf Benutzerpool-Ebene wählen können, wie Lambda-Trigger, Passwortrichtlinien sowie den Inhalt und die Zustellungsmethoden von E-Mail-

und SMS-Nachrichten. Da sich Benutzer in einem gemeinsam genutzten Benutzerpool bei jedem App-Client anmelden können, ist die App-Client-Multi-Tenancy ideal für die Anmeldung mit app-client-specific IdPs oder der Amazon Cognito Cognito-Benutzerpools-API. Die Mehrmandantenfähigkeit von App-Clients eignet sich auch gut für one-to-many Umgebungen, in denen Sie Benutzern den Wechsel zwischen mehreren Anwendungen ermöglichen möchten.

Grad des Aufwands

Die Mehrmandantenfähigkeit von Apps und Clients erfordert einen moderaten Aufwand. Eine große Herausforderung bei der Mehrmandantenfähigkeit zwischen Apps und Clients ist die Möglichkeit für Mandanten, ein gehostetes UI-Cookie zu präsentieren und zwischen Apps zu wechseln. Vermeiden Sie in einer App-Client-Architektur mit mehreren Mandanten die Anmeldung über eine gehostete Benutzeroberfläche, bei der eine Isolierung erforderlich ist. Sie können Ihre mobile App oder Links zu Ihrer Web-App mit integrierter App-Client-Logik verteilen, oder Sie können erste Benutzeroberflächenelemente erstellen, die die Tenancy der Benutzer bestimmen. Der Aufwand ist geringer, da Sie die Konfiguration nicht über mehrere Benutzerpools und Identitätspools hinweg standardisieren und verwalten müssen.

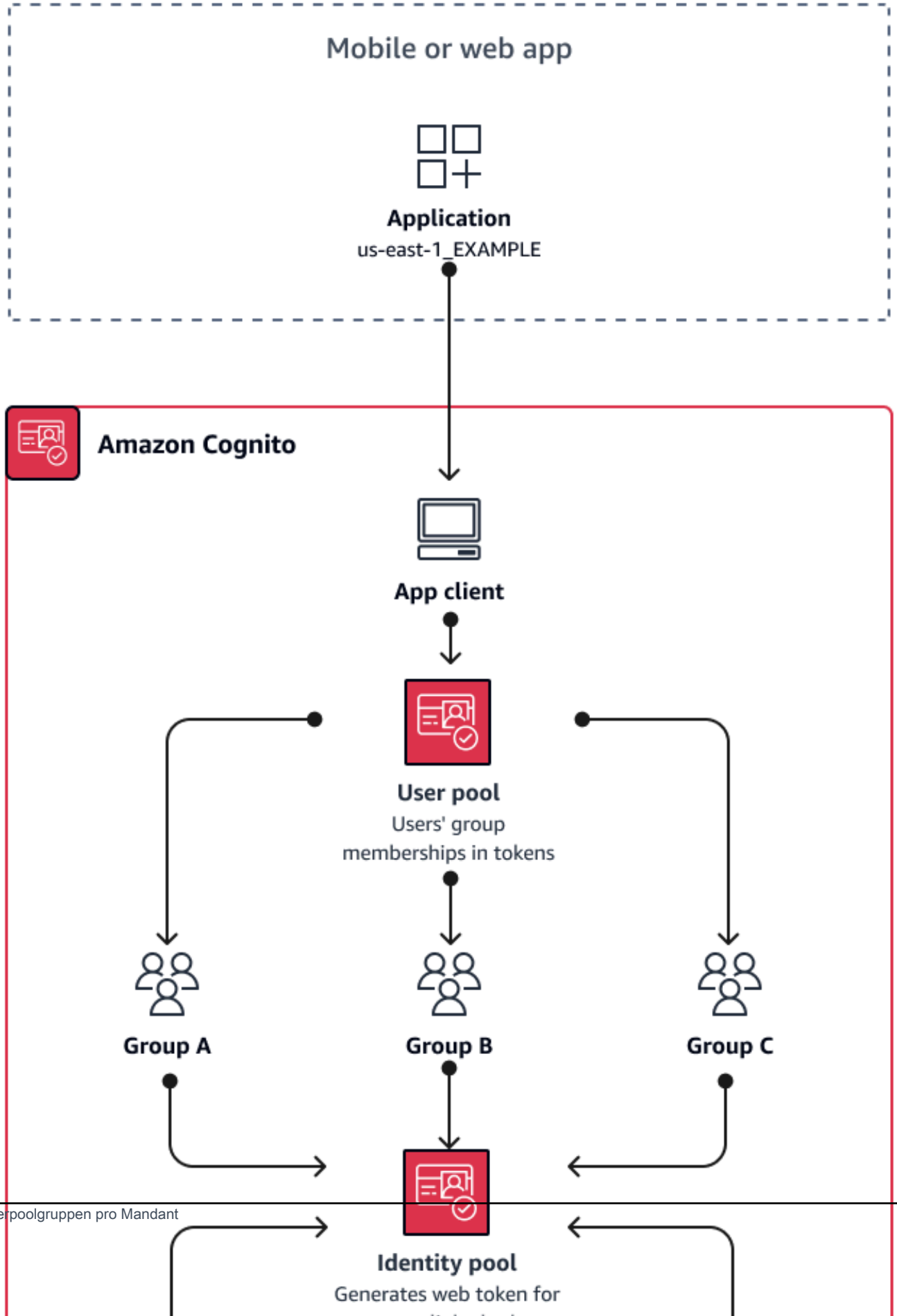
Bewährte Methoden für die Nutzung von Benutzerpools, Gruppen und mehreren Mandanten

Gruppenbasierte Mehrmandantenfähigkeit funktioniert am besten, wenn Ihre Architektur Amazon Cognito Cognito-Benutzerpools mit Identitätspools erfordert.

Die Benutzerpool-ID und die Zugriffstoken enthalten einen Anspruch. `cognito:groups` Darüber hinaus enthalten ID-Token `cognito:preferred_role` Ansprüche `cognito:roles` und Ansprüche. Wenn das primäre Ergebnis der Authentifizierung in Ihrer App temporäre AWS Anmeldeinformationen aus einem Identitätspool sind, können die Gruppenmitgliedschaften Ihrer Benutzer die IAM-Rolle und die Berechtigungen bestimmen, die sie erhalten.

Stellen Sie sich als Beispiel drei Mandanten vor, die jeweils Anwendungsressourcen in ihrem eigenen Amazon S3 S3-Bucket speichern. Weisen Sie die Benutzer jedes Mandanten einer zugehörigen Gruppe zu, konfigurieren Sie eine bevorzugte Rolle für die Gruppe und gewähren Sie dieser Rolle Lesezugriff auf ihren Bucket.

Das folgende Diagramm zeigt Mandanten, die sich einen App-Client und einen Benutzerpool teilen, wobei bestimmte Gruppen im Benutzerpool ihre Eignung für eine IAM-Rolle bestimmen.


Mobile or web app
Application
us-east-1_EXAMPLE
Amazon Cognito
App client
User pool
Users' group
memberships in tokens
Group A
Group B
Group C
Identity pool
Generates web token for

Wann sollte Mehrmandantenfähigkeit für Gruppen implementiert werden

Wenn der Zugriff auf AWS Ressourcen Ihr Hauptanliegen ist. Gruppen in Amazon Cognito Cognito-Benutzerpools Benutzerpools sind ein Mechanismus für die rollenbasierte Zugriffskontrolle (RBAC). Sie können viele Gruppen in einem Benutzerpool konfigurieren und komplexe RBAC-Entscheidungen mit Gruppenpriorität treffen. Identitätspools können Anmeldeinformationen für die Rolle mit der höchsten Priorität, für jede Rolle im Gruppenanspruch oder für andere Ansprüche in den Token eines Benutzers zuweisen.

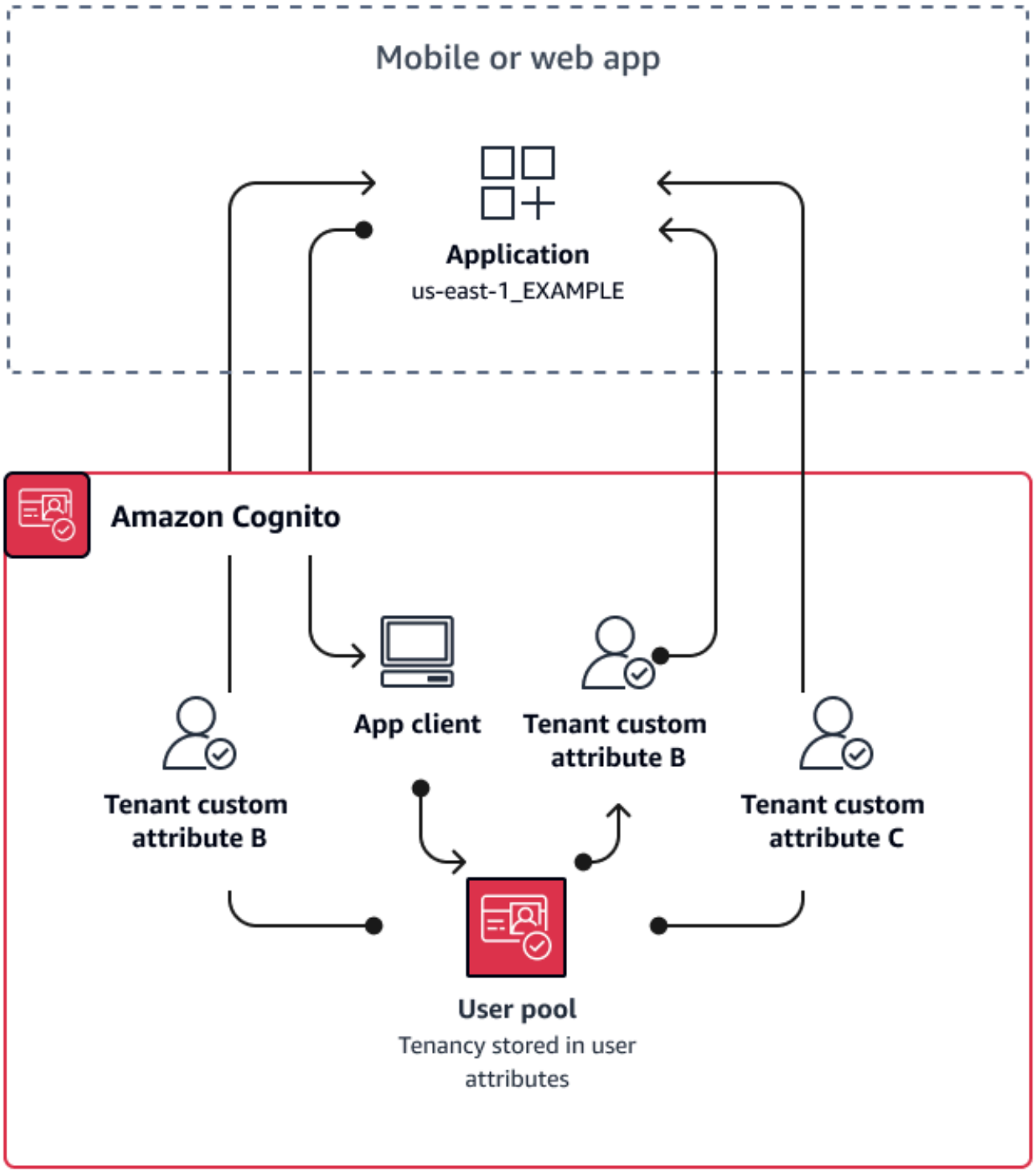
Grad des Aufwands

Der Aufwand, die Mehrmandantenfähigkeit allein durch Gruppenmitgliedschaft aufrechtzuerhalten, ist gering. Um jedoch die Rolle von Benutzerpoolgruppen über die integrierte Kapazität für die IAM-Rollenauswahl hinaus zu erweitern, müssen Sie eine Anwendungslogik erstellen, die die Gruppenmitgliedschaft in Benutzertoken verarbeitet, und festlegen, was im Client zu tun ist. Sie können Amazon Verified Permissions in Ihre Apps integrieren, um clientseitige Autorisierungsentscheidungen zu treffen. Gruppen-IDs werden derzeit nicht in IsAuthorizedWithTokenAPI-Vorgängen für verifizierte Berechtigungen verarbeitet. Sie können jedoch benutzerdefinierten Code entwickeln, der den Inhalt von Token analysiert, einschließlich der Ansprüche auf Gruppenmitgliedschaft.

Bewährte Methoden für Mehrmandantenfähigkeit mit benutzerdefinierten Attributen

Amazon Cognito unterstützt benutzerdefinierte Attribute mit Namen, die Sie wählen. Ein Szenario, in dem benutzerdefinierte Attribute nützlich sind, ist, wenn sie die Tenance von Benutzern in einem gemeinsam genutzten Benutzerpool unterscheiden. Wenn Sie Benutzern einen Wert für ein Attribut wie zuweisen`custom:tenantID`, kann Ihre App den Zugriff auf mandantenspezifische Ressourcen entsprechend zuweisen. Ein benutzerdefiniertes Attribut, das eine Mandanten-ID definiert, sollte für den App-Client unveränderlich oder schreibgeschützt sein.

Das folgende Diagramm zeigt Mandanten, die sich einen App-Client und einen Benutzerpool teilen, wobei benutzerdefinierte Attribute im Benutzerpool den Mandanten angeben, zu dem sie gehören.



Wenn benutzerdefinierte Attribute die Mandantenfähigkeit bestimmen, können Sie eine einzelne Anwendungs- oder Anmelde-URL verteilen. Nachdem sich Ihr Benutzer angemeldet hat, kann Ihre

App den `custom:tenantID` Antrag bearbeiten und festlegen, welche Inhalte geladen werden sollen, welches Branding angewendet werden soll und welche Funktionen angezeigt werden sollen. Für erweiterte Entscheidungen zur Zugriffskontrolle anhand von Benutzerattributen richten Sie Ihren Benutzerpool als Identitätsanbieter in Amazon Verified Permissions ein und generieren Sie Zugriffsentscheidungen anhand der Inhalte von ID- oder Zugriffstoken.

Wann sollte Mehrmandantenfähigkeit mit benutzerdefinierten Attributen implementiert werden

Wenn das Mietverhältnis oberflächlich ist. Ein Mandantenattribut kann zu Branding- und Layoutergebnissen beitragen. Wenn Sie eine deutliche Isolierung zwischen Mandanten erreichen möchten, sind benutzerdefinierte Attribute nicht die beste Wahl. Jeder Unterschied zwischen Mandanten, die auf Benutzerpool- oder App-Client-Ebene konfiguriert werden müssen, wie MFA oder Branding für gehostete Benutzeroberflächen, erfordert, dass Sie Unterscheidungen zwischen Mandanten auf eine Weise erstellen, die benutzerdefinierte Attribute nicht bieten. Bei Identitätspools können Sie sogar die IAM-Rolle Ihrer Benutzer aus dem Anspruch auf benutzerdefinierte Attribute in ihrem ID-Token auswählen.

Grad des Aufwands

Da die Mehrmandantenfähigkeit mit benutzerdefinierten Attributen die Pflicht, mandantenbasierte Autorisierungsentscheidungen zu treffen, auf Ihre App überträgt, ist der Aufwand in der Regel hoch. Wenn Sie sich bereits mit einer Client-Konfiguration auskennen, die OIDC-Anträge analysiert, oder mit Amazon Verified Permissions, ist dieser Ansatz möglicherweise mit dem geringsten Aufwand verbunden.

Sicherheitsempfehlungen für Mehrmandantenfähigkeit

Die folgenden Empfehlungen können helfen, Ihre Anwendung sicherer zu gestalten.

- Bestätigen Sie das Mietverhältnis in Ihrer App mit Amazon Verified Permissions. Erstellen Sie Richtlinien, die Benutzerpools, App-Clients-, Gruppen- oder benutzerdefinierte Attributberechtigungen untersuchen, bevor Sie die Anfrage eines Benutzers in Ihrer Anwendung zulassen. AWS hat Identitätsquellen mit verifizierten Berechtigungen unter Berücksichtigung von Amazon Cognito Cognito-Benutzerpools erstellt. Verified Permissions bietet zusätzliche Anleitungen für die Verwaltung mehrerer Mandanten.
- Verwenden Sie nur eine verifizierte E-Mail-Adresse, um den Benutzerzugriff auf einen Mandanten basierend auf Domänenübereinstimmung zu autorisieren. Vertrauen Sie E-Mail-Adressen und Telefonnummern nur, wenn Ihre App sie überprüft oder der externe IdP Ihnen einen Nachweis

über die Überprüfung erteilt. Weitere Details zum Festlegen dieser Berechtigungen finden Sie unter Attributberechtigungen und -bereiche.

- Verwenden Sie unveränderliche oder schreibgeschützte benutzerdefinierte Attribute für die Benutzerprofilattribute, die Mandanten identifizieren. Sie können den Wert unveränderlicher Attribute nur festlegen, wenn Sie einen Benutzer erstellen oder wenn sich ein Benutzer in Ihrem Benutzerpool anmeldet. Ermöglichen Sie App-Clients schreibgeschützten Zugriff auf die Attribute.
- Verwenden Sie eine 1:1 -Zuordnung zwischen dem externen IdP eines Mandanten und dem Anwendungsclient, um unbefugten mandantenübergreifenden Zugriff zu verhindern. Ein Benutzer, der von einem externen Identitätsanbieter authentifiziert wurde und über ein gültiges Amazon-Cognito-Sitzungscookie verfügt, kann auf andere Mandanten-Apps zugreifen, die demselben Identitätsanbieter vertrauen.
- Stellen Sie beim Implementieren von Mandantenübereinstimmungs- und Autorisierungslogik in Ihrer Anwendung sicher, dass die Kriterien, die zum Autorisieren des Benutzerzugriffs auf die Mandanten verwendet werden, nicht von den Benutzern selbst geändert werden können. Wenn ein externer IdP für den Verbund verwendet wird, beschränken Sie die Mandantenidentitätsanbieter-Administratoren, damit sie den Benutzerzugriff nicht ändern können.

Häufige Amazon-Cognito-Szenarien

In diesem Thema werden sechs typische Szenarien für die Verwendung von Amazon Cognito beschrieben.

Die zwei Hauptkomponenten von Amazon Cognito sind Benutzerpools und Identitäten-Pools. Benutzerpools sind Benutzerverzeichnisse, die Registrierungs- und Anmeldungsoptionen für Ihre Web- und mobilen Anwendungs-Nutzer bereitstellen. Identitätspools stellen temporäre AWS Anmeldeinformationen bereit, um Ihren Benutzern Zugriff auf andere zu gewähren AWS-Services.

Ein Benutzerpool ist ein Benutzerverzeichnis in Amazon Cognito. Ihre App-Benutzer können sich entweder direkt über einen Benutzerpool anmelden oder sich über einen externen Identitätsanbieter (IdP) zusammenschließen. Der Benutzerpool verwaltet den Aufwand für die Verwaltung der Token, die bei der Anmeldung in sozialen Netzwerken über Facebook, Google, Amazon und Apple sowie von OpenID Connect (OIDC) und SAML zurückgegeben werden. IdPs Ganz gleich, ob sich Ihre Benutzer direkt oder über einen Drittanbieter anmelden, alle Mitglieder Ihres Benutzerpools haben ein Verzeichnisprofil, auf das Sie über ein SDK zugreifen können.

Mit einem Identitätspool können Ihre Benutzer temporäre AWS Anmeldeinformationen für den Zugriff auf AWS Dienste wie Amazon S3 und DynamoDB abrufen. Identitätspools unterstützen anonyme Gastbenutzer sowie den Zusammenschluss über Drittanbieter. IdPs

Themen

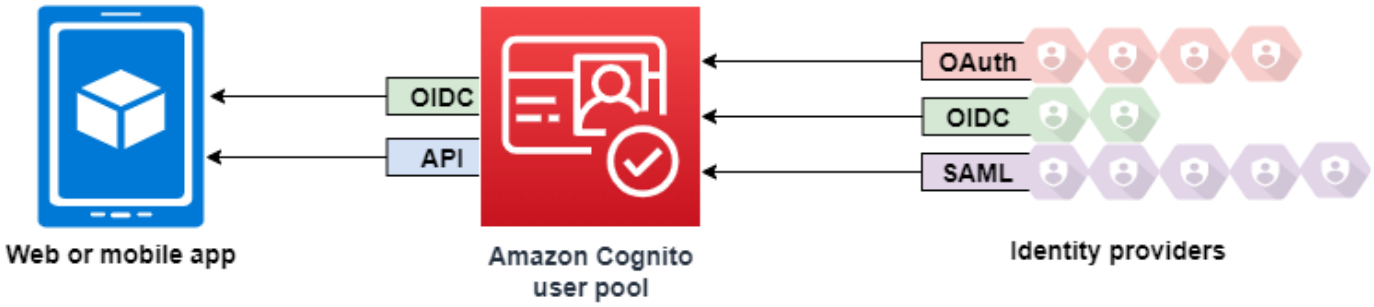
Authentifizierung über einen Benutzerpool

Sie können Ihren Benutzern ermöglichen, sich mit einem Benutzerpool zu authentifizieren. Ihre App-Benutzer können sich entweder direkt über einen Benutzerpool anmelden oder sich über einen externen Identitätsanbieter (IdP) zusammenschließen. Der Benutzerpool verwaltet den Aufwand für

die Verwaltung der Token, die bei der Anmeldung in sozialen Netzwerken über Facebook, Google, Amazon und Apple sowie von OpenID Connect (OIDC) und SAML zurückgegeben werden. IdPs

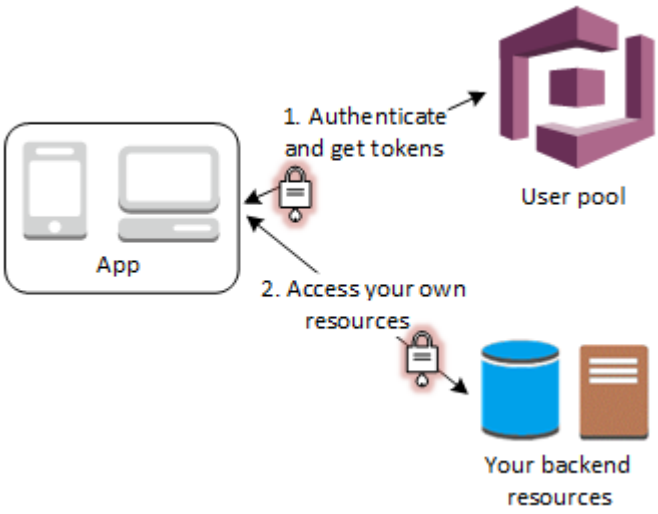
Nach einer erfolgreichen Authentifizierung erhält Ihre Web- oder mobilen Anwendungs-Nutzerpool-Token von Amazon Cognito. Sie können diese Token verwenden, um AWS Anmeldeinformationen abzurufen, mit denen Ihre App auf andere AWS Dienste zugreifen kann, oder Sie können sie verwenden, um den Zugriff auf Ihre serverseitigen Ressourcen oder auf das Amazon API Gateway zu kontrollieren.

Weitere Informationen erhalten Sie unter Ablauf der Authentifizierung in Benutzerpools und Verwenden von Token mit Benutzerpools.



Zugriff auf Ihre serverseitigen Ressourcen mit einem Benutzerpool

Nach einer erfolgreichen Benutzerpool-Anmeldung erhält Ihre Web- oder Mobil-App-Benutzerpool-Token von Amazon Cognito. Sie können mithilfe dieser Token den Zugriff auf Ihre serverseitigen Ressourcen kontrollieren. Sie können auch Benutzerpoolgruppen erstellen, um Berechtigungen zu verwalten und verschiedene Arten von Benutzern darzustellen. Weitere Informationen über die Verwendung von Gruppen zum Steuern des Zugriffs auf Ihre Ressourcen finden Sie unter Hinzufügen von Gruppen zu einem Benutzerpool.



Nachdem eine Domäne für Ihren Benutzerpool konfiguriert wurde, stellt Amazon Cognito eine gehostete Web-Benutzeroberfläche bereit, die Ihnen das Hinzufügen von Registrierungs- und Anmeldeseiten für Ihre App erlaubt. Mit dieser OAuth-2.0-Grundlage können Sie einen eigenen Ressourcenserver erstellen, damit Ihre Benutzer auf geschützte Ressourcen zugreifen können. Weitere Informationen finden Sie unter Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern.

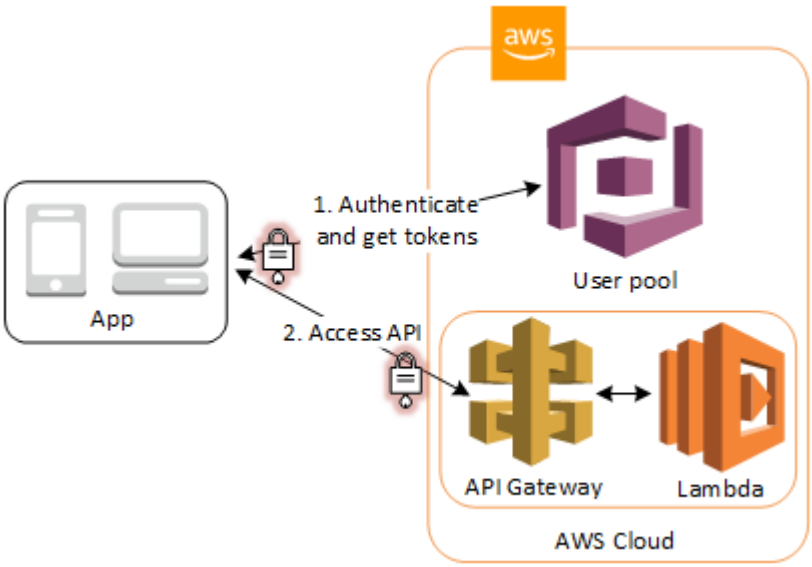
Weitere Informationen über die Benutzerpool-Authentifizierung finden Sie unter Ablauf der Authentifizierung in Benutzerpools und Verwenden von Token mit Benutzerpools.

Zugriff auf Ressourcen mit API Gateway und Lambda mit einem Benutzerpool

Sie können Benutzern ermöglichen, über API Gateway auf Ihre API zuzugreifen. API Gateway validiert die Tokens aus einer erfolgreichen Benutzerpool-Authentifizierung und gewährt damit Ihren Benutzern den Zugriff auf Ressourcen, darunter Lambda-Funktionen oder Ihre eigene API.

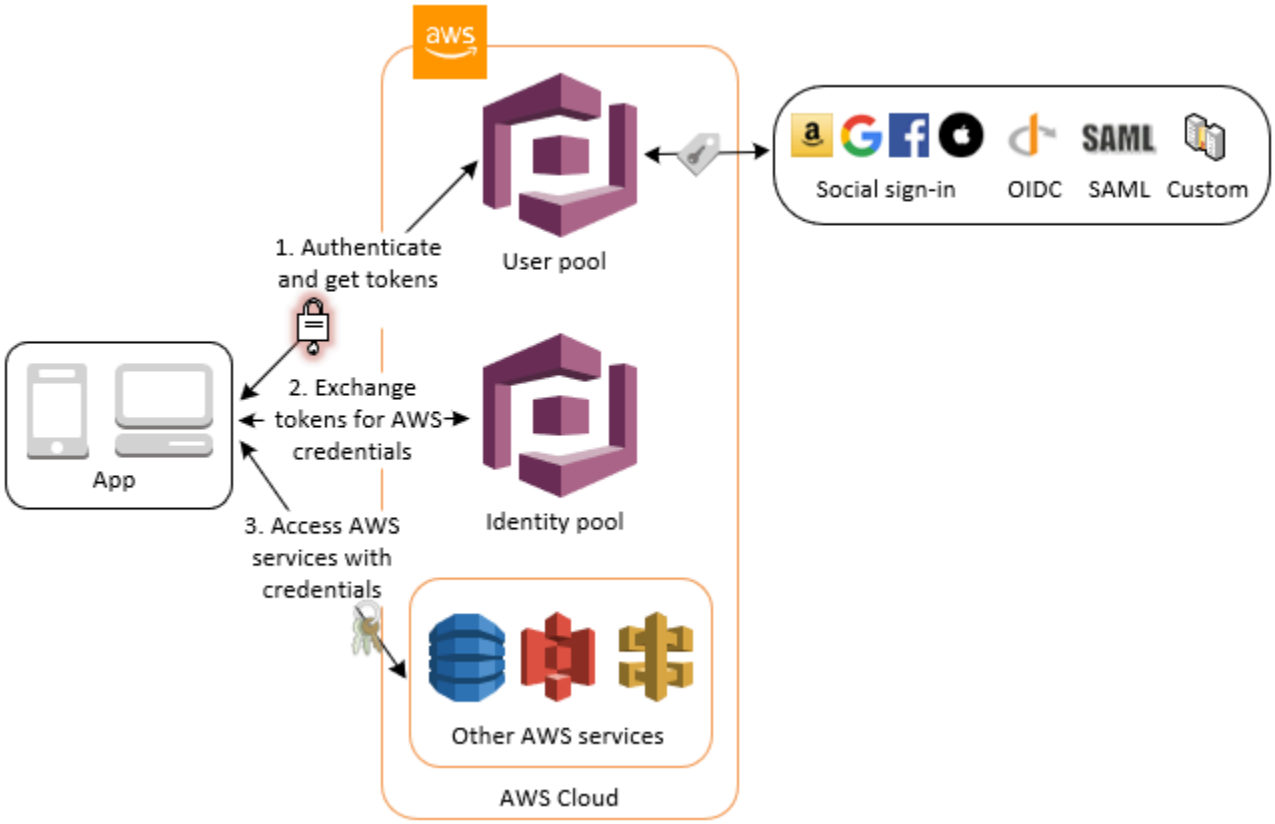
Sie können Gruppen in einem Benutzerpool verwenden, um Berechtigungen für API Gateway zu steuern, indem Sie die Gruppenmitgliedschaft mit IAM-Rollen abgleichen. Die Gruppen, bei denen ein Benutzer ein Mitglied ist, werden in das ID-Token eingeschlossen, das von einem Benutzerpool zur Verfügung gestellt, wenn sich Ihr App-Benutzer anmeldet. Weitere Informationen zu Benutzerpoolgruppen finden Sie unter Hinzufügen von Gruppen zu einem Benutzerpool.

Sie können Ihre Benutzerpool-Tokens mit einer Anfrage an API Gateway übermitteln, damit diese von einer Amazon-Cognito-Authorizer-Lambda-Funktion verifiziert werden. Weitere Informationen zu API Gateway finden Sie unter Verwenden von API Gateway mit Amazon-Cognito-Benutzerpools.



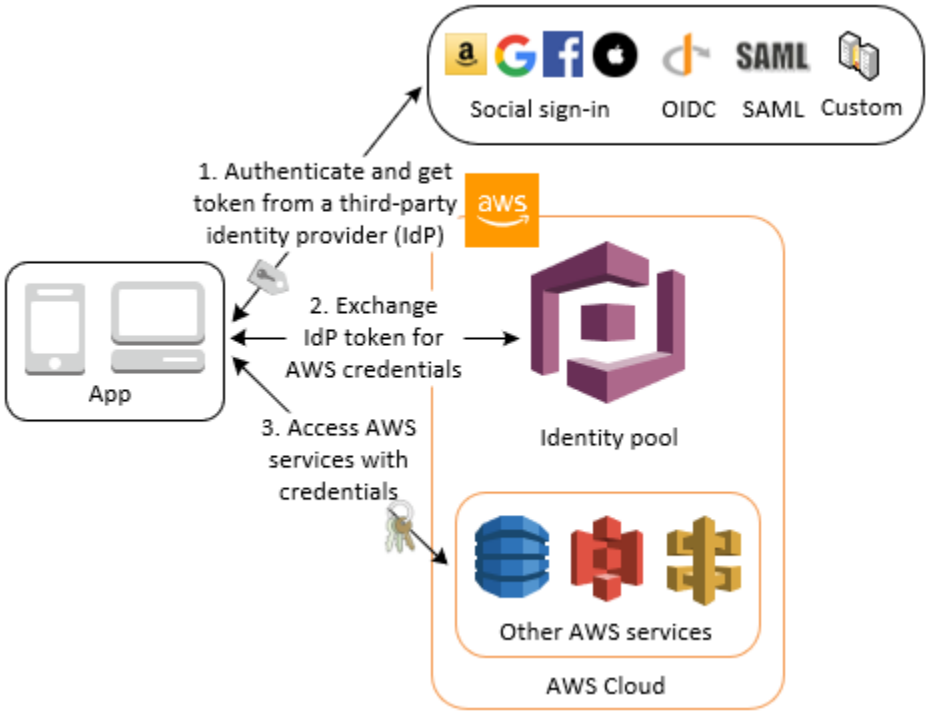
Greifen Sie mit einem Benutzerpool und einem Identitätspool auf AWS Dienste zu

Nach einer erfolgreichen Benutzerpool-Authentifizierung erhält Ihre App Benutzerpool-Token von Amazon Cognito. Sie können sie gegen temporären Zugriff auf andere AWS Dienste mit einem Identitätspool eintauschen. Weitere Informationen finden Sie unter Zugriff AWS-Services über einen Identitätspool nach der Anmeldung und Erste Schritte mit Amazon Cognito Cognito-Identitätspools.



Authentifizierung über einen Drittanbieter und Zugriff auf AWS -Services über einen Identitätspool

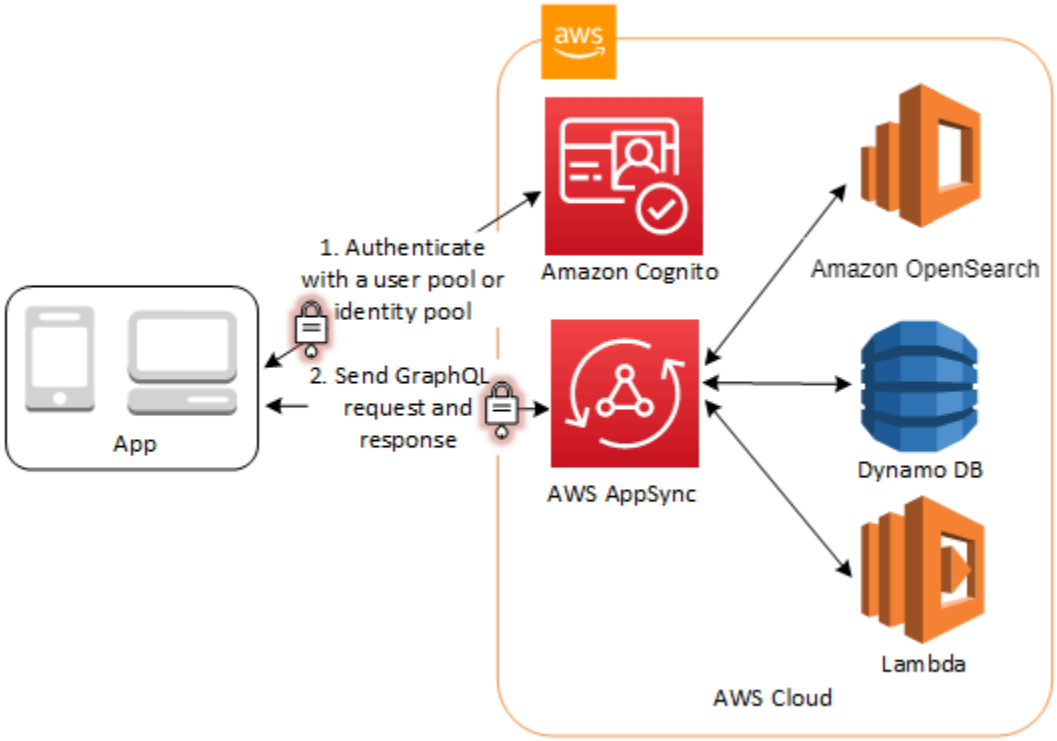
Sie können Ihren Benutzern den Zugriff auf AWS Dienste über einen Identitätspool ermöglichen. Ein Identitäten-Pool erfordert ein IdP-Token von einem Benutzer, der von einem externen Identitätsanbieter authentifiziert wurde (bzw. nichts, wenn es sich um einen anonymen Gast handelt). Im Gegenzug gewährt der Identitätspool temporäre AWS Anmeldeinformationen, mit denen Sie auf andere AWS Dienste zugreifen können. Weitere Informationen finden Sie unter Erste Schritte mit Amazon Cognito Cognito-Identitätspools.



Greifen Sie mit Amazon Cognito auf AWS AppSync Ressourcen zu

Sie können Ihren Benutzern Zugriff auf AWS AppSync Ressourcen mit Tokens aus einer erfolgreichen Amazon Cognito Cognito-Benutzerpool-Authentifizierung gewähren. Weitere Informationen finden Sie unter AMAZON_COGNITO_USER_POOLS-Autorisierung im AWS AppSync -Entwicklerhandbuch.

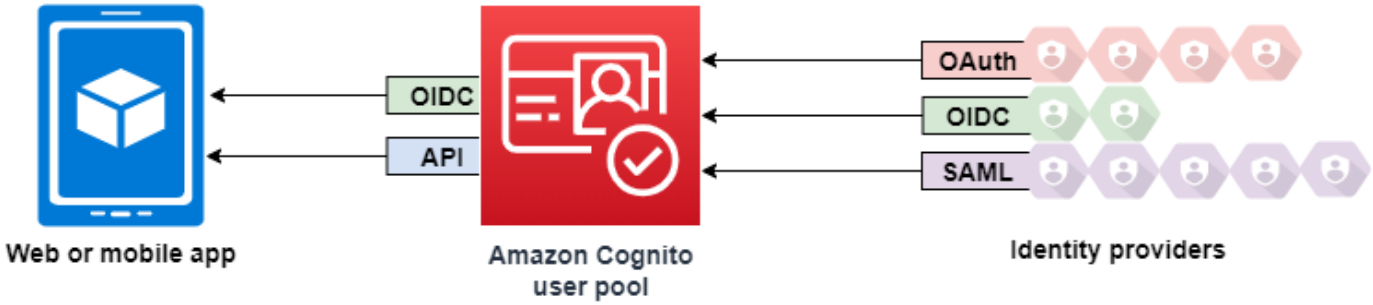
Sie können Anfragen an die AWS AppSync GraphQL-API auch mit den IAM-Anmeldeinformationen signieren, die Sie von einem Identitätspool erhalten. Weitere Informationen finden Sie unter AWS_IAM-Autorisierung.


aws
Amazon Cognito
Amazon OpenSearch
1. Authenticate with a user pool or identity pool
2. Send GraphQL request and response
App
AWS AppSync
Dynamo DB
Lambda
AWS Cloud

Amazon-Cognito-Benutzerpools

Ein Amazon-Cognito-Benutzerpool ist ein Benutzerverzeichnis für die Authentifizierung und Autorisierung von Web- und mobilen Apps. Aus der Sicht Ihrer App ist ein Amazon-Cognito-Benutzerpool ein OIDC (OpenID Connect)-Identitätsanbieter. Ein Benutzerpool bietet zusätzliche Ebenen für Sicherheit, Identitätsverbund, Anwendungsintegration und Anpassung der Benutzerumgebung.

Sie können beispielsweise überprüfen, ob die Sitzungen Ihrer Benutzer aus vertrauenswürdigen Quellen stammen. Sie können das Amazon-Cognito-Verzeichnis mit einem externen Identitätsanbieter kombinieren. Mit Ihrem bevorzugten AWS SDK können Sie das API-Autorisierungsmodell auswählen, das für Ihre App am besten geeignet ist. Dazu können Sie AWS Lambda -Funktionen hinzufügen, die das Standardverhalten von Amazon Cognito ändern oder überarbeiten.



Themen

- Features
- Authentifizierung mit einem Benutzerpool
- Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte
- Aktualisieren der Benutzerpool-Konfiguration
- Einrichtung und Verwendung der gehosteten Benutzeroberfläche und der Verbundendpunkte in Amazon Cognito
- Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern
- Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter
- Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern
- Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools

- Verwalten von Benutzern in Ihrem Benutzerpool
- E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools
- Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools
- Verwenden von Token mit Benutzerpools
- Zugriff auf Ressourcen nach einer erfolgreichen Benutzerpool-Authentifizierung
- Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools

Features

Amazon-Cognito-Benutzerpools haben die folgenden Merkmale.

Registrieren

Amazon-Cognito-Benutzerpools verfügen über benutzergesteuerte, administratorgesteuerte und programmatische Methoden, um Ihrem Benutzerpool Benutzerprofile hinzuzufügen. Amazon-Cognito-Benutzerpools unterstützen die folgenden Anmeldemodelle. Sie können eine beliebige Kombination dieser Modelle in Ihrer App verwenden.

> **Important**
>
> Wenn Sie die Benutzerregistrierung in Ihrem Benutzerpool aktivieren, kann sich jeder im Internet für ein Konto registrieren und bei Ihren Apps anmelden. Aktivieren Sie die Selbstregistrierung in Ihrem Benutzerpool nur dann, wenn Sie die öffentliche Registrierung für Ihre App aktivieren möchten. Um diese Einstellung zu ändern, aktualisieren Sie die Self-Service-Registrierung auf der Registerkarte Anmeldevorgang der Benutzerpool-Konsole oder aktualisieren Sie den Wert `AllowAdminCreateUserOnly` in einer `CreateUserPool` oder `UpdateUserPool`API-Anfrage.
>
> Hinweise zu Sicherheitsfunktionen, die Sie in Ihren Benutzerpools einrichten können, finden Sie unter Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools.

1. Ihre Benutzer können ihre Informationen in Ihre App eingeben und ein Benutzerprofil erstellen, das für Ihren Benutzerpool spezifisch ist. Sie können API-Anmeldevorgänge aufrufen, um Benutzer in Ihrem Benutzerpool zu registrieren. Sie können diese Anmeldevorgänge für jedermann öffnen oder sie mit einem geheimen Kundengeheimnis oder Anmeldeinformationen autorisieren. AWS

2. Sie können Benutzer an einen externen IdP weiterleiten, den sie autorisieren können, ihre Informationen an Amazon Cognito weiterzugeben. Amazon Cognito verarbeitet OIDC-ID-Tokens, OAuth-2.0–`userInfo`Daten und SAML-2.0-Assertions in Benutzerprofilen in Ihrem Benutzerpool. Sie kontrollieren die Attribute, die Amazon Cognito erhalten soll, auf der Grundlage von Regeln für die Attributzuordnung.
3. Sie können die öffentliche oder Verbundregistrierung überspringen und Benutzer auf der Grundlage Ihrer eigenen Datenquelle und Ihres eigenen Schemas erstellen. Fügen Sie Benutzer direkt in der Amazon-Cognito-Konsole oder der API hinzu. Importieren von Benutzern aus einer CSV-Datei. Führen Sie eine just-in-time AWS Lambda Funktion aus, die Ihren neuen Benutzer in einem vorhandenen Verzeichnis sucht und sein Benutzerprofil anhand vorhandener Daten auffüllt.

Nachdem sich Ihre Benutzer angemeldet haben, können Sie sie zu Gruppen hinzufügen, die Amazon Cognito in den Zugriffs- und ID-Token auflistet. Sie können Benutzerpoolgruppen auch mit IAM-Rollen verknüpfen, wenn Sie das ID-Token an einen Identitätspool übergeben.

Verwandte Themen

- Verwalten von Benutzern in Ihrem Benutzerpool
- Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte
- Codebeispiele für Amazon Cognito Identity Provider mit AWS SDKs

Anmelden

Amazon Cognito kann ein eigenständiges Benutzerverzeichnis und ein Identitätsanbieter (IDP) für Ihre App sein. Ihre Benutzer können sich über eine Benutzeroberfläche anmelden, die von Amazon Cognito gehostet wird, oder über Ihre eigene Benutzeroberfläche über die Amazon-Cognito-Benutzerpool-API. Die Anwendungsebene hinter Ihrer benutzerdefinierten Frontend-Benutzeroberfläche kann Anfragen im Backend mit einer von mehreren Methoden autorisieren, um legitime Anfragen zu bestätigen.

Um Benutzer mit einem externen Verzeichnis anzumelden, das optional mit dem in Amazon Cognito integrierten Benutzerverzeichnis kombiniert ist, können Sie die folgenden Integrationen hinzufügen.

1. Melden Sie sich an und importieren Sie Verbrauchernutzerdaten mit OAuth 2.0 Social Sign-in. Amazon Cognito unterstützt die Anmeldung mit Google, Facebook, Amazon und Apple über OAuth 2.0.

2. Melden Sie sich an und importieren Sie Unternehmensnutzerdaten mit SAML- und OIDC-Anmeldung. Sie können Amazon Cognito auch so konfigurieren, dass es Anforderungen von jedem SAML- oder OpenID Connect (OIDC)-Identitätsanbieter (IDP) akzeptiert.
3. Verknüpfen Sie externe Benutzerprofile mit nativen Benutzerprofilen. Ein verknüpfter Benutzer kann sich mit einer Benutzeridentität eines Drittanbieters anmelden und Zugriff erhalten, den Sie einem Benutzer im integrierten Verzeichnis zuweisen.

Verwandte Themen

- Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter
- Verknüpfen von Verbundbenutzern mit einem vorhandenen Benutzerprofil

M-Autorisierung achine-to-machine

Manche Sitzungen sind keine human-to-machine Interaktion. Möglicherweise benötigen Sie ein Servicekonto, das eine Anfrage an eine API durch einen automatisierten Prozess autorisieren kann. Um Zugriffstoken für die machine-to-machine Autorisierung mit OAuth 2.0-Bereichen zu generieren, können Sie einen App-Client hinzufügen, der Berechtigungen für Client-Anmeldeinformationen generiert.

Verwandte Themen

- Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern

Gehostete Benutzeroberfläche

Wenn Sie keine Benutzeroberfläche erstellen möchten, können Sie Ihren Benutzern eine benutzerdefinierte, von Amazon Cognito gehostete Benutzeroberfläche präsentieren. Die gehostete Benutzeroberfläche besteht aus einer Reihe von Webseiten für die Registrierung, die Anmeldung, die Multi-Faktor-Authentifizierung (MFA) und die Passwortrücksetzung. Sie können die gehostete Benutzeroberfläche zu Ihrer vorhandenen Domain hinzufügen oder eine Präfix-ID in einer Subdomain verwenden. AWS

Verwandte Themen

- Einrichtung und Verwendung der gehosteten Benutzeroberfläche und der Verbundendpunkte in Amazon Cognito

- Konfigurieren einer Benutzerpool-Domäne

Sicherheit

Ihre lokalen Benutzer können einen zusätzlichen Authentifizierungsfaktor mit einem Code aus einer SMS-Nachricht oder einer App angeben, die Multi-Faktor-Authentifizierungscodes (MFA) generiert. Sie können Mechanismen erstellen, um MFA in Ihrer App einzurichten und zu verarbeiten, oder dies von der gehosteten Benutzeroberfläche verwalten lassen. Amazon-Cognito-Benutzerpools können MFA umgehen, wenn sich Ihre Benutzer von vertrauenswürdigen Geräten aus anmelden.

Wenn Sie MFA zunächst nicht von Ihren Benutzern verlangen möchten, können Sie dies unter bestimmten Bedingungen anfordern. Mit erweiterten Sicherheitsfunktionen kann Amazon Cognito potenzielle böswillige Aktivitäten erkennen und Ihre Benutzer auffordern, MFA einzurichten oder die Anmeldung blockieren.

Wenn der Netzwerkverkehr zu Ihrem Benutzerpool möglicherweise bösartig ist, können Sie ihn überwachen und mit AWS WAF Web-ACLs Maßnahmen ergreifen.

Verwandte Themen

- Hinzufügen der MFA zu einem Benutzerpool
- Hinzufügen erweiterter Sicherheit zu einem Benutzerpool.
- Eine AWS WAF Web-ACL einem Benutzerpool zuordnen

Benutzerdefinierte Nutzerumgebung

In den meisten Phasen der Registrierung, Anmeldung oder Profilaktualisierung eines Benutzers können Sie einstellen, wie Amazon Cognito die Anfrage behandelt. Mit Lambda-Auslösern können Sie ein ID-Token ändern oder eine Anmeldeanfrage auf der Grundlage benutzerdefinierter Bedingungen ablehnen. Sie können Ihren eigenen benutzerdefinierten Authentifizierungsprozess erstellen.

Sie können benutzerdefiniertes CSS und Logos hochladen, um Ihren Benutzern ein vertrautes Erscheinungsbild der gehosteten Benutzeroberfläche zu geben.

Verwandte Themen

- Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern

- Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen
- Anpassen der integrierten Registrierungs- und Anmeldungswebseiten

Überwachung und Analytik

Amazon-Cognito-Benutzerpools protokollieren API-Anfragen, einschließlich Anfragen an die gehostete Benutzeroberfläche, in AWS CloudTrail. Sie können Leistungskennzahlen in Amazon CloudWatch Logs überprüfen, benutzerdefinierte Protokolle CloudWatch mit Lambda-Triggern per Push übertragen und das Volumen der API-Anfragen in der Service Quotas Quotas-Konsole überwachen.

Sie können auch Geräte- und Sitzungsdaten aus Ihren API-Anfragen für eine Amazon-Pinpoint-Kampagne protokollieren. Mit Amazon Pinpoint können Sie basierend auf Ihrer Analyse der Benutzeraktivitäten Push-Benachrichtigungen von Ihrer App aus senden.

Verwandte Themen

- Protokollieren Amazon Cognito Cognito-API-Aufrufen mit AWS CloudTrail
- Tracking von Kontingenten und Nutzung in CloudWatch Service Quotas
- Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools

Integration von Amazon-Cognito-Identitätspools

Die andere Hälfte von Amazon Cognito sind die Identitätspools. Identitätspools stellen Anmeldeinformationen bereit, mit denen API-Anfragen von Ihren Benutzern autorisiert und überwacht werden können AWS-Services, z. B. an Amazon DynamoDB oder Amazon S3. Sie können identitätsbasierte Zugriffsrichtlinien erstellen, die Ihre Daten, basierend darauf schützen, wie Sie die Benutzer in Ihrem Benutzerpool klassifizieren. Identitätspools können unabhängig von der Benutzerpool-Authentifizierung auch Tokens und SAML-2.0-Assertions von einer Vielzahl von Identitätsanbietern akzeptieren.

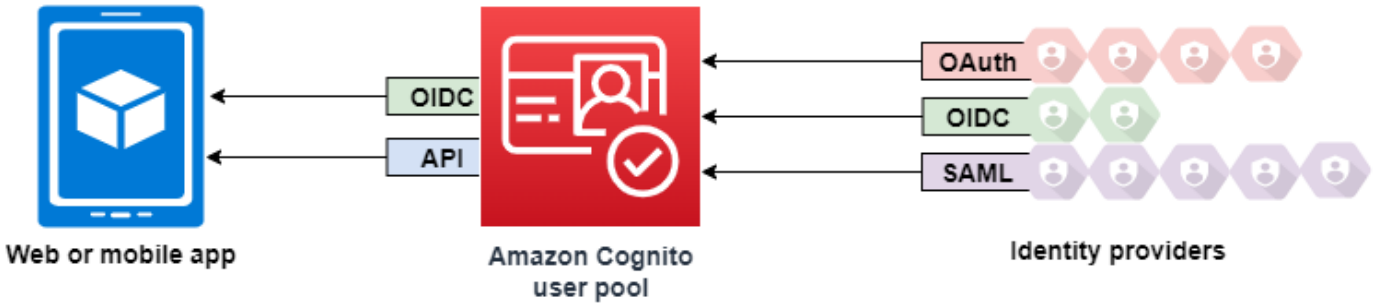
Verwandte Themen

- Zugriff AWS-Services über einen Identitätspool nach der Anmeldung
- Amazon-Cognito-Identitätspools

Authentifizierung mit einem Benutzerpool

Ihre App-Benutzer können sich entweder direkt über einen Benutzerpool anmelden oder sich über einen externen Identitätsanbieter (IdP) zusammenschließen. Der Benutzerpool verwaltet den Aufwand für die Verwaltung der Token, die bei der Anmeldung in sozialen Netzwerken über Facebook, Google, Amazon und Apple sowie von OpenID Connect (OIDC) und SAML zurückgegeben werden. IdPs

Nach einer erfolgreichen Authentifizierung gibt Amazon Cognito Benutzerpool-Token an Ihre App zurück. Sie können dies Tokens verwenden, um Ihren Benutzern den Zugriff auf Ihre eigenen serverseitigen Ressourcen oder auf Amazon API Gateway zu gewähren. Oder Sie können sie gegen AWS Anmeldeinformationen für den Zugriff auf andere Dienste eintauschen. AWS



Die Handhabung und Verwaltung von Benutzerpool-Token für Ihre Web- oder mobile App wird auf der Client-Seite über Amazon-Cognito-SDKs bereitgestellt. Analog dazu aktualisieren Mobile SDK for iOS und Mobile SDK for Android automatisch Ihre ID- und Zugriffstoken, wenn ein gültiges (nicht abgelaufenes) Aktualisierungstoken vorhanden ist, und die ID- und Zugriffstoken eine Gültigkeit von noch mindestens fünf Minuten haben. Informationen zu den SDKs und Beispielcode für JavaScript Android und iOS finden Sie unter Amazon Cognito Cognito-Benutzerpool-SDKs.

Nachdem sich Ihr App-Benutzer erfolgreich angemeldet hat, erstellt Amazon Cognito eine Sitzung und gibt ein ID- , Zugriffs- und Aktualisierungstoken für den authentifizierten Benutzer zurück.

JavaScript

```
// Amazon Cognito creates a session which includes the id, access, and refresh
 tokens of an authenticated user.

var authenticationData = {
        Username : 'username',
        Password : 'password',
```

```
    };
    var authenticationDetails = new
 AmazonCognitoIdentity.AuthenticationDetails(authenticationData);
    var poolData = { UserPoolId : 'us-east-1_ExaMPle',
        ClientId : '1example23456789'
    };
    var userPool = new AmazonCognitoIdentity.CognitoUserPool(poolData);
    var userData = {
        Username : 'username',
        Pool : userPool
    };
    var cognitoUser = new AmazonCognitoIdentity.CognitoUser(userData);
    cognitoUser.authenticateUser(authenticationDetails, {
        onSuccess: function (result) {
            var accessToken = result.getAccessToken().getJwtToken();

            /* Use the idToken for Logins Map when Federating User Pools with
 identity pools or when passing through an Authorization Header to an API Gateway
 Authorizer */
            var idToken = result.idToken.jwtToken;
        },

        onFailure: function(err) {
            alert(err);
        },

});
```

Android

```
// Session is an object of the type CognitoUserSession, and includes the id, access,
 and refresh tokens for a user.

String idToken = session.getIdToken().getJWTToken();
String accessToken = session.getAccessToken().getJWT();
```

iOS - swift

```
// AWSCognitoIdentityUserSession includes id, access, and refresh tokens for a user.

- (AWSTask<AWSCognitoIdentityUserSession *> *)getSession;
```

iOS - objective-C

```
// AWSCognitoIdentityUserSession includes the id, access, and refresh tokens for a
 user.

[[user getSession:@"username" password:@"password" validationData:nil scopes:nil]
 continueWithSuccessBlock:^id _Nullable(AWSTask<AWSCognitoIdentityUserSession *> *
 _Nonnull task) {
    // success, task.result has user session
    return nil;
}];
```

Themen

- Ablauf der Authentifizierung in Benutzerpools
- App-Clients für Benutzerpools
- Verwendung von Benutzergeräten in Ihrem Benutzerpool

Ablauf der Authentifizierung in Benutzerpools

Amazon Cognito umfasst mehrere Methoden zur Authentifizierung Ihrer Benutzer. Alle Benutzerpools, unabhängig davon, ob Sie eine Domain haben oder nicht, können Benutzer in der Benutzerpool-API authentifizieren. Wenn Sie Ihrem Benutzerpool eine Domain hinzufügen, können Sie die Benutzerpool-Endpunkte verwenden. Die Benutzerpool-API unterstützt eine Vielzahl von Autorisierungsmodellen und Anforderungsabläufen für API-Anforderungen.

Zur Überprüfung der Identität von Benutzern unterstützt Amazon Cognito Authentifizierungsabläufe, die zusätzlich zu Passwörtern neue Aufforderungstypen beinhalten. Für die Amazon Cognito-Authentifizierung müssen Sie normalerweise zwei API-Vorgänge in der folgenden Reihenfolge implementieren:

Public authentication

1. InitiateAuth
2. RespondToAuthChallenge

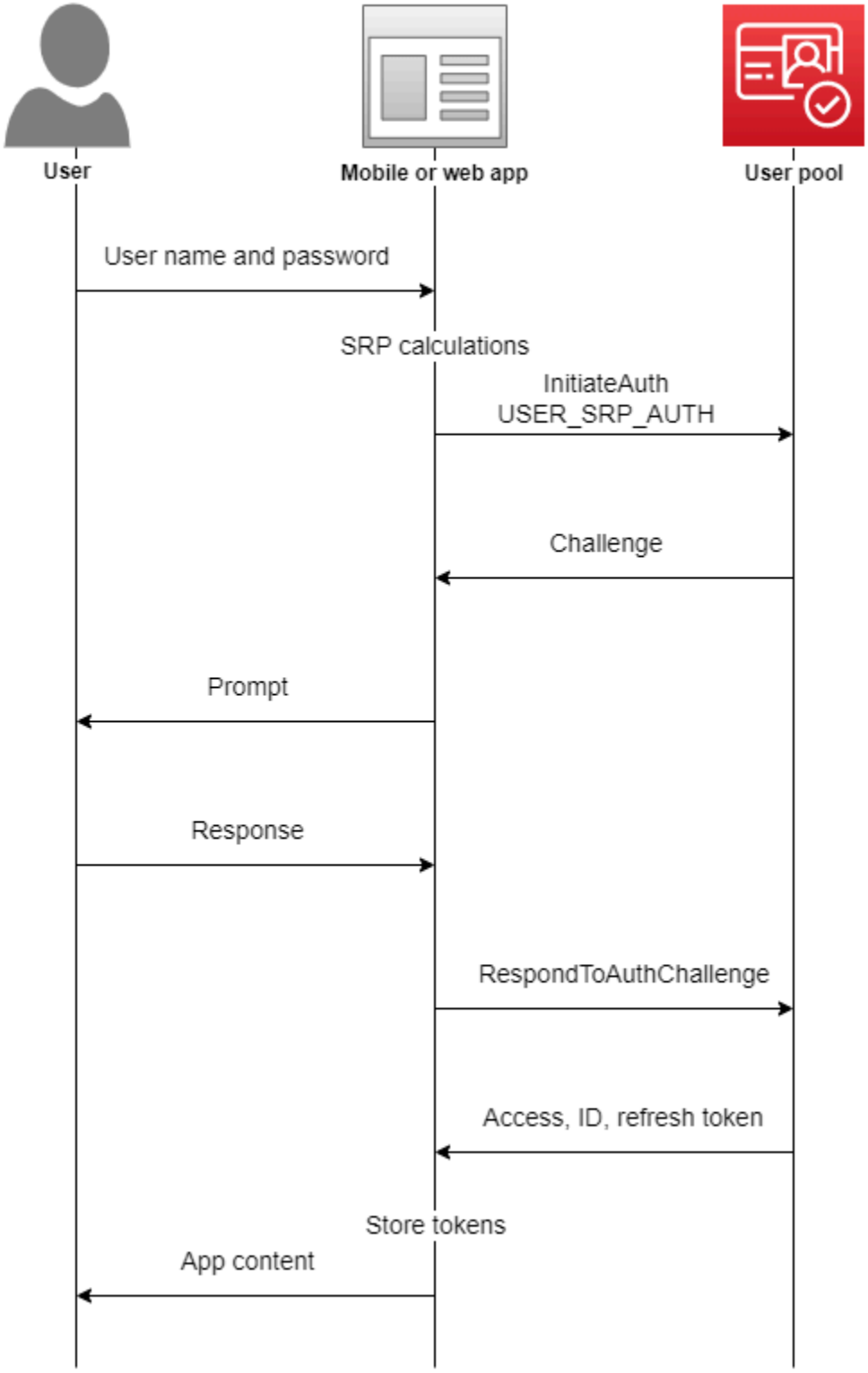
`InitiateAuth` und `RespondToAuthChallenge` sind nicht authentifizierte APIs zur Verwendung mit clientseitigen öffentlichen App-Clients.

Server-side authentication

1. AdminInitiateAuth
2. AdminRespondToAuthChallenge

   `AdminInitiateAuth` und `AdminRespondToAuthChallenge` erfordern IAM-Anmeldeinformationen und eignen sich für serverseitige vertrauliche App-Clients.

Ein Benutzer führt die Authentifizierung durch Beantwortung aufeinanderfolgender Eingabeaufforderungen durch, bis die Authentifizierung entweder fehlschlägt oder Amazon Cognito Token für den Benutzer ausgestellt. Sie können diese Schritte mit Amazon Cognito in einem Prozess wiederholen, der verschiedene Aufforderungen zur Unterstützung benutzerdefinierter Authentifizierungsabläufe beinhaltet.



In der Regel generiert Ihre App eine Aufforderung, Informationen von Ihrem Benutzer zu erfassen, und sendet diese Informationen in einer API-Anforderung an Amazon Cognito. Betrachten Sie

einen `InitiateAuth`-Ablauf in einem Benutzerpool, in dem Sie Ihren Benutzer mit Multi-Faktor-Authentifizierung (MFA) konfiguriert haben.

1. Ihre App fordert den Benutzer zur Eingabe des Benutzernamens und des Passworts auf.
2. Sie fügen den Benutzernamen und das Passwort als Parameter in `InitiateAuth` ein.
3. Amazon Cognito gibt eine `SMS_MFA`-Abfrage und eine Sitzungskennung zurück.
4. Ihre App fordert Ihren Benutzer auf, den MFA-Code von seinem Telefon einzugeben.
5. Sie fügen diesen Code und die Sitzungskennung in die `RespondToAuthChallenge`-Anforderung ein.

Abhängig von den Funktionen Ihres Benutzerpools können Sie am Ende auf verschiedene Abfragen von `InitiateAuth` reagieren, bevor Ihre App Token von Amazon Cognito abruft. Amazon Cognito fügt in der Antwort auf jede Anforderung eine Sitzungszeichenfolge ein. Wenn Sie Ihre API-Anforderungen zu einem Authentifizierungsfluss kombinieren möchten, fügen Sie die Sitzungszeichenfolge aus der Antwort auf die vorherige Anforderung in jede nachfolgende Anforderung ein. Standardmäßig haben Ihre Benutzer für den Abschluss einer Abfrage drei Minuten Zeit, bevor die Sitzungszeichenfolge abläuft. Wenn Sie diesen Zeitraum anpassen möchten, ändern Sie Ihren App-Client Authentication flow session duration (Dauer der Authentifizierungsablaufsitzung). Im folgenden Verfahren wird beschrieben, wie diese Einstellung in Ihrer App-Client-Konfiguration geändert wird.

> **Note**
>
> Die Einstellungen für die Dauer der Sitzung zum Authentifizierungsablauf gelten für die Authentifizierung mit der Amazon-Cognito-Benutzerpool-API. Die von Amazon Cognito gehostete Benutzeroberfläche legt die Sitzungsdauer für die Multi-Faktor-Authentifizierung auf 3 Minuten und für Codes zum Zurücksetzen des Passworts auf 8 Minuten fest.

Amazon Cognito console

So konfigurieren Sie die Dauer der Authentifizierungsablaufsitzung des App-Clients (AWS Management Console)

1. Wählen Sie auf der Registerkarte App integration (App-Integration) in Ihrem Benutzerpool den Namen Ihres App-Clients aus dem Container App clients and analytics (App-Clients und Analytik) aus.

2. Wählen Sie Bearbeiten im Container App-Client-Informationen aus.
3. Ändern Sie den Wert Authentication flow session duration (Dauer der Authentifizierungsablaufsitzung) auf die Gültigkeitsdauer (in Minuten), die Sie für SMS-MFA-Codes wünschen. Damit ändert sich auch die Zeit, die einem Benutzer zur Verfügung steht, um eine Authentifizierungsabfrage in Ihrem App-Client abzuschließen.
4. Wählen Sie Änderungen speichern aus.

Amazon Cognito API

So konfigurieren Sie die Dauer der Authentifizierungsablaufsitzung (Amazon-Cognito-API)

1. Bereiten Sie eine `UpdateUserPoolClient`-Anfrage mit Ihren vorhandenen Benutzerpool-Einstellungen aufgrund einer `DescribeUserPoolClient`-Anfrage vor. Ihre `UpdateUserPoolClient`-Anfrage muss alle vorhandenen App-Client-Eigenschaften enthalten.
2. Ändern Sie den `AuthSessionValidity`-Wert auf die Gültigkeitsdauer (in Minuten), die Sie für SMS-MFA-Codes wünschen. Damit ändert sich auch die Zeit, die einem Benutzer zur Verfügung steht, um eine Authentifizierungsabfrage in Ihrem App-Client abzuschließen.

Weitere Informationen zu App-Clients finden Sie unter App-Clients für Benutzerpools.

Sie können AWS Lambda Auslöser verwenden, um die Authentifizierung von Benutzern anzupassen. Diese Auslöser geben ihre eigenen Eingabeaufforderungen im Rahmen des Authentifizierungsablaufs aus und überprüfen sie.

Sie können auch den Admin-Authentifizierungsablauf für sichere Backend-Server verwenden. Mit dem Authentifizierungsablauf für die Benutzermigration können Sie die Migration von Benutzern ermöglichen, ohne dass Ihre Benutzer ihre Passwörter zurücksetzen müssen.

Sperrverhalten von Amazon Cognito bei fehlgeschlagenen Anmeldeversuchen

Nach fünf fehlgeschlagenen Anmeldeversuchen (ohne Authentifizierung oder mit IAM-Authentifizierung) mit einem Passwort sperrt Amazon Cognito den Benutzer eine Sekunde lang. Die Sperrdauer verdoppelt sich dann nach jedem weiteren fehlgeschlagenen Versuch bis zu einer maximalen Dauer von ca. 15 Minuten. Anmeldeversuche während einer Sperrperiode führen zu einer Ausnahme `Password attempts exceeded` und wirken sich nicht auf die Dauer nachfolgender Sperrperioden aus. Bei einer kumulativen Anzahl fehlgeschlagener Anmeldeversuche n, Ausnahmen

`Password attempts exceeded` nicht eingeschlossen, sperrt Amazon Cognito Ihren Benutzer für 2^(n-5) Sekunden. Um die Sperre zurückzusetzen (n=0), muss sich Ihr Benutzer danach entweder erfolgreich anmelden oder darf 15 Minuten lang keine Anmeldeversuche unternehmen. Änderungen an diesem Verhalten sind vorbehalten. Dieses Verhalten gilt nicht für benutzerdefinierte Challenges, es sei denn, diese führen auch eine passwortbasierte Authentifizierung durch.

Themen

Clientseitiger Authentifizierungsablauf

Der folgende Prozess funktioniert für clientseitige Benutzeranwendungen, die Sie mit AWS Amplify oder den AWS -SDKs erstellen.

1. Der Benutzer gibt den Benutzernamen und das Passwort in der App ein.
2. Die App ruft die `InitiateAuth`-Operation mit dem Benutzernamen und den SRP-Details (Secure Remote Password) des Benutzers auf.

   Diese API-Operation gibt die Authentifizierungsparameter zurück.

   > **Note**
   >
   > Die App generiert SRP-Details mit den Amazon Cognito SRP-Funktionen, die in AWS -SDKs integriert sind.

3. Die App ruft die `RespondToAuthChallenge`-Operation auf. Wenn der Aufruf erfolgreich ist, gibt Amazon Cognito die Token des Benutzers zurück. Damit ist der Authentifizierungsablauf abgeschlossen.

Wenn Amazon Cognito eine weitere Abfrage erfordert, gibt der Aufruf von `RespondToAuthChallenge` keine Token zurück. Stattdessen gibt der Aufruf eine Sitzung zurück.

4. Wenn `RespondToAuthChallenge` eine Sitzung zurückgibt, ruft die App `RespondToAuthChallenge` erneut auf, dieses Mal mit der Sitzung und der Abfrageantwort (z. B. MFA-Code).

Serverseitiger Authentifizierungsablauf

Wenn Sie nicht über eine Benutzeranwendung verfügen, sondern ein sicheres Java-, Ruby- oder Node.js-Backend oder eine serverseitige App verwenden, können Sie die authentifizierte, serverseitige API für Amazon-Cognito-Benutzerpools nutzen.

Für serverseitige Apps ist die Benutzerpoolauthentifizierung mit der Authentifizierung für clientseitige Apps vergleichbar, mit Ausnahme von Folgendem:

- Die serverseitige App ruft die `AdminInitiateAuth`-API-Operation auf (anstelle von `InitiateAuth`). Für diesen Vorgang sind AWS Anmeldeinformationen mit Berechtigungen erforderlich, die `cognito-idp:AdminInitiateAuth` und enthaltencognito-idp:AdminRespondToAuthChallenge. Die Operation gibt die erforderlichen Authentifizierungsparameter zurück.
- Nachdem die serverseitige App über die Authentifizierungsparameter verfügt, ruft sie die `AdminRespondToAuthChallenge`-API-Operation (anstelle von `RespondToAuthChallenge`) auf. Der `AdminRespondToAuthChallenge` API-Vorgang ist nur erfolgreich, wenn Sie AWS Anmeldeinformationen angeben.

Weitere Informationen zum Signieren von Amazon Cognito-API-Anforderungen mit - AWS Anmeldeinformationen finden Sie unter Signaturprozess mit Signaturversion 4 in der AWS Allgemeinen Referenz zu .

Die - `AdminInitiateAuth` und -`AdminRespondToAuthChallenge`API-Operationen können keine username-and-password Benutzeranmeldeinformationen für die Administratoranmeldung akzeptieren, es sei denn, Sie aktivieren dies ausdrücklich auf eine der folgenden Arten:

- Nehmen Sie `ALLOW_ADMIN_USER_PASSWORD_AUTH` (ehemals als `ADMIN_NO_SRP_AUTH` bezeichnet) in den Parameter `ExplicitAuthFlow` beim Aufrufen von `CreateUserPoolClient` oder `UpdateUserPoolClient` auf.

- Fügen Sie `ALLOW_ADMIN_USER_PASSWORD_AUTH` der Liste der Authentifizierungsabläufe für Ihren App-Client hinzu. Konfigurieren Sie App-Clients auf dem Tab App integration (App-Integration) in Ihrem Benutzerpool unter App clients and analytics (App-Clients und Analytik). Weitere Informationen finden Sie unter App-Clients für Benutzerpools.

Benutzerdefinierter Authentifizierungsablauf

Amazon Cognito-Benutzerpools ermöglichen auch die Verwendung benutzerdefinierter Authentifizierungsabläufe, mit denen Sie mithilfe von AWS Lambda Auslösern ein auf Aufforderungen/Antworten basierendes Authentifizierungsmodell erstellen können.

> **Note**
>
> Sie können die erweiterten Sicherheitsfunktionen nicht für kompromittierte Anmeldeinformationen und die adaptive Authentifizierung mit benutzerdefinierten Authentifizierungsabläufen verwenden. Weitere Informationen finden Sie unter Hinzufügen erweiterter Sicherheit zu einem Benutzerpool..

Der benutzerdefinierte Authentifizierungsfluss ermöglicht kundenspezifische Aufforderungs- und Antwortzyklen, um unterschiedliche Anforderungen zu erfüllen. Der Ablauf beginnt mit dem Aufruf der `InitiateAuth`-API-Operation, die den zu verwendenden Authentifizierungstyp angibt und die anfänglichen Authentifizierungsparameter bereitstellt. Amazon Cognito antwortet auf den `InitiateAuth`-Aufruf mit einer der folgenden Arten von Informationen:

- Mit einer Aufforderung für den Benutzer sowie einer Sitzung und Parametern.
- Mit einem Fehler, wenn der Benutzer nicht authentifiziert werden konnte
- Mit einem ID-, Zugriffs- und Aktualisierungstoken, wenn die angegebenen Parameter im `InitiateAuth`-Aufruf ausreichen, um den Benutzer anzumelden. (Normalerweise muss der Benutzer oder die App zuerst eine Herausforderung beantworten, aber Ihr benutzerdefinierter Code muss dies bestimmen.)

Wenn Amazon Cognito auf den `InitiateAuth`-Aufruf mit einer Aufforderung antwortet, sammelt die App weitere Eingaben und ruft die `RespondToAuthChallenge`-Operation auf. Dieser Aufruf liefert die Antworten auf die Aufforderungen und gibt sie an die Sitzung zurück. Amazon Cognito reagiert auf den `RespondToAuthChallenge`-Aufruf ähnlich wie auf den `InitiateAuth`-Aufruf. Wenn sich der Benutzer angemeldet hat, stellt Amazon Cognito Token bereit. Wenn der Benutzer hingegen

nicht angemeldet ist, zeigt Amazon Cognito eine weitere Aufforderung oder einen Fehler an. Wenn Amazon Cognito eine weitere Aufforderung zurückgibt, wiederholt sich die Sequenz und die App ruft `RespondToAuthChallenge` auf, bis sich der Benutzer erfolgreich angemeldet hat oder ein Fehler angezeigt wird. Weitere Details finden Sie in der API-Dokumentation für die API-Operationen `InitiateAuth` und `RespondToAuthChallenge`.

Integrierter Authentifizierungsablauf und Aufforderungen

Amazon Cognito enthält integrierte `AuthFlow`- und `ChallengeName`-Werte, so dass ein standardmäßiger Authentifizierungsablauf einen Benutzernamen und ein Passwort über das Secure Remote Password (SRP)-Protokoll validieren kann. Die AWS SDKs bieten integrierte Unterstützung für diese Flows mit Amazon Cognito .

Der Flow beginnt mit dem Senden von `USER_SRP_AUTH` als `AuthFlow` an `InitiateAuth`. Sie können auch `USERNAME`- und `SRP_A`-Werte in `AuthParameters` senden. Wenn der `InitiateAuth`-Aufruf erfolgreich ist, enthält die Antwort `PASSWORD_VERIFIER` als `ChallengeName` und `SRP_B` in den Herausforderungsparametern. Die App ruft dann `RespondToAuthChallenge` mit dem `PASSWORD_VERIFIER` `ChallengeName` und den erforderlichen Parametern in `ChallengeResponses` auf. Wenn der Aufruf von `RespondToAuthChallenge` erfolgreich ist und sich der Benutzer anmeldet, stellt Amazon Cognito Token aus. Wenn Sie die Multi-Faktor-Authentifizierung (MFA) für den Benutzer aktiviert haben, gibt Amazon Cognito die `ChallengeName` von `SMS_MFA` aus. Die App kann den erforderlichen Code durch einen anderen Anruf an `RespondToAuthChallenge` ausstellen.

Benutzerdefinierter Authentifizierungsablauf und Aufforderungen

Eine App kann einen benutzerdefinierten Authentifizierungsfluss durch Aufrufen von `InitiateAuth` mit `CUSTOM_AUTH` als `Authflow` initiieren. Bei einem benutzerdefinierten Authentifizierungsablauf steuern drei Lambda-Auslöser die Aufforderungen und die Verifizierung der Antworten.

- Der `DefineAuthChallenge`-Lambda-Auslöser verwendet ein Sitzungs-Array mit früheren Aufforderungen und Antworten als Eingabe. Anschließend generiert er den nächsten Aufforderungsnamen und boolesche Werte, die angeben, ob der Benutzer authentifiziert ist und ihm Token gewährt werden können. Dieser Lambda-Auslöser ist ein Zustandsautomat, der den Weg des Benutzers durch die Aufforderungen steuert.
- Der `CreateAuthChallenge`-Lambda-Auslöser verwendet einen Aufforderungsnamen als Eingabe und generiert die Aufforderung sowie Parameter zur Bewertung der Antwort. Wenn `DefineAuthChallenge` `CUSTOM_CHALLENGE` als nächste Aufforderung zurückgibt, ruft der

Authentifizierungsablauf `CreateAuthChallenge` an. Der `CreateAuthChallenge`-Lambda-Trigger übergibt den nächsten Aufforderungstyp im Metadatenparameter der Aufforderung.

- Die `VerifyAuthChallengeResponse`-Lambda-Funktion bewertet die Antwort und gibt einen booleschen Wert zurück, der angibt, ob die Antwort gültig war.

Ein benutzerdefinierter Authentifizierungsablauf kann auch eine Kombination aus integrierten Aufforderungen, z. B. SRP-Passwortverifizierung und MFA via SMS verwenden. Er kann auch benutzerdefinierte Aufforderungen wie CAPTCHA oder geheime Fragen verwenden.

Verwenden der SRP-Passwortverifizierung im benutzerdefinierten Authentifizierungsablauf

Wenn Sie SRP in einen benutzerdefinierten Authentifizierungsablauf einbeziehen möchten, müssen Sie mit SRP beginnen.

- Um die SRP-Passwortüberprüfung in einem benutzerdefinierten Ablauf zu initiieren, ruft die App `InitiateAuth` mit `CUSTOM_AUTH` als `Authflow` auf. Die Anfrage Ihrer App enthält in der `AuthParameters`-Karte `SRP_A:` (den SRP-A-Wert) und `CHALLENGE_NAME: SRP_A`.
- Der `CUSTOM_AUTH`-Ablauf ruft den `DefineAuthChallenge`-Lambda-Auslöser mit einer anfänglichen Sitzung von `challengeName: SRP_A` und `challengeResult: true` auf. Ihre Lambda-Funktion antwortet mit `challengeName: PASSWORD_VERIFIER`, `issueTokens: false` und `failAuthentication: false`.
- Die App muss als Nächstes `RespondToAuthChallenge` mit `challengeName: PASSWORD_VERIFIER` und die anderen Parameter aufrufen, die für SRP in der `challengeResponses`-Zuordnung erforderlich sind.
- Wenn Amazon Cognito das Passwort verifiziert, wird `RespondToAuthChallenge` den `DefineAuthChallenge`-Lambda-Auslöser mit einer zweiten Sitzung von `challengeName: PASSWORD_VERIFIER` und `challengeResult: true` aufrufen. Nun kann der Lambda-Auslöser `DefineAuthChallenge` mit `challengeName: CUSTOM_CHALLENGE` reagieren, um die benutzerdefinierte Aufforderung zu starten.
- Wenn MFA für einen Benutzer aktiviert ist und Amazon Cognito das Passwort überprüft hat, wird Ihr Benutzer aufgefordert, die Einrichtung oder Anmeldung mit MFA vorzunehmen.

> **Note**
>
> Die von Amazon Cognito gehostete Anmeldewebseite kann Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen nicht aktivieren.

Weitere Informationen zu den Lambda-Auslösern, einschließlich Beispielcode, finden Sie unter Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern.

Ablauf der Administratorauthentifizierung

Die bewährte Methode für die Authentifizierung ist die Verwendung der in Benutzerdefinierter Authentifizierungsablauf beschriebenen API-Operationen mit SRP zur Passwortverifizierung. Die AWS SDKs verwenden diesen Ansatz, und dieser Ansatz hilft ihnen bei der Verwendung von SRP. Wenn Sie jedoch SRP-Berechnungen vermeiden möchten, ist ein alternativer Satz von Administrator-API-Operationen für sichere Backend-Server verfügbar. Verwenden Sie für diese Backend-Administrator-Implementierungen `AdminInitiateAuth` anstelle von `InitiateAuth`. Verwenden Sie auch `AdminRespondToAuthChallenge` anstelle von `RespondToAuthChallenge`. Da Sie das Passwort als Klartext übermitteln können, brauchen Sie bei der Verwendung dieser Vorgänge keine SRP-Berechnungen durchzuführen. Ein Beispiel:

```
AdminInitiateAuth Request {
    "AuthFlow":"ADMIN_USER_PASSWORD_AUTH",
    "AuthParameters":{
        "USERNAME":"<username>",
        "PASSWORD":"<password>"
        },
    "ClientId":"<clientId>",
    "UserPoolId":"<userPoolId>"
}
```

Diese Operationen zur Administratorauthentifizierung benötigen Entwickleranmeldeinformationen und verwenden den Signaturprozess mittels AWS Signature Version 4 (SigV4). Diese Operationen sind in Standard- AWS -SDKs verfügbar, einschließlich Node.js, was für Lambda-Funktionen praktisch ist. Damit diese Operationen verwendet werden können und Passwörter als Klartext akzeptieren, müssen Sie sie für die App in der Konsole aktivieren. Alternativ können Sie `ADMIN_USER_PASSWORD_AUTH` für den `ExplicitAuthFlow`-Parameter in Aufrufen von `CreateUserPoolClient` oder `UpdateUserPoolClient` übergeben. Die `InitiateAuth`- und `RespondToAuthChallenge`-Operationen akzeptieren `ADMIN_USER_PASSWORD_AUTH` `AuthFlow` nicht.

In der `AdminInitiateAuth`-Antwort `ChallengeParameters` enthält das Attribut `USER_ID_FOR_SRP` – sofern vorhanden – den tatsächlichen Benutzernamen des Benutzers und keinen Alias (wie etwa E-Mail-Adresse oder Telefonnummer). In dem Aufruf an `AdminRespondToAuthChallenge` in den `ChallengeResponses` müssen Sie diesen Benutzernamen im Parameter `USERNAME` übergeben.

> **Note**
>
> Da der Ablauf der Administratorauthentifizierung für Implementierungen zur Verwaltung über das Backend verwendet wird, unterstützt er die Geräteverfolgung nicht. Wenn Sie die Geräteverfolgung aktiviert haben, ist die Administratorauthentifizierung erfolgreich, aber jeder Aufruf zum Aktualisieren des Zugriffstokens schlägt fehl.

Ablauf der Authentifizierung für die Benutzermigration

Ein Lambda-Auslöser für die Benutzermigration hilft bei der Migration von Benutzern aus einem Legacy-Benutzerverwaltungssystem in Ihren Benutzerpool. Wenn Sie den `USER_PASSWORD_AUTH`-Authentifizierungsablauf auswählen, müssen Benutzer ihre Passwörter während der Benutzermigration nicht zurücksetzen. Dieser Ablauf sendet während der Authentifizierung die Passwörter Ihrer Benutzer über eine verschlüsselte SSL-Verbindung an den Service.

Wenn Sie alle Ihre Benutzer migriert haben, wechseln Sie zu dem sichereren SRP-Ablauf. Der SRP-Ablauf sendet keine Passwörter über das Netzwerk.

Weitere Informationen zu Lambda-Auslösern finden Sie unter Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern.

Weitere Informationen über die Migration von Benutzern über einen Lambda-Auslöser finden Sie unter Importieren von Benutzern in Benutzerpools mit einem Lambda-Auslöser für die Benutzermigration.

App-Clients für Benutzerpools

Ein Benutzerpool-App-Client ist eine Konfiguration innerhalb eines Benutzerpools, die mit einer Mobil- oder Webanwendung interagiert, welche sich bei Amazon Cognito authentifiziert. App-Clients können authentifizierte und nicht authentifizierte API-Operationen aufrufen und einige oder alle Attribute Ihrer Benutzer lesen oder ändern. Ihre App muss sich bei Operationen gegenüber dem

App-Client identifizieren, um sich zu registrieren, anzumelden und mit vergessenen Passwörtern umzugehen. Diese API-Anforderungen müssen die Selbstidentifikation mit einer App-Client-ID und die Autorisierung mit einem optionalen Client-Geheimnis beinhalten. Sie müssen App-Client-IDs oder -Geheimnisse sichern, sodass nur autorisierte Client-Apps diese nicht authentifizierten Operationen aufrufen können. Wenn Sie Ihre App so konfigurieren, dass authentifizierte API-Anfragen mit AWS Anmeldeinformationen signiert werden, müssen Sie Ihre Anmeldeinformationen außerdem vor Benutzereinsicht schützen.

Sie können mehrere Apps für einen Benutzerpool erstellen. Ein App-Client kann mit der Codeplattform einer App oder einem separaten Mandanten in Ihrem Benutzerpool verknüpft sein. Beispielsweise können Sie eine App für eine serverseitige Anwendung und eine Android-App erstellen. Jede Anwendung hat eine eigene App-Client-ID.

Arten von App-Clients

Wenn Sie einen App-Client in Amazon Cognito erstellen, können Sie Optionen basierend auf den Standard-OAuth-Clienttypen öffentlicher Client und vertraulicher Client vorab belegen. Konfigurieren eines vertraulichen Clients mit einem Clientschlüssel. Weitere Informationen zu Client-Arten finden Sie unter IETF RFC 6749 #2.1.

Öffentlicher Client

Ein öffentlicher Client läuft in einem Browser oder auf einem mobilen Gerät. Da er keine vertrauenswürdigen serverseitigen Ressourcen hat, hat es auch keinen Clientschlüssel.

Vertraulicher Client

Ein vertraulicher Client verfügt über serverseitige Ressourcen, denen mit einem Clientschlüssel für nicht authentifizierte API-Vorgänge vertraut werden kann. Die App wird möglicherweise als Daemon- oder Shell-Skript auf Ihrem Backend-Server ausgeführt.

Clientschlüssel

Ein geheimer Client-Schlüssel ist eine feste Zeichenfolge, die Ihre App in allen API-Anfragen an den App-Client verwenden muss. Ihr App-Client muss einen Clientschlüssel haben, um `client_credentials`-Erteilungen auszuführen. Weitere Informationen finden Sie unter IETF RFC 6749 #2.3.1.

Nach der Erstellung einer App können die Schlüssel nicht mehr geändert werden. Sie können eine neue App mit einem neuen Schlüssel erstellen, wenn Sie den Schlüssel rotieren möchten. Ferner

sind Sie in der Lage, eine Anwendung zu löschen und so den Zugriff über Apps zu blockieren, welche die jeweilige App-Client-ID verwenden.

Sie können einen vertraulichen Client und einen Clientschlüssel mit einer öffentlichen App verwenden. Verwenden Sie einen CloudFront Amazon-Proxy, um einen `SECRET_HASH` In-Transit hinzuzufügen. Weitere Informationen finden Sie im AWS Blog unter Schützen öffentlicher Clients für Amazon Cognito mithilfe eines CloudFront Amazon-Proxys.

JSON-Webtoken

Amazon-Cognito-App-Clients können JSON-Webtoken (JWTs) der folgenden Arten ausgeben.

Identitätstoken (ID)

Eine überprüfbare Aussage, dass Ihr Benutzer in Ihrem Benutzerpool authentifiziert wurde. OpenID Connect (OIDC) hat die ID-Token-Spezifikation zu den in OAuth 2.0 definierten Standards für Zugriffs- und Aktualisierungstoken hinzugefügt. Das ID-Token enthält Identitätsinformationen wie Benutzerattribute, die Ihre App verwenden kann, um ein Benutzerprofil zu erstellen und Ressourcen bereitzustellen. Weitere Informationen finden Sie unter Verwenden des ID-Tokens.

Zugriffstoken

Eine überprüfbare Erklärung zu den Zugriffsrechten Ihres Benutzers. Das Zugriffstoken enthält Bereiche, ein Feature von OIDC und OAuth 2.0. Ihre App kann Back-End-Ressourcen Bereiche präsentieren und nachweisen, dass Ihr Benutzerpool einen Benutzer oder eine Maschine autorisiert hat, um auf Daten von einer API oder auf ihre eigenen Benutzerdaten zuzugreifen. Ein Zugriffstoken mit benutzerdefinierten Bereichen, häufig aus einer Erteilung von M2M-Client-Anmeldeinformationen, autorisiert den Zugriff auf einen Ressourcenserver. Weitere Informationen finden Sie unter Verwenden des Zugriffstokens.

Aktualisierungs-Token

Eine verschlüsselte Erklärung zur Erstauthentifizierung, die Ihre App Ihrem Benutzerpool präsentieren kann, wenn die Token Ihrer Benutzer ablaufen. Eine Anforderung für ein Aktualisierungstoken gibt neue, noch nicht abgelaufene Zugriffs- und ID-Token zurück. Weitere Informationen finden Sie unter Verwenden des Aktualisierungs-Tokens.

Sie können den Ablauf dieser Token für jeden App-Client auf der Registerkarte App-Integration Ihres Benutzerpools in der Amazon-Cognito-Konsole festlegen.

App-Client-Bedingungen

Die folgenden Begriffe sind verfügbare Eigenschaften von App-Clients in der Amazon-Cognito-Konsole.

Zulässige Rückruf-URLs

Eine Rückruf-URL gibt an, wohin der Benutzer nach erfolgreicher Anmeldung umgeleitet werden soll. Wählen Sie mindestens eine Rückruf-URL aus. Die Rückruf-URL muss:

- Eine absolute URI sein.
- Vorab bei einem Client registriert worden sein.
- Sie darf keine Fragment-Komponente enthalten.

Weitere Informationen finden Sie unter OAuth 2.0 – redirection endpoint (OAuth 2.0 – Umleitungsendpunkt).

Amazon Cognito fordert `HTTPS` statt `HTTP`, außer nur für Testzwecke für `http://localhost`.

App-Callback-URLs wie `myapp://example` werden ebenfalls unterstützt.

Zulässige Abmelde-URLs

Eine Abmelde-URL gibt an, wohin Ihr Benutzer nach der Abmeldung umgeleitet werden soll.

Festlegen von Lese- und Schreibberechtigungen

Ihr Benutzerpool kann viele Kunden haben, von denen jeder seinen eigenen App-Client hat und IdPs. Sie können Ihren App-Client so konfigurieren, dass er nur Lese- und Schreibzugriff auf die Benutzerattribute hat, die für die App relevant sind. In Fällen wie der machine-to-machine (M2M-) Autorisierung können Sie Zugriff auf keines Ihrer Benutzerattribute gewähren.

Überlegungen zur Konfiguration der Lese- und Schreibberechtigungen für Attribute

- Wenn Sie einen App-Client erstellen und die Lese- und Schreibberechtigungen für Attribute nicht anpassen, gewährt Amazon Cognito Lese- und Schreibberechtigungen für alle Benutzerpool-Attribute.
- Sie können Schreibzugriff für unveränderliche benutzerdefinierte Attribute gewähren. Sie können einen Wert nur dann in ein unveränderliches benutzerdefiniertes Attribut schreiben, wenn Sie einen Benutzer erstellen oder anmelden. Danach können Sie keine Werte mehr in unveränderliche benutzerdefinierte Attribute für den Benutzer schreiben.
- App-Clients müssen Schreibzugriff auf die erforderlichen Attribute in Ihrem Benutzerpool haben. Die Amazon-Cognito-Konsole legt die erforderlichen Attribute automatisch als schreibbar fest.

- Sie können einem App-Client nicht erlauben, Schreibzugriff auf `email_verified` oder `phone_number_verified` zu haben. Ein Benutzerpool-Administrator kann diese Werte ändern. Ein Benutzer kann den Wert dieser Attribute nur durch Attributüberprüfung ändern.

Authentifizierungsabläufe

Die Methoden, die Ihr App-Client für die Anmeldung zulässt. Ihre App kann die Authentifizierung mit Benutzername und Passwort, Secure Remote Password (SRP), benutzerdefinierter Authentifizierung mit Lambda-Triggern und Tokenaktualisierung unterstützen. Verwenden Sie als bewährte Sicherheitsmethode die SRP-Authentifizierung als primäre Anmeldemethode. Die gehostete Benutzeroberfläche meldet Benutzer automatisch mit SRP an.

Benutzerdefinierte Bereiche

Ein benutzerdefinierter Bereich ist ein Bereich, den Sie unter Resource Servers (Ressourcen-Server) für Ihre eigenen Ressourcenserver definieren. Das Format ist *`resource-server-identifier`*/*`scope`*. Siehe Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern.

Standard-Umleitungs-URI

Ersetzt den `redirect_uri` Parameter in Authentifizierungsanfragen für Benutzer durch Drittanbieter IdPs. Konfigurieren Sie diese App-Client-Einstellung mit dem `DefaultRedirectURI` Parameter einer CreateUserPoolClientoder UpdateUserPoolClientAPI-Anfrage. Diese URL muss auch Mitglied des `CallbackURLs` für Ihren App-Client sein. Amazon Cognito leitet authentifizierte Sitzungen an diese URL weiter, wenn:

1. Ihrem App-Client ist ein Identitätsanbieter zugewiesen und mehrere Callback-URLs definiert. Ihr Benutzerpool leitet Authentifizierungsanfragen an den Autorisierungsserver an den Standard-Umleitungs-URI weiter, wenn sie keinen `redirect_uri` Parameter enthalten.
2. Ihrem App-Client ist ein Identitätsanbieter zugewiesen und eine Callback-URL definiert. In diesem Szenario ist es nicht erforderlich, eine Standard-Callback-URL zu definieren. Anfragen, die keinen `redirect_uri` Parameter enthalten, leiten zu der einen verfügbaren Callback-URL weiter.

Identitätsanbieter

Sie können einige oder alle externen Identitätsanbieter (IdPs) Ihres Benutzerpools auswählen, um Ihre Benutzer zu authentifizieren. Ihr App-Client kann auch nur lokale Benutzer in Ihrem Benutzerpool authentifizieren. Wenn Sie Ihrem App-Client einen IdP hinzufügen, können Sie Autorisierungslinks für den IdP generieren und ihn auf Ihrer Anmeldeseite der

gehosteten Benutzeroberfläche anzeigen. Sie können mehrere zuweisen IdPs, müssen jedoch mindestens einen zuweisen. Weitere Informationen zur Verwendung von Extern IdPs finden Sie unterHinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter.

OpenID-Connect-Bereiche

Wählen Sie einen oder mehrere der folgenden `OAuth`-Bereiche aus, um die Zugriffsprivilegien, die für Zugriffs-Token eingestellt werden können, festzulegen.

- Der `openid`-Bereich gibt an, dass Sie ein ID-Token und die eindeutige ID eines Benutzers abrufen möchten. Außerdem werden alle oder einige Benutzerattribute angefordert, je nachdem welche zusätzlichen Bereiche in der Anfrage enthalten sind. Amazon Cognito gibt kein ID-Token zurück, es sei denn, Sie fordern den `openid`-Bereich an. Der `openid`-Bereich autorisiert Ansprüche auf strukturelle ID-Tokens wie Ablauf und Schlüssel-ID und bestimmt die Benutzerattribute, die Sie in einer Antwort von UserInfo-Endpunkt erhalten.
  - Wenn `openid` der einzige angeforderte Bereich ist, füllt Amazon Cognito das ID-Token mit allen Benutzerattributen, die der aktuelle App-Client lesen kann. Die `userInfo`-Antwort auf ein Zugriffs-Token mit nur diesem Bereich gibt alle Benutzerattribute zurück.
  - Wenn Sie `openid` mit anderen Bereichen wie `phone`, `email` oder `profile` anfordern, geben das ID-Token und `userInfo` die eindeutige ID des Benutzers und die durch die zusätzlichen Bereiche definierten Attribute zurück.
- Der `phone`-Bereich genehmigt den Zugriff auf `phone_number` und `phone_number_verified`-Anfragen. Dieser Bereich kann ausschließlich mit dem `openid`-Bereich beantragt werden.
- Der `email`-Bereich genehmigt den Zugriff auf `email` und `email_verified`-Anfragen. Dieser Bereich kann ausschließlich mit dem `openid`-Bereich beantragt werden.
- Der `aws.cognito.signin.user.admin` Geltungsbereich gewährt Zugriff auf API-Operationen für Amazon Cognito Cognito-Benutzerpools, für die Zugriffstoken erforderlich sind, z. B. UpdateUserAttributesund VerifyUserAttribute.
- Der `profile`-Bereich gewährt Zugriff auf alle Benutzerattribute, die vom Client gelesen werden können. Dieser Bereich kann ausschließlich mit dem `openid`-Bereich beantragt werden.

Weitere Informationen über Bereiche finden Sie in der Liste der Standard-OIDC-Bereiche.

Arten von OAuth-Erteilungen

Eine OAuth-Erteilung ist eine Authentifizierungsmethode, mit der Benutzerpool-Token abgerufen werden. Amazon Cognito unterstützt die folgenden Arten von Erteilungen. Um diese OAuth-Erteilungen in Ihre App zu integrieren, müssen Sie Ihrem Benutzerpool eine Domain hinzufügen.

Erteilung des Autorisierungscodes

Durch die Gewährung des Autorisierungscodes wird ein Code generiert, den Ihre App bei der/beim Token-Endpunkt gegen Benutzerpool-Token eintauschen kann. Wenn Sie einen Autorisierungscode austauschen, erhält Ihre App die ID, den Zugriff und die Aktualisierungs-Token. Dieser OAuth-Prozess findet ebenso wie die implizite Erteilung in den Browsern Ihrer Benutzer statt. Die Gewährung eines Autorisierungscodes ist die sicherste Art von Erteilung, die Amazon Cognito bietet, da Token in den Sitzungen Ihrer Benutzer nicht sichtbar sind. Stattdessen generiert Ihre App die Anfrage, die Token zurückgibt, und kann sie im geschützten Speicher zwischenspeichern. Weitere Informationen finden Sie unter Autorisierungscode in IETF RFC 6749 #1.3.1.

> **Note**
>
> Als bewährte Sicherheitsmethode in Apps für öffentliche Clients sollten Sie nur den OAuth-Prozess für die Gewährung von Autorisierungscodes aktivieren und Proof Key for Code Exchange (PKCE) implementieren, um den Austausch von Token einzuschränken. Mit PKCE kann ein Client nur dann einen Autorisierungscode austauschen, wenn er dem Token-Endpunkt denselben geheimen Schlüssel zur Verfügung gestellt hat, das in der ursprünglichen Authentifizierungsanfrage angegeben wurde. Weitere Informationen zu PKCE finden Sie unter IETF RFC 7636.

Implizite Erteilung

Durch die implizite Erteilung wird der Browsersitzung Ihres Benutzers direkt aus der/dem Autorisieren des Endpunkts ein Zugriffs- und ID-Token, jedoch kein Aktualisierungstoken, zur Verfügung gestellt. Durch eine implizite Erteilung muss keine separate Anfrage an den Token-Endpunkt mehr gestellt werden. Sie ist jedoch nicht mit PKCE kompatibel und gibt keine Aktualisierungstoken zurück. Diese Art der Erteilung eignet sich für Testszenarien und Anwendungsarchitekturen, bei denen Autorisierungscode-Erteilungen nicht abgeschlossen werden können. Weitere Informationen finden Sie unter Implizite Erteilung in IETF RFC 6749 #1.3.2. Sie können die Autorisierungscode-Erteilung und die implizite Codeerteilung in einem App-Client aktivieren und dann beide Erteilungen nach Bedarf verwenden.

Erteilung von Client-Anmeldeinformationen

Die Gewährung der Kundenanmeldedaten ist für machine-to-machine (M2M-) Kommunikation vorgesehen. Autorisierungscode- und implizite Erteilungen geben Token an authentifizierte

menschliche Benutzer aus. Kundenanmeldeinformationen gewähren eine bereichsabhängige Autorisierung von einem nicht interaktiven System zu einer API. Ihre App kann Kundenanmeldeinformationen direkt vom Token-Endpunkt anfordern und erhält ein Zugriffstoken. Weitere Informationen finden Sie unter Client-Anmeldeinformationen in IETF RFC 6749 #1.3.4. Sie können die Gewährung von Client-Anmeldeinformationen nur in App-Clients aktivieren, die über einen geheimen Client-Schlüssel verfügen und die weder Autorisierungscode- noch implizite Erteilungen unterstützen.

> **Note**
>
> Da Sie den Prozess für Client-Anmeldeinformationen nicht als Benutzer aufrufen, können Sie mit dieser Erteilung nur benutzerdefinierte Bereiche zu Zugriffstoken hinzufügen. Ein benutzerdefinierter Bereich ist ein Bereich, den Sie für Ihre eigenen Ressourcenserver definieren. Standardbereiche wie `openid` und `profile` gelten nicht für nichtmenschliche Benutzer.
>
> Da es sich bei ID-Token um eine Überprüfung von Benutzerattributen handelt, sind sie für die M2M-Kommunikation nicht relevant und werden nicht durch Erteilungen für Kundenanmeldeinformationen ausgestellt. Siehe Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern.

Bei Zuschüssen mit Kundendaten fallen zusätzliche Kosten auf Ihre AWS Rechnung an. Weitere Informationen finden Sie unter Amazon Cognito – Preise.

Erstellen eines App-Clients

AWS Management Console

So erstellen Sie einen App-Client (Konsole)

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus.
5. Wählen Sie unter App clients (App-Clients) Create an app client (App-Client erstellen) aus.

6. Wählen Sie einen App type (Anwendungstyp): Public client (Öffentlicher Client), Confidential client (Vertraulicher Client) oder Other (Sonstige) aus.
7. Geben Sie einen App-Client-Namen ein.
8. Wählen Sie Client-Geheimnis generieren aus, damit Amazon Cognito ein Client-Geheimnis für Sie erstellt. Clientgeheimnisse werden normalerweise mit vertraulichen Clients verknüpft.
9. Wählen Sie die Authentifizierungsabläufe aus, die Sie in Ihrem App-Client zulassen möchten.
10. Konfigurieren Sie die Authentication flow session duration (Dauer der Authentifizierungsablaufsitzung). Dies ist die Zeitdauer, die Ihren Benutzern für den Abschluss einer Authentifizierungsabfrage zur Verfügung steht, bevor das Sitzungstoken abläuft.
11. (Optional) Wenn Sie den Token-Ablauf konfigurieren möchten, führen Sie die folgenden Schritte aus:

    a. Geben Sie den Ablauf für Aktualisierungs-Token für den App-Client an. Der Standardwert lautet 30 Tage. Sie können dies in jeden Wert zwischen 1 Stunde und 10 Jahren ändern.

    b. Geben Sie den Ablauf für Zugriffs-Token für den App-Client an. Der Standardwert lautet 1 Stunde. Sie können ihn in jeden Wert zwischen 5 Minuten und 24 Stunden ändern.

    c. Geben Sie den Ablauf für ID-Token für den App-Client an. Der Standardwert lautet 1 Stunde. Sie können ihn in jeden Wert zwischen 5 Minuten und 24 Stunden ändern.

    > **Important**
    >
    > Wenn Sie die gehostete Benutzeroberfläche verwenden und Token für weniger als eine Stunde einrichten, kann der Benutzer Token basierend auf seinem Sitzungscookie abrufen, das derzeit auf eine Stunde festgelegt ist.

12. Wählen Sie aus, ob Sie für diesen App-Client die Token-Sperre aktivieren möchten. Dies erhöht die Größe der Token, die Amazon Cognito ausgibt.
13. Wählen Sie aus, ob Sie für diesen App-Client Fehler bei vorhandenen Benutzern verhindern aktivieren möchten. Amazon Cognito antwortet auf Anmeldeanfragen für nicht vorhandene Benutzer mit einer generischen Nachricht, die angibt, dass entweder der Benutzername oder das Passwort falsch waren.
14. Wenn Sie die gehostete Benutzeroberfläche mit diesem App-Client verwenden möchten, konfigurieren Sie die Einstellungen der gehosteten Benutzeroberfläche.

a. Geben Sie eine oder mehrere Erlaubte Callback-URLs ein. Dies sind die Web- oder App-URLs, an die Amazon Cognito Ihre Benutzer weiterleiten soll, nachdem sie die Authentifizierung abgeschlossen haben.

b. Geben Sie eine oder mehrere Erlaubte Abmelde-URLs ein. Dies sind URLs, die Ihre App bei Anfragen an den Logout-Endpunkt akzeptieren soll.

c. Wählen Sie einen oder mehrere Identitätsanbieter aus, mit denen Sie Benutzer für Ihre App anmelden können möchten. Sie können eine beliebige Kombination aus vorhandenen auswählen IdPs. Sie können Benutzer nur mit Ihrem Benutzerpool oder mit einem oder mehreren Drittanbietern authentifizieren, IdPs die Sie in Ihrem Benutzerpool konfiguriert haben.

d. Wählen Sie die OAuth-2.0-Erteilungstypen aus, die Ihr App-Client akzeptieren soll.

   - Wählen Sie Autorisierungscode erteilen aus, um Codes an Ihre App weiterzuleiten, die sie mit dem Token-Endpunkt gegen Tokens einlösen kann.
   - Wählen Sie Implizite Erteilung aus, um ID und Zugriffstokens direkt an Ihre App zu übergeben. Beim Ablauf bei der impliziten Erteilung werden Ihren Benutzern Tokens direkt zur Verfügung gestellt.
   - Wählen Sie Client-Anmeldeinformationen aus, um Zugriffstokens nicht basierend auf der Kenntnis der Benutzeranmeldeinformationen, sondern des Client-Geheimnis an Ihre App zu übergeben. Der Ablauf zur Erteilung von Client-Anmeldeinformationen und die Abläufe bei Autorisierungscode und bei der impliziten Erteilung schließen sich gegenseitig aus.

e. Wählen Sie die OpenID-Connect-Bereiche aus, die Sie für die Verwendung mit diesem App-Client autorisieren möchten. Über die Benutzerpool-API können Sie Zugriffstokens nur mit dem `aws.cognito.signin.user.admin`-Gültigkeitsbereich generieren. Für zusätzliche Bereiche müssen Sie Ihre Zugriffstokens beim Token-Endpunkt anfordern.

f. Wählen Sie die Benutzerdefinierte Bereiche aus, die Sie für diesen App-Client autorisieren möchten. Benutzerdefinierte Bereiche werden am häufigsten verwendet, um den Zugriff auf APIs von Drittanbietern zu autorisieren.

15. Konfigurieren Sie Lese- und Schreibberechtigungen zuweisen für diesen App-Client. Ihr App-Client kann über die Berechtigung verfügen, das gesamte oder eine begrenzte Teilmenge des Zuweisungsschemata Ihres Benutzerpools zu lesen und darin zu schreiben.
16. Wählen Sie Create app client.

17. Notieren Sie sich die Client-ID. Dies identifiziert den App-Client in Registrierungs- und Anmeldungsanfragen.

AWS CLI

```
aws cognito-idp create-user-pool-client --user-pool-id MyUserPoolID --client-
name myApp
```

> **Note**
>
> Verwenden Sie das JSON-Format für Callback- und Abmelde-URLs, um zu verhindern, dass die CLI sie als Remote-Parameterdateien behandelt:
>
> ```
> --callback-urls "["https://example.com"]"
> --logout-urls "["https://example.com"]"
> ```

Weitere Informationen finden Sie in der AWS CLI Befehlsreferenz: create-user-pool-client

Amazon Cognito user pools API

Generieren Sie eine CreateUserPoolClientAPI-Anfrage. Sie müssen einen Wert für alle Parameter angeben, die nicht auf einen Standardwert festgelegt werden sollen.

Aktualisierung eines Benutzerpool-App-Clients (AWS CLI und einer AWS API)

Geben Sie am den AWS CLI folgenden Befehl ein:

```
aws cognito-idp update-user-pool-client --user-pool-id  "MyUserPoolID" --client-id
 "MyAppClientID" --allowed-o-auth-flows-user-pool-client --allowed-o-auth-flows "code"
 "implicit" --allowed-o-auth-scopes "openid" --callback-urls "["https://example.com"]"
 --supported-identity-providers "["MySAMLIdP", "LoginWithAmazon"]"
```

Wenn der Befehl erfolgreich ist, wird eine Bestätigung AWS CLI zurückgegeben:

```
{
    "UserPoolClient": {
        "ClientId": "MyClientID",
        "SupportedIdentityProviders": [
```

```
            "LoginWithAmazon",
            "MySAMLIdP"
        ],
        "CallbackURLs": [
            "https://example.com"
        ],
        "AllowedOAuthScopes": [
            "openid"
        ],
        "ClientName": "Example",
        "AllowedOAuthFlows": [
            "implicit",
            "code"
        ],
        "RefreshTokenValidity": 30,
        "AuthSessionValidity": 3,
        "CreationDate": 1524628110.29,
        "AllowedOAuthFlowsUserPoolClient": true,
        "UserPoolId": "MyUserPoolID",
        "LastModifiedDate": 1530055177.553
    }
}
```

Weitere Informationen finden Sie in der AWS CLI Befehlsreferenz: update-user-pool-client.

AWS API: UpdateUserPoolClient

Informationen über einen Benutzerpool-App-Client (AWS CLI und eine AWS API) abrufen

```
aws cognito-idp describe-user-pool-client --user-pool-id MyUserPoolID --client-
id MyClientID
```

Weitere Informationen finden Sie in der AWS CLI Befehlsreferenz: describe-user-pool-client.

AWS API: DescribeUserPoolClient

Auflistung aller App-Client-Informationen in einem Benutzerpool (AWS CLI und einer AWS API)

```
aws cognito-idp list-user-pool-clients --user-pool-id "MyUserPoolID" --max-results 3
```

Weitere Informationen finden Sie in der AWS CLI Befehlsreferenz: list-user-pool-clients.

AWS API: ListUserPoolClients

Löschen eines Benutzerpool-App-Clients (AWS CLI und einer AWS API)

```
aws cognito-idp delete-user-pool-client --user-pool-id "MyUserPoolID" --client-id
 "MyAppClientID"
```

Weitere Informationen finden Sie in der AWS CLI Befehlsreferenz: delete-user-pool-client

AWS API: DeleteUserPoolClient

Verwendung von Benutzergeräten in Ihrem Benutzerpool

Wenn Sie Benutzer des lokalen Benutzerpools mit der Amazon Cognito-Benutzerpool-API anmelden, können Sie die Aktivitätsprotokolle dieser Benutzer aus den erweiterten Sicherheitsfunktionen jedem ihrer Geräte zuordnen und ihnen ermöglichen, die Multi-Faktor-Authentifizierung (MFA) zu überspringen, wenn sie sich auf einem vertrauenswürdigen Gerät befinden. Amazon Cognito fügt der Antwort auf jede Anmeldung, die noch keine Geräteinformationen enthält, einen Geräteschlüssel hinzu. Der Geräteschlüssel hat das Format *Region*_*UUID*. Mit einem Geräteschlüssel, einer Secure Remote Password (SRP)-Bibliothek und einem Benutzerpool, der die Geräteauthentifizierung ermöglicht, können Sie Benutzer in Ihrer App auffordern, dem aktuellen Gerät zu vertrauen, und bei der Anmeldung nicht mehr nach einem MFA-Code fragen.

Themen

- Einrichten von gespeicherten Geräten
- Geräteschlüssel abrufen
- Anmelden mit einem Gerät
- Geräte anzeigen, aktualisieren und vergessen

Einrichten von gespeicherten Geräten

Mit Amazon Cognito-Benutzerpools können Sie jedes Gerät Ihrer Benutzer mit einer eindeutigen Gerätekennung verknüpfen, also einem Geräteschlüssel. Wenn Sie den Geräteschlüssel vorlegen und die Geräteauthentifizierung bei der Anmeldung durchführen, können Sie zwei Funktionen nutzen.

1. Mit den erweiterten Sicherheitsfunktionen können Sie die Benutzeraktivitäten auf bestimmten Geräten zu Sicherheits- und Analysezwecken überwachen. Wenn sich Benutzer anmelden,

bietet Ihre App die Möglichkeit, alle Benutzer und ihre Geräte zu authentifizieren und Geräteinformationen zu ihren Aktivitätsprotokollen hinzuzufügen.

2. Die Gerätespeicherfunktion unterstützt auch einen Authentifizierungsvorgang für vertrauenswürdige Geräte, bei dem Ihre Benutzer wählen können, ob sie sich für den Zeitraum, der den Sicherheitsanforderungen Ihrer App entspricht, ohne MFA anmelden möchten. Wenn Sie einen Benutzer erneut auffordern möchten, einen MFA-Code einzugeben, können Sie dazu den gespeicherten Status seines Geräts ändern.

   Gespeicherte Geräte können MFA nur in Benutzerpools mit aktivem MFA überschreiben.

Wenn sich ein Benutzer mit einem gespeicherten Gerät anmeldet, müssen Sie während des Authentifizierungsvorgangs eine zusätzliche Geräteauthentifizierung durchführen. Weitere Informationen finden Sie unter Anmelden mit einem Gerät.

Konfigurieren Sie die Gerätespeicherung in Ihrem Benutzerpool auf der Registerkarte Anmeldeerfahrung unter Geräteverfolgung. Wenn Sie die Funktionalität der gespeicherten Geräte über die Amazon-Cognito-Konsole einrichten, haben Sie drei Optionen: Immer, Benutzerabonnement und Nein.

Don’t remember (Nicht speichern)

Ihr Benutzerpool fordert Benutzer nicht auf, sich an Geräte zu erinnern, wenn sie sich anmelden.

Always remember (Immer speichern)

Wenn Ihre App das Gerät eines Benutzers bestätigt, erinnert sich Ihr Benutzerpool immer an das Gerät und gibt bei zukünftigen erfolgreichen Geräteanmeldungen keine MFA-Aufforderung zurück.

User opt-in (Benutzer-Opt-in)

Wenn Ihre App das Gerät eines Benutzers bestätigt, unterdrückt Ihr Benutzerpool nicht automatisch eine MFA-Aufforderung. Fordern Sie Ihre Benutzer auf, auszuwählen, ob sie sich an ihr Gerät erinnern möchten.

Wenn Sie Always remember oder User Opt-In wählen, generiert Amazon Cognito jedes Mal, wenn sich ein Benutzer von einem unbekannten Gerät aus anmeldet, einen Gerätekennungsschlüssel und ein Geheimnis. Der Geräteschlüssel ist die erste Kennung, die Ihre App an Ihren Benutzerpool sendet, wenn Ihr Benutzer die Geräteauthentifizierung durchführt.

Bei jedem bestätigten Benutzergerät, unabhängig davon, ob es automatisch erinnert oder angemeldet wurde, können Sie den Gerätekennungsschlüssel und das Geheimnis verwenden, um ein Gerät bei jeder Benutzeranmeldung zu authentifizieren.

Sie können die Gerätespeichereinstellungen auch in einer API-Anfrage des Typs CreateUserPool oder UpdateUserPool für Ihren Benutzerpool konfigurieren. Weitere Informationen finden Sie unter der Eigenschaft DeviceConfiguration.

Die Benutzerpool-API von Amazon Cognito bietet zusätzliche Funktionen für gespeicherte Geräte.

1. ListDevices und AdminListDevices geben eine Liste der Geräteschlüssel und ihrer Metadaten für einen Benutzer zurück.
2. GetDevice und AdminGetDevice geben den Geräteschlüssel und die Metadaten für ein einzelnes Gerät zurück.
3. UpdateDeviceStatus und AdminUpdateDeviceStatus legen fest, ob das Gerät eines Benutzers gespeichert oder nicht gespeichert wird.
4. ForgetDevice und AdminForgetDevice entfernen das bestätigte Gerät eines Benutzers aus seinem Profil.

API-Operationen, deren Namen mit `Admin` beginnen, sind für die Verwendung in serverseitigen Apps vorgesehen und müssen mit IAM-Anmeldeinformationen autorisiert werden. Weitere Informationen finden Sie unter Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte.

Geräteschlüssel abrufen

Jedes Mal, wenn sich Ihr Benutzer mit der Benutzerpools-API anmeldet und keinen Geräteschlüssel als `DEVICE_KEY` in die Authentifizierungsparameter einbezieht, gibt Amazon Cognito in der Antwort einen neuen Geräteschlüssel zurück. Platzieren Sie den Geräteschlüssel in Ihrer öffentlichen clientseitigen App im App-Speicher, damit Sie ihn in zukünftige Anfragen aufnehmen können. Legen Sie in Ihrer vertraulichen serverseitigen App ein Browser-Cookie oder ein anderes clientseitiges Token mit dem Geräteschlüssel Ihres Benutzers fest.

Bevor sich Ihr Benutzer mit seinem vertrauenswürdigen Gerät anmelden kann, muss Ihre App den Geräteschlüssel bestätigen und zusätzliche Informationen bereitstellen. Generieren Sie eine ConfirmDevice-Anfrage an Amazon Cognito, die das Gerät Ihres Benutzers mit dem Geräteschlüssel, einem benutzerfreundlichen Namen, einer Kennwortverifizierung und einem Salt bestätigt. Wenn Sie Ihren Benutzerpool für die Opt-in-Geräteauthentifizierung konfiguriert haben, beantwortet Amazon

Cognito Ihre `ConfirmDevice`-Anfrage mit der Aufforderung, dass Ihr Benutzer wählen muss, ob das aktuelle Gerät gespeichert werden soll. Antworten Sie mit der Auswahl Ihres Benutzers in einer UpdateDeviceStatus-Anfrage.

Wenn Sie das Gerät Ihres Benutzers bestätigen, es aber nicht als gespeichert festlegen, speichert Amazon Cognito die Zuordnung, fährt aber mit der geräteunabhängigen Anmeldung fort, wenn Sie den Geräteschlüssel angeben. Geräte können Protokolle generieren, die für die Benutzersicherheit und die Fehlerbehebung nützlich sind. Ein bestätigtes Gerät, das nicht gespeichert wurde, nutzt die Anmeldefunktion nicht, wohl aber die Funktion der Sicherheitsüberwachungsprotokolle. Wenn Sie erweiterte Sicherheitsfunktionen für Ihren App-Client aktivieren und einen Geräte-Footprint in Ihrer Anfrage codieren, ordnet Amazon Cognito Benutzerereignisse dem bestätigten Gerät zu.

So rufen Sie einen neuen Geräteschlüssel ab

1. Starten Sie die Anmeldesitzung Ihres Benutzers mit einer InitiateAuth-API-Anfrage.
2. Beantworten Sie alle Authentifizierungsaufgaben mit RespondToAuthChallenge, bis Sie JSON-Webtoken (JWTs) erhalten, die die Anmeldesitzung Ihres Benutzers als abgeschlossen kennzeichnen.
3. Notieren Sie in Ihrer App die Werte, die Amazon Cognito in `NewDeviceMetadata` in seiner `RespondToAuthChallenge`- oder `InitiateAuth`-Antwort zurückgibt: `DeviceGroupKey` und `DeviceKey`.
4. Generieren Sie ein neues SRP-Geheimnis für Ihren Benutzer: eine Salt- und eine Passwortverifizierung. Diese Funktion ist in SDKs verfügbar, die SRP-Bibliotheken bereitstellen.
5. Fordern Sie den Benutzer zur Eingabe eines Gerätenamens auf oder generieren Sie einen anhand der Geräteeigenschaften des Benutzers.
6. Geben Sie das Zugriffstoken, den Geräteschlüssel, den Gerätenamen und das SRP-Geheimnis Ihres Benutzers in einer ConfirmDevice-API-Anfrage ein. Wenn in Ihrem Benutzerpool die Option Always remember aktiviert ist, ist die Registrierung Ihres Benutzers abgeschlossen.
7. Wenn Amazon Cognito `ConfirmDevice` mit `"UserConfirmationNecessary": true` beantwortet hat, fordern Sie Ihren Benutzer auf, auszuwählen, ob das Gerät gespeichert werden soll. Wenn er dies bestätigt, generieren Sie eine UpdateDeviceStatus-API-Anfrage mit dem Zugriffstoken sowie dem Geräteschlüssel Ihres Benutzers und `"DeviceRememberedStatus": "remembered"`.
8. Wenn Sie Amazon Cognito angewiesen haben, das Gerät zu speichern, wird der Benutzer bei der nächsten Anmeldung anstelle einer MFA-Anfrage mit einer `DEVICE_SRP_AUTH`-Aufgabe konfrontiert.

Anmelden mit einem Gerät

Nachdem Sie das Gerät eines Benutzers so konfiguriert haben, dass es gespeichert wird, verlangt Amazon Cognito nicht mehr, dass er einen MFA-Code übermittelt, wenn er sich mit demselben Geräteschlüssel anmeldet. Die Geräteauthentifizierung ersetzt lediglich die MFA-Authentifizierungsaufgabe durch eine Geräteauthentifizierungsaufgabe. Sie können Ihre Benutzer nicht nur mit Geräteauthentifizierung anmelden. Ihre Benutzer müssen zuerst die Authentifizierung mit ihrem Passwort oder einer benutzerdefinierten Aufgabe abschließen. Im Folgenden wird der Authentifizierungsprozess für einen Benutzer auf einem gespeicherten Gerät beschrieben.

Um die Geräteauthentifizierung in einem Verfahren durchzuführen, das Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen verwendet, übermitteln Sie in Ihrer InitiateAuth-API-Anfrage einen `DEVICE_KEY`-Parameter. Nachdem Ihr Benutzer alle Aufgaben erfolgreich gemeistert und die `CUSTOM_CHALLENGE`-Aufgabe den `issueTokens`-Wert `true` zurückgegeben hat, gibt Amazon Cognito eine letzte `DEVICE_SRP_AUTH`-Aufgabe zurück.

So melden Sie sich mit einem Gerät an

1. Rufen Sie den Geräteschlüssel Ihres Benutzers aus dem Client-Speicher ab.
2. Starten Sie die Anmeldesitzung Ihres Benutzers mit einer InitiateAuth-API-Anfrage. Wählen Sie einen `AuthFlow` aus `USER_SRP_AUTH`, `REFRESH_TOKEN_AUTH`, `USER_PASSWORD_AUTH` oder `CUSTOM_AUTH` aus. Fügen Sie in `AuthParameters` den Geräteschlüssel Ihres Benutzers zum `DEVICE_KEY`-Parameter hinzu und ergänzen Sie die anderen erforderlichen Parameter für das ausgewählte Anmeldeverfahren.
   a. Sie können `DEVICE_KEY` auch in den Parametern einer `PASSWORD_VERIFIER`-Antwort an eine Authentifizierungsaufgabe übermitteln.
3. Schließen Sie Aufgaben ab, bis Sie in der Antwort eine `DEVICE_SRP_AUTH`-Aufgabe erhalten.
4. Senden Sie in einer RespondToAuthChallenge-API-Anfrage einen `ChallengeName` von `DEVICE_SRP_AUTH` und Parameter für `USERNAME`,`DEVICE_KEY` und `SRP_A`.
5. Amazon Cognito reagiert mit einer `DEVICE_PASSWORD_VERIFIER`-Aufgabe. Diese Aufgabenantwort enthält Werte für `SECRET_BLOCK` und `SRP_B`.
6. Generieren Sie mit Ihrer SRP-Bibliothek die Parameter `PASSWORD_CLAIM_SIGNATURE`, `PASSWORD_CLAIM_SECRET_BLOCK`, `TIMESTAMP`, `USERNAME` und `DEVICE_KEY` und übermitteln Sie sie. Versenden Sie sie in einer zusätzlichen `RespondToAuthChallenge`-Anfrage.
7. Schließen Sie zusätzliche Aufgaben ab, bis Sie die JWTs des Benutzers erhalten.

Der folgende Pseudocode zeigt, wie Sie Werte für Ihre Antwort auf die `DEVICE_PASSWORD_VERIFIER`-Aufgabe berechnen.

```
PASSWORD_CLAIM_SECRET_BLOCK = SECRET_BLOCK
TIMESTAMP = Tue Sep 25 00:09:40 UTC 2018
PASSWORD_CLAIM_SIGNATURE = Base64(SHA256_HMAC(K_USER, DeviceGroupKey + DeviceKey +
 PASSWORD_CLAIM_SECRET_BLOCK + TIMESTAMP))
K_USER = SHA256_HASH(S_USER)
S_USER = (SRP_B - k * gx)(a + ux)
x = SHA256_HASH(salt + FULL_PASSWORD)
u = SHA256_HASH(SRP_A + SRP_B)
k = SHA256_HASH(N + g)
```

Geräte anzeigen, aktualisieren und vergessen

Mit der Amazon Cognito-API können Sie die folgenden Funktionen in Ihrer App implementieren.

1. Informationen über das aktuelle Gerät eines Benutzers anzeigen.
2. Eine Liste aller Geräte Ihres Benutzers anzeigen.
3. Ein Gerät verwerfen.
4. Den gespeicherten Status eines Geräts aktualisieren.

Die Zugriffstoken, die die API-Anfragen in den folgenden Beschreibungen autorisieren, müssen den Geltungsbereich `aws.cognito.signin.user.admin` enthalten. Amazon Cognito fügt allen Zugriffstoken, die Sie mit der Benutzerpool-API von Amazon Cognito generieren, einen Antrag für diesen Bereich hinzu. Drittanbieter-IdPs müssen Geräte und MFA für ihre Benutzer, die sich bei Amazon Cognito authentifizieren, separat verwalten. Auf der gehosteten Benutzeroberfläche können Sie den Bereich `aws.cognito.signin.user.admin` anfordern, aber die gehostete Benutzeroberfläche fügt automatisch Geräteinformationen zu erweiterten Sicherheitsbenutzerprotokollen hinzu und bietet nicht an, Geräte zu speichern.

Informationen zu einem Gerät anzeigen

Sie können Informationen über das Gerät eines Benutzers abfragen, um festzustellen, ob es derzeit noch verwendet wird. Möglicherweise möchten Sie gespeicherte Geräte deaktivieren, nachdem 90 Tage lang keine Anmeldung stattgefunden hat.

- Um die Geräteinformationen Ihres Benutzers in einer öffentlichen Client-App anzuzeigen, übermitteln Sie den Zugriffsschlüssel und den Geräteschlüssel Ihres Benutzers in einer GetDevice-API-Anfrage.
- Um die Geräteinformationen Ihres Benutzers in einer vertraulichen Client-App anzuzeigen, signieren Sie eine AdminGetDevice-API-Anfrage mit AWS-Anmeldeinformationen und geben Sie den Benutzernamen, den Geräteschlüssel und den Benutzerpool Ihres Benutzers ein.

Eine Liste aller Geräte Ihres Benutzers anzeigen

Sie können eine Liste aller Geräte Ihrer Benutzer und ihrer Eigenschaften anzeigen. Möglicherweise möchten Sie beispielsweise überprüfen, ob das aktuelle Gerät mit einem gespeicherten Gerät übereinstimmt.

- Senden Sie in einer öffentlichen Client-App das Zugriffstoken Ihres Benutzers in einer ListDevices-API-Anfrage.
- Signieren Sie in einer vertraulichen Client-App eine AdminListDevices-API-Anfrage mit AWS-Anmeldeinformationen und geben Sie den Benutzernamen und den Benutzerpool Ihres Benutzers ein.

Ein Gerät verwerfen

Sie können den Geräteschlüssel eines Benutzers löschen. Sie sollten dies tun, wenn Sie feststellen, dass Ihr Benutzer ein Gerät nicht mehr verwendet, oder wenn Sie ungewöhnliche Aktivitäten feststellen und einen Benutzer auffordern möchten, die MFA erneut abzuschließen. Um das Gerät später erneut zu registrieren, müssen Sie einen neuen Geräteschlüssel generieren und speichern.

- Senden Sie in einer öffentlichen Client-App den Geräteschlüssel und das Zugriffstoken Ihres Benutzers in einer ForgetDevice-API-Anfrage.
- Senden Sie in einer vertraulichen Client-App den Geräteschlüssel und das Zugriffstoken Ihres Benutzers in einer AdminForgetDevice-API-Anfrage.

Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte

Für die Registrierung, Anmeldung und Verwaltung von Benutzern in Ihrem Benutzerpool haben Sie zwei Möglichkeiten.

1. Zu Ihren Benutzerpool-Endpunkten gehören die gehostete Benutzeroberfläche und die Verbund-Endpunkte. Sie bilden ein Paket öffentlicher Webseiten, das Amazon Cognito aktiviert, wenn Sie eine Domain für Ihren Benutzerpool auswählen. Verwenden Sie für einen schnellen Einstieg in die Authentifizierungs- und Autorisierungsfunktionen der Amazon-Cognito-Benutzerpools, einschließlich Seiten für die Registrierung, Anmeldung, Passwortverwaltung und Multi-Faktor-Authentifizierung (MFA), die integrierte Benutzeroberfläche der gehosteten Benutzeroberfläche. Die anderen Benutzerpool-Endpunkte ermöglichen die Authentifizierung bei externen Identitätsanbietern (IdPs). Die von ihnen erbrachten Dienstleistungen beinhalten Folgendes.

   a. Callback-Endpunkte für Serviceprovider für authentifizierte Anträge Ihrer IdPs, wie `saml2/idpresponse` und `oauth2/idpresponse`. Wenn Amazon Cognito ein Zwischendienstanbieter (SP) zwischen Ihrer App und Ihrem IdP ist, repräsentieren die Callback-Endpunkte den Service.

   b. Endpunkte, die Informationen über Ihre Umgebung bereitstellen, wie `oauth2/userInfo` und `jwks.json`. Ihre App verwendet diese Endpunkte, wenn sie Token überprüft oder Benutzerprofildaten mit AWS-SDKs und OAuth-2.0-Bibliotheken abruft.

2. Die API für Amazon-Cognito-Benutzerpools ist ein Satz von Tools für Ihre Web- oder Mobil-App, mit dem nach der Erfassung von Anmeldeinformationen in Ihrem eigenen benutzerdefinierten Frontend Benutzer authentifiziert werden können. Die API-Authentifizierung für Benutzerpools erzeugt die folgenden JSON-Web-Token.

   a. Ein Identitäts-Token mit überprüfbaren Attributansprüchen Ihres Benutzers.

   b. Ein Zugriffs-Token, das Ihren Benutzer autorisiert, über Token autorisierte API-Anfragen für einen AWS-Service-Endpunkt zu erstellen.

   > **Note**
   >
   > Zugriffstoken aus der Benutzerpool-API-Authentifizierung enthalten standardmäßig nur den `aws.cognito.signin.user.admin`-Bereich. Um ein Zugriffstoken mit zusätzlichen Bereichen zu generieren, um beispielsweise eine Anfrage an eine Drittanbieter-API zu autorisieren, fragen Sie bei der Authentifizierung über Ihre Benutzerpool-Endpunkte Bereiche an oder fügen Sie in einer Lambda-Auslöser für die Vorab-Generierung von Token benutzerdefinierte Bereiche hinzu. Die Anpassung des Zugriffstokens erhöht Ihre AWS-Rechnung um zusätzliche Kosten.

Sie können einen Verbundbenutzer, der sich normalerweise über die Benutzerpool-Endpunkte anmelden würde, mit einem Benutzer verknüpfen, dessen Profil in Ihrem Benutzerpool lokal ist. Ein lokaler Benutzer existiert ausschließlich in Ihrem Benutzerpool-Verzeichnis ohne Verbund über einen externen IdP. Wenn Sie dessen Verbundidentität in einer AdminLinkProviderForUser-API-Anforderung mit einem lokalen Benutzer verknüpfen, kann er sich mit der Benutzerpool-API anmelden. Weitere Informationen finden Sie unter Verknüpfen von Verbundbenutzern mit einem vorhandenen Benutzerprofil.

Die Amazon-Cognito-Benutzerpool-API hat zwei Verwendungszwecke. Sie erstellt und konfiguriert die Ressourcen für Ihre Amazon-Cognito-Benutzerpools. Sie können beispielsweise Benutzerpools erstellen, AWS Lambda-Auslöser hinzufügen und Ihre gehostete UI-Domain konfigurieren. Die Benutzerpool-API führt auch Operationen zur Anmeldung und Registrierung sowie andere Benutzeroperationen für lokale und verknüpfte Benutzer durch.

Beispielszenario unter Verwendung der Amazon-Cognito-Benutzerpool-API

1. Ihr Benutzer wählt die Schaltfläche zum Erstellen eines Kontos aus, die Sie in Ihrer App erstellt haben. Der Benutzer gibt eine E-Mail-Adresse und ein Passwort ein.
2. Ihre App sendet eine API-Anforderung SignUp und erstellt einen neuen Benutzer in Ihrem Benutzerpool.
3. Ihre App fordert den Benutzer zur Eingabe eines E-Mail-Bestätigungscodes auf. Der Benutzer gibt den Code ein, den er in einer E-Mail-Nachricht erhalten hat.
4. Ihre App sendet eine API-Anforderung ConfirmSignup mit dem Bestätigungscode des Benutzers.
5. Ihre App fordert den Benutzer zur Eingabe seines Benutzernamens und Passworts auf und der Benutzer gibt die Informationen ein.
6. Ihre App sendet eine API-Anforderung InitiateAuth und speichert ein ID-Token, ein Zugriffstoken sowie ein Aktualisierungstoken. Ihre App ruft OIDC-Bibliotheken auf, um die Tokens Ihres Benutzers zu verwalten und eine dauerhafte Sitzung für diesen Benutzer zu unterhalten.

In der Amazon-Cognito-Benutzerpool-API können Sie keine Benutzer anmelden, die über einen IdP verbunden sind. Diese Benutzer müssen Sie über Ihre Benutzerpool-Endpunkte authentifizieren. Weitere Informationen zu den Benutzerpool–Endpunkten, die die gehostete Benutzeroberfläche enthalten, finden Sie unter Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI. Ihre Verbundbenutzer können zunächst in der gehosteten Benutzeroberfläche ihren IdP auswählen, Sie können aber auch die gehostete Benutzeroberfläche überspringen und Ihre Benutzer zur Anmeldung direkt an Ihren IdP weiterleiten. Wenn Ihre API-Anforderung an den Autorisieren des Endpunkts einen

IdP-Parameter enthält, leitet Amazon Cognito Ihren Benutzer im Hintergrund auf die Anmeldeseite des IdP weiter.

Beispielszenario mit Benutzerpool-Endpunkten

1. Ihr Benutzer wählt die Schaltfläche zum Erstellen eines Kontos aus, die Sie in Ihrer App erstellt haben.
2. Sie präsentieren dem Benutzer eine Liste der Social-Identity-Anbieter, bei denen Sie Entwickleranmeldeinformationen registriert haben. Der Benutzer entscheidet sich für Apple.
3. Ihre App leitet eine Anforderung an den Autorisieren des Endpunkts mit dem Anbieternamen `SignInWithApple` ein.
4. Im Browser des Benutzers wird die Apple-OAuth-Autorisierungsseite geöffnet. Der Benutzer legt fest, dass Amazon Cognito seine Profilinformationen lesen darf.
5. Amazon Cognito bestätigt das Apple-Zugriffstoken und fragt das Apple-Profil des Benutzers ab.
6. Der Benutzer präsentiert Ihrer App einen Amazon-Cognito-Autorisierungscode.
7. Ihre App tauscht den Autorisierungscode durch den Token-Endpunkt aus und speichert ein ID-Token, ein Zugriffstoken und ein Aktualisierungstoken. Ihre App ruft OIDC-Bibliotheken auf, um die Tokens Ihres Benutzers zu verwalten und eine dauerhafte Sitzung für diesen Benutzer zu unterhalten.

Die Benutzerpool-API und die Benutzerpool–Endpunkte unterstützen eine Vielzahl von Szenarien, die in diesem Handbuch beschrieben werden. In den folgenden Abschnitten erörtern wir, wie sich die Benutzerpool-API weiter in Klassen unterteilt, die Ihre Anforderungen in Bezug auf die Anmeldung, Registrierung und Ressourcenverwaltung unterstützen.

Authentifizierte und nicht authentifizierte API-Operationen der Amazon-Cognito-Benutzerpools

Die Amazon-Cognito-Benutzerpool-API, die eine Schnittstelle für die Ressourcenverwaltung sowie eine Authentifizierungs- und Autorisierungsschnittstelle für Benutzer bildet, kombiniert die folgenden Autorisierungsmodelle bei ihren Operationen. Je nach API-Operation müssen Sie möglicherweise eine Autorisierung mit IAM-Anmeldeinformationen, einem Zugriffstoken, einem Sitzungstoken, einem Client-Schlüssel oder einer Kombination davon vornehmen. Für viele Operationen zur Authentifizierung und Autorisierung von Benutzern haben Sie die Wahl zwischen authentifizierten und nicht authentifizierten Versionen der Anforderung. Nicht authentifizierte Operationen sind eine

bewährte Sicherheitsmethode für Apps, die Sie an Ihre Benutzer verteilen, z. B. mobile Apps. Dabei müssen Sie keine geheimen Schlüssel in Ihren Code aufnehmen.

Sie können Berechtigungen in IAM-Richtlinien nur für IAM-authentifizierte Verwaltungsoperationen und IAM-authentifizierte Benutzeroperationen zuweisen.

IAM-authentifizierte Verwaltungsoperationen

IAM-authentifizierte Verwaltungsoperationen können die Konfiguration Ihres Benutzerpools und Ihres App-Clients ändern und anzeigen, wie Sie dies in der AWS Management Console tun würden.

Um beispielsweise Ihren Benutzerpool mit einer API-Anforderung UpdateUserPool zu ändern, müssen Sie AWS-Anmeldeinformationen und IAM-Berechtigungen zum Aktualisieren der Ressource angeben.

Um diese Anforderungen in der AWS Command Line Interface (AWS CLI) oder einem AWS-SDK zu autorisieren, konfigurieren Sie Ihre Umgebung mit Umgebungsvariablen oder einer Client-Konfiguration, die Ihrer Anforderung IAM-Anmeldeinformationen hinzufügt. Weitere Informationen finden Sie unter Zugreifen auf AWS mit Ihren AWS-Anmeldeinformationen in der Allgemeine AWS-Referenz. Sie können auch Anforderungen direkt an die Service-Endpunkte für die Benutzerpool-API von Amazon Cognito senden. Sie müssen diese Anforderungen mit AWS-Anmeldeinformationen autorisieren oder signieren, die Sie in den Header Ihrer Anforderung einbetten. Weitere Informationen finden Sie unter Signieren von AWS-API-Anforderungen.

IAM-authentifizierte Verwaltungsoperationen

`AddCustomAttributes`

`CreateGroup`

`CreateIdentityProvider`

`CreateResourceServer`

`CreateUserImportJob`

`CreateUserPool`

`CreateUserPoolClient`

`CreateUserPoolDomain`

IAM-authentifizierte Verwaltungsoperationen

DeleteGroup

DeleteIdentityProvider

DeleteResourceServer

DeleteUserPool

DeleteUserPoolClient

DeleteUserPoolDomain

DescribeIdentityProvider

DescribeResourceServer

DescribeRiskConfiguration

DescribeUserImportJob

DescribeUserPool

DescribeUserPoolClient

DescribeUserPoolDomain

GetCSVHeader

GetGroup

GetIdentityProviderByIdentifier

GetSigningCertificate

GetUICustomization

GetUserPoolMFaconfig

ListGroups

ListIdentityProviders

IAM-authentifizierte Verwaltungsoperationen

`ListResourceServers`

`ListTagsForResource`

`ListUserImportJobs`

`ListUserPoolClients`

`ListUserPools`

`ListUsers`

`ListUsersInGroup`

`SetRiskConfiguration`

`SetUICustomization`

`SetUserPoolMfaConfig`

`StartUserImportJob`

`StopUserImportJob`

`TagResource`

`UntagResource`

`UpdateGroup`

`UpdateIdentityProvider`

`UpdateResourceServer`

`UpdateUserPool`

`UpdateUserPoolClient`

`UpdateUserPoolDomain`

IAM-authentifizierte Benutzeroperationen

IAM-authentifizierte Benutzeroperationen können zum Anmelden, Registrieren, Ändern und Anzeigen Ihrer Benutzer sowie zum Verwalten von deren Anmeldeinformationen verwendet werden.

Sie können beispielsweise eine serverseitige Anwendungsebene haben, die ein Web-Frontend unterstützt. Bei Ihrer serverseitigen App handelt es sich um einen vertraulichen OAuth-Client, dem Sie privilegierten Zugriff auf Ihre Amazon-Cognito-Ressourcen gewähren. Um einen Benutzer in der App zu registrieren, kann Ihr Server AWS-Anmeldeinformationen in eine API-Aufforderung AdminCreateUser aufnehmen. Weitere Informationen zu OAuth-Client-Typen finden Sie unter Client Types in The OAuth 2.0 Authorization Framework.

Um diese Anforderungen in der AWS CLI oder einem AWS-SDK zu autorisieren, konfigurieren Sie Ihre serverseitige App-Umgebung mit Umgebungsvariablen oder einer Client-Konfiguration, die Ihrer Anforderung IAM-Anmeldeinformationen hinzufügt. Weitere Informationen finden Sie unter Zugreifen auf AWS mit Ihren AWS-Anmeldeinformationen in der Allgemeine AWS-Referenz. Sie können auch Anforderungen direkt an die Service-Endpunkte für die Benutzerpool-API von Amazon Cognito senden. Sie müssen diese Anforderungen mit AWS-Anmeldeinformationen autorisieren oder signieren, die Sie in den Header Ihrer Anforderung einbetten. Weitere Informationen finden Sie unter Signieren von AWS-API-Anforderungen.

Wenn Ihr App-Client über einen geheimen Client-Schlüssel verfügt, müssen Sie Ihre IAM-Anmeldeinformationen und, je nach Operation, den Parameter `SecretHash` oder den Wert `SECRET_HASH` in `AuthParameters` angeben. Weitere Informationen finden Sie unter Berechnen von Werten für geheime Hashes.

IAM-authentifizierte Benutzeroperationen

`AdminAddUserToGroup`

`AdminConfirmSignUp`

`AdminCreateUser`

`AdminDeleteUser`

`AdminDeleteUserAttributes`

`AdminDisableProviderForUser`

IAM-authentifizierte Benutzeroperationen

`AdminDisableUser`

`AdminEnableUser`

`AdminForgetDevice`

`AdminGetDevice`

`AdminGetUser`

`AdminInitiateAuth`

`AdminLinkProviderForUser`

`AdminListDevices`

`AdminListGroupsForUser`

`AdminListUserAuthEvents`

`AdminRemoveUserFromGroup`

`AdminResetUserPassword`

`AdminRespondToAuthChallenge`

`AdminSetUserMFAPReference`

`AdminSetUserPassword`

`AdminSetUserSettings`

`AdminUpdateAuthEventFeedback`

`AdminUpdateDeviceStatus`

`AdminUpdateUserAttributes`

`AdminUserLobalSignout`

Nicht authentifizierte Benutzeroperationen

Nicht authentifizierte Benutzeroperationen können zur Registrierung und Anmeldung Ihrer Benutzer sowie zum Zurücksetzen des Passworts der Benutzer genutzt werden. Verwenden Sie nicht authentifizierte oder öffentliche API-Operationen, wenn Sie möchten, dass sich jeder Internetbenutzer bei Ihrer App registrieren und anmelden kann.

Um beispielsweise einen Benutzer in Ihrer App zu registrieren, können Sie einen öffentlichen OAuth-Client verteilen, der keinen privilegierten Zugriff auf geheime Schlüssel bietet. Sie können diesen Benutzer mit der nicht authentifizierten API-Operation SignUp registrieren.

Um diese Anforderungen in einem öffentlichen Client zu senden, den Sie mit einem AWS-SDK entwickelt haben, müssen Sie keine Anmeldeinformationen konfigurieren. Sie können Anforderungen auch direkt an die Service-Endpunkte für die Benutzerpool-API von Amazon Cognito ohne zusätzliche Autorisierung senden.

Wenn Ihr App-Client über einen geheimen Client-Schlüssel verfügt, müssen Sie je nach Operation den Parameter `SecretHash` oder den Wert `SECRET_HASH` in `AuthParameters` angeben. Weitere Informationen finden Sie unter Berechnen von Werten für geheime Hashes.

Nicht authentifizierte Benutzeroperationen

`SignUp`

`ConfirmSignUp`

`ResendConfirmationCode`

`ForgotPassword`

`ConfirmForgotPassword`

`InitiateAuth`

Über Token autorisierte Benutzeroperationen

Über Token autorisierte Benutzeroperationen können zum Abmelden, Ändern und Anzeigen Ihrer Benutzer sowie zur Verwaltung der Anmeldeinformationen für Ihre Benutzer verwendet werden, nachdem sich diese angemeldet bzw. den Anmeldevorgang gestartet haben. Verwenden Sie über Token autorisierte API-Operationen, wenn Sie keine geheimen Schlüssel in Ihrer App verteilen

und Anforderungen mit den eigenen Anmeldeinformationen des Benutzers autorisieren möchten. Wenn Ihr Benutzer die Anmeldung abgeschlossen hat, müssen Sie seine über Token autorisierte API-Anforderung mit einem Zugriffstoken autorisieren. Wenn sich der Benutzer gerade in einem Anmeldevorgang befindet, müssen Sie seine über Token autorisierte API-Anforderung mit einem Sitzungstoken autorisieren, das Amazon Cognito in der Antwort auf die vorherige Anforderung zurückgegeben hat.

Vielleicht möchten Sie beispielsweise in einem öffentlichen Client das Profil eines Benutzers so aktualisieren, dass der Schreibzugriff auf das eigene Profil des Benutzers beschränkt ist. Für diese Aktualisierung kann Ihr Client das Zugriffstoken des Benutzers in eine API-Anforderung UpdateUserAttributes aufnehmen.

Um diese Anforderungen in einem öffentlichen Client zu senden, den Sie mit einem AWS-SDK entwickelt haben, müssen Sie keine Anmeldeinformationen konfigurieren. Fügen Sie Ihrer Anforderung einen Parameter `AccessToken` oder `Session` hinzu. Sie können auch Anforderungen direkt an die Service-Endpunkte für die Benutzerpool-API von Amazon Cognito senden. Um eine Anforderung an einen Service-Endpunkt zu autorisieren, fügen Sie das Zugriffs- oder Sitzungstoken in den POST-Text Ihrer Anforderung ein.

Um eine API-Anforderung für eine über Token autorisierte Operation zu signieren, fügen Sie das Zugriffs-Token als `Authorization`-Header in die Anforderung ein, und zwar im Format `Bearer` *`<Base64-encoded access token>`*.

Über Token autorisierte Benutzeroperationen	AccessToken	Sitzung
`RespondToAuthChallenge`		✓
`ChangePassword`	✓	
`GetUser`	✓	
`UpdateUserAttributes`	✓	
`DeleteUserAttributes`	✓	

Über Token autorisierte Benutzeroperationen	AccessToken	Sitzung
DeleteUser	✓	
ConfirmDevice	✓	
ForgetDevice	✓	
GetDevice	✓	
ListDevices	✓	
UpdateDeviceStatus	✓	
GetUserAttributeVerificationCode	✓	
VerifyUserAttribute	✓	
SetUserSettings	✓	
SetUserMFAPreference	✓	
GlobalSignout	✓	
AssociateSoftwareToken	✓	✓
UpdateAuthEventFeedback		✓
VerifySoftwareToken	✓	✓
RevokeToken [1]		

[1] RevokeToken verwendet ein Aktualisierungstoken als Parameter. Das Aktualisierungstoken dient als Autorisierungstoken und als Zielressource.

Aktualisieren der Benutzerpool-Konfiguration

Um die Einstellungen der Amazon Cognito Cognito-Benutzerpools in der zu ändern AWS Management Console, navigieren Sie durch die funktionsbasierten Registerkarten in Ihren Benutzerpool-Einstellungen und aktualisieren Sie die Felder, wie in anderen Bereichen dieses Handbuchs beschrieben. Nachdem Sie einen Benutzerpool erstellt haben, können Sie einige dieser Einstellungen nicht mehr ändern. Wenn Sie die folgenden Einstellungen ändern möchten, müssen Sie einen neuen Benutzerpool oder App-Client erstellen.

Name des Benutzerpools

Name des API-Parameters: PoolName

Der Anzeigename, den Sie Ihrem Benutzerpool zugewiesen haben. Wenn Sie den Namen eines Benutzerpools ändern möchten, erstellen Sie einen neuen Benutzerpool.

Anmeldeoptionen für den Amazon-Cognito-Benutzerpool

API-Parameternamen: AliasAttributesund UsernameAttributes

Die Attribute, die Ihre Benutzer beim Anmelden als Benutzernamen übergeben können. Wenn Sie einen Benutzerpool erstellen, können Sie die Anmeldung mit Benutzername, E-Mail-Adresse, Telefonnummer oder einem bevorzugten Benutzernamen zulassen. Wenn Sie die Anmeldeoptionen für den Benutzerpool ändern möchten, erstellen Sie einen neuen Benutzerpool.

Make user name case sensitive (Groß- und Kleinschreibung bei Benutzernamen beachten)

API-Parametername: UsernameConfiguration

Wenn Sie einen Benutzernamen erstellen, der mit Ausnahme der Groß-/Kleinschreibung mit einem anderen Benutzernamen übereinstimmt, kann Amazon Cognito diesen entweder als denselben Benutzer oder als eindeutigen Benutzer behandeln. Weitere Informationen finden Sie unter Berücksichtigung der Groß-/Kleinschreibung im Benutzerpool. Wenn Sie die Groß-/Kleinschreibung ändern möchten, erstellen Sie einen neuen Benutzerpool.

Clientschlüssel

Name des API-Parameters: GenerateSecret

Wenn Sie einen App-Client erstellen, können Sie einen Client-Schlüssel generieren, damit nur vertrauenswürdige Quellen Anfragen an Ihren Benutzerpool stellen können. Weitere Informationen finden Sie unter App-Clients für Benutzerpools. Wenn Sie einen Client-Schlüssel ändern möchten, erstellen Sie einen neuen App-Client im selben Benutzerpool.

Erforderliche Attribute

API-Parametername: Schema

Die Attribute, für die Ihre Benutzer Werte angeben müssen, wenn sie sich anmelden oder wenn Sie sie erstellen. Weitere Informationen finden Sie unter Attribute für den Benutzerpool. Wenn Sie die erforderlichen Attribute ändern möchten, erstellen Sie einen neuen Benutzerpool.

Custom attributes (Benutzerdefinierte Attribute)

API-Parametername: Schema

Attribute mit benutzerdefinierten Namen. Sie können den Wert des benutzerdefinierten Attributs eines Benutzers ändern, aber Sie können kein benutzerdefiniertes Attribut aus Ihrem Benutzerpool löschen. Weitere Informationen finden Sie unter Attribute für den Benutzerpool. Wenn Sie die maximale Anzahl benutzerdefinierter Attribute erreicht haben und die Liste ändern möchten, erstellen Sie einen neuen Benutzerpool.

SMS-Konfiguration

Nachdem Sie SMS-Nachrichten in Ihrem Benutzerpool aktiviert haben, können Sie sie nicht mehr deaktivieren.

- Wenn Sie sich beim Erstellen eines Benutzerpools dafür entscheiden, SMS-Nachrichten zu konfigurieren, können Sie SMS nach Abschluss der Einrichtung nicht deaktivieren.
- Sie können SMS-Nachrichten in einem Benutzerpool aktivieren, den Sie erstellt haben, aber danach können Sie SMS nicht mehr deaktivieren.
- Amazon Cognito kann SMS-Nachrichten für die Einladung und Wiederherstellung von Benutzerkonten, die Überprüfung von Attributen und die Multi-Faktor-Authentifizierung (MFA) verwenden. Nachdem Sie SMS-Nachrichten aktiviert haben, können Sie SMS-Nachrichten für diese Funktionen jederzeit ein- oder ausschalten.
- Die SMS-Nachrichtenkonfiguration umfasst eine IAM-Rolle, die Sie an Amazon Cognito delegieren, um Nachrichten mit Amazon SNS zu senden. Sie können die zugewiesene Rolle jederzeit ändern.

Aktualisierung eines Benutzerpools mit einem AWS SDK oder einer REST-API AWS CDK

In der Amazon Cognito Cognito-Konsole können Sie die Einstellungen Ihres Benutzerpools für jeden Parameter ändern. Um beispielsweise einen Lambda-Trigger hinzuzufügen, wählen Sie Lambda-Trigger hinzufügen und wählen die Funktion und den Triggertyp aus. Die Amazon Cognito Cognito-Benutzerpools-API ist so strukturiert, dass Aktualisierungsvorgänge für Benutzerpools und App-Clients alle Parameter für den Benutzerpool erfordern. Die Konsole automatisiert diesen Aktualisierungsvorgang jedoch auf transparente Weise mit Ihren anderen Benutzerpool-Einstellungen.

Möglicherweise stellen Sie manchmal fest, dass eine Änderung an einer anderen Stelle in Ihrem AWS-Konto System dazu führen kann, dass Updates einen Fehler generieren, wenn sie nicht mit der Einstellung zusammenhängen, die Sie ändern möchten. Eine gelöschte Amazon SES SES-Identität oder eine Änderung einer IAM-Berechtigung für AWS WAF, zum Beispiel. Wenn einer der aktuellen Parameter nicht mehr gültig ist, können Sie Ihre Einstellungen erst aktualisieren, wenn Sie ihn behoben haben. Wenn Sie auf einen solchen Fehler stoßen, überprüfen Sie die Fehlerreaktion und überprüfen Sie die darin erwähnte Einstellung.

Die Amazon Cognito Cognito-Benutzerpools AWS Cloud Development Kit (AWS CDK), die REST-API und die AWS SDKs sind Tools für die Automatisierung und programmatische Konfiguration von Amazon Cognito Cognito-Ressourcen. Anfragen mit diesen Tools müssen ebenfalls, wie bei der Amazon Cognito-Konsole, eine Einstellung mit einer vollständigen Ressourcenkonfiguration im Anfragetext aktualisieren. Auf einer höheren Ebene müssen Sie den folgenden Prozess ausführen.

1. Erfassen Sie die Ausgabe eines Vorgangs, der die Konfiguration Ihrer vorhandenen Ressource beschreibt.
2. Ändern Sie die Ausgabe mit Ihren geänderten Einstellungen.
3. Senden Sie die geänderte Konfiguration in einem Vorgang, der Ihre Ressource aktualisiert.

Das folgende Verfahren aktualisiert Ihre Konfiguration mit dem UpdateUserPoolAPI-Vorgang. Derselbe Ansatz mit unterschiedlichen Eingabefeldern gilt für UpdateUserPoolClient.

> **Important**
>
> Wenn Sie keine Werte für vorhandene Parameter angeben, legt Amazon Cognito diese auf Standardwerte fest. Wenn Sie zum Beispiel einen vorhandenen `LambdaConfig`-Wert haben

und einen `UpdateUserPool` mit einer leeren `LambdaConfig` senden, löschen Sie die Zuordnung aller Lambda-Funktionen zu Benutzerpool-Auslösern. Planen Sie entsprechend, wenn Sie Änderungen an Ihrer Benutzerpool-Konfiguration automatisieren möchten.

1. Erfassen Sie den aktuellen Status Ihres Benutzerpools mit DescribeUserPool.
2. Formatieren Sie die Ausgabe von `DescribeUserPool` den Anfrageparametern von `UpdateUserPool` entsprechend. Entfernen Sie die folgenden Felder der obersten Ebene und ihre untergeordneten Objekte aus dem ausgegebenen JSON-Code.
   - `Arn`
   - `CreationDate`
   - `CustomDomain`
     - Aktualisieren Sie dieses Feld mit der UpdateUserPoolDomainAPI-Operation.
   - `Domain`
     - Aktualisieren Sie dieses Feld mit der UpdateUserPoolDomainAPI-Operation.
   - `EmailConfigurationFailure`
   - `EstimatedNumberOfUsers`
   - `Id`
   - `LastModifiedDate`
   - `Name`
   - `SchemaAttributes`
   - `SmsConfigurationFailure`
   - `Status`
3. Bestätigen Sie, dass der resultierende JSON-Code den Anfrageparametern von `UpdateUserPool` entspricht.
4. Ändern Sie alle Parameter, die Sie im resultierenden JSON-Code bearbeiten möchten.
5. Senden Sie eine API-Anfrage `UpdateUserPool` mit Ihrem modifizierten JSON-Code als Anforderungseingabe.

Sie können auch diese modifizierte Ausgabe `DescribeUserPool` im Parameter `--cli-input-json` von `update-user-pool` in der AWS CLI verwenden.

Führen Sie alternativ den folgenden AWS CLI Befehl aus, um JSON mit leeren Werten für die akzeptierten Eingabefelder für `update-user-pool` zu generieren. Anschließend können Sie diese Felder mit den vorhandenen Werten aus Ihrem Benutzerpool füllen.

```
aws cognito-idp update-user-pool --generate-cli-skeleton --output json
```

Führen Sie den folgenden Befehl aus, um dasselbe JSON-Objekt für einen App-Client zu erstellen.

```
aws cognito-idp update-user-pool-client --generate-cli-skeleton --output json
```

Einrichtung und Verwendung der gehosteten Benutzeroberfläche und der Verbundendpunkte in Amazon Cognito

Ein Amazon Cognito Cognito-Benutzerpool mit einer Domain ist ein OAuth-2.0-kompatibler Autorisierungsserver und eine ready-to-use gehostete Benutzeroberfläche (UI) für die Authentifizierung. Der Autorisierungsserver leitet Authentifizierungsanfragen weiter, gibt JSON-Web-Token (JWTs) aus und verwaltet sie und stellt Informationen zu Benutzerattributen zur Verfügung. Bei der gehosteten Benutzeroberfläche handelt es sich um eine Sammlung von Webschnittstellen für grundlegende Aktivitäten zur Registrierung, Anmeldung, Multi-Faktor-Authentifizierung und zum Zurücksetzen von Passwörtern in Ihrem Benutzerpool. Es ist auch ein zentraler Knotenpunkt für die Authentifizierung bei den externen Identitätsanbietern (IdPs), die Sie mit Ihrer App verknüpfen. Ihre App kann die gehostete Benutzeroberfläche und die Autorisierungsendpunkte aufrufen, wenn Sie Benutzer authentifizieren und autorisieren möchten. Sie können das Benutzererlebnis der gehosteten Benutzeroberfläche mit Ihrem eigenen Logo und Ihrer CSS-Anpassung an Ihre Marke anpassen. Weitere Informationen zu den Komponenten der gehosteten Benutzeroberfläche und des Autorisierungsservers finden Sie unter Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI.

> **Note**
>
> Die gehostete Benutzeroberfläche von Amazon Cognito unterstützt keine benutzerdefinierte Authentifizierung mit benutzerdefinierten Authentifizierungs-Challenge-Lambda-Triggern.

Themen

- Einrichtung der gehosteten Benutzeroberfläche mit AWS Amplify

- Einrichten der gehosteten Benutzeroberfläche mit der Amazon-Cognito-Konsole
- Anzeigen Ihrer Anmeldeseite
- Wissenswertes über die gehostete Benutzeroberfläche von Amazon-Cognito-Benutzerpools
- Konfigurieren einer Benutzerpool-Domäne
- Anpassen der integrierten Registrierungs- und Anmeldungswebseiten
- Registrieren und Anmelden mit der gehosteten Benutzeroberfläche

Einrichtung der gehosteten Benutzeroberfläche mit AWS Amplify

Wenn Sie AWS Amplify Ihrer Web- oder mobilen App Authentifizierung hinzufügen, können Sie Ihre gehostete Benutzeroberfläche mithilfe der Befehlszeilenschnittstelle (CLI) und der Bibliotheken im AWS Amplify Framework einrichten. Wenn Sie eine Authentifizierung zu Ihrer Anwendung hinzufügen möchten, verwenden Sie die AWS Amplify -CLI, um dem Projekt die `Auth`-Kategorie hinzuzufügen. Anschließend verwenden Sie in Ihrem Client-Code die AWS Amplify Bibliotheken, um Benutzer mit Ihrem Amazon Cognito Cognito-Benutzerpool zu authentifizieren.

Sie können eine vordefinierte gehostete Benutzeroberfläche anzeigen oder Benutzer über einen OAuth 2.0-Endpunkt verbinden, der eine Umleitung zu einem Anbieter für Social Sign-in durchführt, z. B. Facebook, Google, Amazon oder Apple. Nachdem sich ein Benutzer erfolgreich bei diesem Anbieter authentifiziert hat, erstellt AWS Amplify gegebenenfalls einen neuen Benutzer in Ihrem Benutzerpool und stellt das OIDC-Token des Benutzers in Ihrer Anwendung bereit.

Die folgenden Beispiele zeigen, wie Sie AWS Amplify die gehostete Benutzeroberfläche mit sozialen Anbietern in Ihrer App einrichten können.

- AWS Amplify Authentifizierung für JavaScript.
- AWS Amplify Authentifizierung für Swift.
- AWS Amplify Authentifizierung für Flutter.
- AWS Amplify Authentifizierung für Android.

Einrichten der gehosteten Benutzeroberfläche mit der Amazon-Cognito-Konsole

Erstellen eines App-Clients

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus.
5. Wählen Sie unter App clients (App-Clients) Create an app client (App-Client erstellen) aus.
6. Wählen Sie einen App type (Anwendungstyp): Public client (Öffentlicher Client), Confidential client (Vertraulicher Client) oder Other (Sonstige) aus. Ein öffentlicher Client arbeitet normalerweise von den Geräten Ihrer Benutzer aus und verwendet nicht authentifizierte und mit Token authentifizierte APIs. Ein vertraulicher Client wird in der Regel über eine App auf einem zentralen Server ausgeführt, dem Sie Client-Geheimnisse und API-Anmeldeinformationen anvertrauen, und verwendet Autorisierungsheader und AWS Identity and Access Management Anmeldeinformationen, um Anfragen zu signieren. Wenn sich Ihr Anwendungsfall von den vorkonfigurierten App-Client-Einstellungen für einen öffentlichen Client oder einen vertraulichen Client unterscheidet, wählen Sie Other (Sonstige) aus.
7. Geben Sie einen App-Client-Namen ein.
8. Wählen Sie die Authentifizierungsabläufe aus, die Sie in Ihrem App-Client zulassen möchten.
9. Konfigurieren Sie die Authentication flow session duration (Dauer der Authentifizierungsablaufsitzung). Dies ist die Zeitdauer, die Ihren Benutzern für den Abschluss einer Authentifizierungsabfrage zur Verfügung steht, bevor das Sitzungstoken abläuft.
10. (Optional) Konfigurieren Sie den Token-Ablauf.
    a. Geben Sie den Ablauf für Aktualisierungs-Token für den App-Client an. Der Standardwert lautet 30 Tage. Sie können dies in jeden Wert zwischen 1 Stunde und 10 Jahren ändern.
    b. Geben Sie den Ablauf für Zugriffs-Token für den App-Client an. Der Standardwert lautet 1 Stunde. Sie können ihn in jeden Wert zwischen 5 Minuten und 24 Stunden ändern.
    c. Geben Sie den Ablauf für ID-Token für den App-Client an. Der Standardwert lautet 1 Stunde. Sie können ihn in jeden Wert zwischen 5 Minuten und 24 Stunden ändern.

> **Important**
>
> Wenn Sie die gehostete Benutzeroberfläche verwenden und Token für weniger als eine Stunde einrichten, kann der Benutzer Token basierend auf seinem Sitzungscookie abrufen, das derzeit auf eine Stunde festgelegt ist.

11. Wählen Sie Generate client secret (Kundengeheimnis generieren) aus, damit Amazon Cognito ein Kundengeheimnis für Sie erstellt. Clientgeheimnisse werden normalerweise mit vertraulichen Clients verknüpft.
12. Wählen Sie aus, ob Sie für diesen App-Client die Token-Sperre aktivieren möchten. Dies erhöht den Umfang der Token. Weitere Informationen finden Sie unter Revoking Tokens (Widerrufen von Token).
13. Wählen Sie aus, ob Sie für diesen App-Client Prevent error messages that reveal user existence (Fehlermeldungen vermeiden, die die Benutzerexistenz enthüllen) aktivieren möchten. Amazon Cognito antwortet auf Anmeldeanfragen für nicht vorhandene Benutzer mit einer generischen Nachricht, die angibt, dass entweder der Benutzername oder das Passwort falsch waren.
14. (Optional) Konfigurieren Sie Attribute read and write permissions (Attribut-Lese- und Schreibberechtigungen) für diesen App-Client. Ihr App-Client kann die Berechtigung haben, eine begrenzte Teilmenge des Attributschemas Ihres Benutzerpools zu lesen und zu schreiben.
15. Wählen Sie Create (Erstellen) aus.
16. Notieren Sie sich die Client-ID. Dies identifiziert den App-Client in Registrierungs- und Anmeldungsanfragen.

Konfigurieren Sie die App.

1. Wählen Sie auf der Registerkarte App integration (Anwendungsintegration) unter App clients (App-Client) Ihren App-Client aus. Überprüfen Sie die aktuellen Informationen zu gehosteten Benutzeroberflächen.
2. Fügen Sie eine Rückruf-URL hinzu unter Allowed callback URL(s) (Zugelassene Rückruf-URLs). Eine Rückruf-URL gibt an, wohin der Benutzer nach erfolgreicher Anmeldung umgeleitet wird.
3. Fügen Sie eine Abmelde-URL hinzu unter Allowed sigh-out URL(s) (Zulässige Abmelde-URL(s)). Eine Abmelde-URL gibt an, wohin Ihr Benutzer nach der Abmeldung umgeleitet wird.
4. Fügen Sie mindestens eine Option von der Liste Identity providers (Identitätsanbieter) hinzu.

5. Wählen Sie unter OAuth 2.0 grant types (OAuth 2.0 Gewährungstypen) Authorization code grant (Autorisierungscodegewährung) aus, um einen Autorisierungscode auszugeben, der dann gegen die Benutzerpool-Tokens ausgetauscht wird. Da diese Tokens niemals einem Endbenutzer direkt gezeigt werden, sind sie weniger anfällig gegen Angriffe. Allerdings ist am Backend eine benutzerdefinierte Anwendung erforderlich, um den Autorisierungscode gegen Benutzerpool-Tokens austauschen zu können. Aus Sicherheitsgründen empfehlen wir Ihnen, den Ablauf für die Autorisierungscodegewährung in Verbindung mit Proof Key for Code Exchange (PKCE) für mobile Apps zu verwenden.
6. Wählen Sie unter OAuth 2.0 grant types (OAuth 2.0 Gewährungstypen) Implicit grant (Implizite Gewährung), damit Sie von Amazon Cognito Benutzerpool-JSON-Web-Tokens (JWT) erhalten. Sie können diesen Ablauf verwenden, wenn kein Backend für den Austausch eines Autorisierungscodes gegen Tokens vorhanden ist. Er ist auch für das Debugging von Tokens nützlich.
7. Sie können sowohl den Autorisierungscode als auch die Implizite Codegewährung aktivieren und dann beide Gewährungen nach Bedarf verwenden. Wenn weder Autorisierungscode noch Implizite Zugriffscodegewährungen ausgewählt werden, und Ihr App-Client einen Clientschlüssel hat, können Sie Client credentials grants (Client-Anmeldeinformationsgewährungen) aktivieren. Wählen Sie Client credentials (Client-Anmeldeinformationen) nur dann, wenn Ihre App Zugriffstoken für sich und nicht für einen Benutzer anfordern muss.
8. Wählen Sie die OpenID-Connect-Bereiche aus, die Sie für diesen App-Client autorisieren möchten.
9. Wählen Sie Save Changes.

Eine Domäne konfigurieren.

1. Navigieren Sie zur Registerkarte App integration (Anwendungsintegration) für Ihren Benutzerpool.
2. Gehen Sie dann zu Domain (Domäne) und wählen Sie Actions (Aktionen), Create custom domain (Benutzerdefinierte Domäne erstellen) oder Create Cognito domain (Cognito-Domäne erstellen) aus. Wenn Sie bereits eine Benutzerpool-Domäne konfiguriert haben, wählen Sie Delete Cognito domain (Cognito-Domäne löschen) oder Delete custom domain (Benutzerdefinierte Domäne löschen) aus, bevor Sie eine neue benutzerdefinierte Domäne erstellen.

3. Geben Sie ein verfügbares Domänenpräfix zur Verwendung mit einer Cognito-Domäne ein. Weitere Informationen zum Einrichten einer benutzerdefinierten Domäne finden Sie unter Verwenden der eigenen Domäne für die gehostete Benutzeroberfläche.
4. Wählen Sie Create (Erstellen) aus.

Anzeigen Ihrer Anmeldeseite

Wählen Sie in der Amazon-Cognito-Konsole die Schaltfläche View Hosted UI (Gehostete Benutzeroberfläche anzeigen) in der Konfiguration Ihres App-Clients unter App clients and analytics (App-Clients und Analytik) auf der Registerkarte App integration (App-Integration) aus. Über diese Schaltfläche gelangen Sie auf eine Anmeldeseite in Ihrer gehosteten Benutzeroberfläche mit den folgenden grundlegenden Parametern.

- Die App-Client-ID
- Eine Anfrage zur Erteilung eines Autorisierungscodes
- Eine Anfrage für alle Bereiche, die Sie für den aktuellen App-Client aktiviert haben
- Die erste Rückruf-URL in der Liste für den aktuellen App-Client

Die Schaltfläche View hostet UI (Gehostete Benutzeroberfläche anzeigen) ist nützlich, wenn Sie die grundlegenden Funktionen Ihrer gehosteten Benutzeroberfläche testen möchten. Sie können Ihre Anmelde-URL mit zusätzlichen und geänderten Parametern anpassen. In den meisten Fällen entsprechen die automatisch generierten Parameter des Links View hosted UI (Gehostete Benutzeroberfläche anzeigen) nicht vollständig den Anforderungen Ihrer App. In diesen Fällen müssen Sie die URL anpassen, die Ihre App beim Anmelden Ihrer Benutzer aufruft. Weitere Informationen zu den Parameterschlüsseln und -werten für die Anmeldung finden Sie unter Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI.

Die gehostete UI-Anmeldewebseite verwendet das folgende URL-Format. In diesem Beispiel wird die Erteilung eines Autorisierungscodes mit dem `response_type=code`-Parameter angefordert.

```
https://<your domain>/oauth2/authorize?response_type=code&client_id=<your app client
 id>&redirect_uri=<your callback url>
```

Sie können Ihre Benutzerpool-Domain-Zeichenfolge über die Registerkarte App-Integration abrufen. Auf dieser Registerkarte finden Sie ebenfalls die App-Client-IDs, ihre Callback-URLs, ihre zulässigen Bereiche und andere Konfigurationen unter App-Clients und -Analysen.

Wenn Sie zum `/oauth2/authorize`-Endpunkt mit Ihren benutzerdefinierten Parametern navigieren, leitet Amazon Cognito Sie entweder zum `/oauth2/login`-Endpunkt oder im Hintergrund zu Ihrer IDP-Anmeldeseite um, sofern Sie einen `identity_provider`- oder `idp_identifier`-Parameter angegeben haben. Eine Beispiel-URL, die die gehostete Benutzeroberfläche umgeht, finden Sie unter Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools.

Beispielanforderung der gehosteten Benutzeroberfläche für eine implizite Erteilung

Sie können die Anmeldeseite der gehosteten Benutzeroberfläche mit der folgenden URL für die implizite Codeerteilung anzeigen, für die Folgendes gilt: `response_type=token`. Nach einer erfolgreichen Anmeldung gibt Amazon Cognito Benutzerpool-Token in die Adresszeile Ihres Webbrowsers aus.

```
            https://mydomain.us-east-1.amazoncognito.com/authorize?
response_type=token&client_id=1example23456789&redirect_uri=https://
mydomain.example.com
```

Die Identitäts- und Zugriffstoken werden als Parameter angezeigt, die an Ihre Weiterleitungs-URL angefügt werden.

Das folgende Beispiel ist eine Antwort von einer impliziten Erteilungsanforderung.

```
            https://mydomain.example.com/
#id_token=eyJraaBcDeF1234567890&access_token=eyJraGhIjKlM1112131415&expires_in=3600&token_type=
```

Wissenswertes über die gehostete Benutzeroberfläche von Amazon-Cognito-Benutzerpools

Die gehostete Benutzeroberfläche und die Bestätigung von Benutzern als Administrator

Für lokale Benutzer im Benutzerpool funktioniert die gehostete Benutzeroberfläche am besten, wenn Sie für den Benutzerpool die Einstellung Cognito erlauben, automatisch Nachrichten zur Überprüfung und Bestätigung zu senden aktivieren. Wenn Sie diese Einstellung aktivieren, sendet Amazon Cognito eine Nachricht mit einem Bestätigungscode an Benutzer, die sich registrieren.

Wenn Sie Benutzer stattdessen als Benutzerpool-Administrator bestätigen, zeigt die gehostete Benutzeroberfläche nach der Registrierung eine Fehlermeldung an. In diesem Fall hat Amazon Cognito den neuen Benutzer erstellt, konnte aber keine Bestätigungsnachricht senden. Sie können Benutzer weiterhin als Administrator bestätigen, jedoch wenden die Benutzer sich möglicherweise an Ihren Support, wenn eine Fehlermeldung angezeigt wird. Weitere Informationen zur administrativen Bestätigung finden Sie unter Benutzern erlauben, sich in der Anwendung anzumelden, sie aber als Benutzerpool-Administrator bestätigen.

Die Änderungen an der Konfiguration der gehosteten Benutzeroberfläche anzeigen

Wenn Änderungen an Ihren gehosteten UI-Seiten nicht sofort angezeigt werden, warten Sie ein paar Minuten und aktualisieren Sie dann die Seite.

Benutzerpool-Tokens decodieren

Benutzerpool-Tokens von Amazon Cognito werden unter Verwendung eines RS256-Algorithmus signiert. Sie können Benutzerpool-Token dekodieren und verifizieren mit AWS Lambda, siehe Amazon Cognito JWT-Token dekodieren und verifizieren auf. GitHub

Die gehostete UI- und TLS-Version

Die gehostete Benutzeroberfläche erfordert Verschlüsselung bei der Übertragung. Benutzerpool-Domänen, die von Amazon Cognito bereitgestellt werden, erfordern mindestens eine TLS-Version von 1.2. Benutzerdefinierte Domänen werden unterstützt, erfordern jedoch keine TLS-Version 1.2. Da Amazon Cognito die Konfiguration der gehosteten UI- und Autorisierungsserver-Endpunkte verwaltet, können Sie die TLS-Anforderungen Ihrer Benutzerpool-Domain nicht ändern.

Die gehostete Benutzeroberfläche und CORS-Richtlinien

Die gehostete Benutzeroberfläche von Amazon Cognito unterstützt keine Cross-Origin Resource Sharing (CORS)-Ursprungsrichtlinien. Eine CORS-Richtlinie in der gehosteten Benutzeroberfläche würde verhindern, dass Benutzer Authentifizierungsparameter in ihren Anfragen übergeben. Implementieren Sie stattdessen eine CORS-Richtlinie im Web-Frontend Ihrer App. Amazon Cognito gibt einen `Access-Control-Allow-Origin: *`-Antwort-Header auf Anfragen an die folgenden OAuth-Endpunkte zurück.

1. Token-Endpunkt
2. Widerrufen des Endpunkts
3. UserInfo-Endpunkt

Cookies für gehostete Benutzeroberflächen und Autorisierungsserver

Die Endpunkte des Amazon Cognito Cognito-Benutzerpools setzen Cookies in den Browsern der Benutzer. Die Cookies entsprechen den Anforderungen einiger Browser, dass Websites keine Cookies von Drittanbietern setzen. Sie sind nur auf die Endpunkte Ihres Benutzerpools beschränkt und beinhalten Folgendes:

- Ein `XSRF-TOKEN` Cookie für jede Anfrage.
- Ein `csrf-state` Cookie für die Sitzungskonsistenz, wenn ein Benutzer umgeleitet wird.
- Ein `cognito` Sitzungscookie, das erfolgreiche Anmeldeversuche eine Stunde lang speichert.

Konfigurieren einer Benutzerpool-Domäne

Nach dem Einrichten eines App-Clients konfigurieren Sie die Adresse Ihrer Registrierungs- und Anmeldungs-Webseiten. Sie können eine von Amazon Cognito gehostete Domäne verwenden und ein verfügbares Präfix für Ihre Domäne wählen, oder Sie können Ihre eigenen Web-Adresse als benutzerdefinierte Domäne verwenden.

Weitere Informationen zum Hinzufügen eines App-Client und einer bei Amazon Cognito gehosteten Domäne mit der AWS Management Console finden Sie unter Hinzufügen einer App zur Aktivierung der gehosteten Web-Benutzeroberfläche.

> **Note**
>
> Sie können den Text `aws`, `amazon` oder `cognito` nicht im Domänenpräfix verwenden.

Themen

- Verwenden der Amazon-Cognito-Domäne für die gehostete Benutzeroberfläche
- Verwenden der eigenen Domäne für die gehostete Benutzeroberfläche

Verwenden der Amazon-Cognito-Domäne für die gehostete Benutzeroberfläche

Nach dem Einrichten eines App-Clients konfigurieren Sie die Adresse Ihrer Registrierungs- und Anmeldungs-Webseiten. Sie können die gehostete Amazon-Cognito-Domäne mit Ihrem eigenen Domain-Präfix verwenden.

> **Note**
>
> Um Ihre Amazon-Cognito-Anwendungen sicherer zu machen, werden die übergeordneten Domains der Benutzerpool-Endpunkte in der Public Suffix List (PSL) registriert. Durch die PSL erlangen die Webbrowser Ihrer Benutzer ein einheitliches Verständnis der Endpunkte Ihres Benutzerpools und der von ihnen gesetzten Cookies.
> Die übergeordneten Domains von Benutzerpool-Endpunkten weisen die folgenden Formate auf.
>
> ```
> auth.Region.amazoncognito.com
> auth-fips.Region.amazoncognito.com
> ```

Informationen zum Hinzufügen eines App-Clients und einer von Amazon Cognito gehosteten Domain mit dem AWS Management Console finden Sie unterErstellen eines App-Clients.

Themen

- Voraussetzungen
- Schritt 1: Konfigurieren einer gehosteten Benutzerpool-Domäne
- Schritt 2: Überprüfen Ihrer Anmeldeseite

Voraussetzungen

Bevor Sie anfangen, benötigen Sie:

- Einen Benutzerpool mit einem App-Client. Weitere Informationen finden Sie unter Erste Schritte mit Benutzerpools.

Schritt 1: Konfigurieren einer gehosteten Benutzerpool-Domäne

Gehostete Benutzerpool-Domäne konfigurieren

Sie können entweder die API AWS Management Console oder die AWS CLI oder verwenden, um eine Benutzerpool-Domain zu konfigurieren.

Amazon Cognito console

Eine Domäne konfigurieren.

1. Navigieren Sie zur Registerkarte App integration (Anwendungsintegration) für Ihren Benutzerpool.
2. Navigieren Sie dann zu Domäne und wählen Sie Aktionen, Benutzerdefinierte Domäne erstellen oder Amazon-Cognito-Domäne erstellen aus. Wenn Sie bereits eine Benutzerpool-Domain konfiguriert haben, wählen Sie Amazon-Cognito-Domäne löschen oder Benutzerdefinierte Domäne löschen aus, bevor Sie Ihre neue benutzerdefinierte Domain erstellen.
3. Geben Sie ein verfügbares Domain-Präfix zur Verwendung mit einer Amazon-Cognito-Domäne ein. Weitere Informationen zum Einrichten einer benutzerdefinierten Domäne finden Sie unter Verwenden der eigenen Domäne für die gehostete Benutzeroberfläche.
4. Wählen Sie Create (Erstellen) aus.

CLI/API

Verwenden Sie die folgenden Befehle zum Erstellen eines benutzerdefinierten Domänen-Präfix. Weisen Sie dieses anschließend Ihrem Benutzerpool zu.

Konfigurieren einer Benutzerpool-Domäne

- AWS CLI: `aws cognito-idp create-user-pool-domain`

  Beispiel: `aws cognito-idp create-user-pool-domain --user-pool-id <user_pool_id> --domain <domain_name>`
- AWS API: CreateUserPoolDomain

So rufen Sie Informationen zu einer Domäne auf

- AWS CLI: `aws cognito-idp describe-user-pool-domain`

  Beispiel: `aws cognito-idp describe-user-pool-domain --domain <domain_name>`
- AWS API: DescribeUserPoolDomain

So löschen Sie eine Domain

- AWS CLI: `aws cognito-idp delete-user-pool-domain`

  Beispiel: `aws cognito-idp delete-user-pool-domain --domain <domain_name>`
- AWS API: DeleteUserPoolDomain

Schritt 2: Überprüfen Ihrer Anmeldeseite

- Stellen Sie sicher, dass die Anmeldeseite Ihrer bei Amazon Cognito gehosteten Domäne erreichbar ist.

```
https://<your_domain>/login?
response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callback_url>
```

Ihre Domäne wird auf der Seite Domänenname der Amazon-Cognito-Konsole angezeigt. Ihre Client-ID der App und die Callback-URL werden auf der Seite App client settings (App-Client-Einstellungen) angezeigt.

Verwenden der eigenen Domäne für die gehostete Benutzeroberfläche

Nach dem Einrichten eines App-Clients können Sie Ihren Benutzerpool mit einer benutzerdefinierten Domäne für die in Amazon Cognito gehostete Benutzeroberfläche und Auth-API-Endpunkte konfigurieren. Mit einer benutzerdefinierten Domäne ermöglichen Sie den Benutzern, Ihre eigene Webadresse für die Anmeldung bei Ihrer Anwendung zu verwenden.

Themen

- Hinzufügen einer benutzerdefinierten Domäne zu einem Benutzerpool
- Ändern des SSL-Zertifikats für die benutzerdefinierte Domäne

Hinzufügen einer benutzerdefinierten Domäne zu einem Benutzerpool

Zum Hinzufügen einer benutzerdefinierten Domäne zum Benutzerpool legen Sie den Domänennamen in der Amazon-Cognito-Konsole fest und stellen ein Zertifikat bereit, das Sie mit AWS Certificate Manager (ACM) verwalten. Wenn Sie die Domäne hinzugefügt haben, stellt Amazon Cognito ein Aliasziel zur Verfügung, das Sie zur DNS-Konfiguration hinzufügen.

Voraussetzungen

Bevor Sie anfangen, benötigen Sie:

- Einen Benutzerpool mit einem App-Client. Weitere Informationen finden Sie unter Erste Schritte mit Benutzerpools.
- Eine Web-Domäne, die Sie besitzen. Die übergeordnete Domain muss über einen gültigen A-Eintrag im DNS verfügen. Sie können diesem Datensatz einen beliebigen Wert zuweisen. Die übergeordnete Domain kann die Wurzel der Domain oder eine untergeordnete Domain sein, die in der Domainhierarchie eine Stufe höher ist. Wenn Ihre benutzerdefinierte Domain beispielsweise auth.xyz.example.com lautet, muss Amazon Cognito in der Lage sein, xyz.example.com in eine IP-Adresse aufzulösen. Damit versehentliche Auswirkungen auf die Kundeninfrastruktur verhindert werden, unterstützt Amazon Cognito die Verwendung von Top-Level-Domänen (TLDs) für benutzerdefinierte Domänen nicht. Weitere Informationen finden Sie unter Domänennamen.
- Die Möglichkeit zum Erstellen einer Unterdomäne für die benutzerdefinierte Domäne. Wir empfehlen die Verwendung von auth als die Subdomäne. Zum Beispiel: *`auth.example.com`*.

  > **Note**
  >
  > Möglicherweise müssen Sie ein neues Zertifikat für die Subdomäne Ihrer benutzerdefinierten Domäne beschaffen, wenn Sie kein Platzhalterzertifikat (Wildcard Certificate) haben.

- Ein von ACM verwaltetes Secure-Sockets-Layer-(SSL)-Zertifikat.

  > **Note**
  >
  > Sie müssen die AWS Region in der ACM-Konsole auf USA Ost (Nord-Virginia) ändern, bevor Sie ein Zertifikat anfordern oder importieren können.

- Eine Anwendung, die es Ihrem Benutzerpool-Autorisierungsserver ermöglicht, Cookies zu Benutzersitzungen hinzuzufügen. Amazon Cognito setzt mehrere erforderliche Cookies für die gehostete Benutzeroberfläche. Dies sind beispielsweise `cognito`, `cognito-fl` und `XSRF-TOKEN`. Obwohl jedes einzelne Cookie den Größenbeschränkungen des Browsers entspricht, können Änderungen an Ihrer Benutzerpool-Konfiguration dazu führen, dass gehostete UI-Cookies an Größe zunehmen. Ein Zwischendienst wie ein Application Load Balancer (ALB) vor Ihrer benutzerdefinierten Domain kann eine maximale Header-Größe oder Gesamtcookie-Größe erzwingen. Wenn Ihre Anwendung auch ihre eigenen Cookies setzt, können die Sitzungen Ihrer

Benutzer diese Grenzwerte überschreiten. Um Konflikte mit Größenbeschränkungen zu vermeiden, empfehlen wir, dass Ihre Anwendung keine Cookies in der gehosteten UI-Subdomain setzt.

- Erlaubnis zur Aktualisierung von CloudFront Amazon-Distributionen. Fügen Sie hierzu die folgende IAM-Richtlinienanweisung einem Benutzer in Ihrem AWS-Konto an:

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Sid": "AllowCloudFrontUpdateDistribution",
            "Effect": "Allow",
            "Action": [
                "cloudfront:updateDistribution"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Weitere Informationen zur Autorisierung von Aktionen finden Sie unter Verwenden identitätsbasierter Richtlinien (IAM-Richtlinien) für. CloudFront CloudFront

Amazon Cognito verwendet zunächst Ihre IAM-Berechtigungen, um die CloudFront Verteilung zu konfigurieren, aber die Verteilung wird von verwaltet. AWS Sie können die Konfiguration der CloudFront Distribution, die Amazon Cognito Ihrem Benutzerpool zugeordnet hat, nicht ändern. Sie können also beispielsweise nicht die unterstützten TLS-Versionen in der Sicherheitsrichtlinie aktualisieren.

Schritt 1: Eingeben des benutzerdefinierten Domänennamens

Sie können dem Benutzerpool Ihre Domäne mithilfe der Amazon-Cognito-Konsole oder der API hinzufügen.

Amazon Cognito console

Domäne dem Benutzerpool über die Amazon-Cognito-Konsole hinzufügen:

1. Melden Sie sich bei der Amazon Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS -Anmeldeinformationen ein.
2. Klicken Sie auf User pools (Benutzerpools).
3. Wählen Sie den Benutzerpool aus, dem Sie die Domäne hinzufügen möchten.
4. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus.
5. Gehen Sie als nächstes zu Domain (Domäne), wählen Sie Actions (Aktionen) und dann Create custom domain (Benutzerdefinierte Domäne erstellen) aus.

   > **Note**
   >
   > Wenn Sie bereits eine Benutzerpool-Domäne konfiguriert haben, wählen Sie Delete Cognito domain (Cognito-Domäne löschen) oder Delete custom domain (Benutzerdefinierte Domäne löschen) aus, um eine bestehende Domäne zu löschen, bevor Sie Ihre neue benutzerdefinierte Domäne erstellen.

6. Geben Sie für Custom domain (Benutzerdefinierte Domäne) die URL der Domäne ein, die Sie mit Amazon Cognito verwenden möchten. Der Domänenname darf nur Kleinbuchstaben, Zahlen und Bindestriche enthalten. Verwenden Sie keinen Bindestrich als erstes oder letztes Zeichen. Trennen Sie die Namen von Unterdomänen durch Punkte.
7. Wählen Sie für ACM certificate (ACM-Zertifikat) das SSL-Zertifikat aus, das Sie für die Domäne verwenden möchten. Nur ACM-Zertifikate in USA Ost (Nord-Virginia) können unabhängig von Ihrem Benutzerpool mit einer benutzerdefinierten Amazon Cognito Cognito-Domain verwendet werden. AWS-Region

   Wenn Sie kein verfügbares Zertifikat haben, können Sie ACM verwenden, um eines in USA Ost (Nord-Virginia) bereitzustellen. Weitere Informationen finden Sie unter Erste Schritte im AWS Certificate Manager -Benutzerhandbuch.
8. Wählen Sie Create (Erstellen) aus.
9. Amazon Cognito bringt Sie zurück zur Registerkarte App integration (Anwendungsintegration) zurück. Es wird Ihnen eine Nachricht mit dem Titel Create an alias record in your domain's DNS (Erstellen eines Alias-Datensatzes im DNS der Domäne) angezeigt. Notieren Sie die Domäne und das Alias-Ziel, das in der Konsole angezeigt wird. Sie werden im nächsten Schritt verwendet, um den Datenverkehr auf Ihre benutzerdefinierte Domäne weiterzuleiten.

API

Domäne mit Amazon-Cognito-API zu Benutzerpool hinzufügen:

- Verwenden Sie die Aktion „CreateUserPoolDomain“.

Schritt 2: Hinzufügen eines Alias-Ziels und einer Subdomäne

In diesem Schritt richten Sie einen Alias über Ihren Domain Name Server (DNS)-Serviceanbieter ein, der auf das Alias-Ziel aus dem vorherigen Schritt zurück verweist. Wenn Sie Amazon Route 53 für die DNS-Adresse Auflösung verwenden, wählen Sie den Abschnitt Hinzufügen eines Alias-Ziels und einer Subdomäne mit Route 53.

Hinzufügen eines Alias-Ziels und einer Subdomäne zu Ihrer aktuellen DNS-Konfiguration

- Wenn Sie nicht Route 53 für die DNS-Adressauflösung verwenden, müssen Sie die Konfigurationstools Ihres DNS-Serviceanbieters verwenden, um das Alias-Ziel aus dem vorherigen Schritt zum DNS-Datensatz Ihrer Domäne hinzuzufügen. Ihre DNS-Anbieter muss außerdem die Subdomäne für Ihre benutzerdefinierte Domäne einrichten.

Hinzufügen eines Alias-Ziels und einer Subdomäne mit Route 53

1. Melden Sie sich bei der Route-53-Konsole an. Geben Sie bei Aufforderung Ihre AWS -Anmeldeinformationen ein.
2. Wenn Sie keine gehostete Zone in Route 53 haben, erstellen Sie eine mit einem Stamm, der Ihrer benutzerdefinierten Domain übergeordnet ist. Weitere Informationen finden Sie unter

   a. Wählen Sie Create Hosted Zone.

   b. Geben Sie beispielsweise die übergeordnete Domäne *auth.example.com* Ihrer benutzerdefinierten Domain ein oder beispielsweise *myapp.auth.example.com* aus der Liste Domänenname.

   c. Geben Sie eine Beschreibung für Ihre gehostete Zone ein.

   d. Wählen Sie einen Typ einer gehosteten Zone aus Public hosted zone (Öffentliche gehostete Zone), damit öffentliche Clients die benutzerdefinierte Domäne auflösen können. Die Auswahl von Private hosted zone (Private gehostete Zone) wird nicht unterstützt.

   e. Wenden Sie Tags wie gewünscht an.

   f. Wählen Sie Erstellte gehostete Zone.

> **Note**
>
> Sie können auch eine neue gehostete Zone für Ihre benutzerdefinierte Domäne und einen Delegationssatz in der übergeordneten gehosteten Zone erstellen, der Abfragen an die gehostete Zone der Subdomäne weiterleitet. Andernfalls erstellen Sie einen A-Eintrag. Diese Methode bietet mehr Flexibilität und Sicherheit bei Ihren gehosteten Zonen. Weitere Informationen finden Sie unter Creating a subdomain for a domain hosted through Amazon Route 53 (Erstellen einer Subdomäne für eine über Amazon Route 53 gehostete Domäne).

3. Wählen Sie auf der Seite Hosted Zones (Gehostete Zonen) den Namen Ihrer gehosteten Zone aus.
4. Fügen Sie einen DNS-Eintrag für die übergeordnete Domain Ihrer benutzerdefinierten Domain hinzu, falls Sie noch keinen haben. Fügen Sie einen A DNS-Eintrag für die übergeordnete Domain hinzu und wählen Sie Einträge erstellen. Im Folgenden finden Sie ein Beispieldatensatz für die Domäne *auth.example.com*.

   ```
   auth.example.com. 60 IN A 198.51.100.1
   ```

   > **Note**
   >
   > Amazon Cognito überprüft, dass es einen DNS-Datensatz für die übergeordnete Domäne Ihrer benutzerdefinierten Domäne gibt, um sich vor versehentlichem Hijacking von Produktionsdomänen zu schützen. Wenn Sie keinen DNS-Datensatz für die übergeordnete Domäne haben, gibt Amazon Cognito einen Fehler zurück, wenn Sie versuchen, die benutzerdefinierte Domäne festzulegen. Ein SOA-Eintrag (Start of Authority) ist kein ausreichender DNS-Eintrag für die Überprüfung der übergeordneten Domain.

5. Fügen Sie einen DNS-Eintrag für Ihre benutzerdefinierte Domain hinzu. Ihr Eintrag muss beispielsweise auf das Alias-Ziel der benutzerdefinierten Domain verweisen`123example.cloudfront.net`. Wählen Sie erneut Create record (Datensatz erstellen) aus.
6. Geben Sie einen Datensatznamen ein, der zum Namen Ihrer benutzerdefinierten Domäne passt, beispielsweise *myapp* zum Erstellen eines Datensatzes für *myapp.auth.example.com*.
7. Aktivieren Sie die Option Alias.

8. Wählen Sie unter Route traffic to (Datenverkehr weiterleiten an) die Option Alias to Cloudfront distribution (Alias an Cloudfront Distribution) aus. Geben Sie das von Amazon Cognito bereitgestellte Alias-Ziel ein, nachdem Sie die benutzerdefinierte Domäne erstellt haben.
9. Wählen Sie Create records (Datensätze erstellen) aus.

> **Note**
>
> Es kann etwa 60 Sekunden dauern, bis Ihre neuen Datensätze auf alle Route-53-DNS-Server übertragen werden. Sie können die Route GetChange53-API-Methode verwenden, um zu überprüfen, ob Ihre Änderungen übernommen wurden.

Schritt 3: Überprüfen Ihrer Anmeldeseite

- Stellen Sie sicher, dass die Anmeldeseite Ihrer benutzerdefinierten Domäne erreichbar ist.

  Melden Sie sich bei Ihrer benutzerdefinierten Domäne und Subdomäne an, indem Sie diese Adresse in Ihren Browser eingeben. Dies ist eine Beispiel-URL der benutzerdefinierten Domäne *example.com* mit der Subdomäne *auth*:

```
https://myapp.auth.example.com/login?
response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callback_url>
```

Ändern des SSL-Zertifikats für die benutzerdefinierte Domäne

Falls erforderlich, können Sie mit Amazon Cognito das Zertifikat ändern, das Sie für Ihre benutzerdefinierte Domäne übernommen haben.

Im Falle der routinemäßigen Zertifikaterneuerung mit ACM ist dies in der Regel nicht notwendig. Wenn Sie das vorhandene Zertifikat in ACM erneuern, bleibt der ARN für Ihr Zertifikat gleich und die benutzerdefinierte Domäne verwendet automatisch das neue Zertifikat.

Wenn Sie jedoch das vorhandene Zertifikat durch ein neues ersetzen, gibt ACM dem neuen Zertifikat einen neuen ARN. Sie müssen diesen ARN an Amazon Cognito übermitteln, um das neue Zertifikat für den benutzerdefinierten Domänennamen zu übernehmen.

Nachdem Sie Ihr neues Zertifikat bereitgestellt haben, dauert es max. eine Stunde, bis Amazon Cognito das Zertifikat an die benutzerdefinierte Domäne verteilt hat.

> **Bevor Sie beginnen**
>
> Bevor Sie das Zertifikat in Amazon Cognito ändern können, müssen Sie es ACM hinzufügen. Weitere Informationen finden Sie unter Erste Schritte im AWS Certificate Manager - Benutzerhandbuch.
> Wenn Sie Ihr Zertifikat zu ACM hinzufügen, müssen Sie als AWS -Region USA Ost (Nord-Virginia) auswählen.

Sie können das Zertifikat mit der Amazon-Cognito-Konsole oder der API ändern.

AWS Management Console

Zertifikat über die Amazon-Cognito-Konsole erneuern:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon Cognito Cognito-Konsole unterhttps://console.aws.amazon.com/cognito/home.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie den Benutzerpool aus, für den Sie das Zertifikat aktualisieren möchten.
4. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus.
5. Wählen Sie Actions (Aktionen), Edit ACM certificate (ACM-Zertifikat bearbeiten) aus.
6. Wählen Sie das neue Zertifikat aus, das Sie Ihrer benutzerdefinierten Domäne zuordnen möchten.
7. Wählen Sie Save Changes.

API

So erneuern Sie ein Zertifikat (Amazon-Cognito-API)

- Verwenden Sie die Aktion „UpdateUserPoolDomain“.

Anpassen der integrierten Registrierungs- und Anmeldungswebseiten

Verwenden Sie die AWS Management Console oder AWS CLI oder API, um Anpassungseinstellungen für die integrierte App-Benutzeroberfläche anzugeben. Sie können ein benutzerdefiniertes Logo-Image hochladen, das Sie in der App anzeigen möchten. Sie können auch Cascading Stylesheets (CSS) verwenden, um das Layout der Benutzeroberfläche anzupassen.

Sie können die Einstellungen für die Anpassung der App-Benutzeroberfläche für einen einzelnen Client angeben (mit einem spezifischen `clientId`), oder für alle Clients (durch Festlegen der `clientId` auf `ALL`). Wenn Sie `ALL` angeben, wird die Standard-Konfiguration für jeden Client verwende, für den zuvor keine Anpassung der Benutzeroberfläche eingestellt wurde. Wenn Sie die Benutzeroberflächen-Einstellungen für einen bestimmten Client angeben, greift er nicht mehr auf die Konfiguration `ALL` zurück.

Die Anforderung, die die Anpassung Ihrer Benutzeroberfläche festlegt, darf eine Größe von 135 KB nicht überschreiten. In seltenen Fällen kann es sein, dass die Anforderungs-Header, Ihre CSS-Datei und Ihr Logo zusammen 135 KB überschreiten. Amazon Cognito codiert die Bilddatei in Base64. Dadurch erhöht sich die Größe eines Bilds mit 100 KB auf 130 KB, so dass fünf KB für Anforderungs-Header und Ihre CSS-Datei übrig bleiben. Wenn die Anforderung zu groß ist, gibt die AWS Management Console oder Ihre API-Anforderung `SetUICustomization` einen Fehler `request parameters too large` zurück. Passen Sie Ihr Logobild so an, dass es nicht größer als 100 KB ist, und Ihre CSS-Datei so, dass sie nicht größer als 3 KB ist. Sie können CSS- und Logo-Anpassungen nicht separat festlegen.

> **Note**
>
> Um Ihre Benutzeroberfläche anzupassen, müssen Sie eine Domain für Ihren Benutzerpool einrichten.

Angabe eines benutzerdefinierten Logos für die App

Amazon Cognito zentriert Ihr benutzerdefiniertes Logo über den Eingabefeldern auf dem Login-Endpunkt.

Wählen Sie für Ihr benutzerdefiniertes gehostetes Logo der Benutzeroberfläche eine PNG-, JPG- oder JPEG-Datei, die auf 350 mal 178 Pixel skaliert werden kann. Ihre Logodatei darf nicht größer als 100 KB bzw. 130 KB nach der Codierung in Base64 durch Amazon Cognito sein. Um eine `ImageFile` in SetUICustomization in der API festzulegen, können Sie Ihre Datei in eine Base64-codierte Textzeichenfolge konvertieren oder in der AWS CLI einen Dateipfad angeben und Amazon Cognito die Codierung überlassen.

Angabe von CSS-Anpassungen für die App

Sie können CSS für die gehosteten App-Seiten anpassen. Dabei gelten die folgenden Einschränkungen:

- Sie können einen der folgenden CSS-Klassennamen verwenden:
  - `background-customizable`
  - `banner-customizable`
  - `errorMessage-customizable`
  - `idpButton-customizable`
  - `idpButton-customizable:hover`
  - `idpDescription-customizable`
  - `inputField-customizable`
  - `inputField-customizable:focus`
  - `label-customizable`
  - `legalText-customizable`
  - `logo-customizable`
  - `passwordCheck-valid-customizable`
  - `passwordCheck-notValid-customizable`
  - `redirect-customizable`
  - `socialButton-customizable`
  - `submitButton-customizable`
  - `submitButton-customizable:hover`
  - `textDescription-customizable`
- Eigenschaftswerte können HTML enthalten, mit Ausnahme der folgenden Werte: `@import`, `@supports`, `@page`, `@media`-Anweisungen oder JavaScript.

Sie können die folgenden CSS-Eigenschaften anpassen:

Bezeichnungen

- font-weight ist ein Vielfaches von 100 von 100 bis 900.

Eingabefelder

- width ist die Breite als Prozentsatz des umschließenden Blocks.
- height ist die Höhe des Eingabefeldes in Pixel (px).
- color ist die Textfarbe. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

- background-color ist die Hintergrundfarbe des Eingabefeldes. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- border ist ein standardmäßiger CSS-Rahmenwert, der die Breite, Transparenz und Farbe des Rahmens Ihres App-Fensters angibt. Die Breite kann ein beliebiger Wert zwischen 1px zu 100px sein. Transparenz kann vollständig oder nicht transparent sein. Bei der Farbe kann es sich um jeden standardmäßigen Farbwert handeln.

Textbeschreibungen

- padding-top ist der Abstand oberhalb der Beschreibung.
- padding-bottom ist der Abstand unterhalb der Beschreibung.
- display kann `block` oder `inline` sein.
- font-size ist die Schriftgröße für Textbeschreibungen.

Absenden-Schaltfläche

- font-size ist die Schriftgröße für den Schaltflächentext.
- font-weight ist die Schriftauszeichnung für den Schaltflächentext: `bold`, `italic` oder `normal`.
- margin ist eine Zeichenfolge aus 4 Werten, die die Seitenränder oben, rechts, unten und links für die Schaltfläche angeben.
- font-size ist die Schriftgröße für Textbeschreibungen.
- width ist die Breite des Schaltflächentexts als Prozentwert des umschließenden Blocks.
- height ist die Höhe der Schaltfläche in Pixel (px).
- color ist die Farbe der Schaltfläche. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- background-color ist die Hintergrundfarbe der Schaltfläche. Dabei kann es sich um jeden standardmäßigen Farbwert handeln.

Banner

- padding ist eine Zeichenfolge aus 4 Werten, die die Abstandsgrößen oben, rechts, unten und links für das Banner angeben.
- background-color ist die Hintergrundfarbe der Banners. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

Absenden-Schaltfläche, wenn der Mauszeiger auf sie geschoben wird

- color ist die Vordergrundfarbe der Schaltfläche, wenn Sie den Mauszeiger auf sie schieben. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

- background-color ist die Hintergrundfarbe der Schaltfläche, wenn Sie den Mauszeiger auf sie schieben. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

Identitätsanbieter-Schaltfläche, wenn der Mauszeiger auf sie geschoben wird

- color ist die Vordergrundfarbe der Schaltfläche, wenn Sie den Mauszeiger auf sie schieben. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- background-color ist die Hintergrundfarbe der Schaltfläche, wenn Sie den Mauszeiger auf sie schieben. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

Password-Prüfung nicht gültig

- color ist die Textfarbe der `"Password check not valid"`-Meldung. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

Hintergrund

- background-color ist die Hintergrundfarbe des App-Fensters. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

Fehlermeldungen

- margin ist eine Zeichenfolge aus 4 Werten, die die Seitenränder oben, rechts, unten und links angeben.
- padding ist die Abstandsgröße.
- font-size ist die Schriftgröße.
- width ist die Breite der Fehlermeldung als Prozentsatz des umschließenden Blocks.
- background ist die Hintergrundfarbe der Fehlermeldung. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- border ist eine Zeichenfolge von 3 Werten für die Breite, Transparenz und Farbe des Rahmens angeben.
- color ist die Textfarbe der Fehlermeldung. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- box-sizing wird verwendet, um dem Browser mitzuteilen, welche Größeneigenschaften (Breite und Höhe) verwendet werden sollen.

Identitätsanbieter-Schaltflächen

- height ist die Höhe der Schaltfläche in Pixel (px).
- width ist die Breite des Schaltflächentexts als Prozentwert des umschließenden Blocks.
- text-align ist die Einstellung für die Textausrichtung. Sie kann `left`, `right` oder `center` sein.
- margin-bottom ist die Einstellung für den unteren Rand.

- color ist die Farbe der Schaltfläche. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- background-color ist die Hintergrundfarbe der Schaltfläche. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- border-color ist die Farbe des Schaltflächenrahmens. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

Identitätsanbieter-Beschreibung

- padding-top ist der Abstand oberhalb der Beschreibung.
- padding-bottom ist der Abstand unterhalb der Beschreibung.
- display kann `block` oder `inline` sein.
- font-size ist die Schriftgröße für Beschreibungen.

Rechtliche Hinweise

- color ist die Textfarbe. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- font-size ist die Schriftgröße.

> **Note**
>
> Wenn Sie rechtliche Hinweise anpassen, passen Sie die Nachricht Wir werden auf keinem Ihrer Konten posten, ohne vorher zu fragen an, die auf der Anmeldeseite unter Social-Identity-Anbieter angezeigt wird.

Logo

- max-width ist die maximale Breite als Prozentsatz des umschließenden Blocks.
- max-height ist die maximale Höhe als Prozentsatz des umschließenden Blocks.

Eingabefeld-Fokus

- border-color ist die Farbe des Eingabefelds. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.
- outline ist die Rahmenbreite des Eingabefeldes in Pixel (px).

Sozial-Schaltfläche

- height ist die Höhe der Schaltfläche in Pixel (px).
- text-align ist die Einstellung für die Textausrichtung. Sie kann `left`, `right` oder `center` sein.

- width ist die Breite des Schaltflächentexts als Prozentwert des umschließenden Blocks.
- margin-bottom ist die Einstellung für den unteren Rand.

Password-Prüfung gültig

- color ist die Textfarbe der `"Password check valid"`-Meldung. Dabei kann es sich um jeden standardmäßigen CSS-Farbwert handeln.

Angabe von App-Anpassungseinstellungen für die Benutzeroberfläche eines Benutzerpools (AWS Management Console)

Sie können die AWS Management Console verwenden, um Anpassungen für die App-Benutzeroberflächen-Einstellungen für Ihre App anzugeben.

> **Note**
>
> Sie sehen die gehostete Benutzeroberfläche mit Ihren Anpassungen, indem Sie die folgende URL mit den spezifischen Angaben für Ihren Benutzerpool erstellen und in einen Browser eingeben: `https://<your_domain>/login?response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callbac`
> Möglicherweise müssen Sie bis zu einer Minute warten, um Ihren Browser zu aktualisieren, damit die auf der Konsole vorgenommenen Änderungen angezeigt werden.
> Ihre Domäne wird auf der Registerkarte App integration (Anwendungsintegration) unter Domain (Domäne) angezeigt. Die App-Client-ID und die Rückruf-URL werden unter App clients (App-Clients) angezeigt.

So geben Sie die App-Anpassungseinstellungen für Ihre Benutzeroberfläche an

1. Melden Sie sich bei der Amazon Cognito-Konsole an.
2. Wählen Sie im Navigationsbereich User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus.
4. Um die UI-Einstellungen für alle App-Clients anzupassen, suchen Sie nach Hosted UI customization (Anpassung der gehosteten Benutzeroberfläche) und wählen Edit (Bearbeiten) aus.
5. Um die UI-Einstellungen für einen App-Client anzupassen, suchen Sie nach App-Clients, und wählen Sie den App-Client aus, den Sie ändern möchten. Suchen Sie dann nach Anpassung der

gehosteten Benutzeroberfläche, und wählen Sie Bearbeiten aus. Um für einen App-Client von der Standardkonfiguration des Benutzerpools zur Client-spezifischen Anpassung zu wechseln, wählen Sie Use client-level settings (Client-Level-Einstellungen) aus.

6. Um Ihre eigene Logo-Bilddatei hochzuladen, wählen Sie Choose file (Datei auswählen) oder Replace current file (Aktuelle Datei ersetzen) aus.
7. Um das gehostete UI-CSS anzupassen, laden Sie CSS template.css herunter und ändern Sie die Vorlage mit den Werten, die Sie anpassen möchten. Nur die Schlüssel, die in der Vorlage enthalten sind, können mit der gehosteten Benutzeroberfläche verwendet werden. Hinzugefügte CSS-Schlüssel werden nicht in Ihrer Benutzeroberfläche wiedergegeben. Nachdem Sie die CSS-Datei angepasst haben, wählen Sie Choose file (Datei auswählen) oder Replace current file (Aktuelle Datei ersetzen), um Ihre benutzerdefinierte CSS-Datei hochzuladen.

Angeben von Anpassungseinstellungen für die App-Benutzeroberfläche eines Benutzerpools (AWS CLI- and AWS-API)

Verwenden Sie die folgenden Befehle für die Angabe von Einstellungen der Anpassung der App-Benutzeroberfläche für Ihren Benutzerpool.

Verwenden Sie die folgenden API-Operationen, um die Einstellungen für die Anpassung der Benutzeroberfläche für eine integrierte App-Benutzeroberfläche eines Benutzerpools abzurufen.

- AWS CLI: `aws cognito-idp get-ui-customization`
- AWS-API: GetUICustomization

Verwenden Sie die folgenden API-Operationen, um die Einstellungen für die Anpassung der Benutzeroberfläche für eine integrierte App-Benutzeroberfläche eines Benutzerpools festzulegen.

- AWS CLI von Bilddatei: `aws cognito-idp set-ui-customization --user-pool-id <your-user-pool-id> --client-id <your-app-client-id> --image-file fileb://"<path-to-logo-image-file>" --css ".label-customizable{ color: <color>;}"`
- AWS CLI mit Bild, das als Base64-Binärtext codiert ist: `aws cognito-idp set-ui-customization --user-pool-id <your-user-pool-id> --client-id <your-app-client-id> --image-file <base64-encoded-image-file> --css ".label-customizable{ color: <color>;}"`
- AWS-API: SetUICustomization

Registrieren und Anmelden mit der gehosteten Benutzeroberfläche

Nachdem Sie die von Amazon Cognito gehostete Benutzeroberfläche für Ihren Benutzerpool und Ihre App-Clients konfiguriert und angepasst haben, kann Ihre App sie Ihren Benutzern präsentieren. Die gehostete Benutzeroberfläche unterstützt mehrere Authentifizierungsvorgänge von Amazon Cognito, einschließlich der folgenden Beispiele.

- Sich als neuer Benutzer in Ihrer App anmelden
- Eine E-Mail-Adresse oder Telefonnummer verifizieren
- Multi-Faktor-Authentifizierung (MFA) einrichten
- Melden Sie sich mit einem lokalen Benutzernamen und Passwort an
- Anmelden mit einem externen Identitätsanbieter (IDP)
- Passwort zurücksetzen

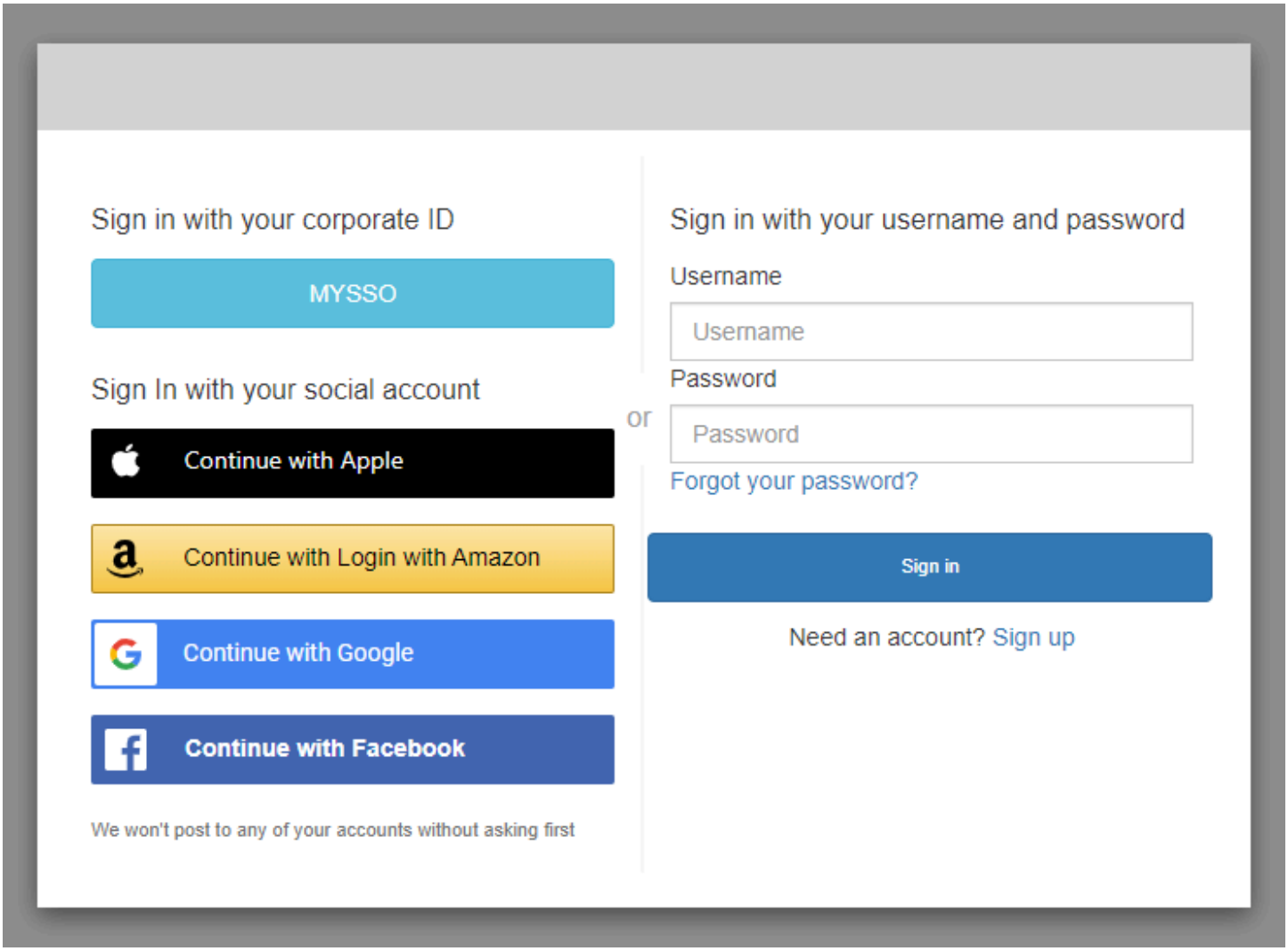
Die von Amazon Cognito gehostete Benutzeroberfläche beginnt am Login-Endpunkt. Die URL zu Ihrer Anmeldeseite ist eine Kombination aus der Domäne, die Sie für Ihren Benutzerpool ausgewählt haben, und Parametern, die die OAuth-2.0-Erteilungen, die Sie ausgeben möchten, Ihren App-Client, den Pfad zu Ihrer App und die OpenID Connect (OIDC)-Bereiche widerspiegeln, die Sie anfordern möchten.

```
https://<your user pool domain>/authorize?client_id=<your app client
 ID>&response_type=<code/token>&scope=<scopes to request>&redirect_uri=<your callback
 URL>
```

Die folgende URL ersetzt die oben angegebenen Platzhalterfelder durch Beispielwerte.

```
https://auth.example.com/authorize? /
client_id=1example23456789 /
&response_type=code /
&scope=aws.cognito.signin.user.admin+email+openid+profile /
&redirect_uri=https%3A%2F%2Faws.amazon.com
```

Die Anmeldeseite für die von Amazon Cognito gehostete Benutzeroberfläche bietet Optionen zum Anmelden über den Benutzerpool oder Identitätsanbieter (IDPs), die Sie dem App-Client zugewiesen haben, den Ihr Benutzer anfordert. Sie enthält auch Links, um sich für ein neues Benutzerkonto im Benutzerpool anzumelden oder ein vergessenes Passwort zurückzusetzen.



Themen

- So registrieren Sie sich für ein neues Konto bei der von Amazon Cognito gehosteten Benutzeroberfläche
- So melden Sie sich bei der von Amazon Cognito gehosteten Benutzeroberfläche an
- So setzen Sie ein Passwort auf der von Amazon Cognito gehosteten Benutzeroberfläche zurück

So registrieren Sie sich für ein neues Konto bei der von Amazon Cognito gehosteten Benutzeroberfläche

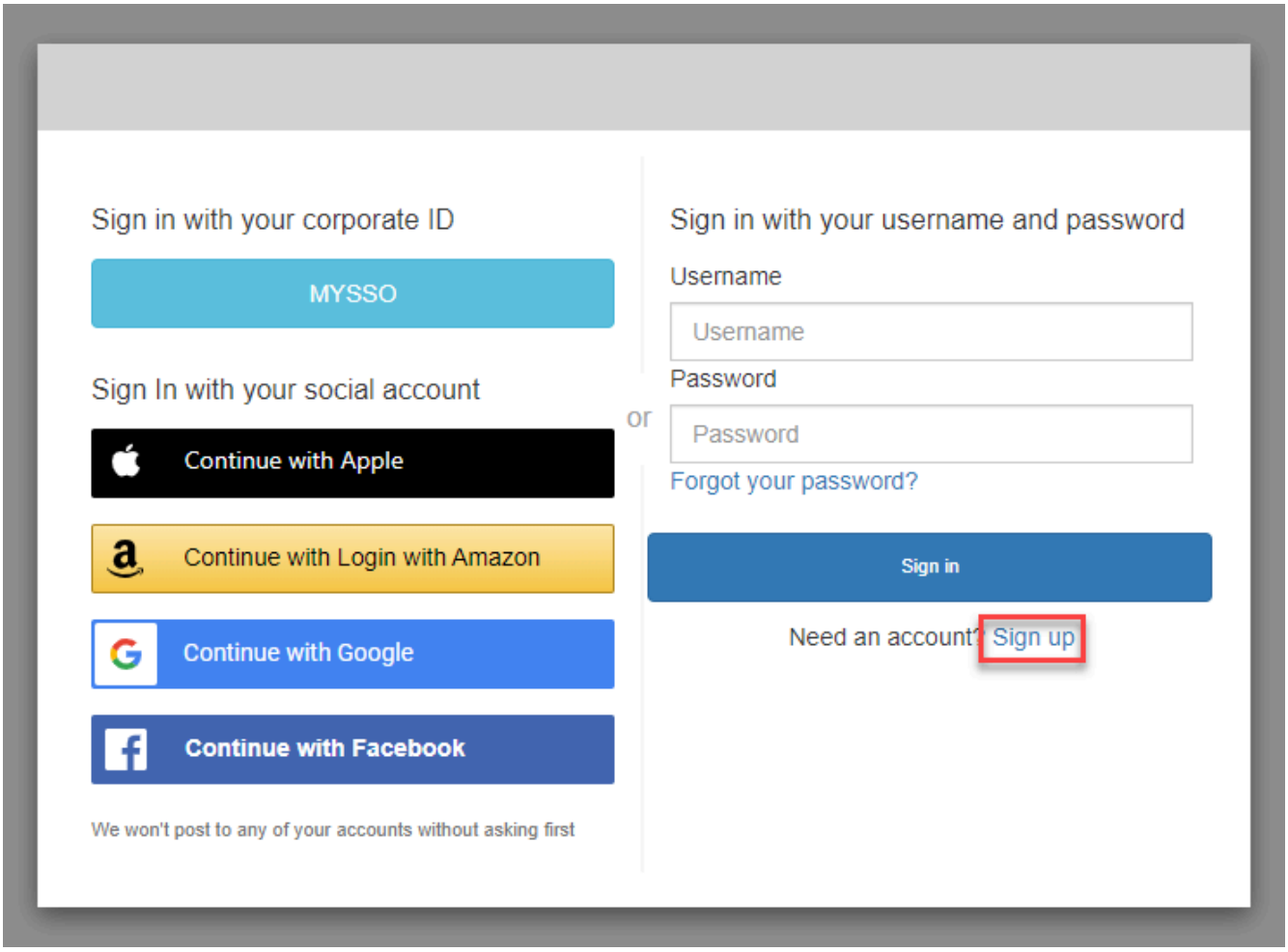
In dieser Anleitung erfahren Sie, wie Sie sich bei Apps, die Amazon Cognito verwenden, für ein Benutzerkonto registrieren.

> **Note**
>
> Wenn Sie sich bei einer App anmelden, die die von Amazon Cognito gehostete Benutzeroberfläche (UI) verwendet, wird möglicherweise eine Seite angezeigt, die der App-Besitzer über die in diesem Handbuch dargestellte Grundkonfiguration hinaus angepasst hat.

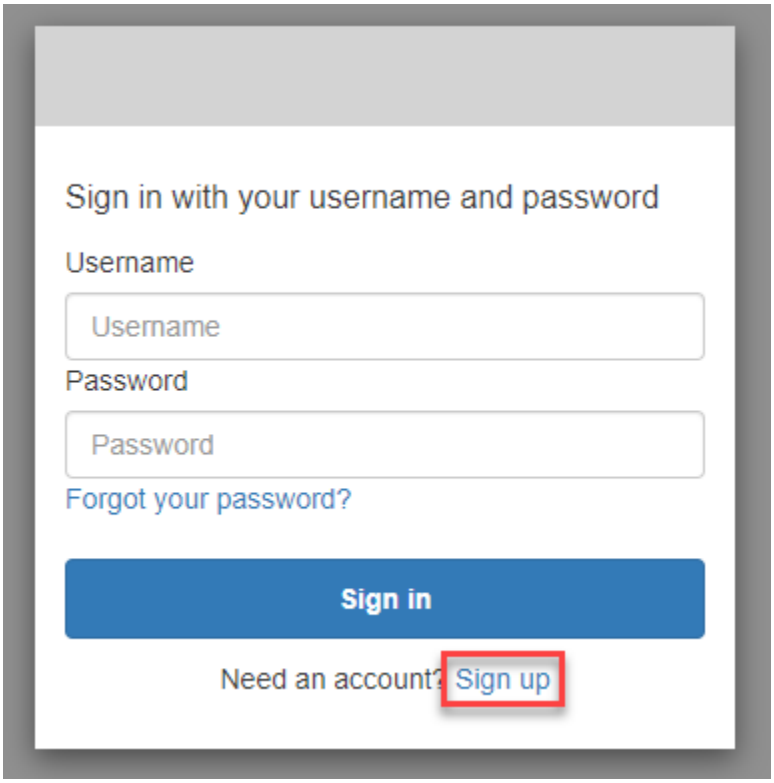
1. Wählen Sie Sign up (Registrieren) auf der Anmeldeseite aus, wenn Sie sich über Amazon Cognito mit einem Benutzernamen und Passwort anmelden möchten, anstatt über einen der externen Identitätsanbieter, die der App-Besitzer aufgeführt hat.

   Wenn es sich bei Ihrem Identitätsanbieter um einen anderen Anbieter als Amazon Cognito handelt, ist Ihre Anmeldung abgeschlossen, nachdem Sie die Schaltfläche für Ihren Drittanbieter ausgewählt haben. Abhängig von den Optionen, die der App-Besitzer ausgewählt hat, stehen Ihnen möglicherweise verschiedene Anbieter zur Auswahl, mit denen Sie sich anmelden können. Eventuell wird nur eine Aufforderung zur Eingabe eines Benutzernamens und Passworts angezeigt.

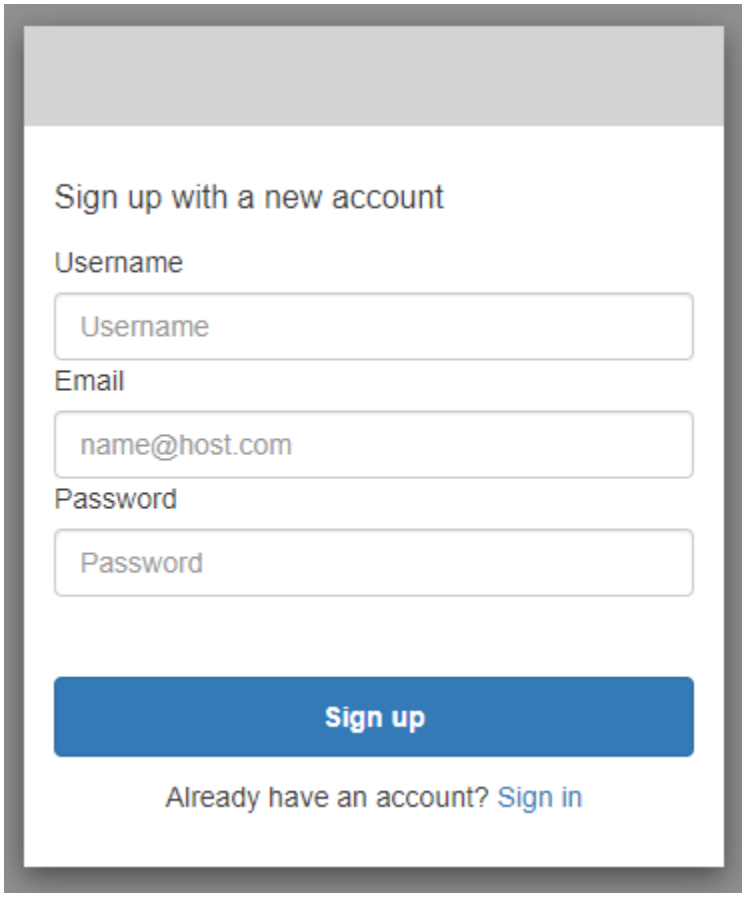
With multiple sign-in providers



With only Amazon Cognito as a sign-in provider

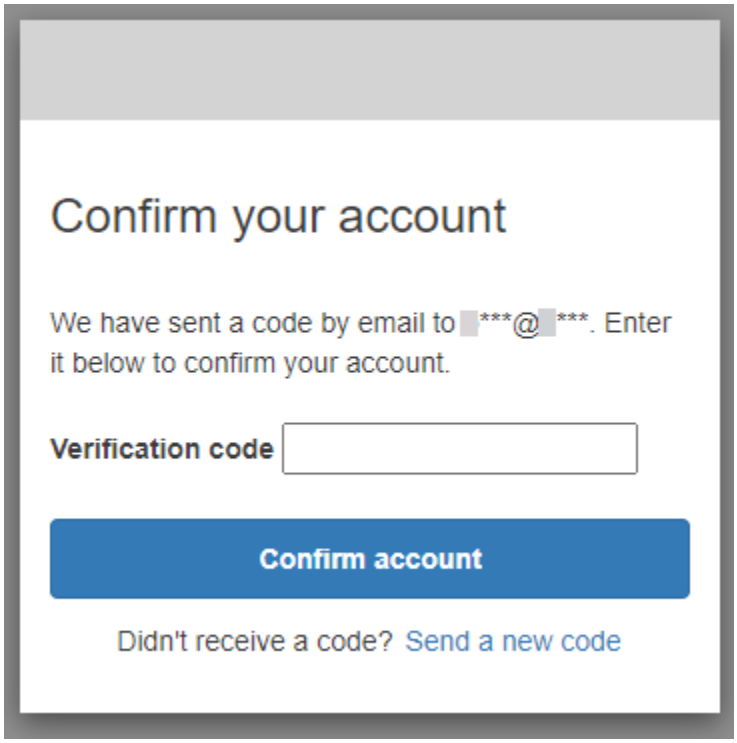


2. Auf der Seite Sign up with a new account (Mit neuem Konto anmelden) fragt der App-Besitzer nach den Informationen, die er für die Anmeldung benötigt. Er fragt möglicherweise nach einem Benutzernamen, einer E-Mail-Adresse oder einer Telefonnummer. Geben Sie die erforderlichen Informationen ein und wählen Sie ein Passwort aus.



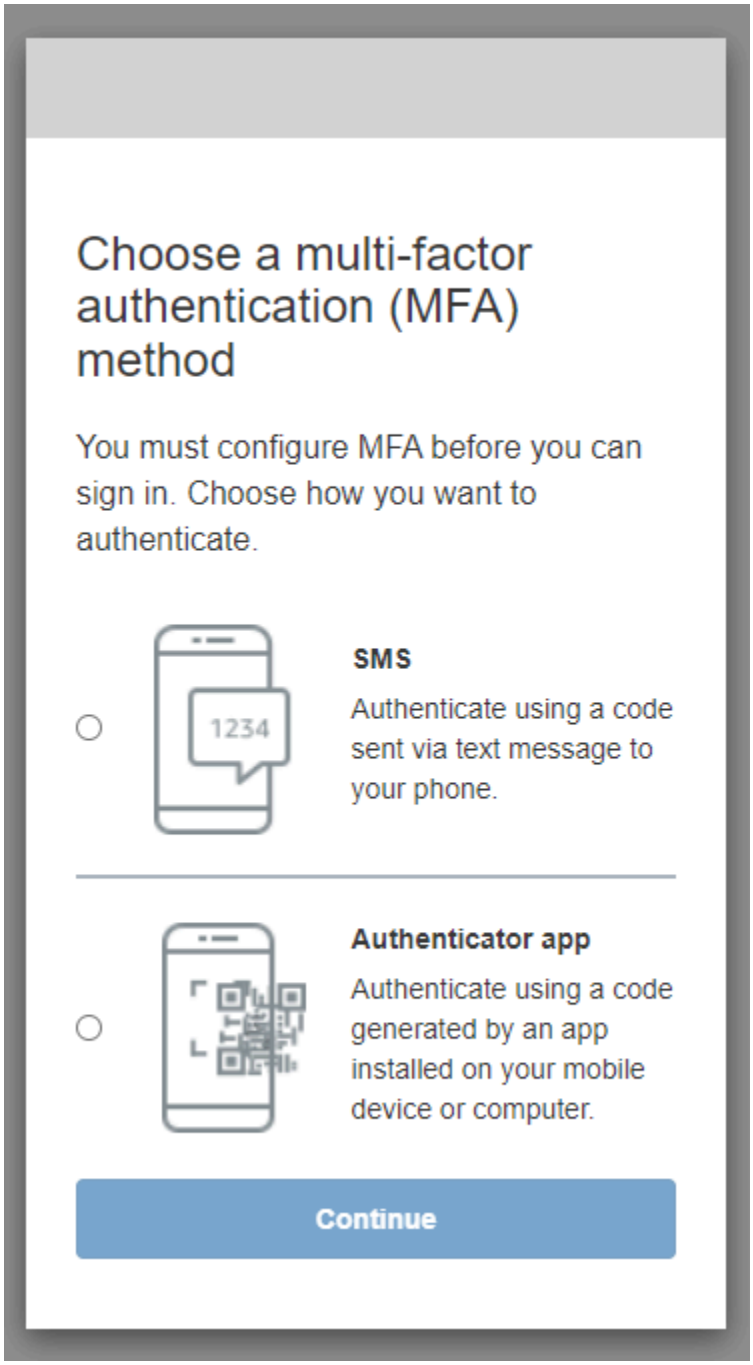
3. Auf der Seite Confirm your account (Konto bestätigen) fordert der App-Besitzer Sie möglicherweise dazu auf, Ihr Konto zu bestätigen, um zu überprüfen, ob Sie Nachrichten über die von Ihnen angegebene E-Mail-Adresse oder Telefonnummer empfangen können.

   Sie erhalten einen Code in Ihrer E-Mail oder in einer SMS-Nachricht. Geben Sie den Code in das Formular ein, um zu bestätigen, dass Sie die richtigen Kontaktinformationen eingegeben haben.



4. Der App-Besitzer verlangt möglicherweise, dass Sie die Multi-Faktor-Authentifizierung (MFA) einrichten. Möglicherweise werden Sie dazu aufgefordert, Ihre MFA-Methode auszuwählen, oder Ihre App springt zum nächsten Schritt.

   Wählen Sie auf der Seite Choose a multi-factor authentication (MFA) method (Eine Multi-Faktor-Authentifizierungsmethode (MFA) auswählen) eine MFA-Methode aus. Wenn Sie SMS auswählen, erhalten Sie MFA-Passcodes in Form von SMS-Nachrichten. Wenn Sie Authenticator app (Authentifizierungs-App) auswählen, müssen Sie eine App auf Ihrem Gerät installieren, um zeitbasierte MFA-Passcodes zu generieren. Sie müssen innerhalb von 3 Minuten eine Wahl treffen.



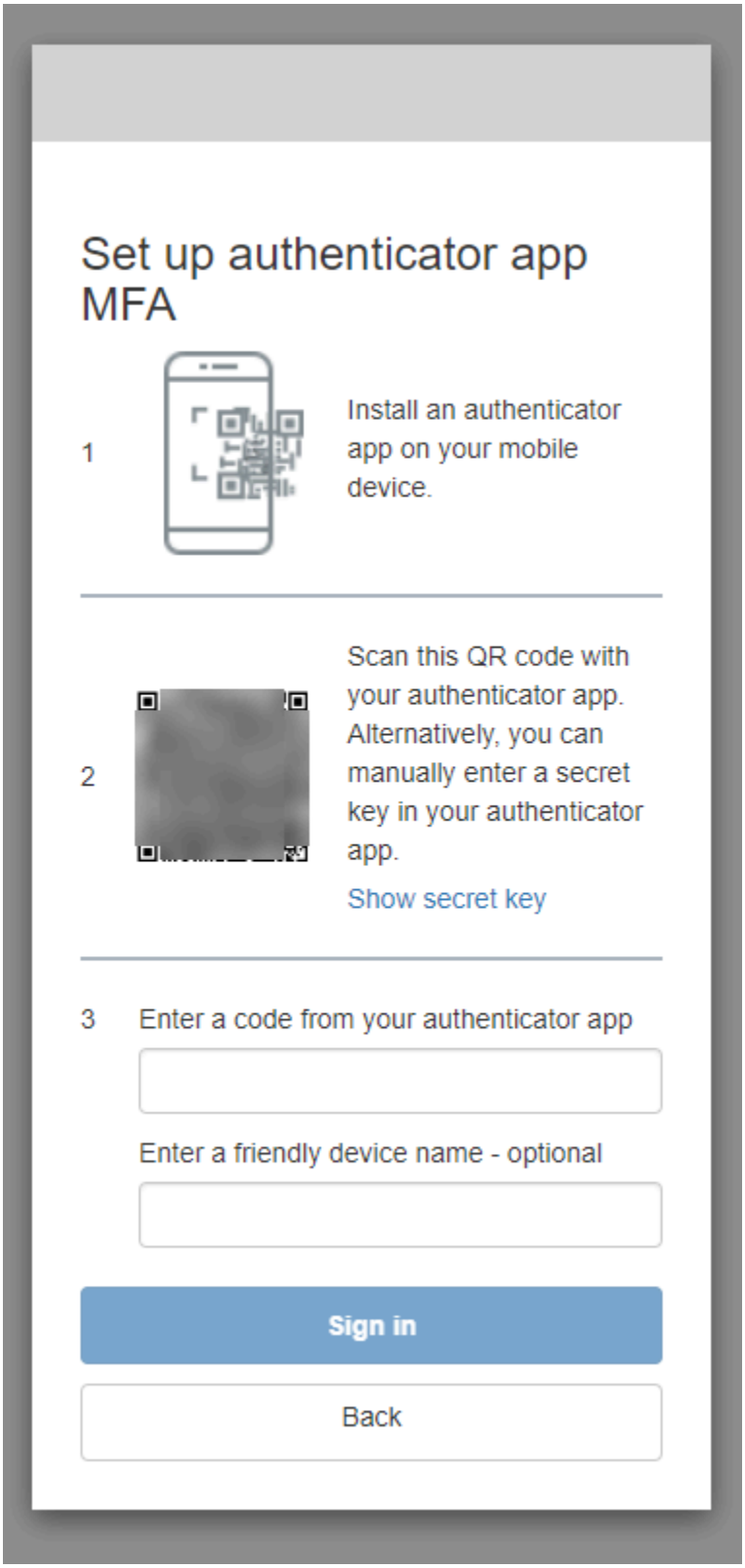
5. Amazon Cognito fragt Sie nach einem Code aus Ihrer Authentifizierungs-App oder SMS-Nachricht. Geben Sie den Code ein, den Sie innerhalb von 3 Minuten erhalten haben.

   Authenticator app

   1. Öffnen Sie die Authentifizierungs-App, die Sie heruntergeladen haben.
   2. Scannen Sie den QR-Code auf der Seite mit Ihrer Kamera. Möglicherweise müssen Sie der App die Verwendung Ihrer Kamera erlauben.

Wenn Sie den QR-Code nicht scannen können, wählen Sie Show secret key (Geheimen Schlüssel anzeigen) aus, um einen Code anzuzeigen, den Sie manuell in Ihre Authentifizierungs-App eingeben können.

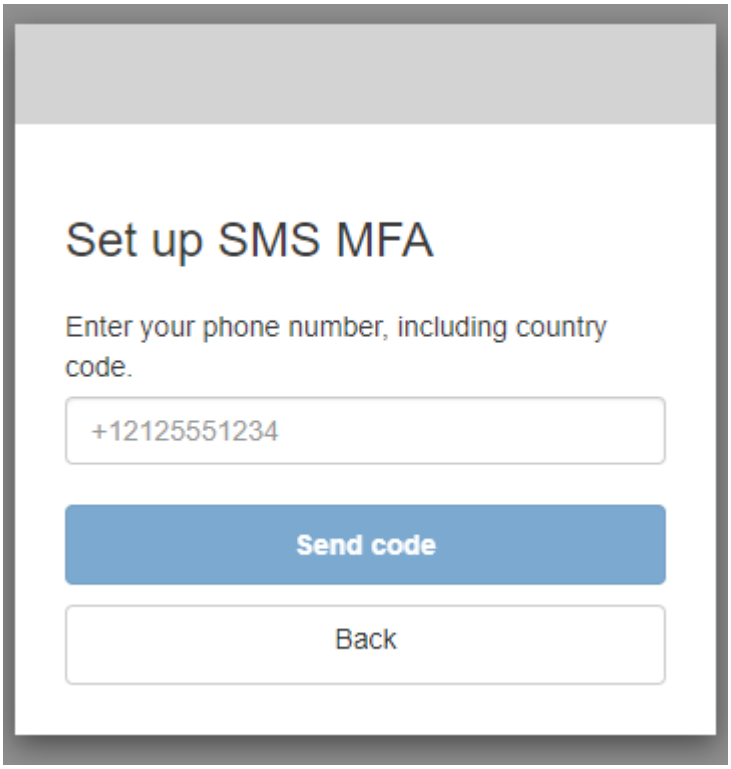
3. Ihre Authentifizierungs-App zeigt Codes an, die sich alle paar Sekunden ändern. Geben Sie einen aktuellen Code aus der App ein.
4. (Optional) Wählen Sie auf der Seite Set up authenticator app MFA (MFA per Authentifizierungs-App einrichten) einen Namen für Ihr Gerät aus. Wenn Sie sich anmelden, werden Sie von Amazon Cognito nach einem Code vom Gerät mit dem Namen gefragt, den Sie hier festlegen.


Set up authenticator app MFA
1
Install an authenticator app on your mobile device.
2
Scan this QR code with your authenticator app. Alternatively, you can manually enter a secret key in your authenticator app.
Show secret key
3
Enter a code from your authenticator app
Enter a friendly device name - optional
Sign in
Back

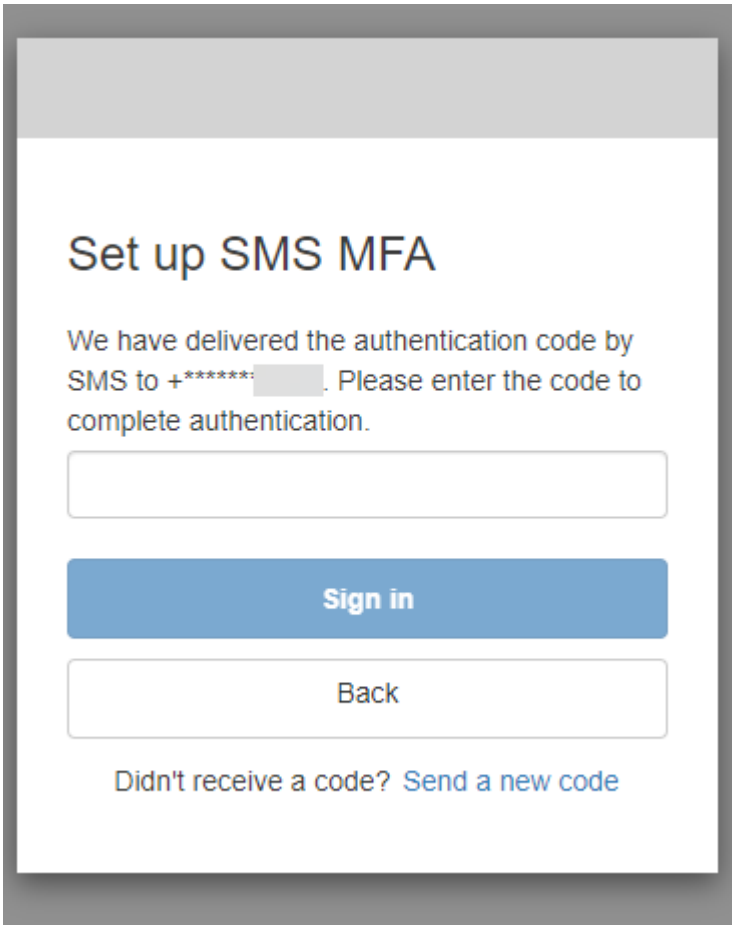
SMS text message

1. Wenn der App-Besitzer Ihre Telefonnummer noch nicht erfasst hat, fordert Amazon Cognito Ihre Telefonnummer an.

   Geben Sie auf der Seite Set up SMS MFA (SMS-MFA einrichten) eine Telefonnummer ein, die ein +-Zeichen und eine Landesvorwahl enthält, zum Beispiel +12125551234.



2. Sie erhalten eine SMS-Nachricht mit einem Code. Geben Sie auf der Seite Set up SMS MFA (SMS-MFA einrichten) den Code ein. Wenn Sie keinen Code erhalten haben und es erneut versuchen möchten, wählen Sie Send a new code (Neuen Code senden) aus. Wählen Sie Back (Zurück) aus, um eine neue Telefonnummer einzugeben.



6. Wenn Sie sich zum ersten Mal anmelden und Ihre Daten bestätigen, gewährt Amazon Cognito Zugriff auf Ihre App, nachdem Sie diesen Vorgang abgeschlossen haben.

So melden Sie sich bei der von Amazon Cognito gehosteten Benutzeroberfläche an

In dieser Anleitung erfahren Sie, wie Sie sich bei Apps, die Amazon Cognito verwenden, anmelden.

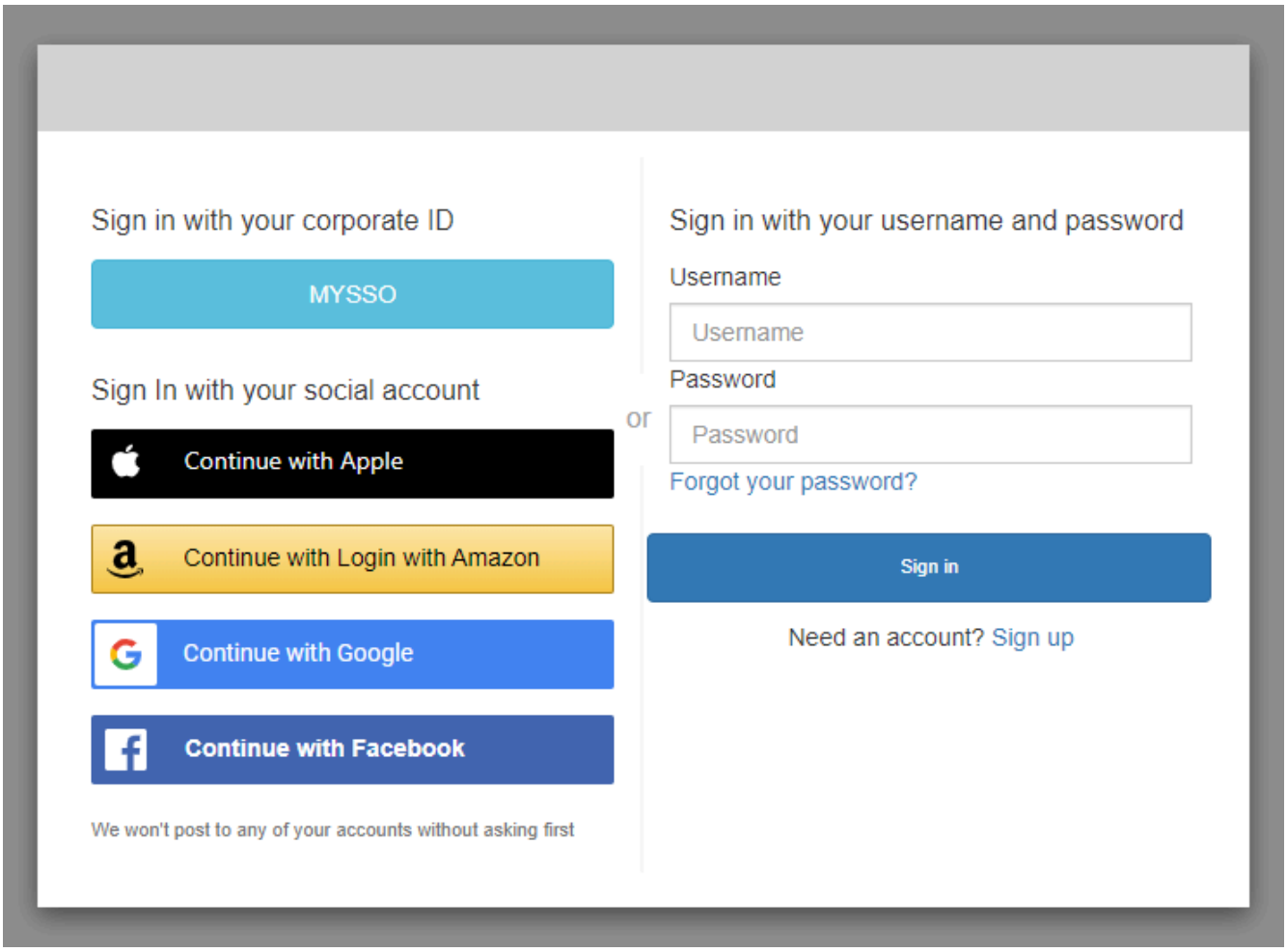
> **Note**
>
> Wenn Sie sich bei einer App anmelden, die die von Amazon Cognito gehostete Benutzeroberfläche (UI) verwendet, wird möglicherweise eine Seite angezeigt, die der App-Besitzer über die in diesem Handbuch dargestellte Grundkonfiguration hinaus angepasst hat.

1. Abhängig von den Optionen, die der App-Besitzer ausgewählt hat, stehen Ihnen möglicherweise verschiedene Anbieter zur Auswahl, mit denen Sie sich anmelden können. Eventuell wird nur eine Aufforderung zur Eingabe eines Benutzernamens und Passworts angezeigt. Wenn Sie

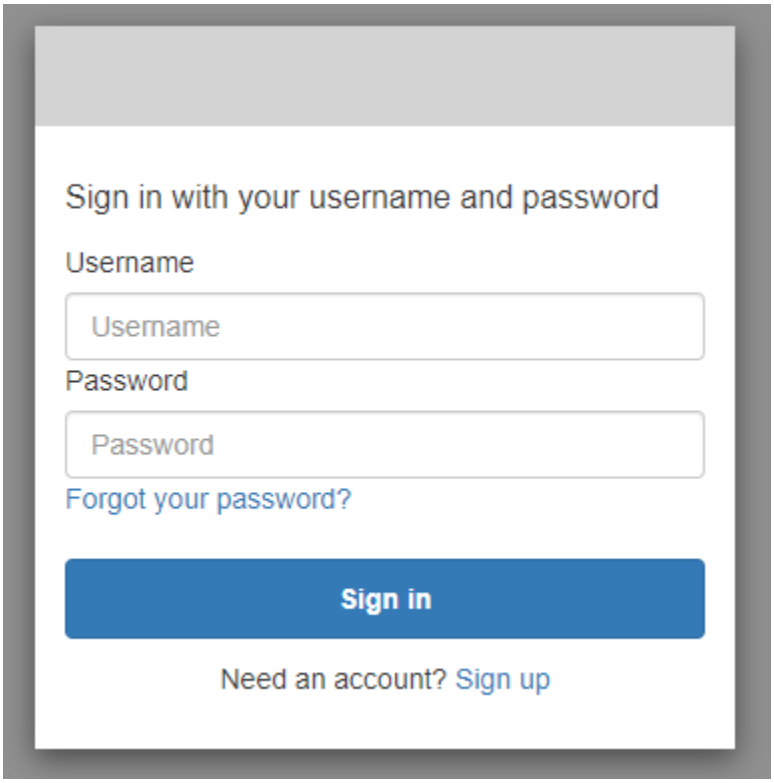
sich auf dieser Seite mit einem Benutzernamen und Passwort anmelden, ist Amazon Cognito Ihr Identitätsanbieter. Andernfalls wird Ihr Identitätsanbieter durch die von Ihnen ausgewählte Schaltfläche dargestellt.

Sie können hier einen Anbieter auswählen oder einen Benutzernamen und ein Passwort eingeben. Sie erhalten sofort Zugriff auf Ihre App. Wenn Amazon Cognito Ihr Identitätsanbieter ist, benötigt der App-Besitzer möglicherweise auch eine Multi-Faktor-Authentifizierung.

With multiple sign-in providers

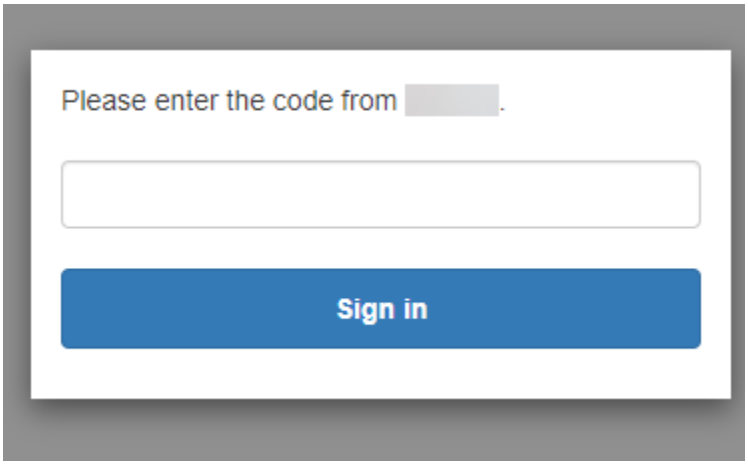


With only Amazon Cognito as a sign-in provider

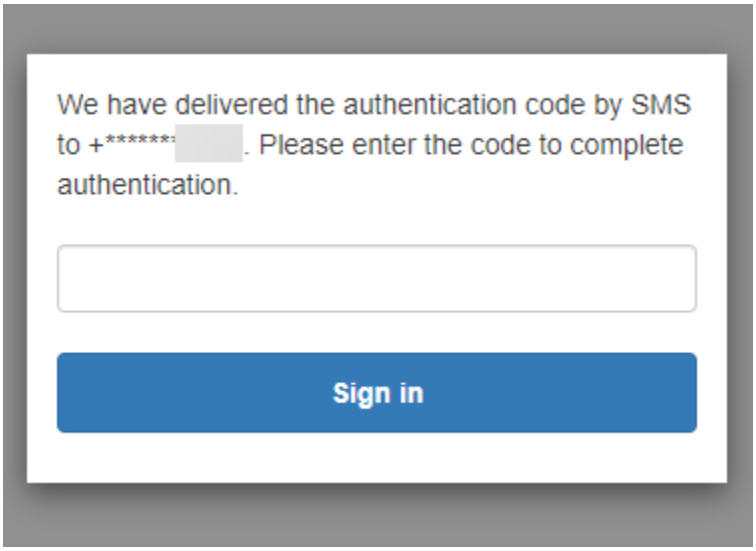


2. Möglicherweise haben Sie die MFA eingerichtet, als Sie sich bei der App registriert haben. Geben Sie Ihren MFA-Code ein, den Sie entweder in einer SMS-Nachricht erhalten haben oder der in Ihrer Authentifizierungs-App angezeigt wird. Sie müssen diesen Code innerhalb von 3 Minuten eingeben.

   With an authenticator app



With an SMS code



3. Nachdem Sie sich angemeldet und die MFA abgeschlossen haben, gewährt Amazon Cognito Zugriff auf Ihre App.

So setzen Sie ein Passwort auf der von Amazon Cognito gehosteten Benutzeroberfläche zurück

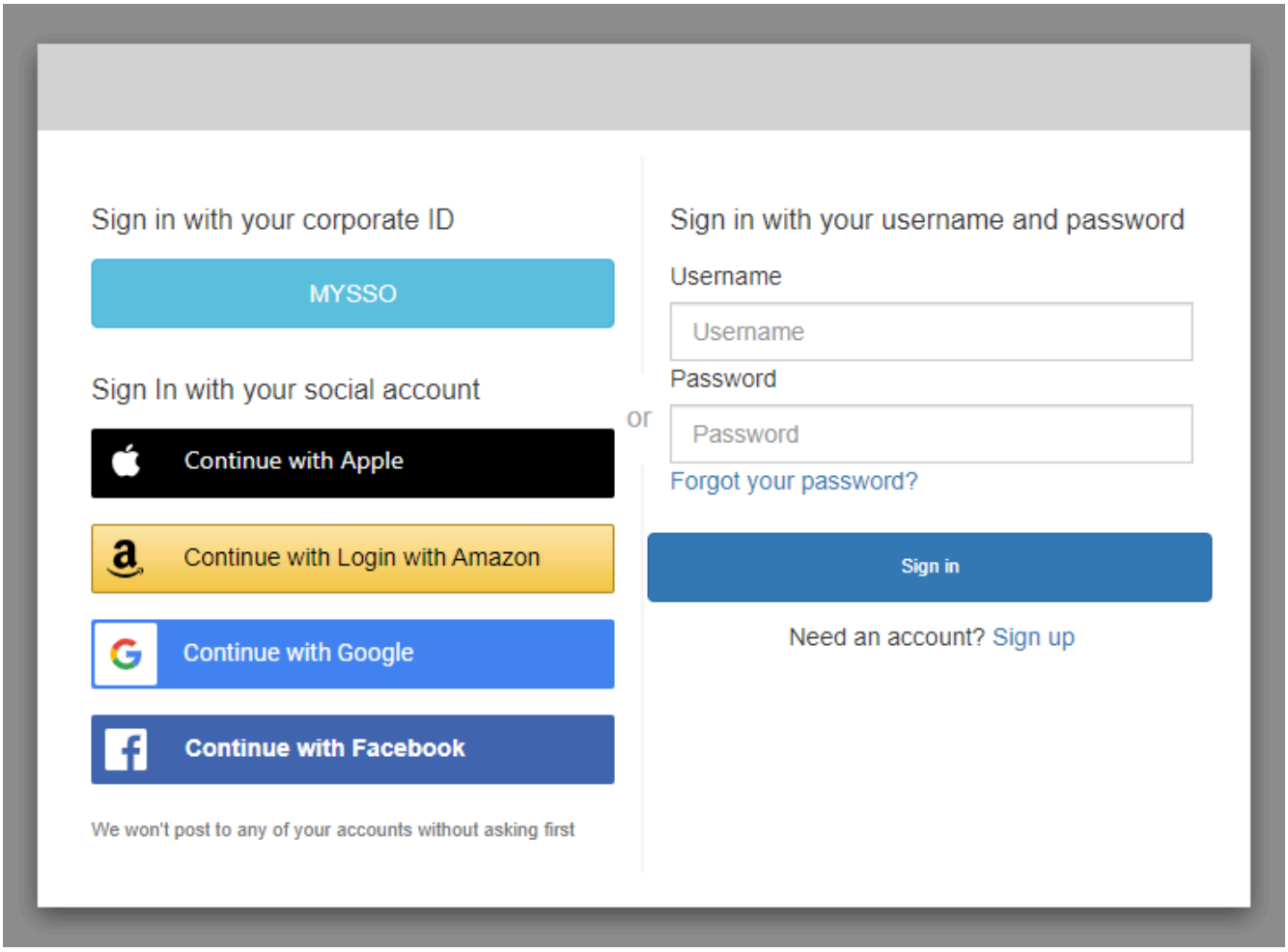
In dieser Anleitung erfahren Sie, wie Sie Ihr Passwort für Apps, die Amazon Cognito verwenden, zurücksetzen.

> **Note**
>
> Wenn Sie sich bei einer App anmelden, die die von Amazon Cognito gehostete Benutzeroberfläche (UI) verwendet, wird möglicherweise eine Seite angezeigt, die der App-Besitzer über die in diesem Handbuch dargestellte Grundkonfiguration hinaus angepasst hat.

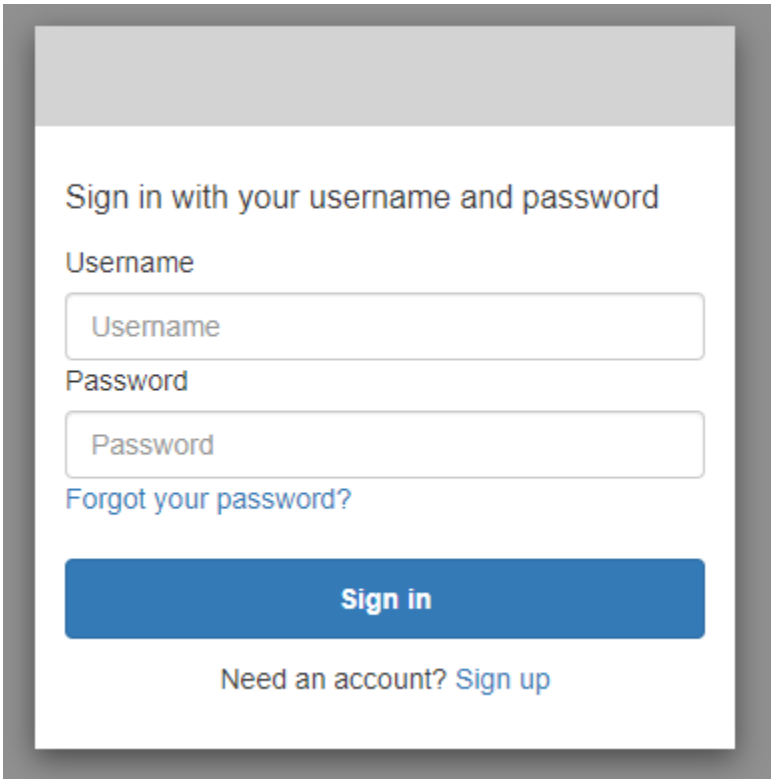
1. Abhängig von den Optionen, die der App-Besitzer ausgewählt hat, stehen Ihnen möglicherweise verschiedene Anbieter zur Auswahl, mit denen Sie sich anmelden können. Eventuell wird nur eine Aufforderung zur Eingabe eines Benutzernamens und Passworts angezeigt. Wenn Sie sich auf dieser Seite mit einem Benutzernamen und Passwort anmelden, ist Amazon Cognito Ihr Identitätsanbieter. Andernfalls wird Ihr Identitätsanbieter durch die von Ihnen ausgewählte Schaltfläche dargestellt.

Wenn Sie wie üblich einen Anbieter auf der Anmeldeseite auswählen und Ihr Passwort nicht funktioniert, befolgen Sie die Anweisungen zum Zurücksetzen Ihres Passworts beim Anbieter. Wenn Amazon Cognito Ihr Identitätsanbieter ist, wählen Sie Forgot your password? (Passwort vergessen?) aus.

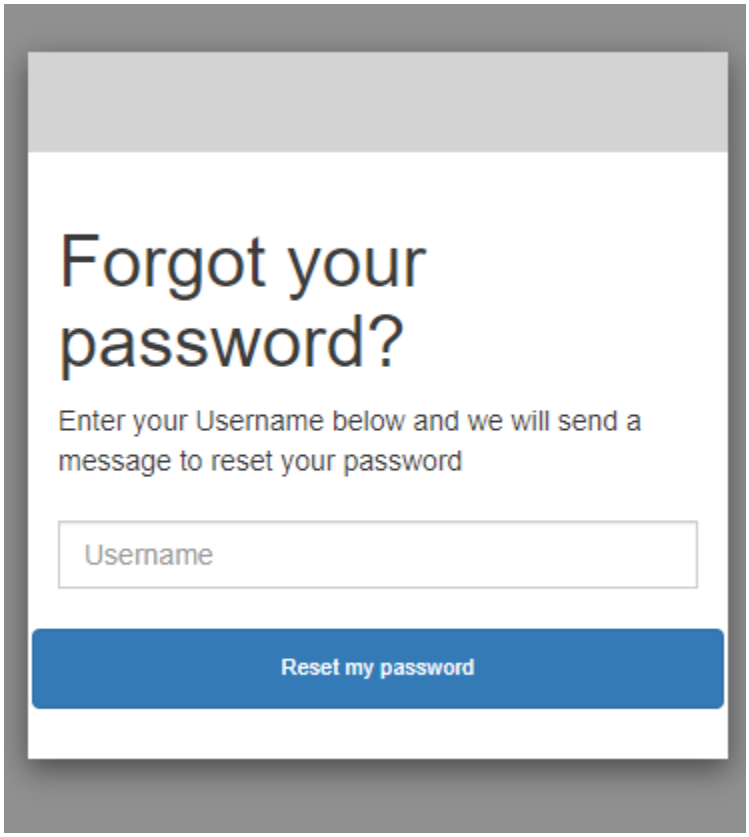
With multiple sign-in providers



With only Amazon Cognito as a sign-in provider

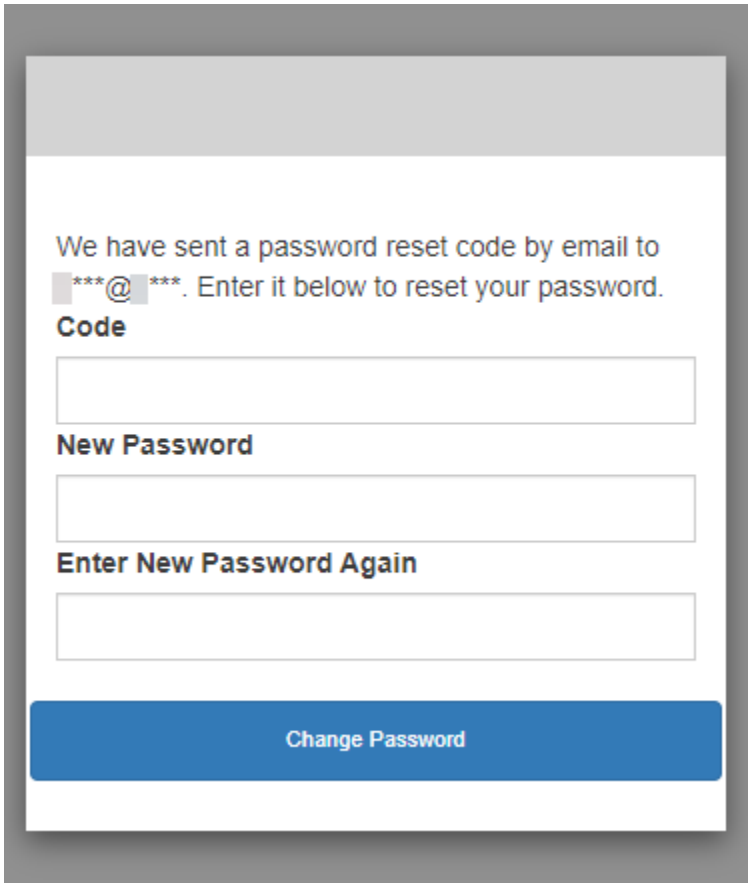


2. Auf der Seite Forgot your password? (Passwort vergessen?) werden Sie von Amazon Cognito zur Eingabe der Informationen aufgefordert, die Sie für die Anmeldung verwenden. Dies kann Ihr Benutzername, Ihre E-Mail-Adresse oder Ihre Telefonnummer sein.



3. Amazon Cognito sendet Ihnen einen Code als E-Mail- oder SMS-Nachricht.

   Geben Sie den Code ein, den Sie erhalten haben, und tragen Sie Ihr neues Passwort zweimal in die dafür vorgesehenen Felder ein. Sie müssen Ihren Reset-Code innerhalb von 8 Minuten eingeben.



4. Nachdem Sie Ihr Passwort geändert haben, kehren Sie zur Anmeldeseite zurück und melden sich mit Ihrem neuen Passwort an.

Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern

Nachdem Sie eine Domain für Ihren Benutzerpool konfiguriert haben, stellt Amazon Cognito automatisch einen OAuth-2.0-Autorisierungsserver und eine gehostete Web-UI mit Registrierungs- und Anmeldeseiten zur Verfügung, die Ihre App den Benutzern präsentieren kann. Weitere Informationen finden Sie unter Fügen Sie einen App-Client mit der gehosteten Benutzeroberfläche hinzu. Sie können die Bereiche auswählen, die der Autorisierungsserver den Zugriffstoken hinzufügen soll. Bereiche autorisieren den Zugriff auf Ressourcenserver und Benutzerdaten.

Ein Ressourcenserver ist ein OAuth-2.0-API-Server. Um zugriffsgeschützte Ressourcen zu sichern, wird überprüft, ob Zugriffstoken aus Ihrem Benutzerpool die Bereiche enthalten, die die angeforderte Methode und den Pfad in der geschützten API autorisieren. Der Aussteller wird anhand der Token-Signatur, der Gültigkeit basierend auf der Token-Ablaufzeit und der Zugriffsebene basierend auf

den Geltungsbereichen in Token-Ansprüchen verifiziert. Die Bereiche des Benutzerpools sind im Anspruch auf Zugriffstoken enthalten. `scope` Weitere Informationen zu den Ansprüchen in Amazon-Cognito-Zugriffstoken finden Sie unter Verwenden des Zugriffstokens.

Mit Amazon Cognito können die Bereiche in Zugriffstoken den Zugriff auf externe APIs oder Benutzerattribute autorisieren. Sie können Zugriffstoken für lokale Benutzer, Verbundbenutzer oder Maschinenidentitäten ausgeben.

M achine-to-machine (M2M) -Autorisierung

Amazon Cognito unterstützt Anwendungen, die auf API-Daten mit Maschinenidentitäten zugreifen. Maschinenidentitäten in Benutzerpools sind vertrauliche Clients, die auf Anwendungsservern ausgeführt werden und eine Verbindung zu Remote-APIs herstellen. Ihr Betrieb erfolgt ohne Benutzerinteraktion: geplante Aufgaben, Datenströme oder Asset-Updates. Wenn diese Clients ihre Anfragen mit einem Zugriffstoken autorisieren, führen sie eine Machine-to-Machine-Autorisierung (M2M) durch. Bei der M2M-Autorisierung ersetzt ein gemeinsam genutzter geheimer Schlüssel die Benutzeranmeldedaten bei der Zugriffskontrolle.

Eine Anwendung, die mit M2M-Autorisierung auf eine API zugreift, muss über eine Client-ID und einen geheimen Clientschlüssel verfügen. In Ihrem Benutzerpool müssen Sie einen App-Client erstellen, der die Gewährung von Client-Anmeldeinformationen unterstützt. Um Client-Anmeldeinformationen zu unterstützen, muss Ihr App-Client über einen geheimen Client-Schlüssel verfügen und Sie müssen über eine Benutzerpool-Domain verfügen. In diesem Ablauf fordert Ihre Computeridentität ein Zugriffstoken direkt von der anToken-Endpunkt. Sie können nur benutzerdefinierte Bereiche von Ressourcenservern in Zugriffstoken für die Gewährung von Client-Anmeldeinformationen autorisieren. Weitere Informationen zum Einrichten von App-Clients finden Sie unter. App-Clients für Benutzerpools

Das Zugriffstoken aus der Gewährung von Kundenanmeldedaten ist eine überprüfbare Aussage über die Vorgänge, die Sie Ihrer Computeridentität von einer API abfragen lassen möchten. Um mehr darüber zu erfahren, wie Zugriffstoken API-Anfragen autorisieren, lesen Sie weiter. Eine Beispielanwendung finden Sie unter Maschine-zu-Maschine-Autorisierung auf Basis von Amazon Cognito und API Gateway mit AWS CDK.

Die M2M-Autorisierung hat ein Abrechnungsmodell, das sich von der Art und Weise unterscheidet, wie monatlich aktive Benutzer (MAUs) in Rechnung gestellt werden. Wenn die Benutzerauthentifizierung mit Kosten pro aktivem Benutzer verbunden ist, spiegelt die M2M-Abrechnung die aktiven Kundenanmeldedaten, die App-Clients und das Gesamtvolumen der Token-Anfragen wider. Weitere Informationen finden Sie unter Amazon Cognito – Preise. Um die Kosten

für die M2M-Autorisierung unter Kontrolle zu halten, sollten Sie die Dauer der Zugriffstoken und die Anzahl der Token-Anfragen, die Ihre Anwendungen stellen, optimieren. Eine Möglichkeit, API-Gateway-Caching zu verwenden, um Anfragen nach neuen Tokens bei der M2M-Autorisierung zu reduzieren, finden Zwischenspeicherung von Token Sie unter.

Informationen zur Optimierung von Amazon Cognito Cognito-Vorgängen, die Ihre AWS Rechnung mit zusätzlichen Kosten belasten, finden Sie unterVerwalten von Kosten.

Grundlegendes zu Bereichen

Ein Bereich ist ein Zugriffsniveau, das eine App von einer Ressource anfordern kann. In einem Zugriffstoken in Amazon Cognito wird der Bereich durch das Vertrauen gesichert, das Sie mit Ihrem Benutzerpool geschaffen haben: ein vertrauenswürdiger Aussteller von Zugriffstoken mit einer bekannten digitalen Signatur. Benutzerpools können Zugriffstoken mit Bereichen generieren, die belegen, dass Ihr Kunde sein eigenes Benutzerprofil ganz oder teilweise verwalten kann oder Daten von einer Back-End-API abrufen darf. Benutzerpools in Amazon Cognito stellen Zugriffstoken mit dem reservierten API-Bereich für Benutzerpools, benutzerdefinierten Bereichen und Standardbereichen aus.

Der reservierte API-Bereich der Benutzerpools

Der `aws.cognito.signin.user.admin`-Bereich autorisiert die Amazon-Cognito-Benutzerpool-API. Es autorisiert den Inhaber eines Zugriffstokens, alle Informationen über einen Benutzerpool-Benutzer abzufragen und zu aktualisieren, z. B. mit den Operationen GetUserund UpdateUserAttributesder API. Wenn Sie Ihren Benutzer mit der Benutzerpools-API in Amazon Cognito authentifizieren, ist dies der einzige Bereich, den Sie im Zugriffstoken erhalten. Dies ist auch der einzige Bereich, den Sie zum Lesen und Schreiben von Benutzerattributen benötigen, die der App-Client lesen und schreiben kann. Sie können diesen Bereich auch in Anfragen an Ihren Autorisieren des Endpunkts anfordern. Dieser Bereich allein reicht nicht aus, um Benutzerattribute von UserInfo-Endpunkt anzufordern. Für Zugriffstoken, die sowohl die Benutzerpool-API als auch *`userInfo`* Anfragen für Ihre Benutzer autorisieren, müssen Sie beide Bereiche *`openid`* und *`aws.cognito.signin.user.admin`* in einer */`oauth2/authorize`* Anfrage anfordern.

Benutzerdefinierte Bereiche

Benutzerdefinierte Bereiche autorisieren Anfragen an die externen APIs, die von Ressourcenservern geschützt werden. Sie können benutzerdefinierte Bereiche mit anderen Arten von Bereichen anfordern. Weitere Informationen zu benutzerdefinierten Bereichen finden Sie auf dieser Seite.

Standardbereiche

Wenn Sie Benutzer mit Ihrem OAuth-2.0-Autorisierungsserver für Ihren Benutzerpool authentifizieren, auch mit der gehosteten Benutzeroberfläche, müssen Sie Bereiche anfordern. Sie können lokale Benutzerpool-Benutzer und externe Verbundbenutzer auf Ihrem Amazon-Cognito-Autorisierungsserver authentifizieren. Standardbereiche von OAuth 2.0 autorisieren Ihre App, Benutzerinformationen aus dem UserInfo-Endpunkt Ihres Benutzerpools zu lesen. Das OAuth-Modell, bei dem Sie Benutzerattribute vom `userInfo`-Endpunkt aus abfragen, kann Ihre App für eine große Anzahl von Anfragen nach Benutzerattributen optimieren. Der `userInfo`-Endpunkt gibt Attribute auf einer Berechtigungsebene zurück, die durch die Bereiche im Zugriffstoken bestimmt wird. Sie können Ihren App-Client autorisieren, Zugriffstoken mit den folgenden OAuth-2.0-Standardbereichen auszustellen.

openid

Der obligatorische Mindestbereich für OpenID-Connect-(OIDC)-Abfragen. Autorisiert das ID-Token, den unique-identifier-Anspruch `sub` und die Möglichkeit, andere Bereiche anzufordern.

> **Note**
>
> Wenn Sie nur den `openid`-Bereich und keine anderen anfordern, enthalten Ihr Benutzerpool-ID-Token und Ihre `userInfo`-Antwort Ansprüche für alle Benutzerattribute, die Ihr App-Client lesen kann. Wenn Sie `openid` und andere Standardbereiche wie `profile`, `email` und `phone` anfordern, wird der Inhalt der ID-Token- und userInfo-Antwort entsprechend der Einschränkungen der zusätzlichen Bereiche beschränkt. Beispielsweise würde eine Anfrage an Autorisieren des Endpunkts mit dem Parameter `scope=openid+email` ein ID-Token mit `sub`, `email` und `email_verified` zurückgeben. Das Zugriffs-Token aus dieser Anfrage gibt dieselben Attribute aus UserInfo-Endpunkt zurück. Eine Anfrage mit Parameter `scope=openid` gibt alle vom Client lesbaren Attribute im ID-Token und aus `userInfo` zurück.

Profil

Autorisiert alle Benutzerattribute, die der App-Client lesen kann.

email

Autorisiert die Benutzerattribute `email` und `email_verified`. Amazon Cognito gibt `email_verified` zurück wenn ein Wert explizit festgelegt wurde.

phone

Autorisiert die Benutzerattribute `phone_number` und `phone_number_verified`.

Grundlegendes zu Ressourcenservern

Eine Ressourcenserver-API kann Zugriff auf die Informationen in einer Datenbank gewähren oder Ihre IT-Ressourcen steuern. Ein Amazon-Cognito-Zugriffstoken kann den Zugriff auf APIs autorisieren, die OAuth 2.0 unterstützen. REST-APIs von Amazon API Gateway bieten integrierte Unterstützung für die Autorisierung mit Amazon-Cognito-Zugriffstoken. Ihre App gibt das Zugriffstoken aus dem API-Aufruf an den Ressourcenserver weiter. Der Ressourcenserver untersucht das Zugriffstoken, um zu festzustellen, ob Zugriff gewährt werden soll.

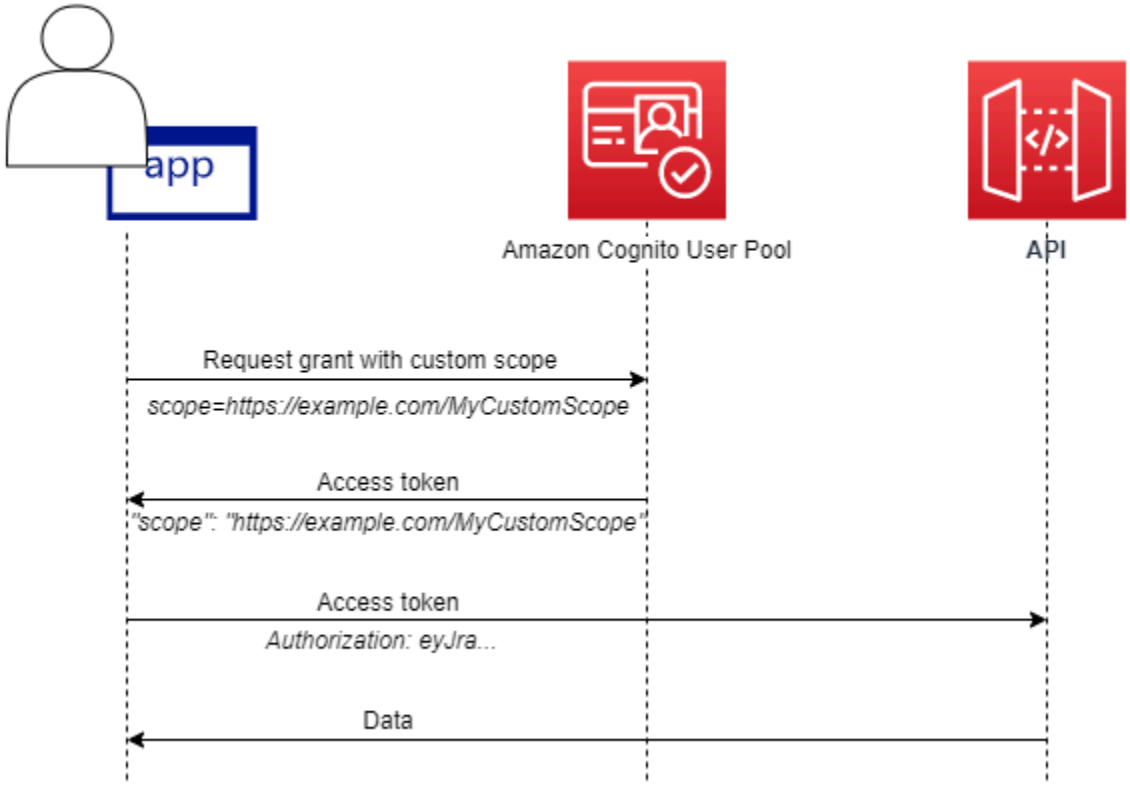
Amazon Cognito wird möglicherweise in Zukunft das Schema der Zugriffstoken für Benutzerpools aktualisieren. Wenn Ihre App den Inhalt des Zugriffstokens analysiert, bevor sie es an eine API weitergibt, müssen Sie Ihren Code so anpassen, dass er Aktualisierungen des Schemas akzeptiert.

Benutzerdefinierte Bereiche werden von Ihnen definiert und erweitern die Autorisierungsfunktionen eines Benutzerpools auf Zwecke, die nichts mit dem Abfragen und Ändern von Benutzern und ihren Attributen zu tun haben. Wenn Sie zum Beispiel einen Ressourcenserver für Fotos haben, könnte er zwei Bereiche definieren: `photos.read` für den Lesezugriff auf die Fotos und `photos.write` für den Schreib-/Lesezugriff. Sie können eine API so konfigurieren, dass sie Zugriffstoken für die Autorisierung akzeptiert und `HTTP GET`-Anforderungen für Zugriffstoken mit `photos.read` im `scope`-Anspruch sowie `HTTP POST`-Anforderungen für Token mit `photos.write` gewährt. Das sind benutzerdefinierte Bereiche.

> **Note**
>
> Ihr Ressourcenserver muss die Signatur und das Ablaufdatum des Zugriffstokens überprüfen, bevor Ansprüche innerhalb des Tokens verarbeitet werden. Weitere Informationen zur Verifizierung von Token finden Sie unter Verifizieren eines JSON-Web-Tokens. Weitere Informationen zur Verifizierung und Verwendung von Benutzerpool-Token finden Sie unter Integration von Amazon-Cognito-Benutzerpools mit API-Gateway im Blog. API Gateway ist gut für die Überprüfung von Zugriffstoken und den Schutz Ihrer Ressourcen geeignet. Weitere Informationen zu API-Gateway-Lambda-Genehmigern finden Sie unter Verwenden von API-Gateway-Lambda-Genehmigern.

Übersicht

Mit Amazon Cognito können Sie OAuth-2.0-Ressourcenserver erstellen und ihnen Benutzerdefinierte Bereiche zuordnen. Benutzerdefinierte Bereiche in einem Zugriffstoken autorisieren spezifische Aktionen in Ihrer API. Sie können jeden App-Client in Ihrem Benutzerpool autorisieren, benutzerdefinierte Bereiche von jedem Ihrer Ressourcenserver aus auszustellen. Ordnen Sie Ihre benutzerdefinierten Bereiche einem App-Client zu und fordern Sie diese Bereiche dann in OAuth-2.0-Autorisierungscode-Erteilungen, impliziten Erteilungen und Client-Anmeldeinformationserteilungen vom Token-Endpunkt an. Amazon Cognito fügt dem `scope`-Anspruch benutzerdefinierte Bereiche in einem Zugriffstoken hinzu. Ein Client kann das Zugriffstoken für seinen Ressourcenserver verwenden, der die Entscheidung über die Autorisierung basierend auf den im Token vorhandenen Bereichen trifft. Weitere Informationen über Zugriffstoken-Bereiche finden Sie unter Verwenden von Token mit Benutzerpools.



Zum Abrufen eines Zugriffstokens mit benutzerdefinierten Bereichen muss Ihre App eine Anfrage an den Token-Endpunkt senden, um einen Autorisierungscode einzulösen oder eine Berechtigung für Client-Anmeldeinformationen anzufordern. In der gehosteten Benutzeroberfläche können Sie auch benutzerdefinierte Bereiche in einem Zugriffstoken aus einer impliziten Erteilung anfordern.

> **Note**
>
> Weil sie für die interaktive Authentifizierung mit dem Benutzerpool als IdP konzipiert sind InitiateAuthund AdminInitiateAuthAnfragen nur einen `scope` Anspruch im Zugriffstoken mit dem einzigen Wert erzeugen. `aws.cognito.signin.user.admin`

Verwalten der Ressourcenserver und benutzerdefinierten Bereiche

Beim Erstellen eines Ressourcenservers müssen Sie einen Ressourcenservernamen und eine Ressourcenserver-ID angeben. Für jeden Bereich, den Sie im Ressourcenserver erstellen, müssen Sie den Namen und die Beschreibung des Bereichs bereitstellen.

- Name des Ressourcenservers: Ein Anzeigename für den Ressourcenserver, z. B. `Solar system object tracker` oder `Photo API`.
- Ressourcenserver-ID: Eine eindeutige ID für den Ressourcenserver. Die ID ist ein beliebiger Name, den Sie Ihrer API zuordnen, wie z. B. `solar-system-data`. Sie können längere IDs konfigurieren, wie z. B. `https://solar-system-data-api.example.com` als direkteren Verweis auf API-URI-Pfade, aber längere Zeichenfolgen machen Zugriffstoken größer.
- Bereichsname: Der Wert, den Sie in Ihren `scope`-Ansprüchen haben wollen. z. B. `sunproximity.read`.
- Beschreibung: Eine Kurzbeschreibung des Bereichs. z. B. `Check current proximity to sun`.

Amazon Cognito kann benutzerdefinierte Bereiche in Zugriffstoken für alle Benutzer einbeziehen, unabhängig davon, ob sie sich lokal in Ihrem Benutzerpool befinden oder mit einem externen Identitätsanbieter verbunden sind. Sie können Bereiche für die Zugriffstoken Ihrer Benutzer während des Authentifizierungsprozesses mit dem OAuth-2.0-Autorisierungsserver auswählen, der die gehostete Benutzeroberfläche enthält. Die Authentifizierung des Benutzers muss bei Autorisieren des Endpunkts beginnen, mit `scope` als einem der Anforderungsparameter. Das folgende Format wird für Ressourcenserver empfohlen. Verwenden Sie als ID einen API-freundlichen Namen. Verwenden Sie für einen benutzerdefinierten Bereich die zu autorisierende Aktion.

```
resourceServerIdentifier/scopeName
```

Sie haben beispielsweise einen neuen Asteroiden im Kuipergürtel entdeckt und möchten ihn über Ihre `solar-system-data`-API registrieren. Der Bereich, der Schreiboperationen in die

Asteroidendatenbank autorisiert, ist `asteroids.add`. Wenn Sie das Zugriffstoken anfordern, das Sie zur Registrierung Ihrer Entdeckung berechtigt, formatieren Sie Ihren `scope`-HTTPS-Anforderungsparameter als `scope=solar-system-data/asteroids.add`.

Durch das Löschen eines Bereichs von einem Ressourcenserver wird nicht die Zuordnung zu allen Clients gelöscht. Stattdessen wird der Geltungsbereich als inaktiv markiert. Amazon Cognito fügt Zugriffstoken keine inaktiven Bereiche hinzu, verfährt jedoch ansonsten wie gewohnt, wenn Ihre App eines anfordert. Wenn Sie Ihrem Ressourcenserver den Gültigkeitsbereich zu einem späteren Zeitpunkt erneut hinzufügen, schreibt Amazon Cognito ihn erneut in das Zugriffstoken. Wenn Sie einen Bereich anfordern, den Sie Ihrem App-Client nicht zugeordnet haben, schlägt die Authentifizierung unabhängig davon fehl, ob Sie ihn vom Ressourcenserver Ihres Benutzerpools gelöscht haben.

Sie können die AWS Management Console API oder die CLI verwenden, um Ressourcenserver und Bereiche für Ihren Benutzerpool zu definieren.

Definition eines Ressourcenservers für Ihren Benutzerpool (AWS Management Console)

Sie können den verwenden AWS Management Console , um einen Ressourcenserver für Ihren Benutzerpool zu definieren.

Definieren eines Ressourcenservers

1. Melden Sie sich bei der Amazon Cognito-Konsole an.
2. Wählen Sie im Navigationsbereich User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus und suchen Sie nach Resource servers (Ressourcen-Server).
4. Wählen Sie Create a resource server (Ressourcenserver erstellen) aus.
5. Geben Sie einen Name für den Ressourcenserver ein. Zum Beispiel , ., `Photo Server`.
6. Geben Sie eine ID für den Ressourcenserver ein. Zum Beispiel , ., `com.example.photos`.
7. Geben Sie benutzerdefinierte Bereiche für Ihre Ressourcen ein, z. B. `read` und `write`.
8. Geben Sie für jeden der Bereichsnamen eine Beschreibung ein, z. B. `view your photos` und `update your photos`.
9. Wählen Sie Create (Erstellen) aus.

Ihre benutzerdefinierten Bereiche können auf der Registerkarte App integration (Anwendungsintegration) unter Resource servers (Ressourcenserver) in der Spalte Custom scopes (Benutzerdefinierte Bereiche) überprüft werden. Benutzerdefinierte Bereiche können für App-Clients auf der Registerkarte App integration (Anwendungsintegration) unter App-Clients aktiviert werden. Wählen Sie einen App-Client aus und suchen Sie nach Hosted UI settings (Einstellungen für gehostete Benutzeroberflächen) und klicken Sie auf Edit (Bearbeiten). Fügen Sie benutzerdefinierte Bereiche hinzu und wählen Sie Save changes (Änderungen speichern) aus.

Definieren Sie einen Ressourcenserver für Ihren Benutzerpool (AWS CLI und Ihre AWS API)

Verwenden Sie die folgenden Befehle für die Angabe von Einstellungen der Ressourcenserver für Ihren Benutzerpool.

Erstellen eines Ressourcenservers

- AWS CLI: `aws cognito-idp create-resource-server`
- AWS API: CreateResourceServer

Informationen über die Einstellungen Ihres Ressourcenservers abrufen

- AWS CLI: `aws cognito-idp describe-resource-server`
- AWS API: DescribeResourceServer

Informationen über alle Ressourcenserver für Ihren Benutzerpool auflisten

- AWS CLI: `aws cognito-idp list-resource-servers`
- AWS API: ListResourceServers

Einen Ressourcenserver löschen

- AWS CLI: `aws cognito-idp delete-resource-server`
- AWS API: DeleteResourceServer

Die Einstellungen für einen Ressourcenserver aktualisieren

- AWS CLI: `aws cognito-idp update-resource-server`

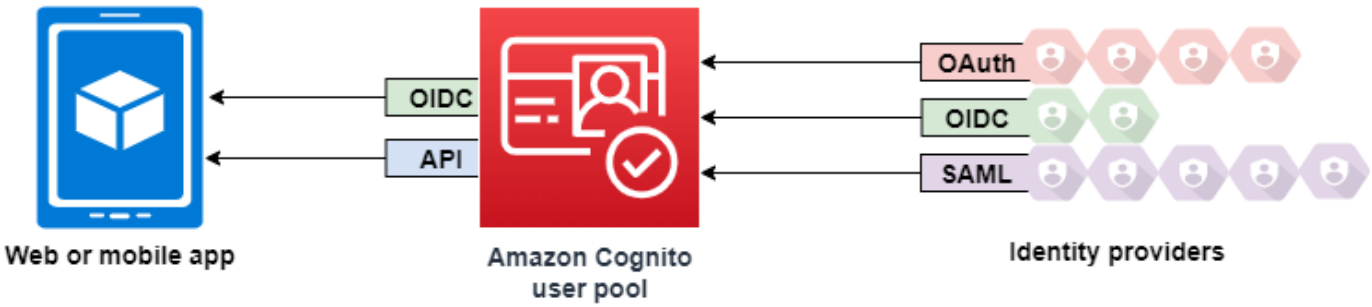
- AWS API: UpdateResourceServer

Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter

Ihre App-Benutzer können sich entweder direkt über einen Benutzerpool anmelden oder sich über einen externen Identitätsanbieter (IdP) zusammenschließen. Der Benutzerpool verwaltet den Aufwand für die Verwaltung der Token, die bei der Anmeldung in sozialen Netzwerken über Facebook, Google, Amazon und Apple sowie von OpenID Connect (OIDC) und SAML zurückgegeben werden. IdPs Mit der integrierten gehosteten Weboberfläche bietet Amazon Cognito die Token-Handhabung und -Verwaltung für authentifizierte Benutzer aller Art. IdPs Auf diese Weise können Ihre Backend-Systeme auf einen Satz von Benutzerpool-Token standardisiert werden.

Funktionsweise der Verbundanmeldung in Amazon-Cognito-Benutzerpools

Die Anmeldung über einen Drittanbieter (Verbund) in Amazon-Cognito-Benutzerpools wird unterstützt. Diese Funktion ist unabhängig von Verbund über Amazon-Cognito-Identitätspools (Verbundidentitäten).



Amazon Cognito ist ein Benutzerverzeichnis und ein OAuth-2.0-Identitätsanbieter (IDP). Wenn Sie lokale Benutzer im Amazon-Cognito-Verzeichnis anmelden, ist Ihr Benutzerpool ein IDP für Ihre App. Ein lokaler Benutzer existiert ausschließlich in Ihrem Benutzerpool-Verzeichnis ohne Verbund über einen externen IdP.

Wenn Sie Amazon Cognito mit Social, SAML oder OpenID Connect (OIDC) verbinden IdPs, fungiert Ihr Benutzerpool als Brücke zwischen mehreren Dienstanbietern und Ihrer App. Für Ihren IDP ist Amazon Cognito ein Serviceanbieter (SP). Sie IdPs übergeben ein OIDC-ID-Token oder eine SAML-Assertion an Amazon Cognito. Amazon Cognito liest die Ansprüche über Ihren Benutzer im Token oder in der Assertion und ordnet diese Ansprüche einem neuen Benutzerprofil in Ihrem Benutzerpool-Verzeichnis zu.

Amazon Cognito erstellt dann ein Benutzerprofil für Ihren Verbundbenutzer in seinem eigenen Verzeichnis. Amazon Cognito fügt Ihrem Benutzer basierend auf den Ansprüchen Ihres IDP Attribute hinzu und bei OIDC- und Social-Identitätsanbietern außerdem einen IDP-betriebenen öffentlichen `userinfo`-Endpunkt. Die Attribute Ihres Benutzers ändern sich in Ihrem Benutzerpool, wenn sich ein zugeordnetes IDP-Attribut ändert. Sie können auch weitere Attribute hinzufügen, die unabhängig von denen des IDP sind.

Nachdem Amazon Cognito ein Profil für Ihren Verbundbenutzer erstellt hat, ändert es seine Funktion und präsentiert sich als IDP für Ihre App, die jetzt der SP ist. Amazon Cognito ist eine Kombination aus OIDC- und OAuth-2.0-IDP. Es generiert Zugriffstoken, ID-Token und Aktualisierungs-Token. Weitere Informationen über Token finden Sie unter Verwenden von Token mit Benutzerpools.

Sie müssen eine App entwickeln, die in Amazon Cognito integriert ist, um Ihre Benutzer zu authentifizieren und zu autorisieren, unabhängig davon, ob es sich um Verbund- oder lokale Benutzer handelt.

Die Verantwortlichkeiten einer App als SP für Amazon Cognito

Die Informationen in den Token überprüfen und verarbeiten

In den meisten Szenarien leitet Amazon Cognito Ihren authentifizierten Benutzer an eine App-URL um, die es mit einem Autorisierungscode anhängt. Ihre App tauscht den Code durch Zugriffs-, ID- und Aktualisierungs-Token aus. Dann muss sie die Gültigkeit der Token überprüfen und basierend auf den Ansprüchen in den Token Informationen an Ihren Benutzer senden.

Antworten auf Authentifizierungsereignisse mit API-Anfragen von Amazon-Cognito

Ihre App muss in der API für Amazon-Cognito-Benutzerpools und in die Authentifizierungs-API-Endpunkte integriert sein. Die Authentifizierungs-API meldet Ihren Benutzer an und ab und verwaltet Token. Die Benutzerpool-API verfügt über eine Vielzahl von Operationen, die Ihren Benutzerpool, Ihre Benutzer und die Sicherheit Ihrer Authentifizierungsumgebung verwalten. Ihre App muss wissen, was als Nächstes zu tun ist, wenn sie eine Antwort von Amazon Cognito erhält.

Wissenswertes über die Anmeldung von Drittanbietern bei Amazon-Cognito-Benutzerpools

- Wenn Sie möchten, dass sich Ihre Benutzer bei Verbundanbietern anmelden, müssen Sie eine Domäne auswählen. Dadurch werden die von Amazon Cognito gehostete Benutzeroberfläche

sowie gehostete UI- und OIDC-Endpunkte eingerichtet. Weitere Informationen finden Sie unter Verwenden der eigenen Domäne für die gehostete Benutzeroberfläche.

- Sie können Verbundbenutzer nicht mit API-Operationen wie und anmelden. InitiateAuthAdminInitiateAuth Verbundbenutzer können sich nur mit dem Login-Endpunkt oder dem Autorisieren des Endpunkts anmelden.
- Autorisieren des Endpunkts ist ein Umleitungsendpunkt. Wenn Sie einen Parameter `idp_identifier` oder `identity_provider` in Ihrer Anforderung angeben, wird diese unter Umgehung der gehosteten Benutzeroberfläche automatisch an Ihren IDP umgeleitet. Andernfalls erfolgt eine Umleitung an den Login-Endpunkt der gehosteten Benutzeroberfläche. Ein Beispiel finden Sie unter Beispielszenario: Amazon Cognito-Apps in einem Unternehmens-Dashboard als Lesezeichen speichern.
- Wenn die gehostete Benutzeroberfläche eine Sitzung an einen Verbund-IDP umleitet, enthält Amazon Cognito den `user-agent`-Header `Amazon/Cognito` in der Anforderung.
- Amazon Cognito leitet das Attribut `username` für ein Verbundbenutzerprofil aus einer Kombination aus einer festen Kennung und dem Namen Ihres IDP ab. Wenn Sie einen Benutzernamen generieren möchten, der Ihren benutzerdefinierten Anforderungen entspricht, erstellen Sie eine Zuordnung zum Attribut `preferred_username`. Weitere Informationen finden Sie unter Wissenswertes über Mappings.

  Beispiel: `MyIDP_bob@example.com`
- Amazon Cognito zeichnet Informationen über die Identität Ihres Verbundbenutzers in einem Attribut und einen Anspruch im ID-Token, mit den Namen `identities`, auf. Dieser Anspruch enthält den Anbieter Ihres Benutzers und seine eindeutige ID vom Anbieter. Sie können das Attribut `identities` nicht direkt in einem Benutzerprofil ändern. Weitere Informationen zur Verknüpfung eines Verbundbenutzers finden Sie unter Verknüpfen von Verbundbenutzern mit einem vorhandenen Benutzerprofil.
- Wenn Sie Ihren IDP in einer API-Aufforderung UpdateIdentityProvider aktualisieren, kann es bis zu einer Minute dauern, bis Ihre Änderungen in der gehosteten Benutzeroberfläche angezeigt werden.
- Amazon Cognito unterstützt bis zu 20 HTTP-Umleitungen zwischen Amazon Cognito und Ihrem IDP.
- Wenn sich Ihr Benutzer mit der gehosteten Benutzeroberfläche anmeldet, speichert sein Browser ein verschlüsseltes Cookie für die Anmeldesitzung, das den Client und den Anbieter aufzeichnet, mit dem er sich angemeldet hat. Wenn er erneut versucht, sich mit denselben Parametern anzumelden, verwendet die gehostete Benutzeroberfläche jede bestehende Sitzung, die noch nicht abgelaufen ist, und der Benutzer authentifiziert sich, ohne erneut Anmeldeinformationen

einzugeben. Wenn sich Ihr Benutzer erneut mit einem anderen IdP anmeldet, einschließlich eines Wechsels zu oder von der lokalen Benutzerpool-Anmeldung, muss er Anmeldeinformationen angeben und eine neue Anmeldesitzung erstellen.

Sie können einen beliebigen Benutzerpool einem IdPs beliebigen App-Client zuweisen, und Benutzer können sich nur mit einem IdP anmelden, den Sie ihrem App-Client zugewiesen haben.

Themen

Konfigurieren von Identitätsanbietern für Ihren Benutzerpool.

Auf der Registerkarte Anmeldeerfahrung unter Anmelden eines Verbundidentitätsanbieters können Sie Ihrem Benutzerpool Identitätsanbieter (IdPs) hinzufügen. Weitere Informationen finden Sie unter Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter.

Themen

Einrichten der Benutzeranmeldung mit einem sozialen IdP

Sie können einen Verbund für Amazon-Cognito-Benutzerpools verwenden, um eine Integration von sozialen Identitäts-Anbietern vorzunehmen, wie Facebook, Google oder Login with Amazon.

Um einen Social Identity-Anbieter hinzuzufügen, erstellen Sie zunächst ein Entwickler-Konto bei dem Identitätsanbieter. Sobald Sie Ihr Entwicklerkonto haben, registrieren Sie Ihre Anwendung beim Identitätsanbieter. Der Identitätsanbieter erstellt eine App-ID und einen geheimen App-Schlüssel für Ihre Anwendung, und Sie konfigurieren diese Werte in Ihrem Amazon-Cognito-Benutzerpool.

- Google Identity Platform
- Facebook for Developers
- Login mit Amazon
- Mit Apple anmelden

So integrieren Sie die Benutzeranmeldung mit einem sozialen IdP

1. Melden Sie sich bei der Amazon Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS-Anmeldeinformationen ein.
2. Wählen Sie im Navigationsbereich erst User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus und suchen Sie nach Federated sign-in (Verbundanmeldung).
4. Wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus oder den Facebook-, Google-, Amazon- oder Apple-Identitätsanbieter, den Sie konfiguriert haben. Suchen Sie danach nach Identity provider information (Informationen zu Identitätsanbietern) und wählen Sie Edit (Bearbeiten) aus. Weitere Informationen zum Hinzufügen eines Social-Identity-Anbieters finden Sie unter Verwenden von Anbietern für soziale Identitäten mit einem Benutzerpool.
5. Geben Sie die Informationen Ihres Social-Identity-Anbieters ein, indem Sie je nach Wahl des IdP einen der folgenden Schritte ausführen:

   Für Facebook, Google und Login with Amazon:

   Geben Sie die App-ID und den geheimen App-Schlüssel ein, den Sie beim Erstellen Ihrer Client-App erhalten haben.

   Mit Apple anmelden

   Geben Sie die Service-ID ein, die Sie bei Apple angegeben haben, sowie die Team-ID, Schlüssel-ID und den privaten Schlüssel, den Sie beim Erstellen Ihres App-Clients erhalten haben.
6. Für Authorize scopes (Bereiche autorisieren) geben Sie den Namen der Social-Identity-Anbieterbereiche ein, die Sie Benutzerpool-Attributen zuordnen möchten. Bereiche definieren, auf welche Benutzerattribute (z. B. Name und E-Mail-Adresse) Sie mit Ihrer Anwendung zugreifen möchten. Verwenden Sie bei der Eingabe von Bereichen die folgenden Richtlinien basierend auf Ihrer Wahl des IdP:

- Facebook – Trennen Sie Bereiche durch Kommas. Beispiel:

  ```
  public_profile, email
  ```

- Google, Login with Amazon und Mit Apple anmelden – Trennen Sie Bereiche durch Leerzeichen. Beispiel:
  - Google: `profile email openid`
  - Login with Amazon: `profile postal_code`
  - Mit Apple anmelden: `name email`

> **Note**
>
> Verwenden Sie für „Mit Apple anmelden“ (Konsole) die Kontrollkästchen zur Auswahl des Bereichs.

7. Wählen Sie Save Changes.
8. Wählen Sie auf der Registerkarte App client integration (App-Client-Integration) einen der App-Clients aus der Liste aus und klicken Sie anschließend auf Edit hosted UI settings (Einstellungen für gehostete UI bearbeiten). Fügen Sie unter Identity providers (Identitätsanbieter) den neuen Social-Identity-Anbieter zum App-Client hinzu.
9. Wählen Sie Save Changes.

Weitere Informationen zu sozialen finden Sie IdPsunter Verwenden von Anbietern für soziale Identitäten mit einem Benutzerpool.

Einrichten der Benutzeranmeldung mit einem OIDC IdP

Sie können die Benutzeranmeldung in einen OpenID Connect (OIDC)-Identitätsanbieter integrieren, z. B. Salesforce oder Ping Identity.

So fügen Sie einem Benutzerpool einen OIDC-Anbieter hinzu

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS-Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus dem Navigationsmenü aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.

4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Federated sign-in (Verbundanmeldung) und wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus.
5. Wählen Sie einen OpenID-Connect-Identitätsanbieter aus.
6. Geben Sie einen eindeutigen Namen in Provider name (Anbietername) ein.
7. Geben Sie die Kunden-ID, die Sie von Ihrem Anbieter erhalten haben, in Client ID (Client-ID) ein.
8. Geben Sie das Kundengeheimnis, das Sie von Ihrem Anbieter erhalten haben, in Client secret (Client-Geheimnis) ein.
9. Geben Sie Authorized scopes (Autorisierte Bereiche) für diesen Anbieter ein. Bereiche definieren, welche Gruppen von Benutzerattributen (z. B. `name` und `email`) Ihre Anwendung von Ihrem Anbieter anfordert. Bereiche müssen durch Leerzeichen getrennt werden, gefolgt von einer OAuth 2.0-Angabe.

   Ihr Benutzer wird aufgefordert, dem Versand dieser Attribute an Ihre Anwendung zuzustimmen.
10. Wählen Sie eine Attributanforderungsmethode aus, um Amazon Cognito die HTTP-Methode (entweder GET oder POST) bereitzustellen, mit der Amazon Cognito die Details des Benutzers vom Endpunkt userInfo abruft, der von Ihrem Anbieter betrieben wird.
11. Wählen Sie eine Einrichtungsmethode, um OpenID-Connect-Endpunkte abzurufen, entweder durch Auto fill through issuer URL (Automatisches Ausfüllen durch Aussteller-URL) oder Manual input (Manuelle Eingabe). Verwenden Sie Auto fill through issuer URL (Automatisches Ausfüllen durch Aussteller-URL), wenn Ihr Anbieter einen öffentlichen `.well-known/openid-configuration`-Endpunkt besitzt, von dem Amazon Cognito die URLs der `authorization`, `token`, `userInfo` und `jwks_uri`-Endpunkte abrufen kann.
12. Geben Sie die Aussteller-URL oder `authorization`-, `token`-, `userInfo`- und `jwks_uri`-Endpunkt-URLs von Ihrem Identitätsanbieter ein.

> **Note**
>
> Sie können nur die Portnummern 443 und 80 über die Erkennung sowie automatisch ausgefüllte und manuell eingegebene URLs verwenden. Benutzeranmeldungen schlagen fehl, wenn Ihr OIDC-Anbieter nicht standardmäßige TCP-Ports verwendet.
> Die Aussteller-URL muss mit `https://` beginnen und darf nicht mit dem Zeichen `/` enden. Beispielsweise verwendet Salesforce diese URL: `https://login.salesforce.com`
> Das mit Ihrer Aussteller-URL verknüpfte `openid-configuration`-Dokument muss HTTPS-URLs für die folgenden Werte enthalten: `authorization_endpoint`,

`token_endpoint`, `userinfo_endpoint` und `jwks_uri` aus. Wenn Sie Manual input (Manuelle Eingabe) auswählen, können Sie auch nur HTTPS-URLs eingeben.

13. Standardmäßig wird sub des OIDC-Anspruchs dem Benutzerpool-Attribut Username (Benutzername) zugeordnet. Sie können Benutzerpool-Attributen weitere OIDC-Ansprüche hinzufügen. Geben Sie den OIDC-Anspruch ein, und wählen Sie das entsprechende Benutzerpool-Attribut aus der Dropdown-Liste aus. Beispielsweise wird der Anspruch email (E-Mail) häufig dem Benutzerpool-Attribut Email (E-Mail) hinzugefügt.
14. Ordnen Sie Ihrem Benutzerpool zusätzliche Attribute von Ihrem Identitätsanbieter zu. Weitere Informationen finden Sie unter Geben Sie die Identitätsanbieter-Attributzuordnungen für Ihren Benutzerpool an.
15. Wählen Sie Create (Erstellen) aus.
16. Wählen Sie auf der Registerkarte App client integration (App-Client-Integration) einen der App-Clients aus der Liste aus und klicken Sie anschließend auf Edit hosted UI settings (Einstellungen für gehostete UI bearbeiten). Fügen Sie unter Identity providers (Identitätsanbieter) den neuen OIDC-Identitätsanbieter zum App-Client hinzu.
17. Wählen Sie Save Changes.

Weitere Informationen zu OIDC finden Sie IdPsunter Verwendung von OIDC-Identitätsanbietern mit einem Benutzerpool.

Einrichten der Benutzeranmeldung mit einem SAML IdP

Sie können einen Verbund für Amazon-Cognito-Benutzerpools zur Integration mit einem SAML-Identitätsanbieter (IdP) verwenden. Sie stellen ein Metadaten-Dokument bereit, entweder durch Hochladen der Datei oder durch das Angeben einer Metadaten-Dokument-Endpunkt-URL. Informationen zum Abrufen von Metadatendokumenten für SAML- von Drittanbietern finden Sie IdPsunter Konfiguration Ihres externen SAML-Identitätsanbieters.

So konfigurieren Sie einen SAML-2.0-Identitätsanbieter in Ihrem Benutzerpool:

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS-Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.

4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Federated sign-in (Verbundanmeldung) und wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus.
5. Wählen Sie einen SAML-Identitätsanbieter aus.
6. Geben Sie IDs durch Kommas getrennt ein. Eine ID weist Amazon Cognito an, dass die E-Mail-Adresse zu überprüfen, die ein Benutzer bei der Anmeldung eingibt. Anschließend wird der Benutzer zu dem Anbieter weitergeleitet, der seiner Domäne entspricht.
7. Wählen Sie Add sign-out flow (Abmeldeablauf hinzufügen) aus, wenn Amazon Cognito signierte Abmeldeanfragen an Ihren Anbieter senden soll, wenn sich ein Benutzer abmeldet. Konfigurieren Sie den SAML-2.0-Identitätsanbieter so, dass er Abmeldeantworten an den `https://`*`mydomain.us-east-1.amazoncognito.com`*`/saml2/logout`-Endpunkt sendet, der bei der Konfiguration der gehosteten Benutzeroberfläche erstellt wird. Dieser `saml2/logout`-Endpunkt verwendet POST-Binding.

   > **Note**
   >
   > Wenn Sie diese Option auswählen und Ihr SAML-Identitätsanbieter eine signierte Abmeldeanforderung erwartet, müssen Sie auch das Signaturzertifikat konfigurieren, das von Amazon Cognito mit Ihrem SAML-IdP bereitgestellt wird.
   > Der SAML-IdP verarbeitet die signierte Abmeldeanforderung und die Abmeldung Ihres Benutzers von der Amazon-Cognito-Sitzung.

8. Wählen Sie eine Metadaten-Dokumentquelle aus. Wenn Ihr Identitätsanbieter SAML-Metadaten unter einer öffentlichen URL anbietet, können Sie Metadata document URL (URL für Metadatendokumente) auswählen und die öffentliche URL eingeben. Wählen Sie andernfalls Upload metadata document (Hochladen eines Metadatendokuments) und anschließend eine Metadatendatei aus, die Sie zuvor von Ihrem Anbieter heruntergeladen haben.

   > **Note**
   >
   > Wenn Ihr Anbieter einen öffentlichen Endpunkt besitzt, empfehlen wir Ihnen, eine Metadaten-Dokument-URL einzugeben und kein Dokument hochzuladen. Wenn Sie die URL verwenden, aktualisiert Amazon Cognito Metadaten automatisch. Normalerweise werden die Metadaten alle sechs Stunden oder bevor sie ablaufen aktualisiert, je nachdem, was zuerst eintritt.

9. Ordnen Sie Attribute zwischen Ihrem SAML-Anbieter und Ihrer Anwendung zu, um SAML-Anbieterattribute dem Benutzerprofil in Ihrem Benutzerpool zuzuordnen. Fügen Sie die erforderlichen Attribute Ihres Benutzerpools in Ihre Attributzuordnung ein.

   Wenn Sie beispielsweise das Benutzerpool-Attribut `email`auswählen, geben Sie den SAML-Attributnamen so ein, wie dieser in der SAML-Assertion Ihres Identitätsanbieters angezeigt wird. Ihr Identitätsanbieter bietet möglicherweise SAML-Assertions als Referenz an. Einige Identitätsanbieter verwenden einfache Namen wie z. B. `email`, während andere URL-formatierte Attributnamen verwenden, die wie folgt aussehen:

   ```
   http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
   ```

10. Wählen Sie Create (Erstellen) aus.

> **Note**
>
> Falls die Meldung `InvalidParameterException` angezeigt wird, während Sie einen SAML-IdP mit HTTPS-Metadaten-Endpunkt-URL erstellen, stellen Sie sicher, dass der Metadaten-Endpunkt eine korrekte SSL hat und ein gültiges SSL-Zertifikat vorliegt.
> Ein Beispiel für diese Fehlermeldung wäre: „Fehler beim Abfragen der Metadaten von <Metadaten-Endpunkt>“.

So richten Sie den SAML-IdP zum Hinzufügen eines Signaturzertifikats ein

- Zum Abrufen des Zertifikats mit dem öffentlichen Schlüssel, mit dem der IdP die signierte Abmeldeanforderung verifiziert, wählen Sie unter Aktive SAML-Anbieter im Dialogfeld SAML unter Identitätsanbieter auf der Konsolenseite Verbund die Option Signaturzertifikat anzeigen aus.

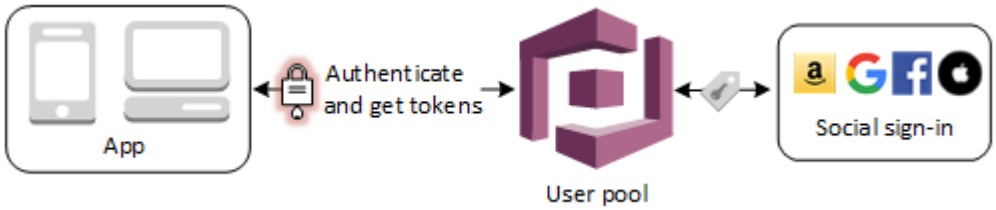
Weitere Informationen zu SAML IdPs finden Sie unter Verwenden von SAML-Identitätsanbietern mit einem Benutzerpool.

Verwenden von Anbietern für soziale Identitäten mit einem Benutzerpool

Die Benutzer Ihrer Web- und mobilen App können sich über Social-Identity-Anbieter (IdP) wie Facebook, Google und Amazon anmelden. Mit der integrierten gehosteten Web-UI bietet Amazon Cognito Token-Handling und Verwaltung für authentifizierte Benutzer aller Identitätsanbieter. Auf diese Weise können Ihre Backend-Systeme auf einen Satz von Benutzerpool-Token standardisiert

werden. Aktivieren Sie die gehostete Benutzeroberfläche, um sie mit unterstützten Social-Identity-Anbietern zu integrieren. Wenn Amazon Cognito Ihre gehostete Benutzeroberfläche erstellt, erstellt es OAuth 2.0-Endpunkte, die Amazon Cognito und Ihr OIDC und Ihre sozialen Netzwerke zum Informationsaustausch verwenden. IdPs Weitere Informationen finden Sie unter Auth-API-Referenz des Amazon-Cognito-Benutzerpools.

Sie können einen sozialen IdP in der hinzufügen AWS Management Console, oder Sie können die AWS CLI oder die Amazon Cognito Cognito-API verwenden.



> **Note**
>
> Die Anmeldung über einen Drittanbieter (Verbund) in Amazon-Cognito-Benutzerpools wird unterstützt. Diese Funktion ist unabhängig von Verbund über Amazon-Cognito-Identitätspools (Verbundidentitäten).

Themen

- Voraussetzungen
- Schritt 1: Registrieren mit einem Social-Identity-Anbieter
- Schritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool
- Schritt 3: Testen der Konfiguration Ihres Social-Identity-Anbieters

Voraussetzungen

Bevor Sie beginnen, muss Folgendes sichergestellt sein:

- Ein Benutzerpool mit einem App-Client und eine Benutzerpool-Domäne. Weitere Informationen finden Sie unter Einen Benutzerpool erstellen.
- Ein Social-IDP.

Schritt 1: Registrieren mit einem Social-Identity-Anbieter

Bevor Sie einen soziale IdP mit Amazon Cognito anlegen, müssen Sie Ihre Anwendung bei dem sozialen IdP registrieren, um eine Kunden-ID und einen geheimen Client-Schlüssel zu erhalten.

Eine App bei Facebook registrieren

1. Erstellen Sie ein Entwickler-Konto bei Facebook.
2. Melden Sie sich mit Ihren Facebook-Anmeldeinformationen an.
3. Wählen Sie im Menü My Apps (Meine Apps) den Eintrag Create New App (Neue App erstellen).
4. Geben Sie einen Namen für Ihre Facebook-App ein und wählen Sie dann Create App ID (App-ID erstellen) aus.
5. Wählen Sie in der linken Navigationsleiste Settings (Einstellungen) aus und dann Basic (Grundlegend).
6. Notieren Sie die App ID und das App Secret (Geheimer Schlüssel für die App). Sie brauchen diese Informationen im nächsten Abschnitt.
7. Wählen Sie unten auf der Seite + Add Platform (+ Plattform hinzufügen).
8. Wählen Sie Website.
9. Geben Sie unter Website den Pfad zur Anmeldeseite Ihrer Anwendung unter Site URL (Website-URL) ein.

   ```
   https://mydomain.us-east-1.amazoncognito.com/login?
   response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
   ```

10. Wählen Sie Save Changes.
11. Geben Sie den Pfad zum Stammverzeichnis Ihrer Benutzerpool-Domäne in App Domains (Anwendungsdomänen) ein.

    ```
    https://mydomain.us-east-1.amazoncognito.com
    ```

12. Wählen Sie Save Changes.
13. Wählen Sie auf der Navigationsleiste Add Product (Produkte hinzufügen) und dann Set up (Einrichten) für Facebook Login (Facebook-Anmeldung).
14. Wählen Sie in der linken Navigationsleiste Facebook Login (Facebook-Anmeldung) und dann Settings (Einstellungen).

Geben Sie den Pfad zum Endpunkt `/oauth2/idpresponse` für Ihre Benutzerpool-Domäne unter Valid OAuth Redirect URIs (Gültige OAuth-Umleitungs-URIs) ein.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

15. Wählen Sie Save Changes.

Eine App bei Amazon registrieren

1. Erstellen Sie ein Entwickler-Konto bei Amazon.
2. Melden Sie sich mit Ihren Amazon-Anmeldeinformationen an.
3. Sie müssen ein Amazon Sicherheitsprofil erstellen, um die Amazon-Client-ID und den geheimen Client-Schlüssel zu erhalten.

   Wählen Sie Apps and Services (Apps und Services) aus der Navigationsleiste oben auf der Seite und wählen Sie dann Login with Amazon (Anmeldung mit Amazon).
4. Wählen Sie Create a Security Profile (Ein Sicherheitsprofil erstellen) aus.
5. Geben Sie einen Security Profile Name (Sicherheitsprofilnamen), eine Security Profile Description (Sicherheitsprofilbeschreibung) und eine Consent Privacy Notice URL (URL zur Zustimmung zum Datenschutzhinweis) ein.
6. Wählen Sie Save (Speichern) aus.
7. Wählen Sie Client ID (Client-ID) und Client Secret (Clientschlüssel), um die Client-ID und den Clientschlüssel anzuzeigen. Sie brauchen diese Informationen im nächsten Abschnitt.
8. Bewegen Sie den Mauszeiger über das Zahnrad, wählen Sie Web Settings (Web-Einstellungen) und dann Edit (Bearbeiten) aus.
9. Geben Sie Ihre Benutzerpool-Domäne in Allowed Origins (Autorisierte Quellen) ein.

   ```
   https://mydomain.us-east-1.amazoncognito.com
   ```
10. Geben Sie Ihre Benutzerpool-Domäne mit dem `/oauth2/idpresponse`-Endpunkt in Allowed Return URLs (Zulässige Rückgabe-URLs) ein.

    ```
    https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
    ```
11. Wählen Sie Save (Speichern) aus.

Eine App bei Google registrieren

Weitere Informationen über OAuth 2.0 auf der Google-Cloud-Plattform finden Sie unter Erfahren Sie mehr über Authentifizierung und Autorisierung in der Dokumentation zu Google Workspace für Entwickler.

1. Erstellen Sie ein Entwickler-Konto bei Google.
2. Melden Sie sich bei der Konsole für Google Cloud Platform an.
3. Klicken Sie in der oberen Navigationsleiste auf Select a project (Projekt auswählen). Wenn Sie bereits ein Projekt auf der Google-Plattform haben, zeigt dieses Menü stattdessen Ihr Standardprojekt an.
4. Wählen Sie NEW PROJECT (NEUES PROJEKT) aus.
5. Geben Sie einen Namen für Ihr Projekt ein und wählen Sie dann CREATE (ERSTELLEN) aus.
6. Wählen Sie in der linken Navigationsleiste APIs and Services (APIs und Services) und dann Oauth consent screen (OAuth-Zustimmungsbildschirm) aus.
7. Geben Sie App-Informationen, eine App domain (App-Domäne), Authorized domains (Autorisierte Domänen) und Developer contact information (Kontaktinformationen für Entwickler) ein. Ihre Authorized domains (Autorisierte Domänen) müssen `amazoncognito.com` und das Stammverzeichnis Ihrer benutzerdefinierten Domäne, z. B. `example.com`, enthalten. Wählen Sie SAVE AND CONTINUE (SPEICHERN UND FORTFAHREN) aus.
8. 1. Wählen Sie unter Scopes (Bereiche) die Option Add or remove scopes (Bereiche hinzufügen oder entfernen) und mindestens die folgenden OAuth-Bereiche aus.

    1. `.../auth/userinfo.email`
    2. `.../auth/userinfo.profile`
    3. openid
9. Wählen Sie unter Test users (Testbenutzer) die Option Add users (Benutzer hinzufügen) aus. Geben Sie Ihre E-Mail-Adresse und weitere autorisierte Testbenutzer ein und wählen Sie dann SAVE AND CONTINUE (SPEICHERN UND FORTFAHREN) aus.
10. Erweitern Sie die linke Navigationsleiste erneut und wählen Sie APIs and Services (APIs und Services) und dann Credentials (Anmeldeinformationen) aus.
11. Klicken Sie auf CREATE CREDENTIALS (ANMELDEINFORMATIONEN ERSTELLEN) und auf OAuth client ID (OAuth-Client-ID).
12. Wählen Sie einen Application type (Anwendungstyp) aus und geben Sie Ihrem Client im Feld Name (Name) einen Namen.

13. Wählen Sie unter Autorisierte JavaScript Ursprünge die Option URI HINZUFÜGEN aus. Geben Sie Ihre Benutzerpool-Domäne ein.

    ```
    https://mydomain.us-east-1.amazoncognito.com
    ```

14. Wählen Sie unter Authorized redirect URIs (Autorisierte Umleitungs-URIs) die Option ADD URI (URI HINZUFÜGEN) aus. Geben Sie den Pfad zum Endpunkt `/oauth2/idpresponse` Ihrer Benutzerpool-Domäne ein.

    ```
    https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
    ```

15. Wählen Sie CREATE (Erstellen) aus.
16. Speichern Sie die Werte sicher, die Google unter Your client ID (Ihre Client-ID) und Your client secret (Ihr Client-Schlüssel) anzeigt. Stellen Sie diese Werte Amazon Cognito zur Verfügung, wenn Sie einen Google-IDP hinzufügen.

Eine App bei Apple registrieren

Die meisten up-to-date Informationen zur Einrichtung von Sign in with Apple finden Sie unter Konfiguration Ihrer Umgebung für die Anmeldung mit Apple in der Apple-Dokumentation für Entwickler.

1. Erstellen Sie ein Entwickler-Konto bei Apple.
2. Melden Sie sich mit Ihren Apple-Anmeldeinformationen an.
3. Wählen Sie in der linken Navigationsleiste Certificates, Identifiers & Profiles (Zertifikate, IDs und Profile) aus.
4. Wählen Sie in der linken Navigationsleiste Kennungen aus.
5. Wählen Sie auf der Seite Kennungen das Symbol + aus.
6. Wählen Sie auf der Seite Neue Kennung registrieren die Option App-IDs und dann Weiter aus.
7. Wählen Sie auf der Seite Select a type (Typ auswählen) die Option App und dann Continue (Weiter) aus.
8. Machen Sie auf der Seite Registrieren einer App-ID das Folgende:

    1. Geben Sie unter Description (Beschreibung) eine Beschreibung ein.
    2. Geben Sie unter App ID Prefix (App-ID-Präfix) eine Bundle ID (Bündel-ID) ein. Notieren Sie sich den Wert unter App ID Prefix (App-ID-Präfix). Sie benötigen diesen Wert, nachdem Sie

Apple als Identitätsanbieter in Schritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool ausgewählt haben.

3. Wählen Sie unter Funktionen die Option Mit Apple anmelden und dann Bearbeiten aus.
4. Wählen Sie auf der Seite Sign in with Apple: App ID Configuration (Mit Apple anmelden: App-ID-Konfiguration) aus, ob Sie die App entweder als primär oder mit anderen App-IDs gruppiert einrichten möchten. Klicken Sie dann auf Save (Speichern).
5. Klicken Sie auf Continue.

9. Wählen Sie auf der Seite App-ID bestätigen die Option Registrieren aus.
10. Wählen Sie auf der Seite Kennungen das Symbol + aus.
11. Wählen Sie auf der Seite Neue Kennung registrieren die Option Services-IDs und dann Weiter aus.
12. Machen Sie auf der Seite Registrieren einer Service-ID das Folgende:

    1. Geben Sie unter Beschreibung eine Beschreibung ein.
    2. Geben Sie unter Kennungen eine Kennung ein. Notieren Sie sich diese Service-ID, da Sie diesen Wert benötigen, nachdem Sie Apple als Identitätsanbieter in ausgewählt haben Schritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool.
    3. Wählen Sie Continue (Weiter) und dann Register (Registrieren) aus.
13. Wählen Sie auf der Seite „Identifiers“ (Bezeichner) die gerade erstellte Service-ID aus.

    1. Wählen Sie Mit Apple anmelden und dann Konfigurieren aus.
    2. Wählen Sie auf der Seite Web Authentication Configuration (Konfiguration der Web-Authentifizierung) die App-ID, die Sie zuvor erstellt haben, als Primary App ID (Primäre App-ID) aus.
    3. Wählen Sie neben Website URLs (Website-URLs) das Symbol + aus.
    4. Geben Sie unter Domains and subdomains (Domänen und Subdomänen) Ihre Benutzerpool-Domäne ohne das Präfix `https://` ein.

       ```
       mydomain.us-east-1.amazoncognito.com
       ```

    5. Geben Sie unter Return URLs (URLs zurückgeben) den Pfad zum Endpunkt `/oauth2/idpresponse` Ihrer Benutzerpool-Domäne ein.

       ```
       https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
       ```

6. Wählen Sie Next (Weiter) und anschließend Done (Fertig) aus. Sie müssen die Domäne nicht verifizieren.
7. Wählen Sie Continue (Weiter) und anschließend Save (Speichern) aus.

14. Wählen Sie in der linken Navigationsleiste die Option Schlüssel aus.
15. Klicken Sie auf der Seite Schlüssel auf das Symbol +.
16. Machen Sie auf der Seite Registrieren eines neuen Schlüssels das Folgende:

    1. Geben Sie unter Key Name (Schlüsselname) einen Schlüsselnamen ein.
    2. Wählen Sie Mit Apple anmelden und dann Konfigurieren aus.
    3. Wählen Sie auf der Seite Configure Key (Schlüssel konfigurieren) die App-ID, die Sie zuvor erstellt haben, als Primary App ID (Primäre App-ID) aus. Wählen Sie Speichern.
    4. Wählen Sie Weiter und dann Registrieren aus.
17. Wählen Sie auf der Seite Download Your Key (Schlüssel herunterladen) die Option Download (Herunterladen) aus, um den privaten Schlüssel herunterzuladen. Notieren Sie die angezeigte Schlüssel-ID und klicken Sie anschließend auf Done (Fertig). Sie benötigen diesen privaten Schlüssel und den auf dieser Seite angezeigten Wert für die Schlüssel-ID, nachdem Sie Apple als Identitätsanbieter in Schritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool ausgewählt haben.

Schritt 2: Hinzufügen eines Social-Identity-Anbieters zu Ihrem Benutzerpool

Um einen Benutzerpool Social IdP mit dem zu konfigurieren AWS Management Console

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Federated sign-in (Verbundanmeldung) und wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus.
5. Wählen Sie einen Social-IdP aus: Facebook, Google, Login with Amazon oder Mit Apple anmelden.
6. Wählen Sie basierend auf Ihrer Wahl des Social-IdP einen der folgenden Schritte aus:

- Google und Login with Amazon – Geben Sie die App-Client-ID und das App-Clientgeheimnis ein, das Sie im vorherigen Abschnitt erstellt haben.
- Facebook – Geben Sie die App-Client-ID und das App-Clientgeheimnis ein, das Sie im vorherigen Abschnitt erstellt haben. Wählen Sie anschließend eine API-Version aus (z. B. Version 2.12). Wir empfehlen, die neueste Version auszuwählen, da jede Facebook-API-Version einen Lebenszyklus und ein Abkündigungsdatum hat. Facebook-Bereiche und Attribute können zwischen API-Versionen variieren. Wir empfehlen, Ihre Social-Identity-Anmeldung mit Facebook zu testen, um sicherzustellen, dass der Verbund wie vorgesehen funktioniert.
- Mit Apple anmelden – Geben Sie die Service-ID, Team-ID, Schlüssel-ID und den privaten Schlüssel ein, die/den Sie im vorherigen Abschnitt erstellt haben.

7. Geben Sie die Namen der autorisierten Bereiche ein, die Sie verwenden möchten. Bereiche definieren, auf welche Benutzerattribute (wie z. B. `name` und `email`) mit Ihrer App zugreifen möchten. Für Facebook müssen diese durch Kommata voneinander getrennt werden. Für Google und "Login with Amazon (Anmelden mit Amazon)" müssen die Werte mit Leerzeichen getrennt werden. Aktivieren Sie für "Sign in with Apple (Mit Apple anmelden)" die Kontrollkästchen der Bereiche, auf die Sie Zugriff benötigen.

Anbieter sozialer Identitäten	Beispiel-Bereiche
Facebook	`public_profile, email`
Google	`profile email openid`
Login with Amazon	`profile postal_code`
Mit Apple anmelden	`email name`

Der App-Benutzer wird aufgefordert, der Bereitstellung dieser Attribute für die App zuzustimmen. Weitere Informationen zu den jeweiligen Bereichen enthält die Dokumentation von Google, Facebook, Login with Amazon oder Mit Apple anmelden.

Bei Verwendung von „Mit Apple anmelden“ werden Bereiche in den folgenden Benutzerszenarien unter Umständen nicht zurückgegeben:

- Bei einem Endbenutzer kommt es nach dem Verlassen der Anmeldeseite von Apple zu Fehlern (interne Fehler von Amazon Cognito oder Probleme in dem vom Entwickler geschriebenen Quelltext).
- Die Service-ID wird über Benutzerpools und/oder andere Authentifizierungs-Services hinweg verwendet.
- Ein Entwickler fügt Bereiche hinzu, nachdem der betreffende Endbenutzer sich angemeldet hat (es werden keine neuen Informationen abgerufen).
- Ein Entwickler löscht den Benutzer und der Benutzer meldet sich dann erneut an, ohne die App zuvor aus seinem Apple-ID-Profil zu entfernen.

8. Ordnen Sie Ihrem Benutzerpool Attribute von Ihrem IdP zu. Weitere Informationen finden Sie unter Geben Sie die Identitätsanbieter-Attributzuordnungen für Ihren Benutzerpool an.
9. Wählen Sie Create (Erstellen) aus.
10. Wählen Sie auf der Registerkarte App client integration (App-Client-Integration) einen der App-Clients aus der Liste aus und klicken Sie anschließend auf Edit hosted UI settings (Einstellungen für gehostete UI bearbeiten). Fügen Sie unter Identity providers (Identitätsanbieter) den neuen Social-IdP zum App-Client hinzu.
11. Wählen Sie Save Changes.

Schritt 3: Testen der Konfiguration Ihres Social-Identity-Anbieters

Unter Verwendung der Elemente aus den vorherigen zwei Abschnitten können Sie eine Anmelde-URL erstellen. Verwenden Sie sie zum Testen der Konfiguration Ihres sozialen IdP.

```
https://mydomain.us-east-1.amazoncognito.com/login?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

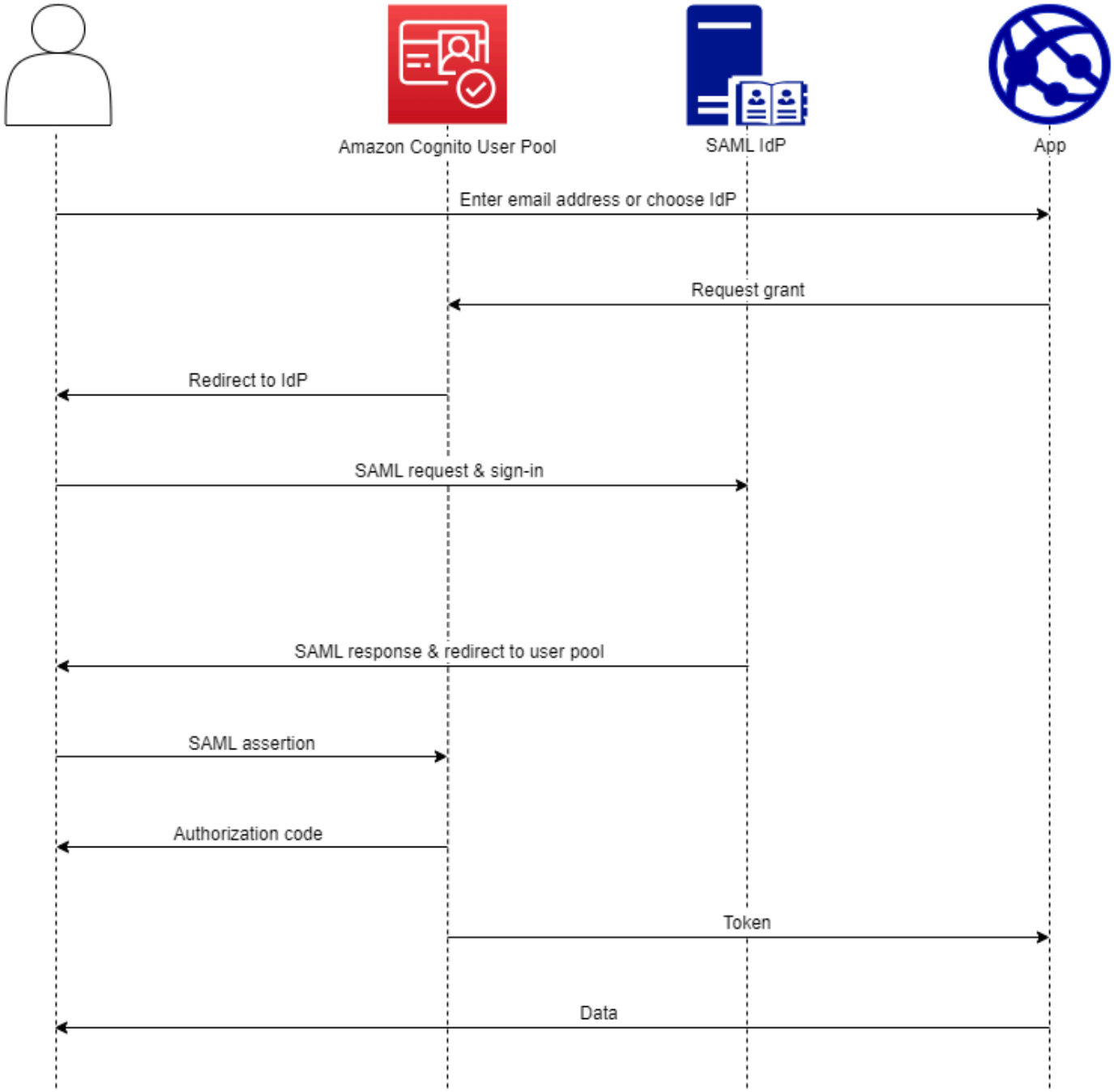
Sie finden Ihre Domäne auf der Konsolenseite Domain name (Domänenname) für den Benutzerpool. Die Client-ID befindet sich auf der Registerkarte App client settings (App-Client-Einstellungen). Verwenden Sie Ihre Callback-URL für den redirect_uri-Parameter. Dies ist die URL der Seite, auf die Ihre Benutzer nach einer erfolgreichen Authentifizierung umgeleitet werden.

> **Note**
>
> Amazon Cognito bricht Authentifizierungsanfragen ab, die nicht innerhalb von 5 Minuten abgeschlossen werden, und leitet den Benutzer an die gehostete Benutzeroberfläche um. Für die Seite wird eine `Something went wrong`-Fehlermeldung angezeigt.

Verwenden von SAML-Identitätsanbietern mit einem Benutzerpool

Sie können wählen, ob sich Ihre Web- und mobilen App-Benutzer über einen SAML-Identitätsanbieter (IdP) wie Microsoft Active Directory Federation Services (ADFS) oder Shibboleth anmelden. Sie müssen einen SAML-IdP auswählen, der den SAML-2.0-Standard unterstützt.

Mit der gehosteten Benutzeroberfläche und den Verbundendpunkten authentifiziert Amazon Cognito lokale und Drittanbieter-IdP-Benutzer und gibt JSON-Web-Tokens (JWTs) aus. Mit den Tokens, die Amazon Cognito ausgibt, können Sie mehrere Identitätsquellen in einem universellen OpenID Connect (OIDC) -Standard für all Ihre Apps konsolidieren. Amazon Cognito kann SAML-Assertionen von Ihren Drittanbietern in diesen SSO-Standard umwandeln. Sie können einen SAML-IdP in der AWS Management Console, über oder mit der Amazon Cognito Cognito-Benutzerpools-API erstellen und verwalten. AWS CLI Informationen zum Erstellen Ihres ersten SAML-IdP in der finden Sie AWS Management Console unter. Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool



> **Note**
>
> Der Verbund mit Anmeldung über einen Drittanbieter-IdP ist eine Funktion der Amazon Cognito Cognito-Benutzerpools. Amazon Cognito-Identitätspools, manchmal auch Amazon Cognito Federated Identities genannt, sind eine Implementierung von Federation, die Sie in jedem Identitätspool separat einrichten müssen. Ein Benutzerpool kann ein Drittanbieter-

IdP für einen Identitätspool sein. Weitere Informationen finden Sie unter Amazon-Cognito-Identitätspools.

Kurzreferenz für die IdP-Konfiguration

Sie müssen Ihren SAML-IdP so konfigurieren, dass er Anfragen akzeptiert und Antworten an Ihren Benutzerpool sendet. Die Dokumentation für Ihren SAML-IdP enthält Informationen darüber, wie Sie Ihren Benutzerpool als vertrauende Partei oder Anwendung für Ihren SAML 2.0-IdP hinzufügen können. Die folgende Dokumentation enthält die Werte, die Sie für die SP-Entitäts-ID und die URL des Assertion Consumer Service (ACS) angeben müssen.

Kurzreferenz zu den SAML-Werten für den Benutzerpool

SP-Entitäts-ID

```
urn:amazon:cognito:sp:us-east-1_EXAMPLE
```

ACS-URL

```
https://Your user pool domain/saml2/idpresponse
```

Sie müssen Ihren Benutzerpool so konfigurieren, dass er Ihren Identitätsanbieter unterstützt. Die allgemeinen Schritte zum Hinzufügen eines externen SAML-IdP lauten wie folgt.

1. Laden Sie SAML-Metadaten von Ihrem IdP herunter oder rufen Sie die URL zu Ihrem Metadaten-Endpunkt ab. Siehe Konfiguration Ihres externen SAML-Identitätsanbieters.
2. Fügen Sie Ihrem Benutzerpool einen neuen IdP hinzu. Laden Sie die SAML-Metadaten hoch oder geben Sie die Metadaten-URL an. Siehe Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool.
3. Weisen Sie den IdP Ihren App-Clients zu. Siehe App-Clients für Benutzerpools

Themen

- Wissenswertes über SAML IdPs in Amazon Cognito Cognito-Benutzerpools
- Groß- und Kleinschreibung von SAML-Benutzernamen
- Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool

- Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools
- Verwenden der SP-initiierten SAML-Anmeldung
- Verwenden der IDP-initiierten SAML-Anmeldung
- Ablauf der SAML-Abmeldung
- SAML-Signatur und Verschlüsselung
- Namen und Kennungen von SAML-Identitätsanbietern
- Konfiguration Ihres externen SAML-Identitätsanbieters

Wissenswertes über SAML IdPs in Amazon Cognito Cognito-Benutzerpools

Amazon Cognito verarbeitet SAML-Assertionen für Sie

Amazon-Cognito-Benutzerpools unterstützen den SAML-2.0-Verbund mit POST-Binding-Endpunkten. Auf diese Weise ist es nicht mehr notwendig, dass Ihre Anwendung SAML-Assertion-Rückmeldungen lädt oder analysiert, da der Benutzerpool die SAML-Rückmeldungen nun direkt von Ihrem IdP über einen Benutzeragenten erhält. Ihr Benutzerpool fungiert als Dienstanbieter (SP) im Namen Ihrer Anwendung. Amazon Cognito unterstützt SP-initiiertes und IdP-initiiertes Single Sign-On (SSO), wie in den Abschnitten 5.1.2 und 5.1.4 der technischen Übersicht über SAML V2.0 beschrieben.

Geben Sie ein gültiges IdP-Signaturzertifikat an

Das Signaturzertifikat in Ihren SAML-Provider-Metadaten darf nicht abgelaufen sein, wenn Sie den SAML-IdP in Ihrem Benutzerpool konfigurieren.

Benutzerpools unterstützen mehrere Signaturzertifikate

Wenn Ihr SAML-IdP mehr als ein Signaturzertifikat in SAML-Metadaten enthält, stellt Ihr Benutzerpool bei der Anmeldung fest, dass die SAML-Assertion gültig ist, sofern sie mit einem Zertifikat in den SAML-Metadaten übereinstimmt. Jedes Signaturzertifikat darf nicht länger als 4.096 Zeichen sein.

Behalten Sie den Relay-State-Parameter bei

Amazon Cognito und Ihr SAML-IdP verwalten Sitzungsinformationen mit einem `relayState`-Parameter.

1. Amazon Cognito unterstützt `relayState`-Werte, die größer als 80 Byte sind. In SAML-Spezifikationen ist zwar angegeben, dass der Wert für `relayState` nicht größer als 80 Byte

sein darf, aber in der Branche wird hiervon derzeit häufiger abgewichen. Die Folge ist, dass die Ablehnung von `relayState`-Werten mit mehr als 80 Byte dazu führt, dass für viele Standardintegrationen von SAML-Anbietern Fehler auftreten.

2. Das `relayState` Token ist ein undurchsichtiger Verweis auf Statusinformationen, die von Amazon Cognito verwaltet werden. Amazon Cognito übernimmt keine Garantie für den Inhalt des `relayState`-Parameters. Parsen Sie den Inhalt nicht so, dass Ihre App vom Ergebnis abhängig ist. Weitere Informationen finden Sie in der Spezifikation zu SAML 2.0.

Identifizieren Sie den ACS-Endpunkt

Ihr SAML-Identitätsanbieter verlangt, dass Sie einen Assertions-Verbraucher-Endpunkt festlegen. Ihr IdP leitet Ihre Benutzer mit ihrer SAML-Assertion an diesen Endpunkt weiter. Konfigurieren Sie den folgenden Endpunkt in Ihrer Benutzerpool-Domäne für SAML-2.0-POST-Binding Ihres SAML-Identitätsanbieters.

```
https://Your user pool domain/saml2/idpresponse
With an Amazon Cognito domain:
https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse
With a custom domain:
https://auth.example.com/saml2/idpresponse
```

Weitere Informationen zum Erstellen von Benutzerpool-Domänen finden Sie unter Konfigurieren einer Benutzerpool-Domäne.

Keine wiederholten Assertionen

Sie können eine SAML-Assertion nicht wiederholen und nicht erneut an Ihren Amazon-Cognito-`saml2/idpresponse`-Endpunkt wiedergeben. Eine erneut wiedergegebene SAML-Zusicherung hat eine Assertion-ID, die die ID einer früheren IdP-Antwort dupliziert.

Die Benutzerpool-ID ist die SP-Entitäts-ID

Sie müssen Ihrem IdP Ihre Benutzerpool-ID im Service Provider (SP) mitteilen`urn`, die auch als Zielgruppen-URI oder SP-Entitäts-ID bezeichnet wird. Der Zielgruppen-URI für Ihren Benutzerpool hat das folgende Format.

```
urn:amazon:cognito:sp:us-east-1_EXAMPLE
```

Sie finden Ihre Benutzerpool-ID unter Benutzerpool-Übersicht in der Amazon Cognito Cognito-Konsole.

Ordnen Sie alle erforderlichen Attribute zu

Konfigurieren Sie Ihren SAML-IdP so, dass dieser Werte für alle in Ihrem Benutzerpool erforderlichen Attribute bereitstellt. Beispielsweise ist `email` ein typisches erforderliches Attribut für Benutzerpools. Bevor sich Ihre Benutzer anmelden können, müssen Ihre SAML-IdP-Assertions einen Anspruch enthalten, den Sie dem Benutzerpool-Attribut `email` zuordnen. Weitere Informationen zu Attributzuordnung finden Sie unter Geben Sie die Identitätsanbieter-Attributzuordnungen für Ihren Benutzerpool an.

Für das Assertion-Format gelten spezifische Anforderungen

Ihr SAML-IdP muss die folgenden Ansprüche in die SAML-Assertion aufnehmen.

1. `NameID`Ein Anspruch. Amazon Cognito ordnet dem Zielbenutzer von eine SAML-Assertion zu. `NameID` Bei `NameID` Änderungen geht Amazon Cognito davon aus, dass die Assertion für einen neuen Benutzer gilt. Das Attribut, auf das Sie `NameID` in Ihrer IdP-Konfiguration festgelegt haben, muss einen dauerhaften Wert haben.

   ```
   <saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:persistent">
     carlos
   </saml2:NameID>
   ```

2. Ein `AudienceRestriction`-Anspruch mit einem `Audience`-Wert, der die SP-Entitäts-ID Ihres Benutzerpools als Ziel der Antwort festlegt.

   ```
   <saml:AudienceRestriction>
     <saml:Audience> urn:amazon:cognito:sp:us-east-1_EXAMPLE
   </saml:AudienceRestriction>
   ```

3. Bei SP-initiiertem Single Sign-On ein `Response` Element mit einem `InResponseTo` Wert der ursprünglichen SAML-Anforderungs-ID.

   ```
   <saml2p:Response Destination="https://mydomain.us-east-1.amazoncognito.com/
   saml2/idpresponse" ID="id123" InResponseTo="_dd0a3436-bc64-4679-
   a0c2-cb4454f04184" IssueInstant="Date-time stamp" Version="2.0"
    xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol" xmlns:xs="http://
   www.w3.org/2001/XMLSchema">
   ```

   > **Note**
   >
   > IDP-initiierte SAML-Assertionen dürfen keinen Wert enthalten. `InResponseTo`

4. Ein `SubjectConfirmationData` Element mit dem `Recipient` Wert Ihres `saml2/idpresponse` Benutzerpool-Endpunkts und, bei SP-initiiertem SAML, einem `InResponseTo` Wert, der der ursprünglichen SAML-Anforderungs-ID entspricht.

```
<saml2:SubjectConfirmationData InResponseTo="_dd0a3436-bc64-4679-a0c2-
cb4454f04184" NotOnOrAfter="Date-time stamp" Recipient="https://mydomain.us-
east-1.amazoncognito.com/saml2/idpresponse"/>
```

SP-initiierte Anmeldeanfragen

Wenn der Autorisieren des Endpunkts Ihren Benutzer zur Anmeldeseite Ihres IDP umleitet, schließt Amazon Cognito eine SAML-Anforderung in einem URL-Parameter der `HTTP GET`-Anforderung ein. Eine SAML-Anfrage enthält Informationen über Ihren Benutzerpool, einschließlich Ihres ACS-Endpunkts. Sie können diese Anfragen optional mit einer kryptografischen Signatur versehen.

Signieren Sie Anfragen und verschlüsseln Sie Antworten

Jeder Benutzerpool mit einem SAML-Anbieter generiert ein asymmetrisches key pair und ein Signaturzertifikat für eine digitale Signatur, die Amazon Cognito SAML-Anfragen zuweist. Jeder externe SAML-IdP, den Sie für die Unterstützung verschlüsselter SAML-Antworten konfigurieren, veranlasst Amazon Cognito, ein neues key pair und ein neues Verschlüsselungszertifikat für diesen Anbieter zu generieren. Um die Zertifikate mit dem öffentlichen Schlüssel anzusehen und herunterzuladen, wählen Sie in der Amazon Cognito Cognito-Konsole auf der Registerkarte Anmeldeerfahrung Ihren IdP aus.

Um SAML-Anfragen aus Ihrem Benutzerpool vertrauenswürdig zu machen, stellen Sie Ihrem IdP eine Kopie Ihres SAML 2.0-Signaturzertifikats Ihres Benutzerpools zur Verfügung. Ihr IdP ignoriert möglicherweise SAML-Anfragen, die Ihr Benutzerpool signiert hat, wenn Sie den IdP nicht so konfigurieren, dass er signierte Anfragen akzeptiert.

1. Amazon Cognito wendet eine digitale Signatur auf SAML-Anfragen an, die Ihr Benutzer an Ihren IdP weiterleitet. Ihr Benutzerpool signiert alle Single Logout (SLO) -Anfragen, und Sie können Ihren Benutzerpool so konfigurieren, dass Single Sign-On-Anfragen (SSO) für jeden externen SAML-IdP signiert werden. Wenn Sie eine Kopie des Zertifikats bereitstellen, kann Ihr IdP die Integrität der SAML-Anfragen Ihrer Benutzer überprüfen.
2. Ihr SAML-IdP kann SAML-Antworten mit dem Verschlüsselungszertifikat verschlüsseln. Wenn Sie einen IdP mit SAML-Verschlüsselung konfigurieren, darf Ihr IdP nur verschlüsselte Antworten senden.

Kodieren Sie nicht-alphanumerische Zeichen

Amazon Cognito akzeptiert keine 4-Byte-UTF-8-Zeichen wie # oder, die Ihr IdP als Attributwert übergibt. Sie können das Zeichen mit Base64 codieren, um es als Text zu übergeben, und es dann in Ihrer App decodieren.

Im folgenden Beispiel wird der Attributanspruch nicht akzeptiert:

```
<saml2:Attribute Name="Name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:unspecified">
  <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="xsd:string">#</saml2:AttributeValue>
</saml2:Attribute>
```

Im Gegenteil zum vorherigen Beispiel wird der Attributanspruch akzeptiert:

```
<saml2:Attribute Name="Name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:unspecified">
  <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="xsd:string">8J+YkA==</saml2:AttributeValue>
</saml2:Attribute>
```

Der Metadaten-Endpunkt muss über eine gültige Transportschichtsicherheit verfügen

Falls die Meldung `InvalidParameterException` angezeigt wird, während Sie einen SAML-IdP mit HTTPS-Metadaten-Endpunkt-URL erstellen, stellen Sie sicher, dass der Metadaten-Endpunkt eine korrekte SSL hat und ein gültiges SSL-Zertifikat vorliegt. Ein Beispiel für diese Fehlermeldung sieht aus wie folgt: „Fehler bei Abfragen der Metadaten von *<Metadaten-Endpunkt>*". Weitere Informationen zur Validierung von Zertifikaten finden Sie unter Was ist ein SSL/TLS-Zertifikat? .

App-Clients mit IDP-initiierter SAML können sich nur mit SAML anmelden

Wenn Sie die Unterstützung für einen SAML 2.0-IdP aktivieren, der die vom IdP initiierte Anmeldung in einem App-Client unterstützt, können Sie diesem App-Client nur andere SAML IdPs 2.0-Dateien hinzufügen. Sie dürfen das Benutzerverzeichnis im Benutzerpool und alle externen Identitätsanbieter, die keine SAML sind, nicht zu einem auf diese Weise konfigurierten App-Client hinzufügen.

Bei Abmeldeantworten muss die POST-Bindung verwendet werden

Der `/saml2/logout` Endpunkt akzeptiert `LogoutResponse` `HTTP POST` AS-Anfragen. Benutzerpools akzeptieren keine `HTTP GET` verbindlichen Abmeldeantworten.

Groß- und Kleinschreibung von SAML-Benutzernamen

Wenn ein Verbundbenutzer versucht, sich anzumelden, übergibt der SAML-Identitätsanbieter (IdP) in der SAML-Assertion des Benutzers einen eindeutigen `NameId` Wert für Amazon Cognito. Amazon Cognito identifiziert einen SAML-Verbundbenutzer anhand seines `NameId`-Anspruchs. Unabhängig von den Einstellungen für die Berücksichtigung von Groß- und Kleinschreibung in Ihrem Benutzerpool erkennt Amazon Cognito einen zurückkehrenden Verbundbenutzer von einem SAML-IdP, wenn er seinen eindeutigen Antrag unter Berücksichtigung der Groß- und Kleinschreibung weitergibt. `NameId` Wenn Sie ein Attribut wie `email` `NameId` zuordnen und Ihr Benutzer seine E-Mail-Adresse ändert, kann er sich nicht bei Ihrer App anmelden.

Ordnen Sie `NameId` in Ihren SAML-Assertionen von einem IDP-Attribut zu, das Werte enthält, die sich nicht ändern.

Carlos verfügt beispielsweise über ein Benutzerprofil in Ihrem Benutzerpool ohne Berücksichtigung der Groß- und Kleinschreibung aus einer SAML-Assertion von Active Directory Federation Services (ADFS), die den `NameId`-Wert `Carlos@example.com` übergibt. Wenn Carlos das nächste Mal versucht, sich anzumelden, übergibt Ihr ADFS-IDP den `NameId`-Wert `carlos@example.com`. Da die Groß- und Kleinschreibung von `NameId` exakt übereinstimmen muss, ist die Anmeldung nicht erfolgreich.

Wenn Ihre Benutzer sich nicht anmelden können, nachdem sich ihre `NameID` geändert hat, löschen Sie ihre Benutzerprofile aus Ihrem Benutzerpool. Amazon Cognito erstellt bei der nächsten Anmeldung neue Benutzerprofile.

Themen

- Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool
- Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools
- Verwenden der SP-initiierten SAML-Anmeldung
- Verwenden der IDP-initiierten SAML-Anmeldung
- Ablauf der SAML-Abmeldung
- SAML-Signatur und Verschlüsselung

- Namen und Kennungen von SAML-Identitätsanbietern
- Konfiguration Ihres externen SAML-Identitätsanbieters

Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool

Die folgenden Verfahren zeigen, wie Sie SAML-Anbieter in einem Amazon Cognito Cognito-Benutzerpool erstellen, ändern und löschen.

AWS Management Console

Sie können den verwenden AWS Management Console , um SAML-Identitätsanbieter zu erstellen und zu löschen (). IdPs

Bevor Sie einen SAML-IdP erstellen, benötigen Sie das SAML-Metadatendokument, das Sie vom Drittanbieter-IdP erhalten. Anweisungen zum Abrufen oder Generieren des erforderlichen SAML-Metadaten-Dokuments finden Sie unter Konfiguration Ihres externen SAML-Identitätsanbieters.

Konfigurieren Sie einen SAML-2.0-IdP in Ihrem Benutzerpool wie folgt

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS -Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Federated sign-in (Verbundanmeldung) und wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus.
5. Wählen Sie einen SAML-IDP aus.
6. Geben Sie einen Anbieternamen ein. Sie können diesen benutzerfreundlichen Namen in einem `identity_provider` Anforderungsparameter an den übergebenAutorisieren des Endpunkts.
7. Geben Sie IDs durch Kommas getrennt ein. Eine ID teilt Amazon Cognito mit, dass die E-Mail-Adresse überprüft werden sollte, die ein Benutzer bei der Anmeldung eingibt. Anschließend werden Benutzer zu dem Anbieter weitergeleitet, der ihrer Domäne entspricht.
8. Wählen Sie Add sign-out flow (Abmeldeablauf hinzufügen) aus, wenn Amazon Cognito signierte Abmeldeanfragen an Ihren Anbieter senden soll, wenn sich ein Benutzer abmeldet. Sie müssen Ihren SAML-2.0-IdP so konfigurieren, dass er Abmeldeantworten an den

`https://mydomain.us-east-1.amazoncognito.com/saml2/logout`-Endpunkt sendet, der erstellt wird, wenn Sie die gehostete Benutzeroberfläche konfigurieren. Dieser `saml2/logout`-Endpunkt verwendet POST-Binding.

> **Note**
>
> Wenn diese Option ausgewählt ist und Ihr SAML-IdP eine signierte Abmeldeanforderung erwartet, müssen Sie Ihrem SAML-IdP auch das Signaturzertifikat aus Ihrem Benutzerpool zur Verfügung stellen.
> Der SAML-IdP verarbeitet die signierte Abmeldeanforderung und meldet Ihren Benutzer von der Amazon-Cognito-Sitzung ab.

9. Wählen Sie Ihre IDP-initiierte SAML-Anmeldekonfiguration. Wählen Sie aus Sicherheitsgründen die Option Nur SP-initiierte SAML-Assertionen akzeptieren. Wenn Sie Ihre Umgebung so vorbereitet haben, dass sie unaufgeforderte SAML-Anmeldesitzungen sicher akzeptiert, wählen Sie SP-initiierte und IdP-initiierte SAML-Assertionen akzeptieren. Weitere Informationen finden Sie unter Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools.
10. Wählen Sie eine Metadaten-Dokumentquelle aus. Wenn Ihr IdP SAML-Metadaten unter einer öffentlichen URL anbietet, können Sie Metadata document URL (URL für Metadatendokumente) auswählen und die öffentliche URL eingeben. Wählen Sie andernfalls Upload metadata document (Hochladen eines Metadatendokuments) und anschließend eine Metadatendatei aus, die Sie zuvor von Ihrem Anbieter heruntergeladen haben.

    > **Note**
    >
    > Wir empfehlen, dass Sie eine URL für ein Metadaten-Dokument eingeben, wenn Ihr Anbieter über einen öffentlichen Endpunkt verfügt, anstatt eine Datei hochzuladen. Amazon Cognito aktualisiert automatisch Metadaten aus der Metadaten-URL. Normalerweise werden die Metadaten alle sechs Stunden oder bevor sie ablaufen aktualisiert, je nachdem, was zuerst eintritt.

11. Ordnen Sie Attribute zwischen Ihrem SAML-Anbieter und Ihrem Benutzerpool zu, um SAML-Anbieterattribute dem Benutzerprofil in Ihrem Benutzerpool zuzuordnen. Fügen Sie die erforderlichen Attribute Ihres Benutzerpools in Ihre Attributzuordnung ein.

    Wenn Sie beispielsweise das Benutzerpool-Attribut `email` auswählen, geben Sie den SAML-Attributnamen so ein, wie dieser in der SAML-Assertion Ihres IdP angezeigt wird. Bietet Ihr

IdP Beispiele für SAML-Assertionen, können Sie sich die Namensfindung erleichtern. Einige IdPs verwenden einfache Namen, z. B.`email`, während andere Namen wie die folgenden verwenden.

```
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

12. Wählen Sie Erstellen.

API/CLI

Verwenden Sie die folgenden Befehle zum Erstellen und Verwalten eines SAML-Identitätsanbieters (IdP).

Erstellen Sie einen IdP und laden Sie ein Metadaten-Dokument wie folgt hoch

- AWS CLI: `aws cognito-idp create-identity-provider`

  Beispiel mit Metadaten-Dokument: : `aws cognito-idp create-identity-provider --user-pool-id` *`us-east-1_EXAMPLE`* `--provider-name=`*`SAML_provider_1`* `--provider-type SAML --provider-details file:///details.json --attribute-mapping` *`email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`*

  Wo `details.json` enthält:

```
"ProviderDetails": {
      "MetadataFile": "<SAML metadata XML>",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

> **Note**
>
> Wenn <SAML metadata XML>das Zeichen vorkommt", müssen Sie es \ als Escape-Zeichen hinzufügen:\".

Beispiel mit Metadaten-URL: : `aws cognito-idp create-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-type SAML --provider-details MetadataURL=https://myidp.example.com/sso/saml/metadata --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

- AWS API: CreateIdentityProvider

So laden Sie ein neues Metadatendokument für einen Identitätsanbieter hoch

- AWS CLI: `aws cognito-idp update-identity-provider`

  Beispiel mit Metadaten-Dokument: : `aws cognito-idp update-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

  Wo `details.json` enthält:

```
"ProviderDetails": {
      "MetadataFile": "<SAML metadata XML>",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

> **Note**
>
> Wenn <SAML metadata XML>das Zeichen vorkommt", müssen Sie \ als Escape-Zeichen Folgendes hinzufügen:\".

  Beispiel mit Metadaten-URL: : `aws cognito-idp update-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-details MetadataURL=https://myidp.example.com/sso/saml/metadata`

```
--attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/
identity/claims/emailaddress
```

- AWS API: UpdateIdentityProvider

Rufen Sie Informationen zu einem bestimmten IDP wie folgt auf

- AWS CLI: `aws cognito-idp describe-identity-provider`

  ```
  aws cognito-idp describe-identity-provider --user-pool-id us-
  east-1_EXAMPLE --provider-name=SAML_provider_1
  ```

- AWS API: DescribeIdentityProvider

Um Informationen über alle aufzulisten IdPs

- AWS CLI: `aws cognito-idp list-identity-providers`

  Beispiel: `aws cognito-idp list-identity-providers --user-pool-id us-east-1_EXAMPLE --max-results 3`

- AWS API: ListIdentityProviders

So löschen Sie einen Identitätsanbieter

- AWS CLI: `aws cognito-idp delete-identity-provider`

  ```
  aws cognito-idp delete-identity-provider --user-pool-id us-
  east-1_EXAMPLE --provider-name=SAML_provider_1
  ```

- AWS API: DeleteIdentityProvider

So richten Sie den SAML-IdP zum Hinzufügen eines Benutzerpools als vertrauende Seite ein

- Der Benutzerpool-Serviceanbieter-URN lautet: `urn:amazon:cognito:sp:us-east-1_EXAMPLE`. Amazon Cognito erfordert einen Wert für die Zielgruppeneinschränkung, der dieser URN in der SAML-Antwort entspricht. Konfigurieren Sie Ihren IdP so, dass er den folgenden POST-Bindungsendpunkt für die IdP-zu-SP-Antwortnachricht verwendet.

  ```
  https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse
  ```

- Ihr SAML-IdP muss alle erforderlichen Attribute für Ihren Benutzerpool in der SAML-Assertion auffüllen`NameID`. `NameID`wird zur eindeutigen Identifizierung Ihres SAML-Verbundbenutzers im Benutzerpool verwendet. Ihr IdP muss die SAML-Namen-ID jedes Benutzers in einem konsistenten Format übergeben, bei dem Groß- und Kleinschreibung beachtet wird. Jede Änderung des Werts der Namen-ID eines Benutzers erstellt ein neues Benutzerprofil.

So stellen Sie ein Signaturzertifikat für Ihren SAML-2.0-IdP bereit

- Um eine Kopie des öffentlichen Schlüssels von Amazon Cognito herunterzuladen, den Ihr IdP zur Validierung von SAML-Abmeldeanfragen verwenden kann, wählen Sie in Ihrem Benutzerpool die Registerkarte Anmeldeerfahrung, wählen Sie Ihren IdP aus und wählen Sie unter Signaturzertifikat anzeigen die Option Als .crt herunterladen aus.

Sie können jeden SAML-Anbieter, den Sie in Ihrem Benutzerpool eingerichtet haben, mit der Amazon-Cognito-Konsole löschen.

So löschen Sie einen SAML-Anbieter

1. Melden Sie sich bei der Amazon Cognito-Konsole an.
2. Wählen Sie im Navigationsbereich User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte Anmeldeerfahrung und suchen Sie nach der Anmeldung mit dem Federated Identity Provider.
4. Wählen Sie das Optionsfeld neben der SAML aus, die IdPs Sie löschen möchten.
5. Wenn Sie aufgefordert werden, den Identitätsanbieter zu löschen, geben Sie den Namen des SAML-Anbieters ein, um das Löschen zu bestätigen. Wählen Sie danach Delete (Löschen) aus.

Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools

Amazon Cognito unterstützt vom Service Provider initiiertes (SP-initiiertes) Single Sign-On (SSO) und IdP-initiiertes SSO. Als bewährte Sicherheitspraxis sollten Sie SP-initiiertes SSO in Ihrem Benutzerpool implementieren. Abschnitt 5.1.2 des SAML V2.0 Technical Overview beschreibt SP-initiiertes SSO. Amazon Cognito ist der Identitätsanbieter (IDP) für Ihre App. Die App ist der Serviceanbieter (SP), der Token für authentifizierte Benutzer abruft. Wenn Sie jedoch einen IdP eines Drittanbieters verwenden, um Benutzer zu authentifizieren, ist Amazon Cognito der SP. Wenn sich

Ihre SAML 2.0-Benutzer mit einem vom SP initiierten Flow authentifizieren, müssen sie immer zuerst eine Anfrage an Amazon Cognito stellen und zur Authentifizierung an den IdP weiterleiten.

In einigen Anwendungsfällen für Unternehmen beginnt der Zugriff auf interne Anwendungen mit einem Lesezeichen in einem Dashboard, das vom Unternehmens-IDP gehostet wird. Wenn ein Benutzer ein Lesezeichen auswählt, generiert der IDP eine SAML-Antwort und sendet sie an den SP, um den Benutzer bei der Anwendung zu authentifizieren.

Sie können einen SAML-IdP in Ihrem Benutzerpool so konfigurieren, dass er IdP-initiiertes SSO unterstützt. Wenn Sie die IDP-initiierte Authentifizierung unterstützen, kann Amazon Cognito nicht überprüfen, ob es die empfangene SAML-Antwort angefordert hat, da Amazon Cognito die Authentifizierung nicht mit einer SAML-Anfrage initiiert. Bei SP-initiiertem SSO legt Amazon Cognito Zustandsparameter fest, die eine SAML-Antwort anhand der ursprünglichen Anfrage validieren. Mit der SP-initiierten Anmeldung können Sie sich auch vor Cross-Site Request Forgery (CSRF) schützen.

Ein Beispiel für die Erstellung von SP-initiiertem SAML in einer Umgebung, in der Sie nicht möchten, dass Ihre Benutzer mit der vom Benutzerpool gehosteten Benutzeroberfläche interagieren, finden Sie unter. Beispielszenario: Amazon Cognito-Apps in einem Unternehmens-Dashboard als Lesezeichen speichern

Themen

Beispielszenario: Amazon Cognito-Apps in einem Unternehmens-Dashboard als Lesezeichen speichern

Sie können in Ihren SAML- oder OIDC-IdP-Dashboards Lesezeichen erstellen, die Amazon Cognito Cognito-Benutzerpools den SSO-Zugriff auf Webanwendungen ermöglichen. Sie können eine Verknüpfung mit Amazon Cognito so erstellen, dass Benutzer sich nicht bei der gehosteten Benutzeroberfläche anmelden müssen. Fügen Sie dazu Ihrem Portal ein Anmeldelesezeichen hinzu, das im folgenden Format zu Ihrem Amazon Cognito Cognito-Benutzerpool weiterleitet. Autorisieren des Endpunkts

```
https://mydomain.us-east-1.amazoncognito.com/authorize?
response_type=code&identity_provider=MySAMLIdP&client_id=1example23456789&redire
www.example.com
```

> **Note**
>
> Sie können anstelle eines `identity_provider`-Parameters auch einen `idp_identifier`-Parameter in Ihrer Anfrage an den Autorisierungsendpunkt verwenden. Eine IdP-ID ist ein alternativer Name oder eine alternative E-Mail-Domain, die Sie konfigurieren können, wenn Sie einen Identitätsanbieter in Ihrem Benutzerpool erstellen. Siehe Namen und Kennungen von SAML-Identitätsanbietern.

Wenn Sie die entsprechenden Parameter in Ihrer Anfrage an `/authorize` verwenden, beginnt Amazon Cognito stillschweigend mit dem SP-initiierten Anmeldeablauf und leitet Ihren Benutzer zur Anmeldung bei Ihrem IDP um.

Fügen Sie zunächst einen SAML-IdP zu Ihrem Benutzerpool hinzu. Erstellen Sie einen App-Client, der Ihren SAML-IDP für die Anmeldung verwendet und die URL für Ihre App als autorisierte Rückruf-URL enthält. Weitere Informationen zu App-Clients finden Sie unter App-Clients für Benutzerpools.

Bevor Sie diesen authentifizierten Zugriff auf Ihr Portal bereitstellen, testen Sie die SP-initiierte Anmeldung bei Ihrer App über Ihre gehostete Benutzeroberfläche. Weitere Informationen zum Konfigurieren eines SAML-IDP in Amazon Cognito finden Sie unter Konfiguration Ihres externen SAML-Identitätsanbieters.

Das folgende Diagramm zeigt einen Authentifizierungsablauf, der IDP-initiiertes SSO emuliert. Ihre Benutzer können sich über einen Link in Ihrem Unternehmensportal bei Amazon Cognito authentifizieren.

Nachdem Sie die Anforderungen erfüllt haben, erstellen Sie ein Lesezeichen für Ihr Konto, Autorisieren des Endpunkts das entweder einen Parameter oder einen Parameter enthält. `identity_provider` `idp_identifier` Die Benutzerauthentifizierung läuft wie folgt ab.

1. Ihr Benutzer meldet sich beim SSO-IDP-Dashboard an. Unternehmensanwendungen, auf die der Benutzer zugreifen darf, werden in diesem Dashboard angezeigt.
2. Ihr Benutzer wählt den Link zu der Anwendung aus, die sich mit Amazon Cognito authentifiziert. In vielen SSO-Portalen können Sie einen benutzerdefinierten App-Link hinzufügen. Jede Funktion, mit der Sie einen Link zu einer öffentlichen URL in Ihrem SSO-Portal erstellen können, funktioniert.
3. Ihr benutzerdefinierter App-Link im SSO-Portal leitet den Benutzer zum Benutzerpool Autorisieren des Endpunkts weiter. Der Link enthält Parameter für `response_type`,

`client_id`, `redirect_uri` und `identity_provider`. Der `identity_provider`-Parameter ist der Name, den Sie dem IDP in Ihrem Benutzerpool gegeben haben. Sie können auch einen `idp_identifier`-Parameter anstelle des `identity_provider`-Parameters verwenden. Ein Benutzer greift über einen Link, der entweder einen `idp_identifier` `identity_provider` Oder-Parameter enthält, auf Ihren Verbundendpunkt zu. Dieser Benutzer umgeht die Anmeldeseite und navigiert direkt zur Authentifizierung bei Ihrem IDP. Weitere Informationen zur Benennung von SAML finden Sie IdPs unter. Namen und Kennungen von SAML-Identitätsanbietern

Beispiel-URL

```
https://mydomain.us-east-1.amazoncognito.com/authorize?
response_type=code&
identity_provider=MySAMLIdP&
client_id=1example23456789&
redirect_uri=https://www.example.com
```

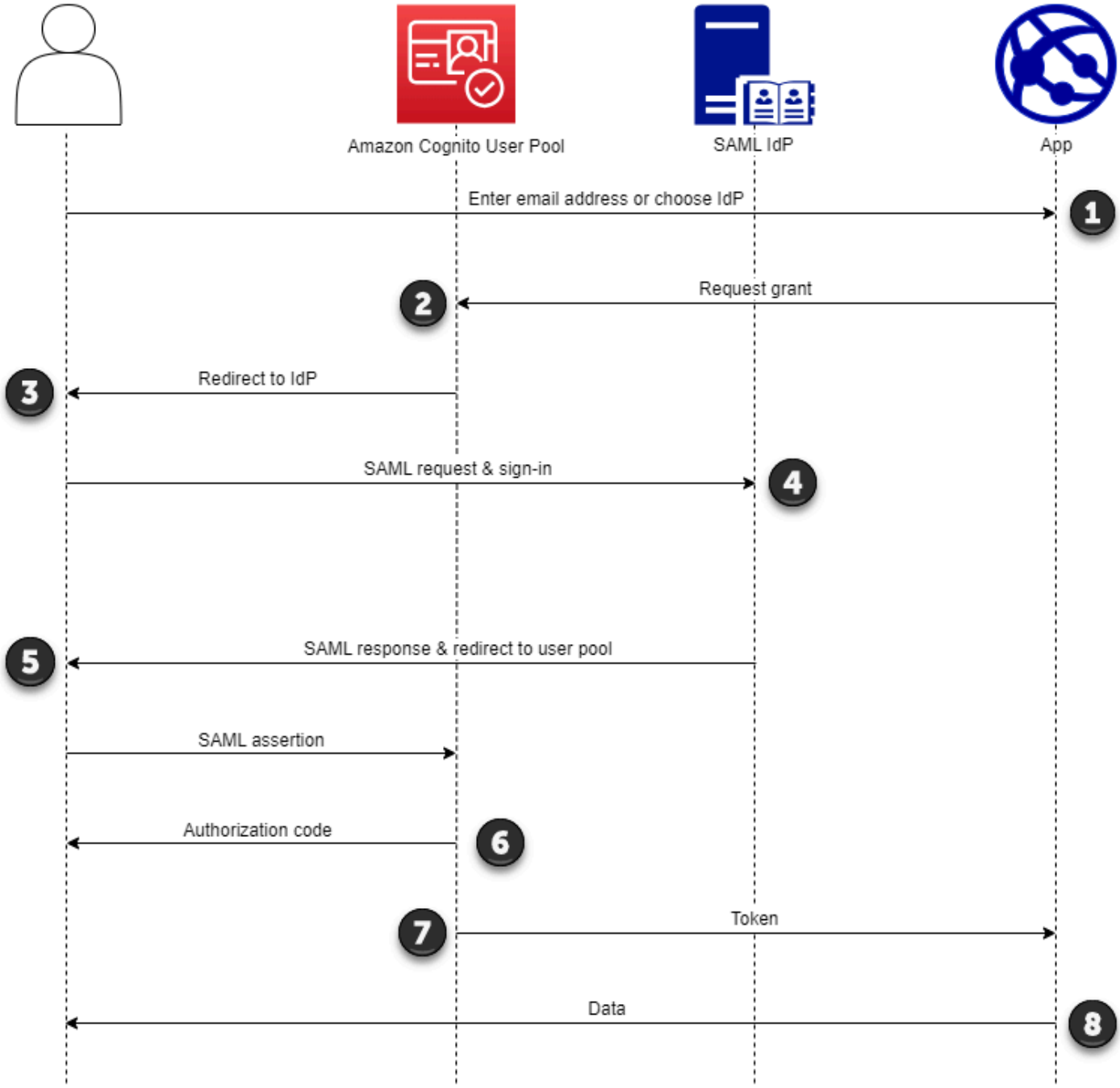
4. Amazon Cognito leitet die Benutzersitzung mit einer SAML-Anfrage an Ihren IDP um.
5. Ihr Benutzer hat möglicherweise ein Sitzungscookie von Ihrem IDP erhalten, als er sich im Dashboard angemeldet hat. Ihr IDP verwendet dieses Cookie, um den Benutzer stillschweigend zu validieren und mit einer SAML-Antwort an den `idpresponse`-Endpunkt von Amazon Cognito umzuleiten. Wenn keine aktive Sitzung vorhanden ist, authentifiziert Ihr IdP den Benutzer erneut, bevor er die SAML-Antwort veröffentlicht.
6. Amazon Cognito validiert die SAML-Antwort und erstellt oder aktualisiert das Benutzerprofil basierend auf der SAML-Assertion.
7. Amazon Cognito leitet den Benutzer mit einem Autorisierungscode zu Ihrer internen App um. Sie haben Ihre interne App-URL als autorisierte Umleitungs-URL für Ihren App-Client konfiguriert.
8. Ihre App tauscht den Autorisierungscode gegen Amazon-Cognito-Token aus. Weitere Informationen finden Sie unter Token-Endpunkt.

Verwenden der SP-initiierten SAML-Anmeldung

Es hat sich bewährt, eine service-provider-initiated (vom SP initiierte) Anmeldung bei Ihrem Benutzerpool zu implementieren. Amazon Cognito initiiert die Sitzung Ihres Benutzers und leitet ihn an Ihren IdP weiter. Mit dieser Methode haben Sie die größte Kontrolle darüber, wer Anmeldeanfragen stellt. Unter bestimmten Bedingungen können Sie auch die vom IDP initiierte

Anmeldung zulassen. Weitere Informationen finden Sie unter [Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools].

Der folgende Prozess zeigt, wie sich Benutzer über einen SAML-Anbieter bei Ihrem Benutzerpool anmelden.



1. Ihr Benutzer gibt seine E-Mail-Adresse auf einer Anmeldeseite ein. Um die Weiterleitung Ihres Benutzers zu seinem IdP zu ermitteln, können Sie seine E-Mail-Adresse in einer benutzerdefinierten App erfassen oder die gehostete Benutzeroberfläche in der Webansicht aufrufen. Sie können Ihre gehostete Benutzeroberfläche so konfigurieren, dass eine Liste von E-Mail-Adressen angezeigt wird IdPs oder dass Sie nur zur Eingabe einer E-Mail-Adresse aufgefordert werden.
2. Ihre App ruft Ihren Benutzerpool-Weiterleitungsendpunkt auf und fordert eine Sitzung mit der Client-ID an, die der App entspricht, und der IdP-ID, die dem Benutzer entspricht.
3. Amazon Cognito leitet Ihren Benutzer mit einer SAML-Anfrage, optional signiert, in einem Element an den IdP weiter. `AuthnRequest`
4. Der IdP authentifiziert den Benutzer interaktiv oder mit einer gespeicherten Sitzung in einem Browser-Cookie.
5. Der IdP leitet Ihren Benutzer mit der optional verschlüsselten SAML-Assertion in seiner POST-Payload an den SAML-Antwortendpunkt Ihres Benutzerpools weiter.

   > **Note**
   >
   > Amazon Cognito storniert Sitzungen, die innerhalb von 5 Minuten keine Antwort erhalten, und leitet den Benutzer zur gehosteten Benutzeroberfläche weiter. Wenn Ihr Benutzer dieses Ergebnis feststellt, erhält er eine `Something went wrong` Fehlermeldung.

6. Nachdem es die SAML-Assertion verifiziert und Benutzerattribute den Ansprüchen in der Antwort zugeordnet hat, erstellt oder aktualisiert Amazon Cognito intern das Benutzerprofil im Benutzerpool. In der Regel gibt Ihr Benutzerpool einen Autorisierungscode an die Browsersitzung Ihres Benutzers zurück.
7. Ihr Benutzer präsentiert seinen Autorisierungscode in Ihrer App, die den Code gegen JSON-Webtoken (JWTs) eintauscht.
8. Ihre App akzeptiert und verarbeitet das ID-Token Ihres Benutzers als Authentifizierung, generiert autorisierte Anfragen an Ressourcen mit seinem Zugriffstoken und speichert dessen Aktualisierungstoken.

Wenn sich ein Benutzer authentifiziert und einen Autorisierungscode erhält, gibt der Benutzerpool ID-, Zugriffs- und Aktualisierungstoken zurück. Das ID-Token ist ein Authentifizierungsobjekt für die OIDC-basierte Identitätsverwaltung. Das Zugriffstoken ist ein Autorisierungsobjekt mit OAuth 2.0-Bereichen. Das Aktualisierungstoken ist ein Objekt, das neue ID- und Zugriffstoken generiert, wenn

die aktuellen Token Ihres Benutzers abgelaufen sind. Sie können die Dauer der Benutzertoken in Ihrem Benutzerpool-App-Client konfigurieren.

Sie können auch die Dauer der Aktualisierungstoken wählen. Nachdem das Aktualisierungstoken eines Benutzers abgelaufen ist, muss er sich erneut anmelden. Wenn sie sich über einen SAML-IdP authentifiziert haben, wird die Sitzungsdauer Ihrer Benutzer durch den Ablauf ihrer Token festgelegt, nicht durch den Ablauf ihrer Sitzung mit ihrem IdP. Ihre App muss das Aktualisierungstoken jedes Benutzers speichern und seine Sitzung erneuern, wenn sie abläuft. Die gehostete Benutzeroberfläche verwaltet Benutzersitzungen in einem Browser-Cookie, das 1 Stunde lang gültig ist.

Verwenden der IDP-initiierten SAML-Anmeldung

Wenn Sie Ihren Identitätsanbieter für die IDP-initiierte SAML 2.0-Anmeldung konfigurieren, können Sie SAML-Assertionen dem `saml2/idpresponse` Endpunkt in Ihrer Benutzerpool-Domäne präsentieren, ohne die Sitzung am starten zu müssen. Autorisieren des Endpunkts Ein Benutzerpool mit dieser Konfiguration akzeptiert IDP-initiierte SAML-Assertionen von einem externen Identitätsanbieter für Benutzerpools, den der angeforderte App-Client unterstützt. In den folgenden Schritten wird der Gesamtprozess zur Konfiguration und Anmeldung bei einem vom IDP initiierten SAML 2.0-Anbieter beschrieben.

1. Erstellen oder bestimmen Sie einen Benutzerpool und einen App-Client.
2. Erstellen Sie einen SAML 2.0-IdP in Ihrem Benutzerpool.
3. Konfigurieren Sie Ihren IdP so, dass er die IdP-Initiierung unterstützt. Von IDP initiiertes SAML führt zu Sicherheitsüberlegungen, denen andere SSO-Anbieter nicht unterliegen. Aus diesem Grund können Sie keinem App-Client IdPs, der einen SAML-Anbieter mit IDP-initiierter Anmeldung verwendet, Nicht-SAML, einschließlich des Benutzerpools selbst, hinzufügen.
4. Ordnen Sie Ihren IDP-initiierten SAML-Anbieter einem App-Client in Ihrem Benutzerpool zu.
5. Leiten Sie Ihren Benutzer zur Anmeldeseite für Ihren SAML-IdP IdP und rufen Sie eine SAML-Assertion ab.
6. Leiten Sie Ihren Benutzer mit seiner SAML-Assertion zu Ihrem `saml2/idpresponse` Benutzerpool-Endpunkt weiter.
7. Empfangen Sie JSON-Webtoken (JWTs).

Um unaufgeforderte SAML-Assertions in Ihrem Benutzerpool zu akzeptieren, müssen Sie die Auswirkungen auf die Sicherheit Ihrer App berücksichtigen. Anforderungs-Spoofing und CSRF-

Versuche sind wahrscheinlich, wenn Sie vom IdP initiierte Anfragen annehmen. Obwohl Ihr Benutzerpool eine vom IdP initiierte Anmeldesitzung nicht verifizieren kann, validiert Amazon Cognito Ihre Anforderungsparameter und SAML-Assertionen.

Darüber hinaus darf Ihre SAML-Assertion keinen `InResponseTo` Anspruch enthalten und muss innerhalb der letzten 6 Minuten ausgestellt worden sein.

Sie müssen Anfragen mit IDP-initiierter SAML an Ihren senden. `/saml2/idpresponse` Für SP-initiierte und gehostete UI-Autorisierungsanfragen müssen Sie Parameter angeben, die Ihren angeforderten App-Client, Bereiche, Umleitungs-URI und andere Details als Abfragezeichenfolgenparameter in Anfragen identifizieren. `HTTP GET` Bei IDP-initiierten SAML-Assertionen müssen die Details Ihrer Anfrage jedoch als `RelayState` Parameter im Hauptteil einer Anfrage formatiert werden. `HTTP POST` Der Anfragetext muss auch Ihre SAML-Assertion als Parameter enthalten. `SAMLResponse`

Im Folgenden finden Sie eine Beispielanforderung für einen IDP-initiierten SAML-Anbieter.

```
POST /saml2/idpresponse HTTP/1.1
User-Agent: USER_AGENT
Accept: */*
Host: example.auth.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded

SAMLResponse=[Base64-encoded SAML assertion]&RelayState=identity_provider
%3DMySAMLIdP%26client_id%3D1example23456789%26redirect_uri%3Dhttps%3A%2F
%2Fwww.example.com%26response_type%3Dcode%26scope%3Demail%2Bopenid%2Bphone

HTTP/1.1 302 Found
Date: Wed, 06 Dec 2023 00:15:29 GMT
Content-Length: 0
x-amz-cognito-request-id: 8aba6eb5-fb54-4bc6-9368-c3878434f0fb
Location: https://www.example.com?code=[Authorization code]
```

AWS Management Console

So konfigurieren Sie einen IdP für IdP-initiiertes SAML

1. Erstellen Sie einen Benutzerpool, einen App-Client und einen SAML-Identitätsanbieter.
2. Trennen Sie alle Social Media- und OIDC-Identitätsanbieter von Ihrem App-Client, falls welche verknüpft sind.
3. Navigieren Sie in Ihrem Benutzerpool zur Registerkarte „Anmeldeerfahrung“.

4. Bearbeiten Sie unter Anmeldung beim Federated Identity Provider einen SAML-Anbieter oder fügen Sie ihn hinzu.
5. Wählen Sie unter IDP-initiierte SAML-Anmeldung die Option SP-initiierte und IDP-initiierte SAML-Assertionen akzeptieren aus.
6. Wählen Sie Änderungen speichern aus.

API/CLI

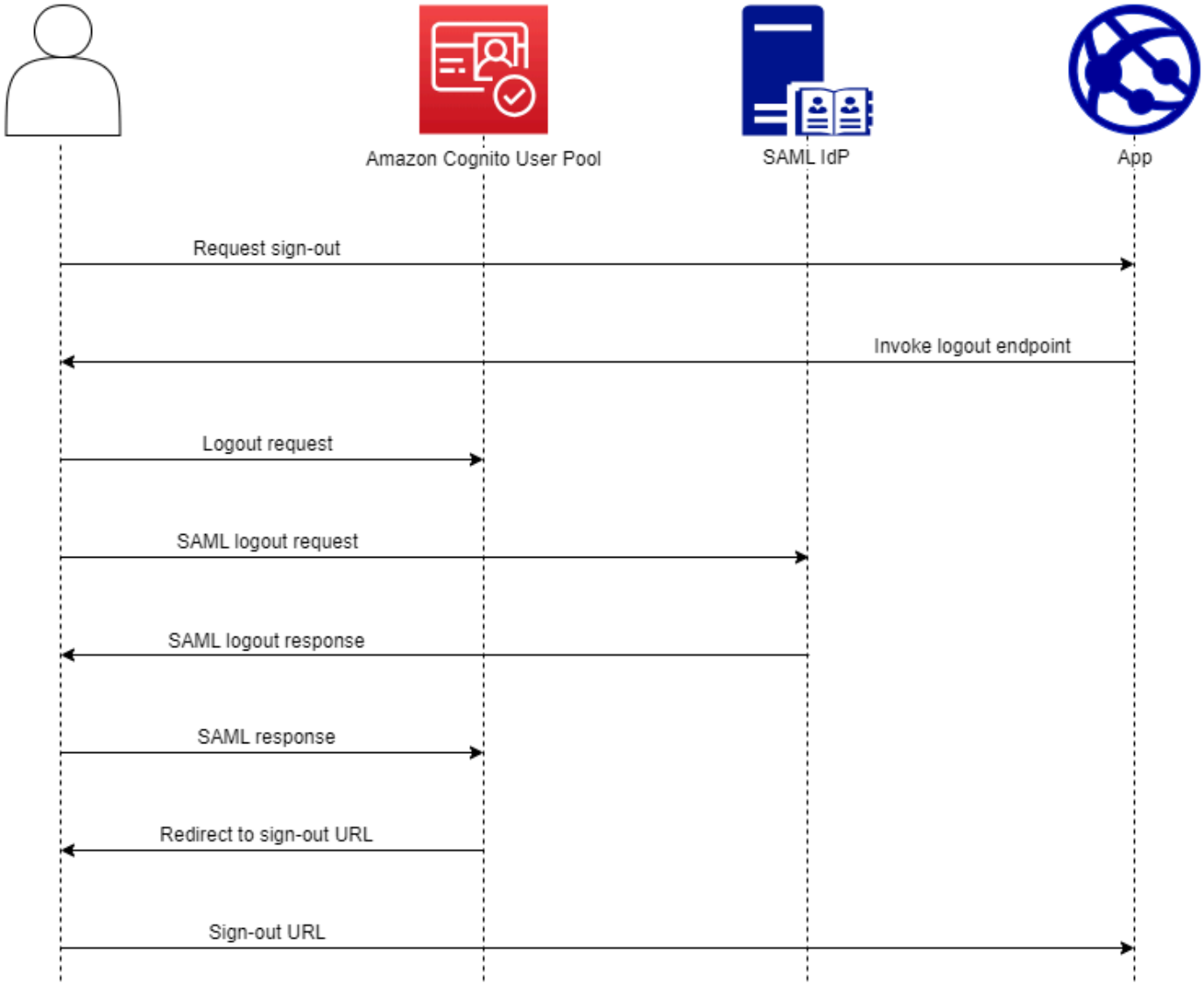
So konfigurieren Sie einen IdP für IdP-initiiertes SAML

Konfigurieren Sie IDP-initiiertes SAML mit dem `IDPInit` Parameter in einer CreateIdentityProvideroder UpdateIdentityProviderAPI-Anfrage. Im Folgenden finden Sie ein Beispiel für einen IdP, `ProviderDetails` der IdP-initiiertes SAML unterstützt.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

Ablauf der SAML-Abmeldung

Amazon Cognito unterstützt SAML 2.0 Single Logout. Wenn Sie Ihren SAML-IdP so konfigurieren, dass er den Abmeldefluss unterstützt, leitet Amazon Cognito Ihren Benutzer mit einer signierten SAML-Abmeldeanfrage an Ihren IdP weiter. Amazon Cognito bestimmt den Ort der Weiterleitung anhand der `SingleLogoutService` URL in Ihren IdP-Metadaten. Amazon Cognito signiert die Abmeldeanforderung mit Ihrem Benutzerpool-Signaturzertifikat.



Wenn Sie einen Benutzer mit einer SAML-Sitzung an Ihren `/logout` Benutzerpool-Endpunkt weiterleiten, leitet Amazon Cognito Ihren SAML-Benutzer mit der folgenden Anfrage an den SLO-Endpunkt weiter, der in den IdP-Metadaten angegeben ist.

```
https://[SingleLogoutService endpoint]?
SAMLRequest=[encoded SAML request]&
RelayState=[RelayState]&
SigAlg=http://www.w3.org/2001/04/xmldsig-more#rsa-sha256&
Signature=[User pool RSA signature]
```

Ihr Benutzer kehrt dann mit einem `LogoutResponse` von seinem IdP zu Ihrem `saml2/logout` Endpunkt zurück. Ihr IdP muss eine `HTTP POST` Anfrage einreichen. `LogoutResponse` Amazon Cognito leitet sie dann von ihrer ursprünglichen Abmeldeanfrage an das Weiterleitungsziel weiter.

Ihr SAML-Anbieter sendet möglicherweise eine `LogoutResponse` mit mehr als einer`AuthnStatement`. Das `sessionIndex` in der ersten Antwort dieses Typs muss mit dem `AuthnStatement` `sessionIndex` in der SAML-Antwort, mit der der Benutzer ursprünglich authentifiziert wurde, übereinstimmen. Wenn `sessionIndex` sich das in einem anderen befindet`AuthnStatement`, erkennt Amazon Cognito die Sitzung nicht und Ihr Benutzer wird nicht abgemeldet.

AWS Management Console

Um die SAML-Abmeldung zu konfigurieren

1. Erstellen Sie einen Benutzerpool, einen App-Client und einen SAML-IdP.
2. Wenn Sie Ihren SAML-Identitätsanbieter erstellen oder bearbeiten, aktivieren Sie unter Informationen zum Identitätsanbieter das Kästchen mit dem Titel Abmeldefluss hinzufügen.
3. Wählen Sie auf der Registerkarte Anmeldeerfahrung Ihres Benutzerpools unter Federated Identity Provider-Anmeldung Ihren IdP aus und suchen Sie das Signaturzertifikat.
4. Wählen Sie Als .crt herunterladen aus.
5. Konfigurieren Sie Ihren SAML-Anbieter so, dass er SAML Single Logout und Request Signing unterstützt, und laden Sie das Signaturzertifikat für den Benutzerpool hoch. Ihr IdP muss zu `/saml2/logout` Ihrer Benutzerpool-Domain weiterleiten.

API/CLI

Um die SAML-Abmeldung zu konfigurieren

Konfigurieren Sie Single Logout mit dem `IDPSignout` Parameter einer CreateIdentityProvideroder UpdateIdentityProviderAPI-Anfrage. Im Folgenden finden Sie ein Beispiel für einen IdP, `ProviderDetails` der SAML Single Logout unterstützt.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",,
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
```

```
      "IDPInit" : "true"
}
```

SAML-Signatur und Verschlüsselung

Amazon Cognito unterstützt signierte SAML-Anfragen und verschlüsselte SAML-Antworten für die An- und Abmeldung. Alle kryptografischen Operationen während SAML-Vorgängen im Benutzerpool müssen Signaturen und Chiffretext mit user-pool-provided Schlüsseln generieren, die Amazon Cognito generiert. Derzeit können Sie einen Benutzerpool nicht so konfigurieren, dass er Anfragen signiert oder verschlüsselte Assertionen mit einem externen Schlüssel akzeptiert.

> **Note**
>
> Ihre Benutzerpoolzertifikate sind 10 Jahre gültig. Einmal pro Jahr generiert Amazon Cognito neue Signatur- und Verschlüsselungszertifikate für Ihren Benutzerpool. Amazon Cognito gibt das neueste Zertifikat zurück, wenn Sie das Signaturzertifikat anfordern, und signiert Anfragen mit dem neuesten Signaturzertifikat. Ihr IdP kann SAML-Assertionen mit jedem Benutzerpool-Verschlüsselungszertifikat verschlüsseln, das nicht abgelaufen ist. Ihre vorherigen Zertifikate sind weiterhin für ihre gesamte Laufzeit gültig. Es hat sich bewährt, das Zertifikat in Ihrer Anbieterkonfiguration jährlich zu aktualisieren.

Themen

- Verschlüsselte SAML-Antworten von Ihrem IdP akzeptieren
- SAML-Anfragen signieren

Verschlüsselte SAML-Antworten von Ihrem IdP akzeptieren

Amazon Cognito und Ihr IdP können SAML-Antworten vertraulich behandeln, wenn sich Benutzer an- und abmelden. Amazon Cognito weist jedem externen SAML-Anbieter, den Sie in Ihrem Benutzerpool konfigurieren, ein öffentlich-privates RSA-Schlüsselpaar und ein Zertifikat zu. Wenn Sie die Antwortverschlüsselung für Ihren SAML-Anbieter für Ihren Benutzerpool aktivieren, müssen Sie Ihr Zertifikat auf einen IdP hochladen, der verschlüsselte SAML-Antworten unterstützt. Ihre Benutzerpool-Verbindung zu Ihrem SAML-IdP funktioniert nicht, bevor Ihr IdP beginnt, alle SAML-Assertionen mit dem bereitgestellten Schlüssel zu verschlüsseln.

Im Folgenden finden Sie einen Überblick über den Ablauf einer verschlüsselten SAML-Anmeldung.

1. Ihr Benutzer beginnt mit der Anmeldung und wählt seinen SAML-IdP aus.
2. Ihr Benutzerpool Autorisieren des Endpunkts leitet Ihren Benutzer mit einer SAML-Anmeldeanfrage zu seinem SAML-IdP weiter. Ihr Benutzerpool kann dieser Anfrage optional eine Signatur beifügen, die eine Integritätsprüfung durch den IdP ermöglicht. Wenn Sie SAML-Anfragen signieren möchten, müssen Sie Ihren IdP so konfigurieren, dass er Anfragen akzeptiert, die Ihr Benutzerpool mit dem öffentlichen Schlüssel im Signaturzertifikat signiert hat.
3. Der SAML-IdP meldet Ihren Benutzer an und generiert eine SAML-Antwort. Der IdP verschlüsselt die Antwort mit dem öffentlichen Schlüssel und leitet Ihren Benutzer zu Ihrem `/saml2/idpresponse` Benutzerpool-Endpunkt weiter. Der IdP muss die Antwort gemäß der SAML 2.0-Spezifikation verschlüsseln. Weitere Informationen finden Sie unter `Element <EncryptedAssertion>` Assertionen und Protokolle für die OASIS Security Assertion Markup Language (SAML) V2.0.
4. Ihr Benutzerpool entschlüsselt den Chiffretext in der SAML-Antwort mit dem privaten Schlüssel und meldet Ihren Benutzer an.

> **Important**
>
> Wenn Sie die Antwortverschlüsselung für einen SAML-IdP in Ihrem Benutzerpool aktivieren, muss Ihr IdP alle Antworten mit einem öffentlichen Schlüssel verschlüsseln, der für den Anbieter spezifisch ist. Amazon Cognito akzeptiert keine unverschlüsselten SAML-Antworten von einem externen SAML-IdP, den Sie für die Unterstützung von Verschlüsselung konfigurieren.

Jeder externe SAML-IdP in Ihrem Benutzerpool kann die Antwortverschlüsselung unterstützen, und jeder IdP erhält sein eigenes key pair.

AWS Management Console

Um die SAML-Antwortverschlüsselung zu konfigurieren

1. Erstellen Sie einen Benutzerpool, einen App-Client und einen SAML-IdP.
2. Wenn Sie Ihren SAML-Identitätsanbieter erstellen oder bearbeiten, aktivieren Sie unter Anfragen signieren und Antworten verschlüsseln das Kästchen mit dem Titel Verschlüsselte SAML-Assertionen von diesem Anbieter anfordern.

3. Wählen Sie auf der Registerkarte Anmeldeerfahrung Ihres Benutzerpools unter Federated Identity Provider-Anmeldung Ihren SAML-IdP aus und wählen Sie Verschlüsselungszertifikat anzeigen aus.
4. Wählen Sie Als .crt herunterladen und stellen Sie die heruntergeladene Datei Ihrem SAML-IdP zur Verfügung. Konfigurieren Sie Ihren SAML-IdP so, dass SAML-Antworten mit dem Schlüssel im Zertifikat verschlüsselt werden.

API/CLI

Um die SAML-Antwortverschlüsselung zu konfigurieren

Konfigurieren Sie die Antwortverschlüsselung mit dem `EncryptedResponses` Parameter einer CreateIdentityProvideroder UpdateIdentityProviderAPI-Anfrage. Im Folgenden finden Sie ein Beispiel für einen IdP, `ProviderDetails` der das Signieren von Anfragen unterstützt.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

SAML-Anfragen signieren

Die Möglichkeit, die Integrität von SAML 2.0-Anfragen an Ihren IdP nachzuweisen, ist ein Sicherheitsvorteil der von Amazon Cognito SP initiierten SAML-Anmeldung. Jeder Benutzerpool mit einer Domain erhält ein X.509-Signaturzertifikat für den Benutzerpool. Mit dem öffentlichen Schlüssel in diesem Zertifikat wenden Benutzerpools eine kryptografische Signatur auf die Abmeldeanfragen an, die Ihr Benutzerpool generiert, wenn Ihre Benutzer einen SAML-IdP auswählen. Sie können Ihren App-Client optional so konfigurieren, dass er SAML-Anmeldeanfragen signiert. Wenn Sie Ihre SAML-Anfragen signieren, kann Ihr IdP überprüfen, ob die Signatur in den XML-Metadaten Ihrer Anfragen mit dem öffentlichen Schlüssel in dem von Ihnen bereitgestellten Benutzerpoolzertifikat übereinstimmt.

AWS Management Console

Um das Signieren von SAML-Anfragen zu konfigurieren

1. Erstellen Sie einen Benutzerpool, einen App-Client und einen SAML-IdP.
2. Wenn Sie Ihren SAML-Identitätsanbieter erstellen oder bearbeiten, aktivieren Sie unter Anfragen signieren und Antworten verschlüsseln das Kästchen mit dem Titel SAML-Anfragen an diesen Anbieter signieren.
3. Wählen Sie auf der Registerkarte Anmeldeerfahrung in Ihrem Benutzerpool unter Anmeldung mit dem Federated Identity Provider die Option Signaturzertifikat anzeigen aus.
4. Wählen Sie Als .crt herunterladen und stellen Sie die heruntergeladene Datei Ihrem SAML-IdP zur Verfügung. Konfigurieren Sie Ihren SAML-IdP, um die Signatur eingehender SAML-Anfragen zu überprüfen.

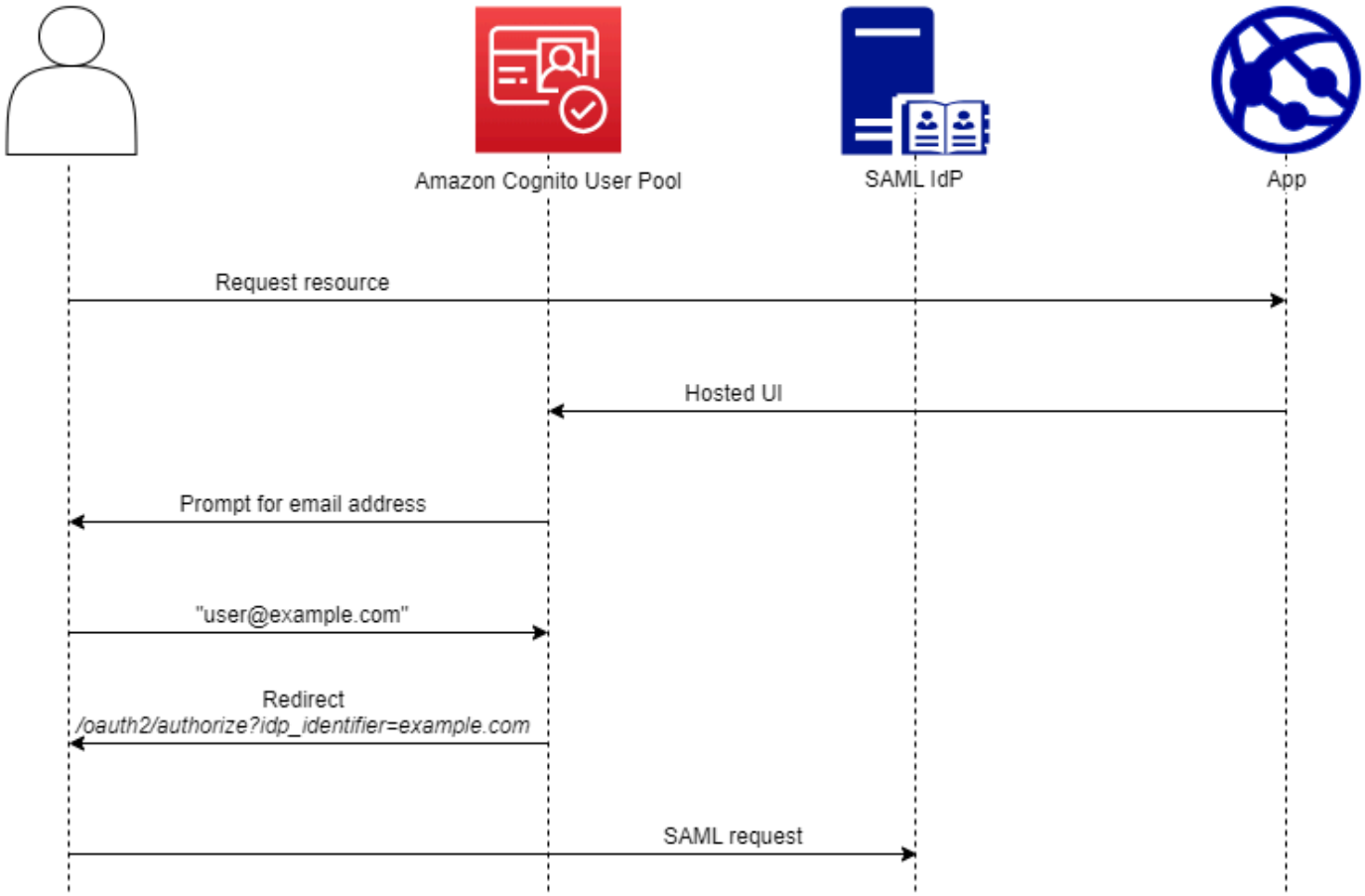
API/CLI

So konfigurieren Sie das Signieren von SAML-Anfragen

Konfigurieren Sie die Anforderungssignatur mit dem `RequestSigningAlgorithm` Parameter einer CreateIdentityProvideroder UpdateIdentityProviderAPI-Anfrage. Im Folgenden finden Sie ein Beispiel für einen IdP, `ProviderDetails` der das Signieren von Anfragen unterstützt.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

Namen und Kennungen von SAML-Identitätsanbietern



Wenn Sie Ihre SAML-Identitätsanbieter (IdPs) benennen und IdP-Identifikatoren zuweisen, können Sie den Fluss von SP-initiierten Anmelde- und Abmeldeanfragen an diesen Anbieter automatisieren. Informationen zu Zeichenketteneinschränkungen für den Anbieternamen finden Sie in der Eigenschaft von. `ProviderName` CreateIdentityProvider

Sie können auch bis zu 50 Identifikatoren für Ihre SAML-Anbieter auswählen. Ein Identifier ist ein benutzerfreundlicher Name für einen IdP in Ihrem Benutzerpool und muss innerhalb des Benutzerpools eindeutig sein. Wenn Ihre SAML-Identifikatoren mit den E-Mail-Domains Ihrer Benutzer übereinstimmen, fordert die von Amazon Cognito gehostete Benutzeroberfläche die E-Mail-Adresse jedes Benutzers an, bewertet die Domain in seiner E-Mail-Adresse und leitet ihn an den IdP weiter, der seiner Domain entspricht. Da dieselbe Organisation mehrere Domains besitzen kann, kann ein einzelner IdP mehrere Identifikatoren haben.

Unabhängig davon, ob Sie E-Mail-Domain-IDs verwenden oder nicht, können Sie Identifikatoren in einer Mehrmandanten-App verwenden, um Benutzer zum richtigen IdP weiterzuleiten. Wenn

Sie die gehostete Benutzeroberfläche vollständig umgehen möchten, können Sie die Links, die Sie Benutzern präsentieren, so anpassen, dass sie Autorisieren des Endpunkts direkt zu ihrem IdP weiterleiten. Um Ihre Benutzer mit einer Kennung anzumelden und zu ihrem IdP weiterzuleiten, fügen Sie die Kennung in das Format `idp_identifier=myidp.example.com` in die Anforderungsparameter ihrer ersten Autorisierungsanfrage ein.

Eine andere Methode, um einen Benutzer an Ihren IdP weiterzuleiten, besteht darin, den Parameter `identity_provider` mit dem Namen Ihres IdP im folgenden URL-Format zu füllen.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
identity_provider=MySAMLIdP&
client_id=1example23456789&
redirect_uri=https://www.example.com
```

Nachdem sich ein Benutzer mit Ihrem SAML-IdP angemeldet hat, leitet Ihr IdP ihn mit einer SAML-Antwort im `HTTP POST` Text an Ihren Endpunkt weiter. `/saml2/idpresponse` Amazon Cognito verarbeitet die SAML-Assertion und leitet, wenn die Ansprüche in der Antwort den Erwartungen entsprechen, zu Ihrer App-Client-Callback-URL weiter. Nachdem Ihr Benutzer die Authentifizierung auf diese Weise abgeschlossen hat, hat er nur mit Webseiten für Ihren IdP und Ihre App interagiert.

Mit IdP-Identifikatoren in einem Domain-Format fordert die von Amazon Cognito gehostete Benutzeroberfläche bei der Anmeldung E-Mail-Adressen an und leitet die Benutzer dann, wenn die E-Mail-Domain mit einer IdP-ID übereinstimmt, auf die Anmeldeseite für ihren IdP weiter. Sie erstellen beispielsweise eine App, für die sich Mitarbeiter zweier verschiedener Unternehmen anmelden müssen. Das erste Unternehmen, AnyCompany A, besitzt `exampleA.com` und`exampleA.co.uk`. Das zweite Unternehmen, AnyCompany B, besitzt`exampleB.com`. Für dieses Beispiel haben Sie zwei eingerichtet IdPs, eine für jedes Unternehmen, wie folgt:

- Für IdP A definieren Sie die IDs `exampleA.com` und `exampleA.co.uk`.
- Für IdP-B definieren Sie die ID `exampleB.com`.

Rufen Sie in Ihrer App die gehostete Benutzeroberfläche für Ihren App-Client auf, um jeden Benutzer zur Eingabe seiner E-Mail-Adresse aufzufordern. Amazon Cognito leitet die Domain von der E-Mail-Adresse ab, korreliert die Domain mit einem IdP mit einer Domain-ID und leitet Ihren Benutzer mit einer Anfrage an den, die einen Anforderungsparameter enthältAutorisieren des Endpunkts, an den richtigen IdP weiter. `idp_identifier` Wenn ein Benutzer beispielsweise eintritt, ist die

nächste Seitebob@exampleA.co.uk, mit der er interagiert, die IdP-Anmeldeseite unter. `https://auth.exampleA.co.uk/sso/saml`

Sie können dieselbe Logik auch unabhängig voneinander implementieren. In Ihrer App können Sie ein benutzerdefiniertes Formular erstellen, das Benutzereingaben sammelt und sie gemäß Ihrer eigenen Logik mit dem richtigen IdP korreliert. Sie können benutzerdefinierte App-Portale für jeden Ihrer App-Mandanten generieren, wobei jedes Portal mit der ID des Mandanten in den Anforderungsparametern auf den Autorisierungsendpunkt verweist.

Um eine E-Mail-Adresse zu erfassen und die Domain in der gehosteten Benutzeroberfläche zu analysieren, weisen Sie jedem SAML-IdP, den Sie Ihrem App-Client zugewiesen haben, mindestens eine Kennung zu. Standardmäßig wird auf dem Anmeldebildschirm für die gehostete Benutzeroberfläche eine Schaltfläche für jedes Element angezeigt IdPs , das Sie Ihrem App-Client zugewiesen haben. Wenn Sie jedoch erfolgreich Kennungen zugewiesen haben, sieht Ihre gehostete UI-Anmeldeseite wie in der folgenden Abbildung aus.

Das Domain-Parsing in der gehosteten Benutzeroberfläche erfordert, dass Sie Domains als Ihre IdP-Identifikatoren verwenden. Wenn Sie jeder SAML IdPs für einen App-Client einen Bezeichner eines beliebigen Typs zuweisen, zeigt die gehostete Benutzeroberfläche für diese App keine IDP-Auswahlschaltflächen mehr an. Fügen Sie IdP-Identifikatoren für SAML hinzu, wenn Sie E-Mail-Parsing oder benutzerdefinierte Logik zum Generieren von Weiterleitungen verwenden möchten. Wenn Sie unbeaufsichtigte Weiterleitungen generieren möchten und auch möchten, dass auf Ihrer gehosteten Benutzeroberfläche eine Liste davon angezeigt wird IdPs, weisen Sie keine Kennungen zu und verwenden Sie den `identity_provider` Anforderungsparameter in Ihren Autorisierungsanfragen.

- Wenn Sie Ihrem App-Client nur einen SAML-IDP zuweisen, wird auf der Anmeldeseite der gehosteten Benutzeroberfläche eine Schaltfläche angezeigt, um sich bei diesem IDP anzumelden.
- Wenn Sie jedem SAML-IdP, den Sie für Ihren App-Client aktivieren, eine Kennung zuweisen, wird auf der gehosteten UI-Anmeldeseite eine Benutzereingabeaufforderung für eine E-Mail-Adresse angezeigt.
- Wenn Sie mehrere haben IdPs und nicht allen eine ID zuweisen, wird auf der gehosteten UI-Anmeldeseite eine Schaltfläche angezeigt, mit der Sie sich bei jedem zugewiesenen IdP anmelden können.
- Wenn Sie Ihrem Identifier zugewiesen haben IdPs und Sie möchten, dass Ihre gehostete Benutzeroberfläche eine Auswahl von IdP-Schaltflächen anzeigt, fügen Sie Ihrem App-Client einen neuen IdP hinzu, der keine ID hat, oder erstellen Sie einen neuen App-Client. Sie können

auch einen vorhandenen IdP löschen und ihn ohne ID erneut hinzufügen. Wenn Sie einen neuen IdP erstellen, erstellen Ihre SAML-Benutzer neue Benutzerprofile. Diese doppelte Anzahl aktiver Benutzer kann sich in dem Monat, in dem Sie Ihre IdP-Konfiguration ändern, auf die Abrechnung auswirken.

Weitere Informationen zur IdP-Einrichtung finden Sie unter Konfigurieren von Identitätsanbietern für Ihren Benutzerpool..

Konfiguration Ihres externen SAML-Identitätsanbieters

Um SAML 2.0-Identitätsanbieter (IdP) -Lösungen von Drittanbietern so zu konfigurieren, dass sie mit dem Verbund für Amazon Cognito Cognito-Benutzerpools funktionieren, müssen Sie Ihren SAML-IdP so konfigurieren, dass er zur folgenden Assertion Consumer Service (ACS) -URL umleitet:. `https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse` Wenn Ihr Benutzerpool über eine Amazon-Cognito-Domain verfügt, finden Sie den Domainpfad für Ihren Benutzerpool auf der Registerkarte App integration (App-Integration) in der Amazon-Cognito-Konsole.

Bei einigen SAMLs IdPs müssen Sie die auch als `urn` Zielgruppen-URI oder SP-Entitäts-ID bezeichnete URL im Formular angeben. `urn:amazon:cognito:sp:us-east-1_EXAMPLE` Sie finden Ihre Benutzerpool-ID unter Benutzerpool-Übersicht in der Amazon Cognito Cognito-Konsole.

Sie müssen Ihren SAML-IdP auch so konfigurieren, dass er Werte für alle Attribute bereitstellt, die Sie in Ihrem Benutzerpool als erforderliche Attribute festgelegt haben. In der Regel `email` ist dies ein erforderliches Attribut für Benutzerpools. In diesem Fall muss der SAML-IdP in seiner SAML-Assertion eine Form von `email` Anspruch angeben, und Sie müssen den Anspruch dem Attribut für diesen Anbieter zuordnen.

Die folgenden Konfigurationsinformationen für SAML 2.0-IdP-Lösungen von Drittanbietern sind ein guter Ausgangspunkt, um mit der Einrichtung eines Verbunds mit Amazon Cognito Cognito-Benutzerpools zu beginnen. Die aktuellsten Informationen finden Sie direkt in der Dokumentation Ihres Anbieters.

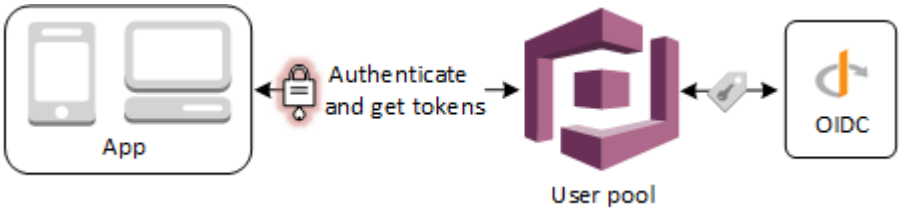
Um SAML-Anfragen zu signieren, müssen Sie Ihren IdP so konfigurieren, dass er Anfragen vertraut, die mit Ihrem Benutzerpool-Signaturzertifikat signiert wurden. Um verschlüsselte SAML-Antworten zu akzeptieren, müssen Sie Ihren IdP so konfigurieren, dass er alle SAML-Antworten an Ihren Benutzerpool verschlüsselt. Ihr Anbieter wird über eine Dokumentation zur Konfiguration dieser Funktionen verfügen. Ein Beispiel von Microsoft finden Sie unter Microsoft Entra SAML-Tokenverschlüsselung konfigurieren.

> **Note**
>
> Amazon Cognito benötigt nur das Metadatendokument Ihres Identitätsanbieters. Ihr Anbieter bietet möglicherweise Konfigurationsinformationen für den AWS-Konto Verbund mit SAML 2.0 an. Diese Informationen sind für die Amazon Cognito Cognito-Integration nicht relevant.

Lösung	Weitere Informationen
Microsoft Active Directory Federation Services (AD FS)	Föderations-Metadaten-Explorer
Okta	So laden Sie die IdP-Metadaten und SAML-Signaturzertifikate für eine SAML-App-Integration herunter
Auth0	Konfigurieren Sie Auth0 als SAML-Identitätsanbieter
Ping-Identität () PingFederate	Exportieren von SAML-Metadaten von PingFederate
JumpCloud	Hinweise zur SAML-Konfiguration
SecureAuth	Integration von SAML-Anwendungen

Verwendung von OIDC-Identitätsanbietern mit einem Benutzerpool

Sie können Ihren Benutzern, die bereits Konten bei OpenID Connect (OIDC) -Identitätsanbietern (IdPs) haben, ermöglichen, den Anmeldeschritt zu überspringen und sich mit einem vorhandenen Konto bei Ihrer Anwendung anzumelden. Mit der integrierten gehosteten Web-UI bietet Amazon Cognito Token-Handling und Verwaltung für authentifizierte Benutzer aller Identitätsanbieter. Auf diese Weise können Ihre Backend-Systeme auf einen Satz von Benutzerpool-Token standardisiert werden.



> **Note**
>
> Die Anmeldung über einen Drittanbieter (Verbund) in Amazon-Cognito-Benutzerpools wird unterstützt. Diese Funktion ist unabhängig von Verbund über Amazon-Cognito-Identitätspools (Verbundidentitäten).

Sie können Ihrem Benutzerpool in der, über die oder mit der AWS Management Console Benutzerpool-API-Methode einen OIDC-IdP hinzufügen. AWS CLICreateIdentityProvider

Themen

- Voraussetzungen
- Schritt 1: Registrieren mit einem OIDC-IdP
- Schritt 2: Hinzufügen eines OIDC-IdP zu Ihrem Benutzerpool
- Schritt 3: Testen Ihrer OIDC-IdP-Konfiguration
- Authentifizierungsablauf für den OIDC-Benutzerpool-IdP

Voraussetzungen

Bevor Sie beginnen, muss Folgendes sichergestellt sein:

- Ein Benutzerpool mit einem App-Client und eine Benutzerpool-Domäne. Weitere Informationen finden Sie unter Einen Benutzerpool erstellen.
- Ein OIDC IdP mit folgender Konfiguration:
  - Unterstützt die `client_secret_post` Clientauthentifizierung. Amazon Cognito überprüft nicht den `token_endpoint_auth_methods_supported`-Antrag am OIDC-Erkennungsendpunkt für Ihren IdP. Amazon Cognito unterstützt nicht die `client_secret_basic` Clientauthentifizierung. Weitere Informationen zur Clientauthentifizierung finden Sie unter Clientauthentifizierung in der Dokumentation zu OpenID Connect.

- Verwendet HTTPS nur für OIDC-Endpunkte wie `openid_configuration`, `userInfo` und `jwks_uri`.
- Verwendet nur die TCP-Ports 80 und 443 für OIDC-Endpunkte.
- Signiert ID-Token nur mit HMAC-SHA-, ECDSA- oder RSA-Algorithmen.
- Veröffentlicht einen Schlüssel-ID`kid`-Anspruch am `jwks_uri` und enthält einen `kid`-Anspruch in seinen Token.

Schritt 1: Registrieren mit einem OIDC-IdP

Bevor Sie einen OIDC-IdP mit Amazon Cognito anlegen, müssen Sie Ihre Anwendung bei dem OIDC-IdP registrieren, um eine Kunden-ID und einen geheimen Client-Schlüssel zu erhalten.

Registrieren mit einem OIDC-IdP

1. Erstellen Sie ein Entwickler-Konto bei dem OIDC-IdP.

   Links zu OIDC IdPs

OIDC-IdP	Installieren	URL für die OIDC-Erkennung
Salesforce	Installieren eines Salesforce-Identitätsanbieters	https://login.salesforce.com
Ping Identity	Installieren eines Ping Identity-Identitätsanbieters	https://*Ihre Ping-Domänenadresse* :9031/idp/userinfo.openid Beispiel: `https://pf.company.com:9031/idp/userinfo.openid`
Okta	Installieren eines Okta-Identitätsanbieters	https://*Ihre Okta-Subdomäne* .oktapreview.com oder `https://`*Your Okta subdomain* `.okta.com`

OIDC-IdP	Installieren	URL für die OIDC-Erkennung
Microsoft Azure Active Directory (Azure AD)	Installieren eines Microsoft Azure AD-Identitätsanbieters	`https://login.microsoftonline.com/ {tenant}/v2.0`
Google	Installieren eines Google-Identitätsanbieters	https://accounts.google.com **Note** Amazon Cognito bietet Google als integrierten sozialen Anmelde-IdP. Wir empfehlen, den integrierten IdP zu verwenden. Siehe Verwenden von Anbietern für soziale Identitäten mit einem Benutzerpool.

2. Registrieren Sie Ihre Benutzerpool-Domänen-URL mit dem Endpunkt `/oauth2/idpresponse` bei Ihrem OIDC-Identitätsanbieter. Auf diese Weise wird sichergestellt, dass der OIDC-Identitätsanbieter sie von Amazon Cognito später akzeptiert, wenn es Benutzer authentifiziert.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

3. Registrieren Sie Ihre Rückruf-URL in Ihrem Amazon-Cognito-Benutzerpool. Dies ist die URL der Seite, auf die Amazon Cognito Ihre Benutzer nach einer erfolgreichen Authentifizierung umleitet.

```
https://www.example.com
```

4. Wählen Sie Ihre Bereiche aus. Der Bereich openid ist erforderlich. Der Bereich email ist erforderlich, um Zugriff auf die Ansprüche email und email_verified zu erteilen.
5. Der OIDC-Identitätsanbieter stellt Ihnen eine Client-ID und einen geheimen Client-Schlüssel bereit. Sie verwenden diese, wenn Sie einen OIDC-Identitätsanbieter in Ihrem Benutzerpool einrichten.

Beispiel: Verwendung von Salesforce als OIDC-Identitätsanbieter für Ihren Benutzerpool

Sie verwenden einen OIDC-IdP, wenn Sie eine Vertrauenstellung zwischen einem OIDC-kompatiblen IdP wie Salesforce und Ihrem Benutzerpool herstellen möchten.

1. Erstellen Sie ein Konto auf der Salesforce-Entwickler-Website.
2. Melden Sie sich über Ihr Entwickler-Konto an, das Sie im vorherigen Schritt eingerichtet haben.
3. Führen Sie auf Ihrer Salesforce-Seite einen der folgenden Schritte aus:
   - Wenn Sie Lightning Experience verwenden, wählen Sie das Zahnradsymbol für die Einrichtung und dann Setup Home (Einrichtung Startseite) aus.
   - Wenn Sie Salesforce Classic verwenden und Setup (Einstellung) in der Kopfzeile der Benutzeroberfläche sehen, wählen Sie es aus.
   - Wenn Sie Salesforce Classic verwenden und Setup (Einstellung) nicht angezeigt wird, wählen Sie Ihren Namen in der oberen Navigationsleiste und wählen dann Setup (Einstellung) aus der Dropdown-Liste aus.
4. Wählen Sie in der linken Navigationsleiste Company Settings (Unternehmenseinstellungen).
5. Wählen Sie in der Navigationsleiste Domain (Domäne), geben Sie eine Domäne ein und wählen Sie Create (Erstellen).
6. Wählen Sie in der linken Navigationsleiste Platform Tools (Plattform-Tools) und wählen Sie Apps (Anwendungen).
7. Wählen Sie App Manager.
8. a. Wählen Sie new connected app (neue verbundene App) aus.
   b. Füllen Sie die Pflichtfelder aus.

      Geben Sie unter Start URL (Start-URL) eine URL am `/authorize`-Endpunkt für die Benutzerpool-Domäne ein, die sich bei Ihrem Salesforce-IDP anmeldet. Wenn Ihre Benutzer auf Ihre verbundene App zugreifen, leitet Salesforce sie an diese URL weiter, um die Anmeldung abzuschließen. Dann leitet Salesforce die Benutzer an die Rückruf-URL um, die Sie Ihrem App-Client zugeordnet haben.

      ```
      https://mydomain.us-east-1.amazoncognito.com/authorize?
      response_type=code&client_id=<your_client_id>&redirect_uri=https://
      www.example.com&identity_provider=CorpSalesforce
      ```

   c. Aktivieren Sie OAuth settings (OAuth-Einstellungen) und geben Sie die URL des `/oauth2/idpresponse`-Endpunkts für Ihre Benutzerpool-Domäne unter Callback URL (Rückruf-

URL) ein. Dies ist die URL, mit der SalesForce den Autorisierungscode ausgibt, den Amazon Cognito gegen ein OAuth-Token eintauscht.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

9. Wählen Sie Ihre Bereiche aus. Sie müssen den Bereich openid einschließen. Fügen Sie den Bereich email hinzu, um Zugriff auf die Ansprüche email und email_verified zu erteilen. Trennen Sie Bereiche durch Leerzeichen.
10. Wählen Sie Create (Erstellen) aus.

    In Salesforce wird die Client-ID als Consumer Key (Verbraucherschlüssel) bezeichnet, der geheime Client-Schlüssel als Consumer Secret (Verbrauchergeheimnis). Notieren Sie die Client-ID und den geheimen Client-Schlüssel. Sie brauchen diese Informationen im nächsten Abschnitt.

Schritt 2: Hinzufügen eines OIDC-IdP zu Ihrem Benutzerpool

In diesem Abschnitt konfigurieren Sie Ihren Benutzerpool, um OIDC-basierte Authentifizierungsanforderungen von einem OIDC IdP zu verarbeiten.

Einen OIDC IdP hinzufügen (Amazon-Cognito-Konsole)

Einen OIDC-IdP hinzufügen

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus dem Navigationsmenü aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Federated sign-in (Verbundanmeldung) und wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) aus.
5. Wählen Sie einen OpenID Connect-IDP aus.
6. Geben Sie einen eindeutigen Namen in Provider name (Anbietername) ein.
7. Geben Sie die Kunden-ID, die Sie von Ihrem Anbieter erhalten haben, in Client ID (Client-ID) ein.
8. Geben Sie das Kundengeheimnis, das Sie von Ihrem Anbieter erhalten haben, in Client secret (Client-Geheimnis) ein.

9. Geben Sie Authorized scopes (Autorisierte Bereiche) für diesen Anbieter ein. Bereiche definieren, welche Gruppen von Benutzerattributen (z. B. `name` und `email`) Ihre Anwendung von Ihrem Anbieter anfordert. Bereiche müssen durch Leerzeichen getrennt werden, gefolgt von einer OAuth 2.0-Angabe.

   Der Endbenutzer wird aufgefordert, der Bereitstellung dieser Attribute für die Anwendung zuzustimmen.

10. Wählen Sie eine Attributanforderungsmethode, um Amazon Cognito die HTTP-Methode (entweder GET oder POST) bereitzustellen, mit der die Details des Benutzers vom Endpunkt userInfo, der von Ihrem Anbieter betrieben wird, abgerufen werden.

11. Wählen Sie eine Einrichtungsmethode, um OpenID-Connect-Endpunkte abzurufen, entweder durch Auto fill through issuer URL (Automatisches Ausfüllen durch Aussteller-URL) oder Manual input (Manuelle Eingabe). Verwenden Sie Auto fill through issuer URL (Automatisches Ausfüllen durch Aussteller-URL), wenn Ihr Anbieter einen öffentlichen `.well-known/openid-configuration`-Endpunkt besitzt, von dem Amazon Cognito die URLs der `authorization`, `token`, `userInfo` und `jwks_uri`-Endpunkte abrufen kann.

12. Geben Sie die Aussteller-URL oder `authorization`-, `token`-, `userInfo`- und `jwks_uri`-Endpunkt-URLs von Ihrem Identitätsanbieter ein.

    > **Note**
    >
    > Die URL sollte mit `https://` beginnen und nicht mit einem Slash `/` enden. Nur die Portnummern 443 und 80 können mit dieser URL verwendet werden. Beispielsweise verwendet Salesforce diese URL:
    > `https://login.salesforce.com`
    > Wenn Sie das automatische Ausfüllen auswählen, muss das Erkennungsdokument HTTPS für die folgenden Werte verwenden: `authorization_endpoint`, `token_endpoint`, `userinfo_endpoint` und `jwks_uri`. Andernfalls schlägt die Anmeldung fehl.

13. Standardmäßig wird sub des OIDC-Anspruchs dem Benutzerpool-Attribut Username (Benutzername) zugeordnet. Sie können Benutzerpool-Attributen weitere OIDC-Ansprüche hinzufügen. Geben Sie den OIDC-Anspruch ein, und wählen Sie das entsprechende Benutzerpool-Attribut aus der Dropdown-Liste aus. Beispielsweise wird der Anspruch email (E-Mail) häufig dem Benutzerpool-Attribut Email (E-Mail) hinzugefügt.

14. Ordnen Sie Ihrem Benutzerpool Attribute von Ihrem IdP zu. Weitere Informationen finden Sie unter Geben Sie die Identitätsanbieter-Attributzuordnungen für Ihren Benutzerpool an.
15. Wählen Sie Create (Erstellen) aus.
16. Wählen Sie auf der Registerkarte App client integration (App-Client-Integration) einen der App-Clients aus der Liste aus und klicken Sie anschließend auf Edit hosted UI settings (Einstellungen für gehostete UI bearbeiten). Fügen Sie unter Identity providers (Identitätsanbieter) den neuen OIDC-IdP zum App-Client hinzu.
17. Wählen Sie Save Changes.

Einen OIDC IdP hinzufügen (AWS CLI)

- Weitere Informationen finden Sie in den Parameterbeschreibungen für die CreateIdentityProviderAPI-Methode.

```
aws cognito-idp create-identity-provider
--user-pool-id string
--provider-name string
--provider-type OIDC
--provider-details map


--attribute-mapping string
--idp-identifiers (list)
--cli-input-json string
--generate-cli-skeleton string
```

Verwenden Sie diese Zuordnung der Anbieterdetails:

```
{
  "client_id": "string",
  "client_secret": "string",
  "authorize_scopes": "string",
  "attributes_request_method": "string",
  "oidc_issuer": "string",
```

```
  "authorize_url": "string",
  "token_url": "string",
  "attributes_url": "string",
  "jwks_uri": "string"
}
```

Schritt 3: Testen Ihrer OIDC-IdP-Konfiguration

Sie können die Berechtigungs-URL erstellen, indem Sie die Elemente aus den beiden vorhergehenden Abschnitten verwenden und damit Ihre OIDC IdP-Konfiguration testen.

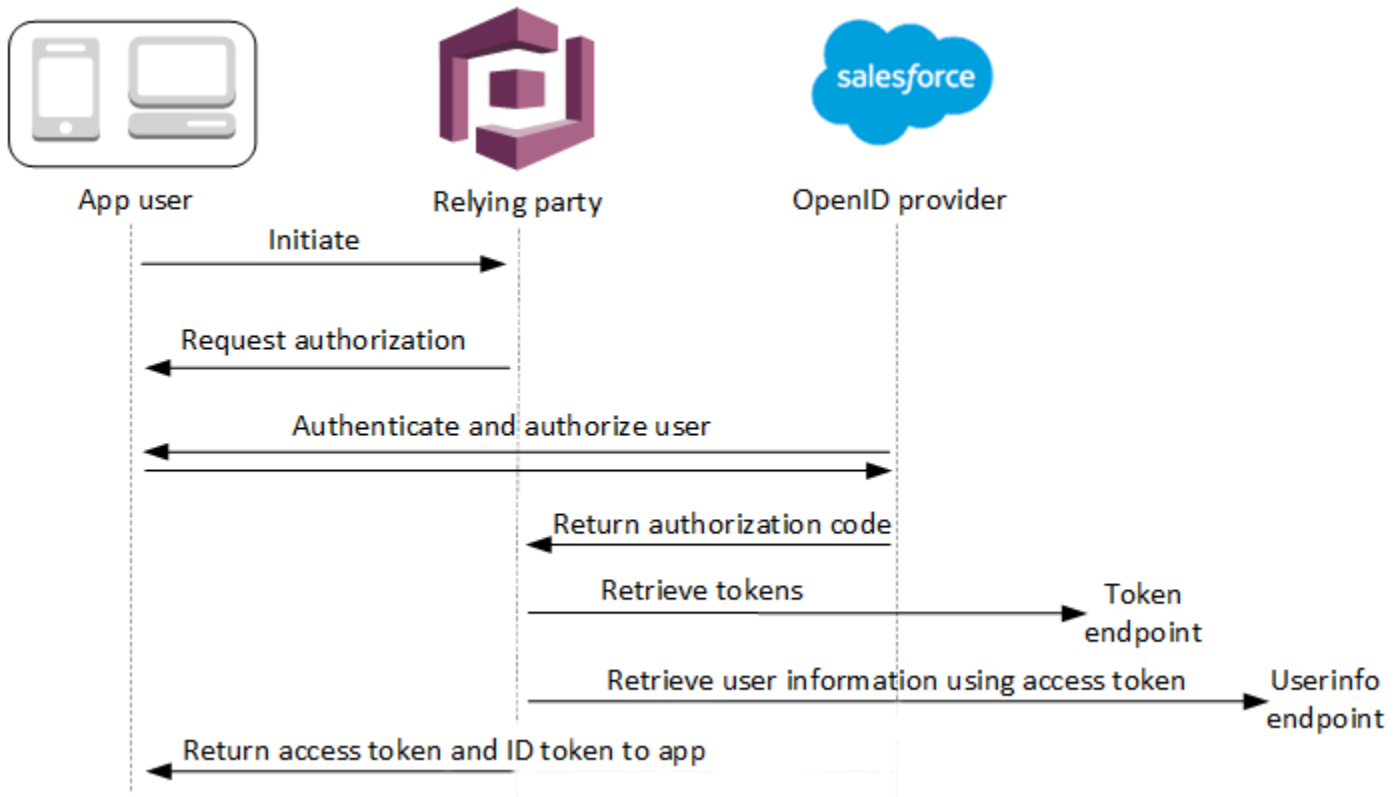
```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Sie finden Ihre Domäne auf der Konsolenseite Domain name (Domänenname) für den Benutzerpool. Die client_id befindet sich auf der Seite General settings (Allgemeine Einstellungen). Verwenden Sie Ihre Callback-URL für den redirect_uri-Parameter. Dies ist die URL der Seite, auf die Ihre Benutzer nach einer erfolgreichen Authentifizierung umgeleitet werden.

Authentifizierungsablauf für den OIDC-Benutzerpool-IdP

Wenn sich Ihr Benutzer mit einem OIDC IdP bei Ihrer Anwendung anmeldet, durchläuft er diesen Authentifizierungsablauf.

1. Ihr Benutzer landet auf der integrierten Anmeldeseite von Amazon Cognito und erhält die Möglichkeit, sich über einen OIDC-IdP wie Salesforce anzumelden.
2. Ihr Benutzer wird an den `authorization`-Endpunkt des OIDC-IdP umgeleitet.
3. Nachdem Ihr Benutzer authentifiziert wurde, leitet der OIDC IdP ihn mit einem Autorisierungscode zu Amazon Cognito weiter.
4. Amazon Cognito tauscht den Autorisierungscode mit dem OIDC-IdP für ein Zugriffstoken aus.
5. Amazon Cognito erstellt oder aktualisiert das Benutzerkonto in Ihrem Benutzerpool.
6. Amazon Cognito stellt die Bearer-Token für Ihre Anwendung aus, wobei es sich unter anderem um Identitäts-, Zugriffs- und Aktualisierungs-Token handeln kann.



> **Note**
>
> Amazon Cognito bricht Authentifizierungsanfragen ab, die nicht innerhalb von 5 Minuten abgeschlossen werden, und leitet den Benutzer an die gehostete Benutzeroberfläche um. Für die Seite wird eine `Something went wrong`-Fehlermeldung angezeigt.

OIDC ist eine Identitätsebene, die auf OAuth 2.0 aufsetzt und Identitätstoken im JSON-Format (JWT) spezifiziert, die von OIDC-Client-Apps (Relying Parties) ausgegeben IdPs werden. In der Dokumentation zu Ihrem OIDC IdP finden Sie Informationen darüber, wie Sie Amazon Cognito als OIDC-vertrauende Partei hinzufügen können.

Wenn sich ein Benutzer mit einer Autorisierungscode-Erteilung authentifiziert, gibt der Benutzerpool das ID-, Zugriffs- und Aktualisierungstoken zurück. Das ID-Token ist ein Standard-OIDC-Token für die Identitätsverwaltung, und das Zugriffs-Token ist ein OAuth 2.0-Standardtoken. Weitere Informationen zu den Erteilungsarten, die Ihr Benutzerpool-App-Client unterstützen kann, finden Sie unter Autorisieren des Endpunkts.

So verarbeitet ein Benutzerpool Anträge eines OIDC-Anbieters

Wenn Ihr Benutzer die Anmeldung bei einem OIDC-Drittanbieter abschließt, ruft die von Amazon Cognito gehostete Benutzeroberfläche einen Autorisierungscode vom IdP ab. Ihr Benutzerpool tauscht den Autorisierungscode für Zugriffs- und ID-Tokens mit dem `token`-Endpunkt Ihres IdP aus. Ihr Benutzerpool gibt diese Token nicht an Ihren Benutzer oder Ihre App weiter, sondern verwendet sie, um ein Benutzerprofil mit Daten zu erstellen, die in Form von Anträgen in eigenen Tokens dargestellt werden.

Amazon Cognito validiert das Zugriffstoken nicht unabhängig. Stattdessen fordert es Benutzerattributinformationen vom `userInfo`-Endpunkt des Anbieters an und erwartet, dass die Anfrage abgelehnt wird, wenn das Token nicht gültig ist.

Amazon Cognito validiert das Anbieter-ID-Token mit den folgenden Prüfungen:

1. Prüfen, ob der Anbieter das Token mit einem Algorithmus aus dem folgenden Satz signiert hat: RSA, HMAC, Elliptic Curve.
2. Wenn der Anbieter das Token mit einem asymmetrischen Signaturalgorithmus signiert hat, prüfen, ob die Signaturschlüssel-ID im Token-`kid`-Antrag am Endpunkt `jwks_uri` des Anbieters aufgeführt ist.
3. Die ID-Tokensignatur mit der Signatur vergleichen, die auf der Grundlage der Anbieter-Metadaten erwartet wird.
4. Den `iss`-Antrag mit dem für den IdP konfigurierten OIDC-Aussteller vergleichen.
5. Vergleichen, ob der `aud`-Antrag mit der auf dem IdP konfigurierten Client-ID übereinstimmt oder ob er die konfigurierte Client-ID enthält, wenn der `aud`-Antrag mehrere Werte enthält.
6. Sicherstellen, dass der Zeitstempel im `exp`-Antrag nicht vor der aktuellen Uhrzeit liegt.

Ihr Benutzerpool validiert das ID-Token und versucht dann, mit dem Anbieter-Zugriffstoken eine Anfrage an den Anbieter-`userInfo`-Endpunkt zu stellen. Er ruft alle Benutzerprofilinformationen ab, zu deren Lesen die Bereiche im Zugriffstoken berechtigen. Ihr Benutzerpool sucht dann nach den Benutzerattributen, die Sie in Ihrem Benutzerpool als erforderlich festgelegt haben. Sie müssen in Ihrer Anbieterkonfiguration Attributzuordnungen für die erforderlichen Attribute erstellen. Ihr Benutzerpool überprüft das Provider-ID-Token und die `userInfo`-Antwort. Ihr Benutzerpool schreibt alle Anträge, die den Zuordnungsregeln entsprechen, den Benutzerattributen im Benutzerprofil des Benutzerpools zu. Ihr Benutzerpool ignoriert Attribute, die einer Zuordnungsregel entsprechen, aber nicht erforderlich sind und nicht in den Anträgen des Anbieters enthalten sind.

Geben Sie die Identitätsanbieter-Attributzuordnungen für Ihren Benutzerpool an

Sie können die AWS Management Console API oder die AWS CLI oder verwenden, um Attributzuordnungen für den Identitätsanbieter (IdP) Ihres Benutzerpools anzugeben.

Wissenswertes über Mappings

Bevor Sie mit der Einrichtung der Zuordnung von Benutzerattributen beginnen, sollten Sie sich die folgenden wichtigen Details ansehen.

- Bei der Anmeldung eines Verbundbenutzers bei Ihrer Anwendung muss ein Mapping für jedes Benutzerpool-Attribut vorhanden sein, das Ihr Benutzerpool verlangt. Beispiel: Wenn Ihr Benutzerpool ein `email`-Attribut für die Anmeldung erfordert, ordnen Sie dieses Attribut seiner Entsprechung vom IDP zu.
- Standardmäßig werden zugeordnete E-Mail-Adressen nicht überprüft. Sie können eine zugeordnete E-Mail-Adresse nicht mit einem einmaligen Code verifizieren. Ordnen Sie stattdessen ein Attribut von Ihrem IdP zu, um den Überprüfungsstatus zu erhalten. Google und die meisten OIDC-Anbieter enthalten beispielsweise das `email_verified`-Attribut.
- So können in Ihrem Benutzerpool Identitätsanbieter(IDP)-Token benutzerdefinierten Attributen zuordnen. Anbieter sozialer Netzwerke präsentieren ein Zugriffstoken, und OIDC-Anbieter präsentieren ein Zugriffs- und ein ID-Token. Um ein Token zuzuordnen, fügen Sie ein benutzerdefiniertes Attribut mit einer maximalen Länge von 2048 Zeichen hinzu, gewähren Sie Ihrem App-Client Schreibzugriff auf das Attribut und ordnen Sie dem benutzerdefinierten Attribut `access_token` oder `id_token` des IDP zu.
- Für jedes zugeordnete Benutzerpool-Attribut muss die maximale Länge des Wertes (2048 Zeichen) für den Wert groß genug sein, den Amazon Cognito vom IDP abruft. Andernfalls meldet Amazon Cognito einen Fehler, wenn sich Benutzer bei Ihrer Anwendung anmelden. Amazon Cognito unterstützt die Zuordnung von IDP-Token zu benutzerdefinierten Attributen nicht, wenn die Token mehr als 2048 Zeichen lang sind.
- Amazon Cognito leitet das `username` Attribut im Profil eines Verbundbenutzers von bestimmten Ansprüchen ab, die Ihr föderierter IdP erfüllt, wie in der folgenden Tabelle dargestellt. Amazon Cognito stellt diesem Attributwert beispielsweise den Namen Ihres IdP voran. `MyOIDCIdP_[sub]` Wenn Sie möchten, dass Ihre Verbundbenutzer ein Attribut haben, das genau mit einem Attribut in Ihrem externen Benutzerverzeichnis übereinstimmt, ordnen Sie dieses Attribut einem Amazon Cognito Cognito-Anmeldeattribut wie zu. `preferred_username`

Identitätsanbieter	Quellattribut **`username`**
Facebook	`id`
Google	`sub`
Login with Amazon	`user_id`
Mit Apple anmelden	`sub`
SAML-Anbieter	`NameID`
Open ID Connect (OIDC)-Anbieter	`sub`

- Amazon Cognito muss Ihre zugeordneten Benutzerpool-Attribute aktualisieren können, wenn sich Benutzer bei Ihrer Anwendung anmelden. Wenn sich ein Benutzer über einen IdP anmeldet, aktualisiert Amazon Cognito die zugeordneten Attribute mit den neuesten Informationen vom IdP. Amazon Cognito aktualisiert die einzelnen zugeordneten Attribute. Dies gilt auch dann, wenn der aktuelle Wert bereits den neuesten Informationen entspricht. Um sicherzustellen, dass Amazon Cognito die Attribute aktualisieren kann, überprüfen Sie die folgenden Voraussetzungen:
  - Alle benutzerdefinierten Attribute des Benutzerpools, die Sie von Ihrem IDP aus zuordnen, müssen veränderbar sein. Sie können veränderbare benutzerdefinierte Attribute jederzeit aktualisieren. Im Gegensatz dazu können Sie nur einen Wert für das unveränderliche benutzerdefinierte Attribut festlegen, wenn Sie das Benutzerprofil erstellen. Wenn Sie ein veränderbares benutzerdefiniertes Attribut in der Amazon-Cognito-Konsole erstellen möchten, aktivieren Sie das Kontrollkästchen Mutable (Veränderbar) für das Attribut, das Sie beim Auswählen von Add custom attributes (Benutzerdefinierte Attribute hinzufügen) auf der Registerkarte Sign-up experience (Anmeldeerlebnis) hinzufügen. Oder, wenn Sie Ihren Benutzerpool mithilfe der CreateUserPoolAPI-Operation erstellen, können Sie den `Mutable` Parameter für jedes dieser Attribute auf setzen. `true` Wenn Ihr IdP einen Wert für ein zugeordnetes unveränderliches Attribut sendet, gibt Amazon Cognito einen Fehler zurück und die Anmeldung schlägt fehl.
  - In den App-Client-Einstellungen für Ihre Anwendung müssen zugeordnete Attribute beschreibbar sein. Sie können festlegen, welche Attribute auf der Seite App-Clients in der Amazon-Cognito-Konsole beschreibbar sind. Wenn Sie den App-Client mithilfe der API-Operation `CreateUserPoolClient` erstellen, können Sie diese Attribute zum Array `WriteAttributes`

hinzufügen. Wenn Ihr IdP einen Wert für ein zugeordnetes nicht beschreibbares Attribut sendet, legt Amazon Cognito den Attributwert nicht fest und fährt mit der Authentifizierung fort.

- Wenn IdP-Attribute mehrere Werte enthalten, reduziert Amazon Cognito alle Werte zu einer einzigen kommagetrennten Zeichenfolge und die Werte, die nicht-alphanumerische Zeichen enthalten (mit Ausnahme der Zeichen ", ", ", " und '`.`'), werden per URL formenkodiert. `-` `*` `_` Sie müssen die einzelnen Werte vor der Verwendung in Ihrer App decodieren und analysieren.

Geben Sie die Identitätsanbieter-Attributzuordnungen für Ihren Benutzerpool an (AWS Management Console)

Sie können die verwenden AWS Management Console , um Attributzuordnungen für den IdP in Ihrem Benutzerpool anzugeben.

> **Note**
>
> Amazon Cognito ordnet eingehende Ansprüche nur dann Benutzerpoolattributen zu, wenn die Ansprüche im eingehenden Token vorhanden sind. Wenn ein zuvor zugeordneter Anspruch nicht mehr im eingehenden Token vorhanden ist, wird er nicht gelöscht oder geändert. Wenn Ihre Anwendung einen Abgleich von gelöschten Anforderungen erfordert, können Sie vor der Authentifizierung den Lambda-Auslöser verwenden, um das benutzerdefinierte Attribut während der Authentifizierung zu löschen und diesen Attributen das erneute Auffüllen aus dem eingehenden Token zu ermöglichen.

Geben Sie eine Attributzuordnung für einen Social-IDP wie folgt an

1. Melden Sie sich bei der Amazon Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre Anmeldeinformationen ein. AWS
2. Wählen Sie im Navigationsbereich erst User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus und suchen Sie nach Federated sign-in (Verbundanmeldung).
4. Wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) oder den Facebook-, Google-, Amazon- oder Apple-IdP aus, den Sie konfiguriert haben. Suchen Sie nach Attribute mapping (Attributzuordnung) und wählen Sie Edit (Bearbeiten) aus.

Weitere Informationen zum Hinzufügen eines Social-IdP finden Sie unter Verwenden von Anbietern für soziale Identitäten mit einem Benutzerpool.

5. Für jedes Attribut, das Sie zuordnen müssen, führen Sie die folgenden Schritte aus:

   a. Wählen Sie ein Attribut aus der Spalte User pool attribute (Benutzerpoolattribut) aus. Dies ist das Attribut, das dem Benutzerprofil in Ihrem Benutzerpool zugewiesen ist. Benutzerdefinierte Attribute werden nach Standardattributen aufgeführt.

   b. Wählen Sie ein Attribut aus der Spalte ***<provider>***-Attribut aus. Dies ist das Attribut, das aus dem Anbieterverzeichnis übergeben wird. Bekannte Attribute des Social-Identity-Anbieters werden in einer Dropdown-Liste bereitgestellt.

   c. Um zusätzliche Attribute zwischen Ihrem IdP und Amazon Cognito zuzuordnen, wählen Sie Add another attribute (Weiteres Attribut hinzufügen) aus.

6. Wählen Sie Save Changes.

So geben Sie das Attribut-Mapping für einen SAML-Anbieter an

1. Melden Sie sich bei der Amazon Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie im Navigationsbereich erst User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus und suchen Sie nach Federated sign-in (Verbundanmeldung).
4. Wählen Sie Add an identity provider (Identitätsanbieter hinzufügen) oder den SAML-IdP aus, den Sie konfiguriert haben. Suchen Sie nach Attribute mapping (Attributzuordnung) und wählen Sie Edit (Bearbeiten) aus. Weitere Informationen zum Hinzufügen eines SAML-IdP finden Sie unter Verwenden von SAML-Identitätsanbietern mit einem Benutzerpool.
5. Für jedes Attribut, das Sie zuordnen müssen, führen Sie die folgenden Schritte aus:

   a. Wählen Sie ein Attribut aus der Spalte User pool attribute (Benutzerpoolattribut) aus. Dies ist das Attribut, das dem Benutzerprofil in Ihrem Benutzerpool zugewiesen ist. Benutzerdefinierte Attribute werden nach Standardattributen aufgeführt.

   b. Wählen Sie ein Attribut aus der Spalte SAML attribute (SAML-Attribut) aus. Dies ist das Attribut, das aus dem Anbieterverzeichnis übergeben wird.

Ihr IdP bietet möglicherweise SAML-Assertions als Referenz an. Einige IdPs verwenden einfache Namen, wie z. B.`email`, während andere URL-formatierte Attributnamen verwenden, die den folgenden ähneln:

```
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

c. Um zusätzliche Attribute zwischen Ihrem IdP und Amazon Cognito zuzuordnen, wählen Sie Add another attribute (Weiteres Attribut hinzufügen) aus.

6. Wählen Sie Save Changes.

Angeben von Zuordnungen von Identitätsanbieter-Attributen für Ihren Benutzerpool (und Ihre API)AWS CLIAWS

Verwenden Sie die folgenden Befehle für die Angabe von Attributzuordnungen für den IdP Ihres Benutzerpools

So geben Sie Attributzuordnungen auf Anbieter-Erstellungszeit an

- AWS CLI: `aws cognito-idp create-identity-provider`

  Beispiel mit Metadaten-Dokument: : `aws cognito-idp create-identity-provider --user-pool-id` *`<user_pool_id>`* `--provider-name=SAML_provider_1 --provider-type SAML --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

  Wo `details.json` enthält:

```
{
    "MetadataFile": "<SAML metadata XML>"
}
```

> **Note**
>
> Falls das *`<SAML metadata XML>`* Anführungsstriche (") enthält, müssen sie durch Escapezeichen (\") geschützt werden.

  Beispiel mit Metadaten-URL:

```
aws cognito-idp create-identity-provider \
--user-pool-id us-east-1_EXAMPLE \
--provider-name=SAML_provider_1 \
--provider-type SAML \
--provider-details MetadataURL=https://myidp.example.com/saml/metadata \
--attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/
emailaddress
```

- AWS API: CreateIdentityProvider

Geben Sie Attributzuordnungen für einen vorhandenen Identitätsanbieter wie folgt an

- AWS CLI: `aws cognito-idp update-identity-provider`

  Beispiel: `aws cognito-idp update-identity-provider --user-pool-id <user_pool_id> --provider-name <provider_name> --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`
- AWS API: UpdateIdentityProvider

Rufen Sie Informationen über das Attribut-Mapping für einen bestimmten IdP wie folgt auf

- AWS CLI: `aws cognito-idp describe-identity-provider`

  Beispiel: `aws cognito-idp describe-identity-provider --user-pool-id <user_pool_id> --provider-name <provider_name>`
- AWS API: DescribeIdentityProvider

Verknüpfen von Verbundbenutzern mit einem vorhandenen Benutzerprofil

Oft hat derselbe Benutzer ein Profil mit mehreren Identitätsanbietern (IdPs), die Sie mit Ihrem Benutzerpool verbunden haben. Amazon Cognito kann jedes Vorkommen eines Benutzers mit demselben Benutzerprofil in Ihrem Verzeichnis verknüpfen. Auf diese Weise kann eine Person mit mehreren IDP-Benutzern von einer konsistenten Erfahrung in Ihrer App profitieren. AdminLinkProviderForUserweist Amazon Cognito an, die eindeutige ID eines Benutzers in Ihrem Verbundverzeichnis als Benutzer im Benutzerpool zu erkennen. Ein Benutzer in Ihrem Benutzerpool zählt als ein monatlicher aktiver Benutzer (MAU) für die Abrechnung, wenn Sie null oder mehr Verbundidentitäten mit dem Benutzerprofil verknüpft haben.

Wenn sich ein Verbundbenutzer zum ersten Mal an Ihrem Benutzerpool anmeldet, sucht Amazon Cognito nach einem lokalen Profil, das Sie mit seiner Identität verknüpft haben. Wenn kein verknüpftes Profil vorhanden ist, erstellt Ihr Benutzerpool ein neues Profil. Sie können jederzeit vor der ersten Anmeldung ein lokales Profil erstellen und es mit Ihrem Verbundbenutzer verknüpfen, und zwar in einer `AdminLinkProviderForUser` API-Anfrage, entweder in einer geplanten Prestaging-Aufgabe oder in einer. Lambda-Auslöser für die Vorab-Registrierung Nachdem sich Ihr Benutzer angemeldet hat und Amazon Cognito ein verknüpftes lokales Profil findet, liest Ihr Benutzerpool die Ansprüche Ihres Benutzers und vergleicht sie mit den Zuordnungsregeln für den IdP. Ihr Benutzerpool aktualisiert dann das verknüpfte lokale Profil mit den Ansprüchen, die bei der Anmeldung zugeordnet wurden. Auf diese Weise können Sie das lokale Profil mit Zugriffsansprüchen konfigurieren und deren Identitätsansprüche bei Ihrem Anbieter behalten. up-to-date Nachdem Amazon Cognito Ihren Verbundbenutzer einem verknüpften Profil zugeordnet hat, meldet er sich immer bei diesem Profil an. Anschließend können Sie weitere Anbieteridentitäten Ihres Benutzers mit demselben Profil verknüpfen, um Kunden ein einheitliches Erlebnis in Ihrer App zu bieten. Um einen Verbundbenutzer zu verknüpfen, der sich zuvor angemeldet hat, müssen Sie zunächst das vorhandene Profil löschen. Sie können vorhandene Profile an ihrem Format erkennen: *`[Provider name]`*`_identifier`. z. B. `LoginWithAmazon_amzn1.account.AFAEXAMPLE`. Ein Benutzer, den Sie erstellt und dann mit einer Benutzeridentität eines Drittanbieters verknüpft haben, hat den Benutzernamen, mit dem er erstellt wurde, und ein `identities` Attribut, das die Details seiner verknüpften Identitäten enthält.

> **Important**
>
> Da `AdminLinkProviderForUser` es einem Benutzer mit einer externen föderierten Identität ermöglicht wird, sich als vorhandener Benutzer im Benutzerpool anzumelden, ist es wichtig, dass er nur mit externen Attributen IdPs und Anbieterattributen verwendet wird, denen der Anwendungsbesitzer vertraut hat.

Angenommen, Sie sind ein Managed Service Provider (MSP) mit einer App, die Sie mit mehreren Kunden teilen. Jeder Kunde meldet sich über Active Directory Federation Services (ADFS) bei Ihrer App an. Ihr IT-Administrator, Carlos, verfügt in jeder Domäne Ihrer Kunden über ein Konto. Sie möchten, dass Carlos unabhängig vom IDP bei jeder Anmeldung als App-Administrator erkannt wird.

Ihr ADFS enthält IdPs die E-Mail-Adresse von Carlos `msp_carlos@example.com` in der `email` Reklamation der SAML-Behauptungen von Carlos an Amazon Cognito. Sie erstellen einen Benutzer

in Ihrem Benutzerpool mit dem Benutzernamen `Carlos`. Die folgenden Befehle AWS Command Line Interface (AWS CLI) verknüpfen Carlos' Identitäten mit ADFS1, ADFS2 und ADFS3. IdPs

> **Note**
>
> Sie können einen Benutzer basierend auf bestimmten Attributansprüchen verknüpfen. Diese Fähigkeit gibt es nur bei OIDC und SAML. IdPs Für andere Anbietertypen müssen Sie eine Verknüpfung basierend auf einem festen Quellattribut herstellen. Weitere Informationen finden Sie unter. AdminLinkProviderForUser Sie müssen `ProviderAttributeName` auf `Cognito_Subject` festlegen, wenn Sie einen Social-Identity-IDP mit einem Benutzerprofil verknüpfen. `ProviderAttributeValue` muss die eindeutige Kennung des Benutzers bei Ihrem IDP sein.

```
aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
 ProviderName=ADFS1,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com

aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
 ProviderName=ADFS2,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com

aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
 ProviderName=ADFS3,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com
```

Das Benutzerprofil `Carlos` in Ihrem Benutzerpool verfügt jetzt über das folgende `identities`-Attribut.

```
[{
    "userId": "msp_carlos@example.com",
    "providerName": "ADFS1",
    "providerType": "SAML",
    "issuer": "http://auth.example.com",
```

```
    "primary": false,
    "dateCreated": 111111111111111
}, {
    "userId": "msp_carlos@example.com",
    "providerName": "ADFS2",
    "providerType": "SAML",
    "issuer": "http://auth2.example.com",
    "primary": false,
    "dateCreated": 111111111111111
}, {
    "userId": "msp_carlos@example.com",
    "providerName": "ADFS3",
    "providerType": "SAML",
    "issuer": "http://auth3.example.com",
    "primary": false,
    "dateCreated": 111111111111111
}]
```

Wissenswertes zur Verknüpfung von Verbundbenutzern

- Sie können bis zu fünf Verbundbenutzer mit jedem Benutzerprofil verknüpfen.
- Sie können Verbundbenutzer entweder mit einem vorhandenen Verbundbenutzerprofil oder mit einem lokalen Benutzer verknüpfen.
- Sie können Anbieter nicht mit Benutzerprofilen in der verknüpfen AWS Management Console.
- Das ID-Token Ihres Benutzers enthält alle zugehörigen Anbieter im `identities`-Anspruch.
- Sie können in einer API-Anfrage ein Passwort für das automatisch erstellte Verbundbenutzerprofil festlegen. AdminSetUserPassword Der Status dieses Benutzers ändert sich dann von `EXTERNAL_PROVIDER` zu `CONFIRMED`. Ein Benutzer in diesem Status kann sich als Verbundbenutzer anmelden und Authentifizierungsabläufe in der API wie ein verknüpfter lokaler Benutzer initiieren. Sie können ihr Passwort und ihre Attribute auch in API-Anfragen mit Token-Authentifizierung wie und ändern. ChangePasswordUpdateUserAttributes Als bewährte Sicherheitsmethode und zur Synchronisierung von Benutzern mit Ihrem externen IdP sollten Sie keine Passwörter für Verbundbenutzerprofile festlegen. Verknüpfen Sie Benutzer stattdessen mit lokalen Profilen mit `AdminLinkProviderForUser`.
- Amazon Cognito füllt Benutzerattribute in ein verknüpftes lokales Benutzerprofil ein, wenn sich der Benutzer über seinen IdP anmeldet. Amazon Cognito verarbeitet Identitätsanforderungen im ID-Token eines OIDC-IdP und überprüft auch den `userInfo`-Endpunkt sowohl von OAuth-2.0- als auch von OIDC-Anbietern. Amazon Cognito priorisiert Informationen in einem ID-Token gegenüber Informationen von`userInfo`.

Wenn Sie erkennen, dass Ihr Benutzer kein externes Benutzerkonto mehr verwendet, das Sie mit seinem Profil verknüpft haben, können Sie die Zuordnung dieses Benutzerkontos zu Ihrem Benutzerpool-Benutzer aufheben. Bei der Verknüpfung des Benutzers haben Sie in der Anfrage den Attributnamen, den Attributwert und den Anbieternamen des Benutzers angegeben. Um ein Profil zu entfernen, das Ihr Benutzer nicht mehr benötigt, stellen Sie eine AdminDisableProviderForUserAPI-Anfrage mit entsprechenden Parametern.

AdminLinkProviderForUserWeitere Befehlssyntax und Beispiele finden Sie in den AWS SDKs.

Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern

Amazon Cognito nutzt AWS Lambda-Funktionen, um das Authentifizierungsverhalten Ihres Benutzerpools zu ändern. Sie können Ihren Benutzerpool so konfigurieren, dass Lambda-Funktionen vor der ersten Registrierung, nach Abschluss der Authentifizierung und in mehreren Phasen dazwischen automatisch aufgerufen werden. Ihre Funktionen können das Standardverhalten Ihres Authentifizierungsprozesses ändern, API-Anfragen zur Änderung Ihres Benutzerpools oder anderer AWS-Ressourcen stellen und mit externen Systemen kommunizieren. Der Code in Ihren Lambda-Funktionen ist Ihr eigener. Amazon Cognito sendet Ereignisdaten an Ihre Funktion, wartet, bis die Funktion die Daten verarbeitet, und erwartet in den meisten Fällen ein Antwortereignis, das alle Änderungen widerspiegelt, die Sie an der Sitzung vornehmen möchten.

Innerhalb des Systems mit Anfrage- und Antwortereignissen können Sie Ihre eigenen Authentifizierungs-Challenges hinzufügen, Benutzer zwischen Ihrem Benutzerpool und einem anderen Identitätsspeicher migrieren, Nachrichten anpassen und JSON-Webtoken (JWTs) ändern.

Lambda-Trigger können die Antwort anpassen, die Amazon Cognito an Ihren Benutzer sendet, nachdem er eine Aktion in Ihrem Benutzerpool ausgelöst hat. Sie können beispielsweise die Anmeldung eines Benutzers verhindern, der sich andernfalls erfolgreich anmelden könnte. Sie können auch Laufzeit-Operationen für Ihre AWS-Umgebung, externe APIs, Datenbanken oder Identitätsspeicher ausführen. Der Trigger „Benutzer migrieren“ kann beispielsweise eine externe Aktion mit einer Änderung in Amazon Cognito kombinieren: Sie können Benutzerinformationen in einem externen Verzeichnis nachschlagen und dann auf der Grundlage dieser externen Informationen Attribute für einen neuen Benutzer festlegen.

Wenn Sie Ihrem Benutzerpool einen Lambda-Trigger zugewiesen haben, unterbricht Amazon Cognito seinen Standardablauf, um Informationen von Ihrer Funktion anzufordern. Amazon Cognito generiert ein JSON--Ereignis und übergibt es an Ihre Funktion. Das Ereignis enthält Informationen über die Anfrage Ihres Benutzers, ein Benutzerkonto zu erstellen, sich anzumelden, ein

Passwort zurückzusetzen oder ein Attribut zu aktualisieren. Ihre Funktion hat dann die Möglichkeit, Maßnahmen zu ergreifen oder das Ereignis unverändert zurückzusenden.

In der folgenden Tabelle sind einige der Möglichkeiten zusammengefasst, wie Sie Lambda-Auslöser verwenden können, um Benutzerpool-Operationen anzupassen:

Benutzerpool-Ablauf	Operation	Beschreibung
Benutzerdefinierter Authentifizierungsfluss	Authentifizierungsaufforderung definieren	Bestimmt die nächste Aufforderung in einem benutzerdefinierten Authentifizierungsablauf
	Authentifizierungsaufforderung erstellen	Erstellt eine Aufforderung in einem benutzerdefinierten Authentifizierungsablauf
	Antwort auf Authentifizierungsaufforderung überprüfen	Bestimmt, ob eine Antwort in einem benutzerdefinierten Authentifizierungsablauf richtig ist
Authentifizierungsereignisse	the section called “Lambda-Auslöser für die Vorab-Authentifizierung”	Benutzerdefinierte Validierung zur Akzeptierung oder Ablehnung einer Anmeldeanforderung
	the section called “Lambda-Auslöser nach der Authentifizierung”	Protokolliert Ereignisse für benutzerdefinierte Analysen
	the section called “Lambda-Auslöser für die Vorab-Generierung von Token”	Erhöht oder unterdrückt Token-Ansprüche
Registrieren	the section called “Lambda-Auslöser für die Vorab-Registrierung”	Führt eine benutzerdefinierte Validierung durch, die die Anmeldeanforderung akzeptiert oder ablehnt

Benutzerpool-Ablauf	Operation	Beschreibung
	the section called “Lambda-Auslöser nach der Bestätigung”	Fügt benutzerdefinierte Begrüßungsnachrichten oder Ereignisprotokollierung für benutzerdefinierte Analysen hinzu
	the section called “Lambda-Auslöser für die Benutzermigration.”	Migriert Benutzer aus einem vorhandenen Benutzerverzeichnis in Benutzerpools
Nachrichten	the section called “Lambda-Auslöser für benutzerdefinierte Nachrichten”	Führt eine erweiterte Anpassung und Lokalisierung von Nachrichten durch
Erstellung von Tokens	the section called “Lambda-Auslöser für die Vorab-Generierung von Token”	Fügt Attribute in ID-Token hinzu oder entfernt sie
Drittanbieter von E-Mails und SMS	the section called “Benutzerdefinierter Lambda-Auslöser für Sender”	Sendet SMS- und E-Mail-Nachrichten mit einem Drittanbieter

Themen

- Wichtige Überlegungen
- Hinzufügen eines Lambda-Auslösers für einen Benutzerpool
- Lambda-Auslöserereignis für einen Benutzerpool
- Allgemeine Parameter von Lambda-Auslösern für Benutzerpools
- Verbinden von API-Operationen mit Lambda-Triggern
- Verbinden von Lambda-Triggern mit funktionalen Benutzerpool-Vorgängen
- Lambda-Auslöser für die Vorab-Registrierung
- Lambda-Auslöser nach der Bestätigung
- Lambda-Auslöser für die Vorab-Authentifizierung
- Lambda-Auslöser nach der Authentifizierung

- Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen
- Lambda-Auslöser für die Vorab-Generierung von Token
- Lambda-Auslöser für die Benutzermigration.
- Lambda-Auslöser für benutzerdefinierte Nachrichten
- Benutzerdefinierter Lambda-Auslöser für Sender

Wichtige Überlegungen

Wenn Sie Ihre Benutzerpools für Lambda-Funktionen vorbereiten, sollten Sie Folgendes berücksichtigen:

- Die Ereignisse, die Amazon Cognito an Ihre Lambda-Trigger sendet, können sich ändern, wenn neue Features hinzugefügt werden. Die Positionen der Antwort- und Anforderungselemente in der JSON-Hierarchie können sich ändern, oder es können weitere Elementnamen hinzugefügt werden. In Ihrer Lambda-Funktion können Sie erwarten, die in diesem Handbuch beschriebenen Schlüssel-Wert-Paare aus Eingabeelementen zu erhalten, jedoch kann eine strengere Eingabevalidierung dazu führen, dass Ihre Funktionen fehlschlagen.
- Sie können eine aus mehreren Versionen der Ereignisse auswählen, die Amazon Cognito an verschiedene Trigger sendet. Bei einigen Versionen müssen Sie möglicherweise eine Änderung Ihrer Amazon-Cognito-Preisgestaltung akzeptieren. Weitere Informationen zu Preisen finden Sie unter Amazon-Cognito-Preise. Wenn Sie Zugriffs-Token in einem Lambda-Auslöser für die Vorab-Generierung von Token anpassen möchten, müssen Sie Ihren Benutzerpool mit erweiterten Sicherheitsfunktionen konfigurieren und Ihre Lambda-Trigger-Konfiguration so aktualisieren, dass die Ereignisversion 2 verwendet wird.
- Mit Ausnahme von Benutzerdefinierter Lambda-Auslöser für Sender ruft Amazon Cognito Lambda-Funktionen synchron auf. Wenn Amazon Cognito Ihre Lambda-Funktion aufruft, muss sie innerhalb von 5 Sekunden reagieren. Wenn dies nicht der Fall ist und der Aufruf wiederholt werden kann, versucht Amazon Cognito den Aufruf erneut. Nach drei nicht erfolgreichen Versuchen erzeugt die Funktion ein Timeout. Sie können diesen Timeout-Wert von fünf Sekunden nicht ändern. Weitere Informationen finden Sie unter Lambda-Programmiermodell im Entwicklerhandbuch zu AWS Lambda.

  Amazon Cognito wiederholt keine Funktionsaufrufe, die einen Invoke-Fehler mit dem HTTP-Statuscode 500-599 zurückgeben. Diese Codes weisen auf ein Konfigurationsproblem hin, aufgrund dessen Lambda die Funktion nicht starten kann. Weitere Informationen finden Sie unter Fehlerbehandlung und automatische Wiederholungen in AWS Lambda.

- Sie können in Ihrer Lambda-Trigger-Konfiguration keine Funktionsversion deklarieren. Amazon-Cognito-Benutzerpools rufen standardmäßig die neueste Version Ihrer Funktion auf. Sie können jedoch eine Funktionsversion mit einem Alias verknüpfen und Ihren Trigger `LambdaArn` in einer API-Anfrage CreateUserPool oder UpdateUserPool auf den Alias-ARN setzen. Diese Option ist in der AWS Management Console nicht verfügbar. Weitere Informationen zu Aliassen finden Sie unter Lambda-Funktionsaliase im Entwicklerhandbuch für AWS Lambda.
- Wenn Sie einen Lambda-Auslöser löschen, müssen Sie den entsprechenden Auslöser im Benutzerpool aktualisieren. Wenn Sie zum Beispiel den Nachauthentifizierungs-Auslöser löschen, müssen Sie den Nachauthentifizierungs-Auslöser im entsprechenden Benutzerpool auf none (keine) setzen.
- Wenn Ihre Lambda-Funktion die Anforderungs- und Antwortparameter nicht an Amazon Cognito zurückgibt oder einen Fehler ausgibt, ist das Authentifizierungsereignis nicht erfolgreich. Sie können einen Fehler in Ihrer Funktion zurückgeben, um die Registrierung, Authentifizierung, Token-Generierung oder jede andere Phase des Authentifizierungsablaufs eines Benutzers zu verhindern, die den Lambda-Trigger aufruft.

  Die von Amazon Cognito gehostete Benutzeroberfläche gibt Fehler, die von Lambda-Triggern generiert wurden, als Fehlertext oberhalb der Anmeldeaufforderung zurück. Die API von Amazon-Cognito-Benutzerpools gibt Trigger-Fehler im Format `*[trigger]* failed with error *[error text from response]*` zurück. Generieren Sie als bewährte Methode nur Fehler in Ihren Lambda-Funktionen, die Ihre Benutzer sehen sollen. Verwenden Sie Ausgabemethoden wie `print()`, um sensible oder Debugging-Informationen in CloudWatch Logs zu protokollieren. Ein Beispiel finden Sie unter Beispiel für die Vorab-Registrierung: Anmeldung ablehnen, wenn der Benutzername weniger als fünf Zeichen enthält.
- Sie können eine Lambda-Funktion in einem anderen AWS-Konto als Auslöser für Ihren Benutzerpool hinzufügen. Sie müssen kontoübergreifende Auslöser mit den API-Operationen CreateUserPool und UpdateUserPool oder deren Entsprechungen in AWS CloudFormation und der AWS CLI hinzufügen. Sie können in der AWS Management Console keine kontoübergreifenden Funktionen hinzufügen.
- Wenn Sie in der Amazon-Cognito-Konsole einen Lambda-Auslöser hinzufügen, fügt Amazon Cognito Ihrer Funktion eine ressourcenbasierte Richtlinie hinzu, die es Ihrem Benutzerpool ermöglicht, die Funktion aufzurufen. Wenn Sie einen Lambda-Auslöser außerhalb der Amazon-Cognito-Konsole erstellen, müssen Sie der ressourcenbasierten Richtlinie der Lambda-Funktion Berechtigungen hinzufügen. Wenn Sie Berechtigungen hinzugefügt haben, kann Amazon Cognito die Funktion im Namen Ihres Benutzerpools aufrufen. Sie können Berechtigungen von der Lambda-Konsole hinzufügen oder die Lambda-API-Funktion AddPermission verwenden.

Beispiel für ressourcenbasierte Lambda-Richtlinie

Die folgende ressourcenbasierte Lambda-Richtlinie gewährt Amazon Cognito eine eingeschränkte Möglichkeit, eine Lambda-Funktion aufzurufen. Amazon Cognito kann nur dann die Funktion verwenden, wenn es diese Funktion für den Benutzerpool in der `aws:SourceArn`-Bedingung als auch für das Konto in der `aws:SourceAccount`-Bedingung übernimmt.

```
{
    "Version": "2012-10-17",
    "Id": "default",
    "Statement": [
        {
            "Sid": "lambda-allow-cognito",
            "Effect": "Allow",
            "Principal": {
                "Service": "cognito-idp.amazonaws.com"
            },
            "Action": "lambda:InvokeFunction",
            "Resource": "<your Lambda function ARN>",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "AWS:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
}
```

Hinzufügen eines Lambda-Auslösers für einen Benutzerpool

Hinzufügen eines Lambda-Auslösers für einen Benutzerpool mit der Konsole

1. Erstellen Sie eine Lambda-Funktion mit der Lambda-Konsole. Weitere Informationen finden Sie unter Lambda-Funktionsfehler im AWS Lambda-Lambda-Entwicklerhandbuch.
2. Navigieren Sie zur Amazon-Cognito-Konsole und wählen Sie User Pools (Benutzerpools) aus.

3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte User pool properties (Benutzerpool-Eigenschaften) und suchen Sie dort nach Lambda-Auslösern.
5. Wählen Sie Add a Lambda trigger (Lambda-Auslöser hinzufügen) aus.
6. Wählen Sie eine Kategorie für den Lambda-Auslöser basierend auf der Authentifizierungsphase aus, die Sie anpassen möchten.
7. Wählen Sie Assign Lambda function (Lambda-Funktion zuweisen) und dann eine Funktion in der gleichen AWS-Region, in der sich auch Ihr Benutzerpool befindet, aus.

   > **Note**
   >
   > Wenn Ihre AWS Identity and Access Management (IAM)-Anmeldeinformationen berechtigt sind, die Lambda-Funktion zu aktualisieren, fügt Amazon Cognito eine Ressourcenbasierte Lambda-Richtlinie hinzu. Mit dieser Richtlinie kann Amazon Cognito die von Ihnen ausgewählte Funktion aufrufen. Wenn die Anmeldeinformationen nicht über ausreichende IAM-Berechtigungen verfügen, müssen Sie die ressourcenbasierte Richtlinie separat aktualisieren. Weitere Informationen finden Sie unter the section called “Wichtige Überlegungen”.

8. Wählen Sie Änderungen speichern.
9. Sie können Ihre Lambda-Funktion mit CloudWatch in der Lambda-Konsole protokollieren. Weitere Informationen finden Sie unter Zugriff auf CloudWatch Logs für Lambda.

Lambda-Auslöserereignis für einen Benutzerpool

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Lambda-Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Dieses Ereignis zeigt die allgemeinen Parameter für den Lambda-Auslöser:

JSON

```
{
    "version": "string",
    "triggerSource": "string",
    "region": AWSRegion,
    "userPoolId": "string",
```

```
    "userName": "string",
    "callerContext":
        {
            "awsSdkVersion": "string",
            "clientId": "string"
        },
    "request":
        {
            "userAttributes": {
                "string": "string",
                ....
            }
        },
    "response": {}
}
```

Allgemeine Parameter von Lambda-Auslösern für Benutzerpools

Version

Die Versionsnummer der Lambda-Funktion.

triggerSource

Der Name des Ereignisses, das die Lambda-Funktion ausgelöst hat. Eine Beschreibung jeder triggerSource finden Sie unter Verbinden von Lambda-Triggern mit funktionalen Benutzerpool-Vorgängen.

region (Region)

AWS-Region als eine `AWSRegion`-Instance.

userPoolId

Die ID des Benutzerpools.

userName

Der Benutzername des aktuellen Benutzers.

callerContext

Metadaten über die Anfrage und die Codeumgebung. Diese enthalten die Felder awsSdkVersion und clientId.

awsSdkVersion

Die Version des AWS-SDK, das die Anfrage generiert hat.

clientId

Die Benutzerpool-App-Client-ID.

request

Details der API-Anfrage Ihres Benutzers. Diese enthalten die folgenden Felder und alle Anforderungsparameter, die für den Trigger spezifisch sind. Beispielsweise enthält ein Ereignis, das Amazon Cognito an einen Auslöser vor der Authentifizierung sendet, auch einen `userNotFound`-Parameter. Sie können den Wert dieses Parameters verarbeiten, um eine benutzerdefinierte Aktion auszuführen, wenn Ihr Benutzer versucht, sich mit einem nicht registrierten Benutzernamen anzumelden.

userAttributes

Ein oder mehrere Schlüssel-Wert-Paar(e) aus Benutzerattributnamen und -werten, beispielsweise `"email": "john@example.com"`.

response

Dieser Parameter enthält keine Informationen in der ursprünglichen Anfrage. Ihre Lambda-Funktion muss das gesamte Ereignis an Amazon Cognito zurückgeben und der `response` alle Rückgabeparameter hinzufügen. Welche Rückgabeparameter Ihre Funktion enthalten kann, finden Sie in der Dokumentation des Triggers, den Sie verwenden möchten.

Verbinden von API-Operationen mit Lambda-Triggern

In den folgenden Abschnitten werden die Lambda-Trigger beschrieben, die Amazon Cognito anhand der Aktivität in Ihrem Benutzerpool aufruft.

Wenn Ihre App Benutzer über die Amazon-Cognito-Benutzerpool-API, die gehostete Benutzeroberfläche oder Benutzerpool-Endpunkte anmeldet, ruft Amazon Cognito Ihre Lambda-Funktionen auf der Grundlage des Sitzungskontexts auf. Weitere Informationen zur Amazon-Cognito-Benutzerpool-API und zu Benutzerpool-Endpunkten finden Sie unter Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte. In den Tabellen in den folgenden Abschnitten werden Ereignisse beschrieben, die Amazon Cognito veranlassen, eine Funktion aufzurufen, sowie die `triggerSource`-Zeichenfolge, die Amazon Cognito in die Anfrage einbezieht.

Themen

Lambda-Trigger in der Amazon-Cognito-API

In der folgenden Tabelle werden die Quellzeichenfolgen für die Lambda-Trigger beschrieben, die Amazon Cognito aufrufen kann, wenn Ihre App einen lokalen Benutzerpool-Benutzer erstellt, anmeldet oder aktualisiert.

Lokale Benutzer-Auslöser-Quellen in der Amazon-Cognito-API

API-Vorgang	Lambda-Trigger	Trigger-Quelle
AdminCreateUser	Voranmeldung	`PreSignUp_AdminCreateUser`
	Generierung von Pre-Token	`TokenGeneration_NewPasswordChallenge`
	Benutzerdefinierte Nachricht	`CustomMessage_AdminCreateUser`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_AdminCreateUser`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_AdminCreateUser`
SignUp	Voranmeldung	`PreSignUp_SignUp`
	Benutzerdefinierte Nachricht	`CustomMessage_SignUp`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_SignUp`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_SignUp`

API-Vorgang	Lambda-Trigger	Trigger-Quelle
ConfirmSignUp AdminConfirmSignUp	Nachbestätigung	`PostConfirmation_ConfirmSignUp`
InitiateAuth AdminInitiateAuth	Vorauthentifizierung	`PreAuthentication_Authentication`
	Authentifizierungsaufforderung definieren	`DefineAuthChallenge_Authentication`
	Authentifizierungsaufforderung erstellen	`CreateAuthChallenge_Authentication`

	Generierung von Pre-Token	`TokenGeneration_Authentication`
		
		`TokenGeneration_AuthenticateDevice`
		
		`TokenGeneration_RefreshTokens`

	Benutzer migrieren	`UserMigration_Authentication`
	Benutzerdefinierte Nachricht	`CustomMessage_Authentication`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_AccountTakeOverNotification`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_Authentication`
ForgotPassword	Benutzer migrieren	`UserMigration_ForgotPassword`

API-Vorgang	Lambda-Trigger	Trigger-Quelle
	Benutzerdefinierte Nachricht	`CustomMessage_ForgotPassword`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_ForgotPassword`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_ForgotPassword`
ConfirmForgotPassword	Nachbestätigung	`PostConfirmation_ConfirmForgotPassword`
UpdateUserAttributes AdminUpdateUserAttributes	Benutzerdefinierte Nachricht	`CustomMessage_UpdateUserAttribute`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_UpdateUserAttribute`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_UpdateUserAttribute`
VerifyUserAttributes	Benutzerdefinierte Nachricht	`CustomMessage_VerifyUserAttribute`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_VerifyUserAttribute`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_VerifyUserAttribute`

Lambda-Auslöser für lokale Amazon-Cognito-Benutzer in der gehosteten Benutzeroberfläche

In der folgenden Tabelle werden die Quellzeichenfolgen für die Lambda-Auslöser beschrieben, die Amazon Cognito aufrufen kann, wenn sich ein lokaler Benutzer mit der gehosteten Benutzeroberfläche bei Ihrem Benutzerpool anmeldet.

Auslöserquellen für lokale Benutzer in der gehosteten Benutzeroberfläche

Gehostete UI-URI	Lambda-Trigger	Trigger-Quelle
`/signup`	Voranmeldung	`PreSignUp_SignUp`
	Benutzerdefinierte Nachricht	`CustomMessage_SignUp`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_SignUp`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_SignUp`
`/confirmuser`	Nachbestätigung	`PostConfirmation_ConfirmSignUp`
`/login`	Vorauthentifizierung	`PreAuthentication_Authentication`
	Authentifizierungsaufforderung definieren	`DefineAuthChallenge_Authentication`
	Authentifizierungsaufforderung erstellen	`CreateAuthChallenge_Authentication`
	Generierung von Pre-Token	`TokenGeneration_Authentication` `TokenGeneration_AuthenticateDevice`

Gehostete UI-URI	Lambda-Trigger	Trigger-Quelle
		`TokenGeneration_RefreshTokens`
	Benutzer migrieren	`UserMigration_Authentication`
	Benutzerdefinierte Nachricht	`CustomMessage_Authentication`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_AccountTakeOverNotification`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_Authentication`
`/forgotpassword`	Benutzer migrieren	`UserMigration_ForgotPassword`
	Benutzerdefinierte Nachricht	`CustomMessage_ForgotPassword`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_ForgotPassword`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_ForgotPassword`
`/confirmforgotpassword`	Nachbestätigung	`PostConfirmation_ConfirmForgotPassword`

Lambda-Auslöser für Verbundbenutzer

Sie können die folgenden Lambda-Auslöser verwenden, um Ihre Benutzerpool-Workflows für Benutzer anzupassen, die sich bei einem Verbundanbieter anmelden.

> **Note**
>
> Verbundbenutzer können die von Amazon Cognito gehostete Benutzeroberfläche verwenden, um sich anzumelden, oder Sie können eine Anfrage an den Autorisieren des Endpunkts generieren, der sie im Hintergrund auf die Anmeldeseite ihres Identitätsanbieters weiterleitet. Sie können mit der Amazon-Cognito-Benutzerpool-API keine Verbundbenutzer anmelden.

Verbundbenutzern-Trigger-Quellen

Anmeldeereignisse	Lambda-Trigger	Trigger-Quelle
Erste Anmeldung	Voranmeldung	`PreSignUp_ExternalProvider`
	Nachbestätigung	`PostConfirmation_ConfirmSignUp`
	Generierung von Pre-Token	`TokenGeneration_HostedAuth`
Nachfolgende Anmeldungen	Vorauthentifizierung	`PreAuthentication_Authentication`
	Nachauthentifizierung	`PostAuthentication_Authentication`
	Generierung von Pre-Token	`TokenGeneration_HostedAuth`

Die Verbundanmeldung ruft keine Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen, Lambda-Auslöser für die Benutzermigration., Lambda-Auslöser für benutzerdefinierte Nachrichten oder Benutzerdefinierter Lambda-Auslöser für Sender in Ihrem Benutzerpool auf.

Verbinden von Lambda-Triggern mit funktionalen Benutzerpool-Vorgängen

Jeder Lambda-Trigger hat eine funktionale Rolle in Ihrem Benutzerpool. Ein Trigger kann beispielsweise Ihren Anmeldeablauf ändern oder eine benutzerdefinierte

Authentifizierungsherausforderung hinzufügen. Das Ereignis, das Amazon Cognito an eine Lambda-Funktion sendet, kann eine von mehreren Aktionen widerspiegeln, aus denen sich diese funktionale Rolle zusammensetzt. Amazon Cognito ruft beispielsweise einen Trigger vor der Registrierung auf, wenn sich Ihr Benutzer anmeldet und wenn Sie einen Benutzer erstellen. Jeder dieser verschiedenen Fälle für dieselbe funktionale Rolle hat seinen eigenen `triggerSource`-Wert. Ihre Lambda-Funktion kann eingehende Ereignisse je nach dem Vorgang, der sie aufgerufen hat, unterschiedlich verarbeiten.

Amazon Cognito ruft auch alle zugewiesenen Funktionen auf, wenn ein Ereignis einer Trigger-Quelle entspricht. Wenn sich ein Benutzer beispielsweise bei einem Benutzerpool anmeldet, dem Sie die Trigger „Benutzer migrieren“ und „Vorauthentifizierung“ zugewiesen haben, werden beide aktiviert.

Registrierungs-, Bestätigungs- und Anmeldungsauslöser (Authentifizierungsauslöser)

Auslöser	triggerSource-Wert	Veranstaltung
Voranmeldung	`PreSignUp_SignUp`	Voranmeldung.
Voranmeldung	`PreSignUp_AdminCreateUser`	Voranmeldung, wenn ein Administrator einen neuen Benutzer erstellt.
Voranmeldung	`PreSignUp_ExternalProvider`	Vorab-Registrierung für externe Identitätsanbieter.
Nachbestätigung	`PostConfirmation_ConfirmSignUp`	Nachbestätigung der Anmeldung.
Nachbestätigung	`PostConfirmation_ConfirmForgotPassword`	Nachbestätigung für vergessenes Passwort.
Vorauthentifizierung	`PreAuthentication_Authentication`	Vorauthentifizierung.
Nachauthentifizierung	`PostAuthentication_Authentication`	Nachauthentifizierung.

Auslöser für benutzerdefinierte Authentifizierungsaufforderungen

Auslöser	triggerSource-Wert	Veranstaltung
Authentifizierungsaufforderung definieren	`DefineAuthChallenge_Authentication`	Authentifizierungsaufforderung definieren.
Authentifizierungsaufforderung erstellen	`CreateAuthChallenge_Authentication`	Authentifizierungsaufforderung erstellen.
Überprüfung der Authentifizierungsaufforderung	`VerifyAuthChallengeResponse_Authentication`	Antwort auf Authentifizierungsaufforderung überprüfen.

Auslöser für die Generierung von Pre-Token

Auslöser	triggerSource-Wert	Veranstaltung
Generierung von Pre-Token	`TokenGeneration_HostedAuth`	Amazon Cognito authentifiziert den Benutzer über die Anmeldeseite der gehosteten Benutzeroberfläche.
Generierung von Pre-Token	`TokenGeneration_Authentication`	Abläufe der Benutzerauthentifizierung sind abgeschlossen.
Generierung von Pre-Token	`TokenGeneration_NewPasswordChallenge`	Admin erstellt den Benutzer. Amazon Cognito ruft diesen Parameter auf, wenn der Benutzer ein temporäres Passwort ändern muss.
Generierung von Pre-Token	`TokenGeneration_AuthenticateDevice`	Ende der Authentifizierung eines Benutzergeräts.
Generierung von Pre-Token	`TokenGeneration_RefreshTokens`	Der Benutzer versucht, die Identitäts- und Zugriffs-Token zu aktualisieren.

Migration von Benutzerauslösern

Auslöser	triggerSource-Wert	Veranstaltung
Benutzermigration	`UserMigration_Authentication`	Benutzermigration zum Zeitpunkt der Anmeldung.
Benutzermigration	`UserMigration_ForgotPassword`	Benutzermigration während des Ablaufs bei vergessenem Passwort.

Auslöser für benutzerdefinierte Nachrichten

Auslöser	triggerSource-Wert	Veranstaltung
Benutzerdefinierte Nachricht	`CustomMessage_SignUp`	Benutzerdefinierte Nachricht , wenn sich ein Benutzer in Ihrem Benutzerpool anmeldet.
Benutzerdefinierte Nachricht	`CustomMessage_AdminCreateUser`	Benutzerdefinierte Nachricht , wenn Sie einen Benutzer als Administrator erstellen und Amazon Cognito ihm ein temporäres Passwort sendet.
Benutzerdefinierte Nachricht	`CustomMessage_ResendCode`	Benutzerdefinierte Nachricht , wenn Ihr vorhandener Benutzer einen neuen Bestätigungscode anfordert.
Benutzerdefinierte Nachricht	`CustomMessage_ForgotPassword`	Benutzerdefinierte Nachricht , wenn Ihr Benutzer sein Passwort zurücksetzen möchte.
Benutzerdefinierte Nachricht	`CustomMessage_UpdateUserAttribute`	Benutzerdefinierte Nachricht , wenn ein Benutzer seine E-Mail-Adresse oder Telefonnummer ändert und Amazon

Auslöser	triggerSource-Wert	Veranstaltung
		Cognito einen Verifizierungscode sendet.
Benutzerdefinierte Nachricht	`CustomMessage_VerifyUserAttribute`	Benutzerdefinierte Nachricht, wenn ein Benutzer eine E-Mail-Adresse oder Telefonnummer hinzufügt und Amazon Cognito einen Verifizierungscode sendet.
Benutzerdefinierte Nachricht	`CustomMessage_Authentication`	Benutzerdefinierte Nachricht, wenn sich ein Benutzer anmeldet, der SMS MFA konfiguriert hat.

Lambda-Auslöser für die Vorab-Registrierung

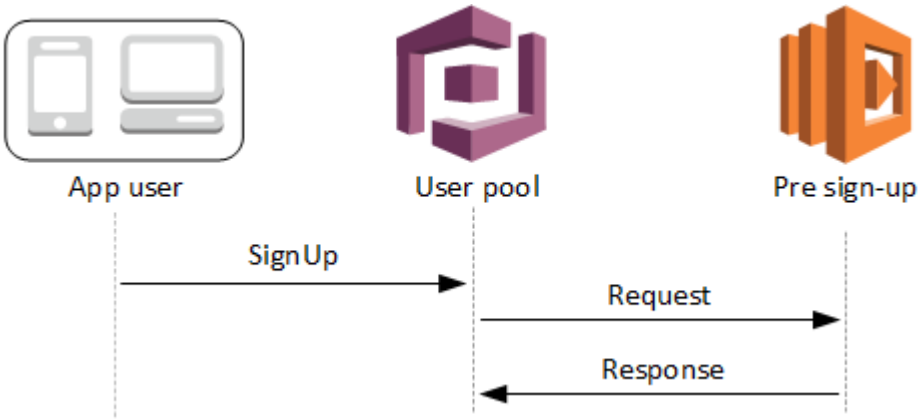
Kurz bevor Amazon Cognito einen neuen Benutzer registriert, wird die AWS Lambda-Funktion zur Vorab-Registrierung aktiviert. Im Rahmen des Anmeldeprozesses können Sie diese Funktion verwenden, um eine benutzerdefinierte Validierung durchzuführen und die Registrierungsanfrage basierend auf den Ergebnissen Ihrer Validierung zu akzeptieren oder abzulehnen.

Themen

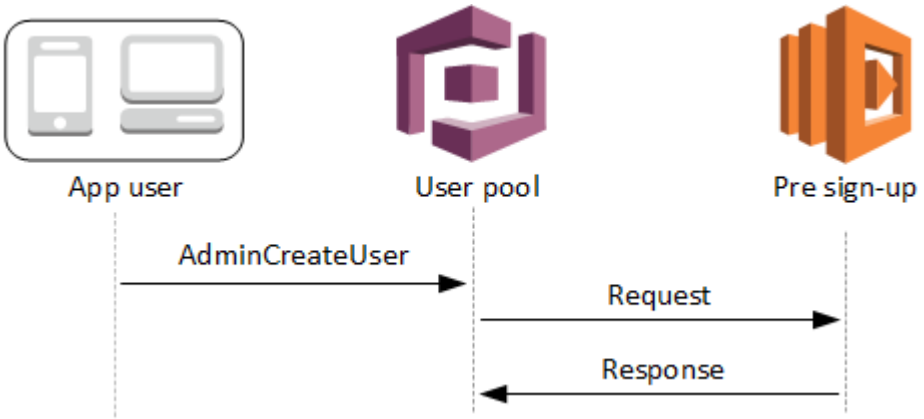
- Lambda-Abläufe für die Vorab-Registrierung
- Lambda-Auslöserparameter für die Vorab-Registrierung
- Tutorials für die Registrierung
- Beispiel für Vorab-Registrierung: Automatische Bestätigung von Benutzern aus einer registrierten Domäne
- Beispiel für Voranmeldung: Automatische Bestätigung und Verifizierung aller Benutzer
- Beispiel für die Vorab-Registrierung: Anmeldung ablehnen, wenn der Benutzername weniger als fünf Zeichen enthält

Lambda-Abläufe für die Vorab-Registrierung

Ablauf der Client-Registrierung



Ablauf der Server-Registrierung



Die Anforderung enthält Validierungsdaten vom Client. Diese Daten stammen aus den `ValidationData` Werten, die an den Benutzerpool SignUp und die AdminCreateUser API-Methoden übergeben wurden.

Lambda-Auslöserparameter für die Vorab-Registrierung

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
```

```
            . . .
        },
        "validationData": {
            "string": "string",
            . . .
         },
        "clientMetadata": {
            "string": "string",
            . . .
         }
    },

    "response": {
        "autoConfirmUser": "boolean",
        "autoVerifyPhone": "boolean",
        "autoVerifyEmail": "boolean"
    }
}
```

Anforderungsparameter für die Vorab-Registrierung

userAttributes

Ein oder mehrere Name-Wert-Paare, die Benutzerattribute darstellen. Die Attributnamen sind die Schlüssel.

validationData

Ein oder mehrere Schlüssel/Wert-Paare mit Benutzerattributdaten, die Ihre App in der Anfrage zur Erstellung eines neuen Benutzers an Amazon Cognito übergeben hat. Senden Sie diese Informationen an Ihre Lambda-Funktion im - ValidationData Parameter Ihrer - AdminCreateUser oder SignUp-API-Anfrage.

Amazon Cognito legt Ihre ValidationData Daten nicht als Attribute des Benutzers fest, den Sie erstellen. ValidationData ist temporäre Benutzerinformationen, die Sie für die Zwecke Ihres Lambda-Auslösers vor der Registrierung angeben.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion bereitstellen können, die Sie für den Auslöser für die Vorab-Registrierung angeben. Sie können diese Daten an Ihre Lambda-Funktion übergebenAdminCreateUser,

indem Sie den ClientMetadata Parameter in den folgenden API-Aktionen verwenden: AdminRespondToAuthChallenge, ForgotPassword, und SignUp.

Antwortparameter für die Vorab-Registrierung

In der Antwort können Sie `autoConfirmUser` auf `true` setzen, wenn der Benutzer automatisch bestätigt werden soll. Sie können die Einstellung `autoVerifyEmail` auf `true` setzen und somit die E-mail-Adresse des Benutzers automatisch überprüfen. Sie könne die Einstellung `autoVerifyPhone` auf `true` setzen und somit die Telefonnummer des Benutzers automatisch überprüfen.

> **Note**
>
> Die Antwortparameter `autoVerifyPhone`, `autoVerifyEmail` und `autoConfirmUser` werden von Amazon Cognito ignoriert, wenn die Lambda-Funktion vor der Registrierung durch die `AdminCreateUser`-API ausgelöst wird.

autoConfirmUser

Setzen Sie diesen Parameter auf `true`, wenn der Benutzer automatisch bestätigt werden soll. Setzen Sie ihn andernfalls auf `false`.

autoVerifyEmail

Legen Sie diesen Parameter auf `true` fest, um die Verifizierung der E-Mail-Adresse eines Benutzers, der sich gerade anmeldet, zu bestätigen. Legen Sie ihn andernfalls auf `false` fest. Falls `autoVerifyEmail` auf `true` gesetzt ist, muss das Attribut `email` einen gültigen Wert besitzen, bei dem es sich nicht um Null handeln darf. Andernfalls tritt ein Fehler auf und der Benutzer wird die Registrierung nicht abschließen können.

Wenn das Attribut `email` als Alias ausgewählt ist, wird ein Alias für die E-Mail-Adresse des Benutzers erstellt. Dabei muss `autoVerifyEmail` aktiv sein. Wenn bereits ein Alias mit dieser E-Mail-Adresse vorhanden ist, wird der Alias dem neuen Benutzer zugewiesen. Die E-Mail-Adresse des vorherigen Benutzers wird als nicht bestätigt gekennzeichnet. Weitere Informationen finden Sie unter Anpassen von Anmeldeattributen.

autoVerifyPhone

Setzen Sie diesen Parameter auf `true` um die Verifizierung der Telefonnummer eines Benutzers, der sich grade anmeldet, zu bestätigen. Setzen Sie ihn andernfalls auf `false`. Falls

`autoVerifyPhone` auf `true` gesetzt ist, muss das Attribut `phone_number` einen gültigen Wert besitzen, bei dem es sich nicht um Null handeln darf. Andernfalls tritt ein Fehler auf und der Benutzer wird die Registrierung nicht abschließen können.

Wenn das Attribut `phone_number` als Alias ausgewählt ist, wird ein Alias für die Telefonnummer des Benutzers erstellt. Dabei muss `autoVerifyPhone` aktiv sein. Wenn ein Alias mit dieser Telefonnummer bereits existiert, wird das Alias einem neuen Benutzer zugewiesen. Die Telefonnummer des vorherigen Benutzers wird dabei als nicht-verifiziert gekennzeichnet. Weitere Informationen finden Sie unter Anpassen von Anmeldeattributen.

Tutorials für die Registrierung

Die Lambda-Funktion für die Vorab-Registrierung wird ausgelöst, unmittelbar bevor ein neuer Benutzer registriert wird. Sehen Sie sich diese Amazon Cognito-Anmeldetutorials für JavaScript, Android und iOS an.

Plattform	Tutorial
JavaScript Identity SDK	Registrieren von Benutzern mit JavaScript
Android Identity SDK	Benutzer mit Android registrieren
iOS Identity SDK	Benutzer mit iOS registrieren

Beispiel für Vorab-Registrierung: Automatische Bestätigung von Benutzern aus einer registrierten Domäne

Mit dem Lambda-Auslöser für die Vorab-Registrierung können Sie eine benutzerdefinierte Logik hinzufügen, die neue Benutzer validiert, die sich für Ihren Benutzerpool registrieren. Dies ist ein JavaScript Beispielprogramm, das zeigt, wie ein neuer Benutzer registriert wird. Sie ruft im Zuge der Authentifizierung einen Lambda-Auslöser für die Vorab-Registrierung auf.

JavaScript

```
var attributeList = [];
var dataEmail = {
  Name: "email",
  Value: "...", // your email here
```

```
};
var dataPhoneNumber = {
  Name: "phone_number",
  Value: "...", // your phone number here with +country code and no delimiters in
 front
};

var dataEmailDomain = {
  Name: "custom:domain",
  Value: "example.com",
};
var attributeEmail = new AmazonCognitoIdentity.CognitoUserAttribute(dataEmail);
var attributePhoneNumber = new AmazonCognitoIdentity.CognitoUserAttribute(
  dataPhoneNumber
);
var attributeEmailDomain = new AmazonCognitoIdentity.CognitoUserAttribute(
  dataEmailDomain
);

attributeList.push(attributeEmail);
attributeList.push(attributePhoneNumber);
attributeList.push(attributeEmailDomain);

var cognitoUser;
userPool.signUp(
  "username",
  "password",
  attributeList,
  null,
  function (err, result) {
    if (err) {
      alert(err);
      return;
    }
    cognitoUser = result.user;
    console.log("user name is " + cognitoUser.getUsername());
  }
);
```

Dies ist ein Beispiel für einen Lambda-Auslöser, der unmittelbar vor der Registrierung beim Benutzerpool durch den Lambda-Auslöser für die Vorab-Registrierung aufgerufen wird. Sie verwendet ein benutzerdefiniertes Attribut custom:domain, um neue Benutzer von einer

bestimmten E-Mail-Domäne automatisch zu bestätigen. Alle neuen Benutzer, die sich nicht in der benutzerdefinierten Domäne befinden, werden dem Benutzerpool hinzugefügt, aber nicht automatisch bestätigt.

Node.js

```
exports.handler = (event, context, callback) => {
  // Set the user pool autoConfirmUser flag after validating the email domain
  event.response.autoConfirmUser = false;

  // Split the email address so we can compare domains
  var address = event.request.userAttributes.email.split("@");

  // This example uses a custom attribute "custom:domain"
  if (event.request.userAttributes.hasOwnProperty("custom:domain")) {
    if (event.request.userAttributes["custom:domain"] === address[1]) {
      event.response.autoConfirmUser = true;
    }
  }

  // Return to Amazon Cognito
  callback(null, event);
};
```

Python

```
def lambda_handler(event, context):
    # It sets the user pool autoConfirmUser flag after validating the email domain
    event['response']['autoConfirmUser'] = False

    # Split the email address so we can compare domains
    address = event['request']['userAttributes']['email'].split('@')

    # This example uses a custom attribute 'custom:domain'
    if 'custom:domain' in event['request']['userAttributes']:
        if event['request']['userAttributes']['custom:domain'] == address[1]:
            event['response']['autoConfirmUser'] = True

    # Return to Amazon Cognito
    return event
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
    "request": {
        "userAttributes": {
            "email": "testuser@example.com",
            "custom:domain": "example.com"
        }
    },
    "response": {}
}
```

Beispiel für Voranmeldung: Automatische Bestätigung und Verifizierung aller Benutzer

Dieses Beispiel bestätigt alle Benutzer und setzt die Attribute `email` und `phone_number` des Benutzers auf verifiziert, wenn das Attribut vorhanden ist. Wenn Aliasse unterstützt werden, werden für `phone_number` und `email` außerdem Aliasse erstellt, wenn die automatische Bestätigung festgelegt ist.

> **Note**
>
> Wenn bereits ein Alias mit derselben Telefonnummer vorhanden ist, wird der Alias dem neuen Benutzer zugewiesen. Die `phone_number` des vorherigen Benutzers wird als nicht bestätigt gekennzeichnet. Dies gilt auch für E-Mail-Adressen. Um dies zu verhindern, können Sie die Benutzerpool-ListUsers API verwenden, um festzustellen, ob bereits ein Benutzer vorhanden ist, der die Telefonnummer oder E-Mail-Adresse des neuen Benutzers als Alias verwendet.

Node.js

```
const handler = async (event) => {
  // Confirm the user
  event.response.autoConfirmUser = true;
```

```
  // Set the email as verified if it is in the request
  if (event.request.userAttributes.hasOwnProperty("email")) {
    event.response.autoVerifyEmail = true;
  }

  // Set the phone number as verified if it is in the request
  if (event.request.userAttributes.hasOwnProperty("phone_number")) {
    event.response.autoVerifyPhone = true;
  }

  return event;
};

export { handler };
```

Python

```
def lambda_handler(event, context):
    # Confirm the user
    event['response']['autoConfirmUser'] = True

    # Set the email as verified if it is in the request
    if 'email' in event['request']['userAttributes']:
        event['response']['autoVerifyEmail'] = True

    # Set the phone number as verified if it is in the request
    if 'phone_number' in event['request']['userAttributes']:
        event['response']['autoVerifyPhone'] = True

    # Return to Amazon Cognito
    return event
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
  "request": {
```

```
    "userAttributes": {
      "email": "user@example.com",
      "phone_number": "+12065550100"
    }
  },
  "response": {}
}
```

Beispiel für die Vorab-Registrierung: Anmeldung ablehnen, wenn der Benutzername weniger als fünf Zeichen enthält

In diesem Beispiel wird die Länge des Benutzernamens in einer Anmeldeanforderung überprüft. Das Beispiel gibt einen Fehler zurück, wenn der Benutzer einen Namen mit weniger als fünf Zeichen eingegeben hat.

Node.js

```
exports.handler = (event, context, callback) => {
    // Impose a condition that the minimum length of the username is 5 is imposed on
 all user pools.
    if (event.userName.length < 5) {
        var error = new Error("Cannot register users with username less than the
 minimum length of 5");
        // Return error to Amazon Cognito
        callback(error, event);
    }
    // Return to Amazon Cognito
    callback(null, event);
};
```

Python

```
def lambda_handler(event, context):
    if len(event['userName']) < 5:
        raise Exception("Cannot register users with username less than the minimum
 length of 5")
    # Return to Amazon Cognito
    return event
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
  "userName": "rroe",
  "response": {}
}
```

Lambda-Auslöser nach der Bestätigung

Amazon Cognito ruft diesen Trigger auf, nachdem ein angemeldeter Benutzer sein Benutzerkonto bestätigt hat. In Ihrer Lambda-Funktion nach der Bestätigung können Sie benutzerdefinierte Nachrichten senden oder benutzerdefinierte API-Anforderungen hinzufügen. Sie können beispielsweise ein externes System abfragen und dem Benutzer zusätzliche Attribute zuweisen. Amazon Cognito ruft diesen Trigger nur für Benutzer auf, die sich in Ihrem Benutzerpool anmelden, nicht für Benutzerkonten, die Sie mit Ihren Administrator-Anmeldeinformationen erstellen.

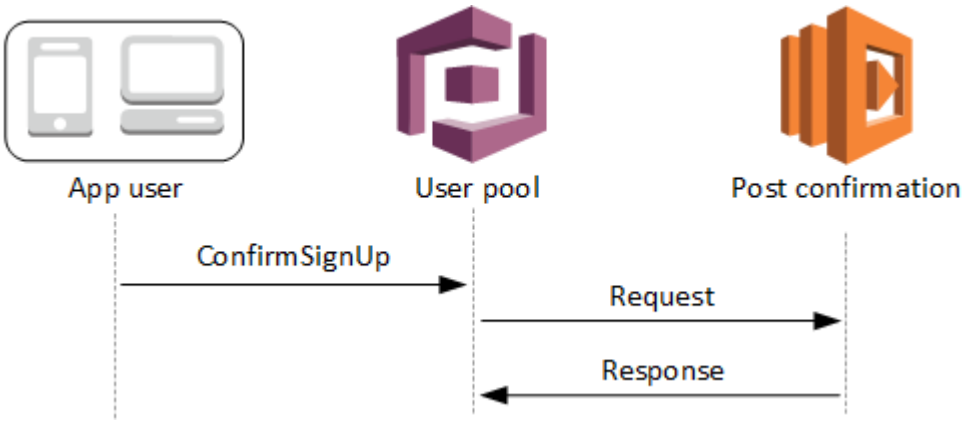
Die Anforderung enthält die aktuellen Attribute für den bestätigten Benutzer.

Themen

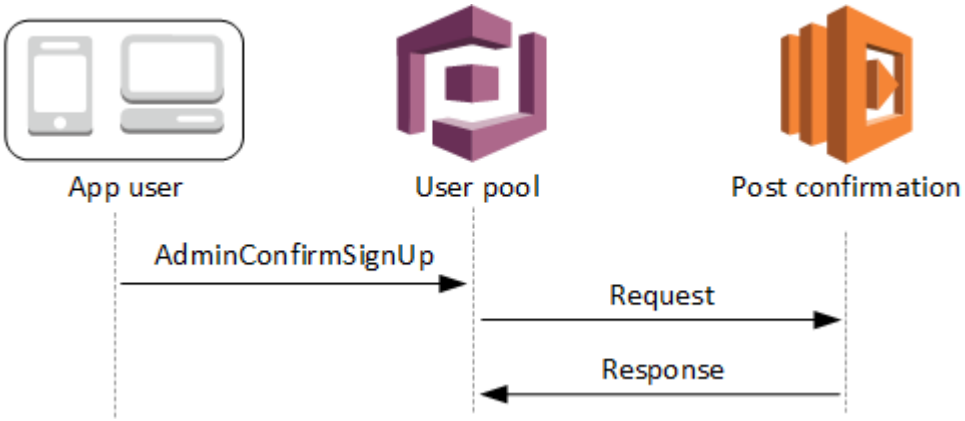
- Lambda-Abläufe nach der Bestätigung
- Lambda-Auslöserparameter nach der Bestätigung
- Tutorials für die Benutzerbestätigung
- Beispiel für „Nachbestätigung“

Lambda-Abläufe nach der Bestätigung

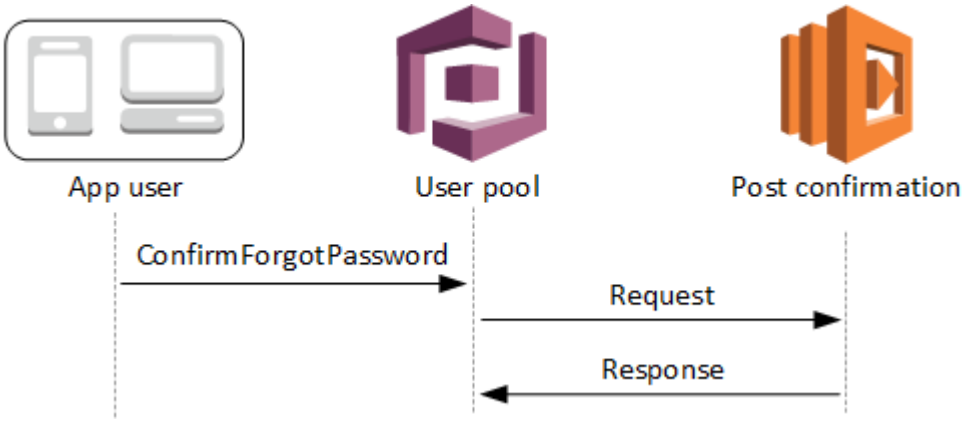
Ablauf der Bestätigung der Client-Registrierung



Ablauf der Bestätigung der Server-Registrierung



Ablauf der Bestätigung für vergessenes Passwort



Lambda-Auslöserparameter nach der Bestätigung

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
            "userAttributes": {
                "string": "string",
                . . .
            },
            "clientMetadata": {
             "string": "string",
              . . .
            }
        },
    "response": {}
}
```

Anforderungsparameter nach der Bestätigung

userAttributes

Ein oder mehrere Schlüssel-Wert-Paare, die Benutzerattribute darstellen.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion bereitstellen können, die Sie für den Auslöser nach der Bestätigung angeben. Sie können diese Daten an Ihre Lambda-Funktion übergeben, indem Sie den Parameter ClientMetadata in den folgenden API-Aktionen verwenden: AdminConfirmSignUp, ConfirmForgotPassword, ConfirmSignUp und SignUp.

Antwortparameter nach der Bestätigung

Erwartungsgemäß enthält die Antwort keine weiteren Informationen.

Tutorials für die Benutzerbestätigung

Die Lambda-Funktion für nach der Bestätigung wird ausgelöst, unmittelbar nachdem Amazon Cognito einen neuen Benutzer bestätigt. Weitere Informationen finden Sie in diesen Tutorials für die Benutzerbestätigung für JavaScript, Android und iOS.

Plattform	Tutorial
JavaScript Identity SDK	Benutzer mit JavaScript bestätigen
Android Identity SDK	Benutzer mit Android bestätigen
iOS Identity SDK	Benutzer mit iOS bestätigen

Beispiel für „Nachbestätigung“

Diese Lambda-Beispielfunktion sendet eine Bestätigungs-E-Mail-Nachricht an Ihre Benutzer unter Verwendung von Amazon SES. Weitere Informationen finden Sie im Entwicklerhandbuch für Amazon Simple Storage Service.

Node.js

```
// Import required AWS SDK clients and commands for Node.js. Note that this requires
// the `@aws-sdk/client-ses` module to be either bundled with this code or included
// as a Lambda layer.
import { SES, SendEmailCommand } from "@aws-sdk/client-ses";
const ses = new SES();

const handler = async (event) => {
  if (event.request.userAttributes.email) {
    await sendTheEmail(
      event.request.userAttributes.email,
      `Congratulations ${event.userName}, you have been confirmed.`
    );
  }
  return event;
};

const sendTheEmail = async (to, body) => {
  const eParams = {
    Destination: {
```

```
      ToAddresses: [to],
    },
    Message: {
      Body: {
        Text: {
          Data: body,
        },
      },
      Subject: {
        Data: "Cognito Identity Provider registration completed",
      },
    },
    // Replace source_email with your SES validated email address
    Source: "<source_email>",
  };
  try {
    await ses.send(new SendEmailCommand(eParams));
  } catch (err) {
    console.log(err);
  }
};

export { handler };
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
    "request": {
        "userAttributes": {
            "email": "user@example.com",
            "email_verified": true
        }
    },
    "response": {}
}
```

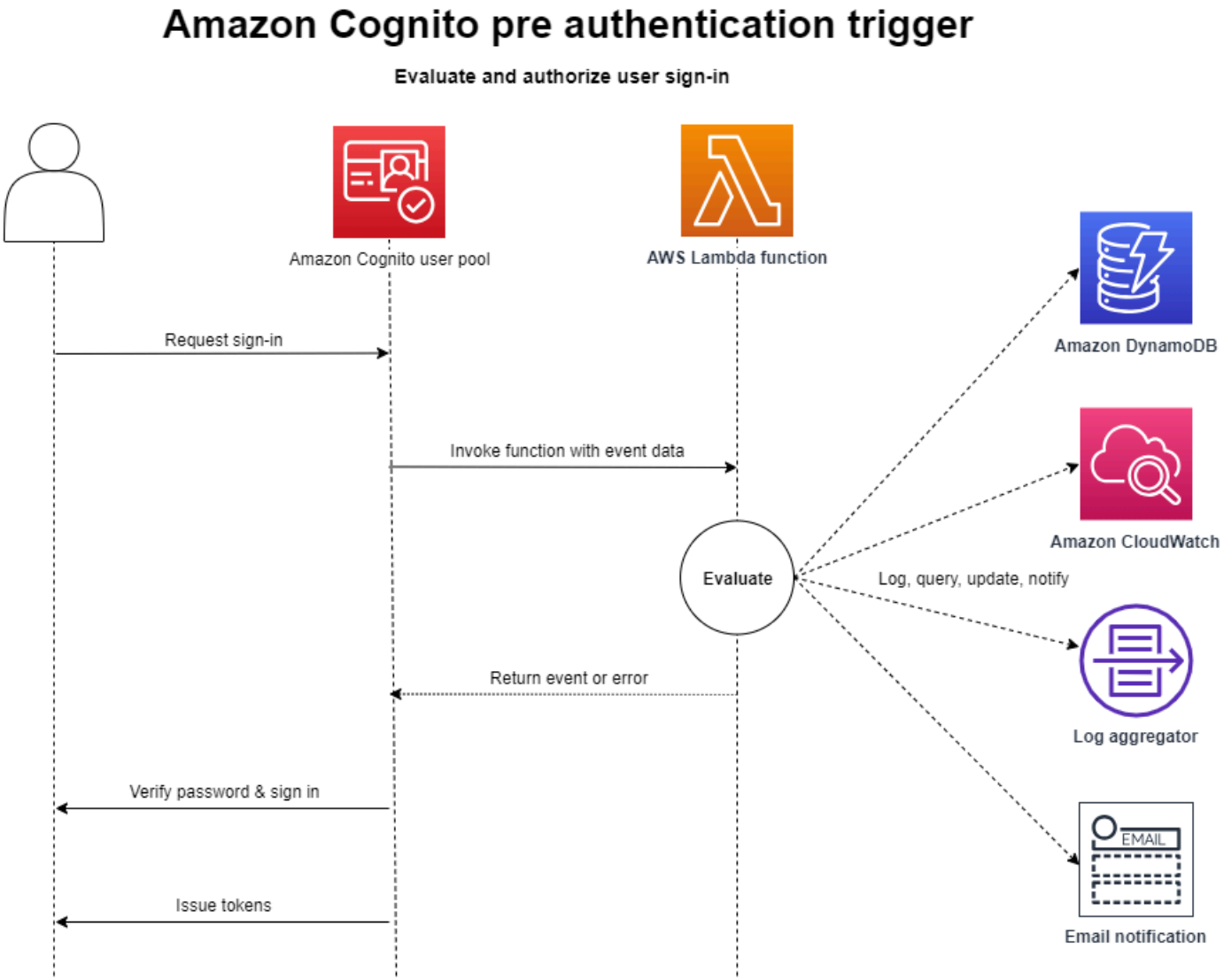
Lambda-Auslöser für die Vorab-Authentifizierung

Amazon Cognito ruft diesen Auslöser auf, wenn ein Benutzer sich anmelden möchte. Sie erhalten auf diese Weise die Möglichkeit, eine benutzerdefinierte Validierung zu erstellen, die vorbereitende Aktionen durchführt. Beispielsweise können Sie die Authentifizierungsanfrage ablehnen oder Sitzungsdaten für ein externes System aufzeichnen.

> **Note**
>
> Dieser Lambda-Trigger wird nicht aktiviert, wenn ein Benutzer nicht existiert oder bereits eine Sitzung in Ihrem Benutzerpool hat. Wenn die `PreventUserExistenceErrors`-Einstellung eines Benutzerpool-App-Clients auf `ENABLED` gesetzt ist, wird der Lambda-Trigger aktiviert.

Themen

Authentifizierungsprozess – Übersicht



Die Anfrage enthält Validierungsdaten des Clients, die aus den `ClientMetadata`-Werten stammen, die Ihre App an die Benutzerpool-API-Vorgänge `InitiateAuth` und `AdminInitiateAuth` übergibt.

Weitere Informationen finden Sie unter Ablauf der Authentifizierung in Benutzerpools.

Lambda-Auslöseparameter für die Vorab-Authentifizierung

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "validationData": {
            "string": "string",
            . . .
        },
        "userNotFound": boolean
    },
    "response": {}
}
```

Anforderungsparameter für die Vorab-Authentifizierung

userAttributes

Ein oder mehrere Name-Wert-Paare, die Benutzerattribute darstellen.

userNotFound

Amazon Cognito gibt diesen booleschen Wert ein, wenn Sie `PreventUserExistenceErrors` für Ihren Benutzerpool-Client auf `ENABLED` festgelegt haben.

validationData

Ein oder mehrere Schlüssel-Wert-Paare, die die Validierungsdaten in der Anmeldeanforderung des Benutzers enthalten. Um diese Daten an Ihre Lambda-Funktion zu übergeben, verwenden Sie den Parameter ClientMetadata in den API-Aktionen InitiateAuth und AdminInitiateAuth.

Antwortparameter für die Vorab-Authentifizierung

Amazon Cognito erwartet keine zusätzlichen Rückgabeinformationen in der Antwort. Ihre Funktion kann einen Fehler anzeigen, um den Anmeldeversuch abzulehnen oder API-Operationen zum Abfragen und Ändern Ihrer Ressourcen zu verwenden.

Beispiel für „Vorauthentifizierung“

Diese Beispielfunktion verhindert, dass sich Benutzer von einem bestimmten App-Client in Ihrem Benutzerpool anmelden. Da die Lambda-Funktion vor der Authentifizierung nicht aufgerufen wird, wenn Ihr Benutzer eine bestehende Sitzung hat, verhindert diese Funktion nur neue Sitzungen mit der App-Client-ID, die Sie blockieren möchten.

Node.js

```
const handler = async (event) => {
  if (
    event.callerContext.clientId === "user-pool-app-client-id-to-be-blocked"
  ) {
    throw new Error("Cannot authenticate users from this user pool app client");
  }

  return event;
};

export { handler };
```

Python

```
def lambda_handler(event, context):
    if event['callerContext']['clientId'] == "<user pool app client id to be
 blocked>":
        raise Exception("Cannot authenticate users from this user pool app client")

    # Return to Amazon Cognito
    return event
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
    "callerContext": {
```

```
        "clientId": "<user pool app client id to be blocked>"
    },
    "response": {}
}
```

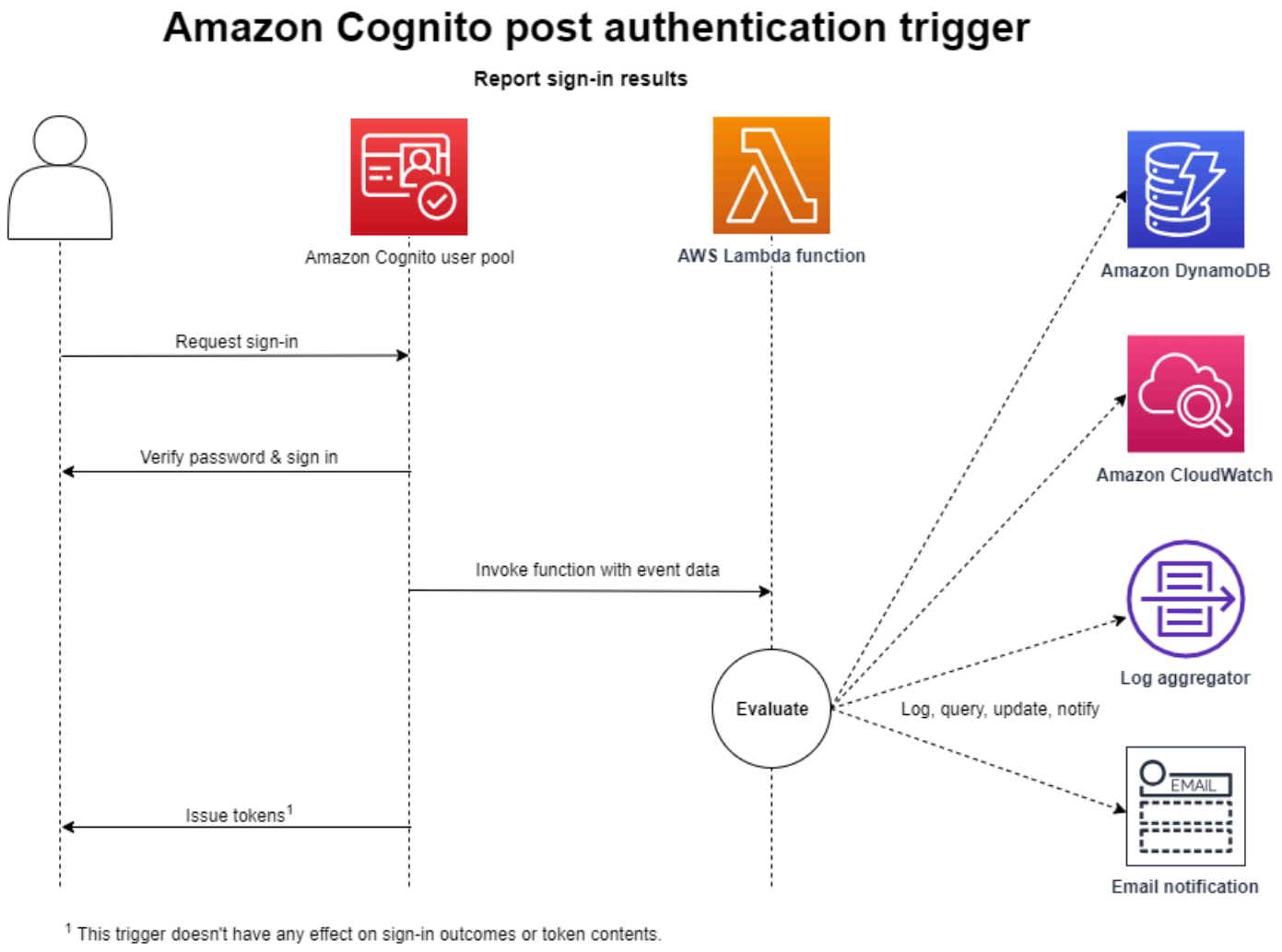
Lambda-Auslöser nach der Authentifizierung

Amazon Cognito ruft diesen Auslöser nach der Anmeldung eines Benutzers auf, sodass Sie benutzerdefinierte Logik für nach der Authentifizierung des Benutzers durch Amazon Cognito hinzufügen können.

Themen

- Authentifizierungsprozess – Übersicht
- Lambda-Auslöseparameter nach der Authentifizierung
- Tutorials für die Authentifizierung
- Beispiel für „Nachauthentifizierung“

Authentifizierungsprozess – Übersicht



Weitere Informationen finden Sie unter Ablauf der Authentifizierung in Benutzerpools.

Lambda-Auslöseparameter nach der Authentifizierung

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "userAttributes": {
```

```
            "string": "string",
            . . .
        },
        "newDeviceUsed": boolean,
        "clientMetadata": {
            "string": "string",
            . . .
           }
       },
    "response": {}
}
```

Anforderungsparameter nach der Authentifizierung

newDeviceUsed

Dieses Flag zeigt an, ob Sich der Benutzer an einem neuen Gerät angemeldet hat. Amazon Cognito setzt dieses Flag nur, wenn der Wert des Benutzerpools für gespeicherte Geräte auf `Always` oder `User Opt-In` gesetzt ist.

userAttributes

Ein oder mehrere Name-Wert-Paare, die Benutzerattribute darstellen.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion bereitstellen können, die Sie für den Auslöser nach der Authentifizierung angeben. Sie können den ClientMetadata-Parameter in den API-Aktionen AdminRespondToAuthChallenge und RespondToAuthChallenge verwenden, um diese Daten an Ihre Lambda-Funktion zu übergeben. Amazon Cognito enthält keine Daten aus dem ClientMetadata-Parameter in AdminInitiateAuth und InitiateAuth-API-Operationen in der Anforderung, die es an die Funktion nach der Authentifizierung übergibt.

Antwortparameter nach der Authentifizierung

Amazon Cognito erwartet keine zusätzlichen Rückgabeinformationen in der Antwort. Ihre Funktion kann API-Operationen verwenden, um Ihre Ressourcen abzufragen und zu ändern oder Ereignismetadaten in einem externen System aufzuzeichnen.

Tutorials für die Authentifizierung

Unmittelbar nachdem Amazon Cognito einen Benutzer anmeldet, aktiviert es die Lambda-Funktion nach der Authentifizierung. Weitere Informationen finden Sie in diesen Tutorials für die Anmeldung für JavaScript, Android und iOS.

Plattform	Tutorial
JavaScript Identity SDK	Benutzer mit JavaScript anmelden
Android Identity SDK	Benutzer mit Android anmelden
iOS Identity SDK	Benutzer mit iOS anmelden

Beispiel für „Nachauthentifizierung“

Dieser Lambda-Beispielfunktion für nach der Authentifizierung sendet Daten aus einer erfolgreichen Anmeldung an CloudWatch Logs.

Node.js

```
const handler = async (event) => {
  // Send post authentication data to Amazon CloudWatch logs
  console.log("Authentication successful");
  console.log("Trigger function =", event.triggerSource);
  console.log("User pool = ", event.userPoolId);
  console.log("App client ID = ", event.callerContext.clientId);
  console.log("User ID = ", event.userName);

  return event;
};

export { handler }
```

Python

```
import os
def lambda_handler(event, context):

    # Send post authentication data to Cloudwatch logs
```

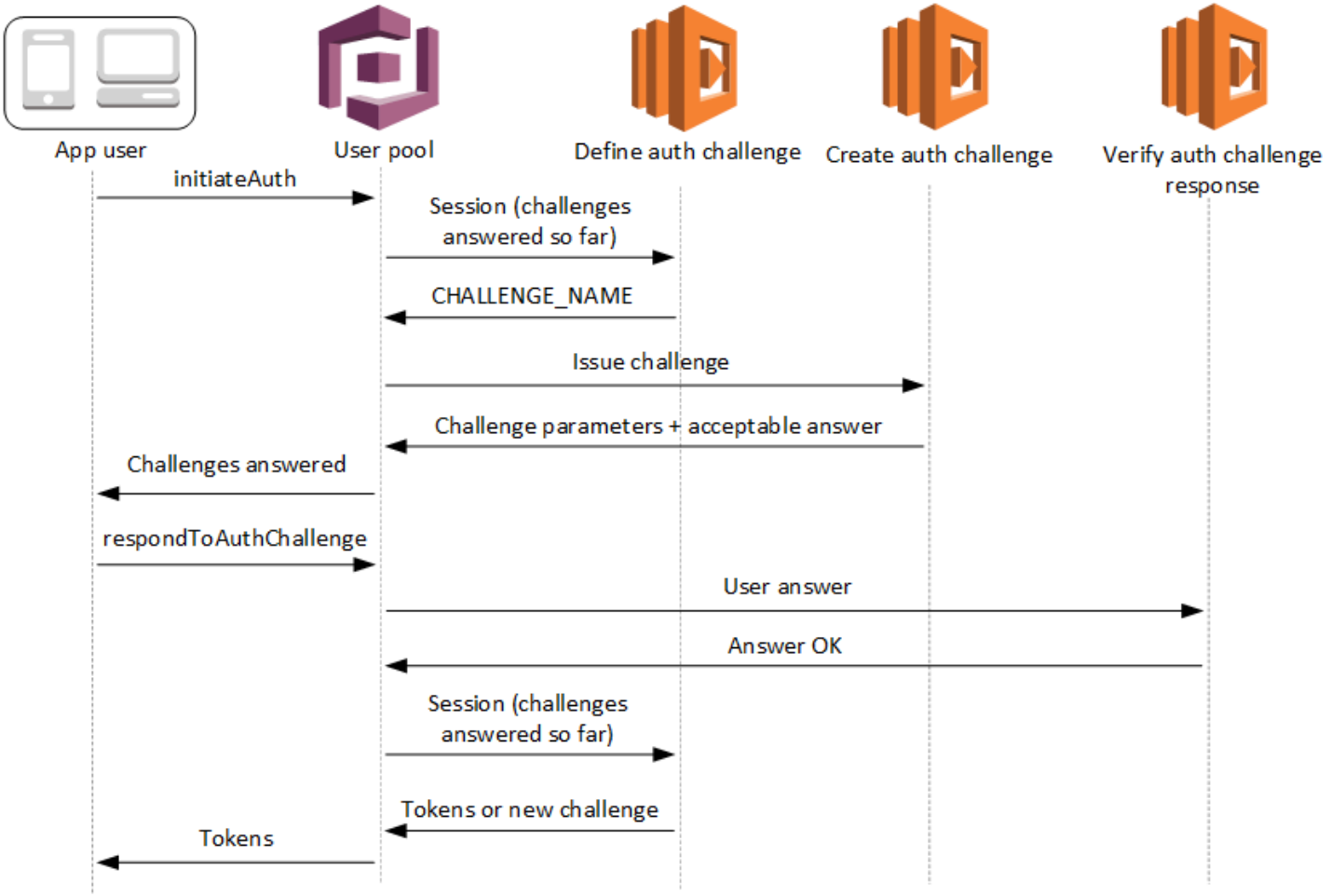
```
    print ("Authentication successful")
    print ("Trigger function =", event['triggerSource'])
    print ("User pool = ", event['userPoolId'])
    print ("App client ID = ", event['callerContext']['clientId'])
    print ("User ID = ", event['userName'])

    # Return to Amazon Cognito
    return event
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
  "triggerSource": "testTrigger",
  "userPoolId": "testPool",
  "userName": "testName",
  "callerContext": {
      "clientId": "12345"
  },
  "response": {}
}
```

Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen



Diese Lambda-Auslöser erstellen und verifizieren ihre eigenen Aufforderungen im Rahmen eines benutzerdefinierten Authentifizierungsablaufs für den Benutzerpool.

Authentifizierungsaufforderung definieren

Amazon Cognito ruft diesen Auslöser auf, um den benutzerdefinierten Authentifizierungsablauf zu initiieren.

Authentifizierungsaufforderung erstellen

Amazon Cognito aktiviert diesen Auslöser nach Authentifizierungsaufforderung definieren, um eine benutzerdefinierte Aufforderung zu erstellen.

Antwort auf Authentifizierungsaufforderung überprüfen

Amazon Cognito ruft diesen Auslöser auf, um zu überprüfen, ob die Antwort des Endbenutzers auf eine benutzerdefinierte Aufforderung gültig ist.

Mit diesen Lambda-Auslösern für Aufforderungen können Sie neue Aufforderungstypen einbinden. Beispielsweise könnten diese Aufforderungstypen CAPTCHAs oder dynamische Aufforderungsfragen enthalten.

Mit den API-Methoden InitiateAuth und RespondToAuthChallenge können Sie die Authentifizierung in zwei allgemeine Schritte verallgemeinern.

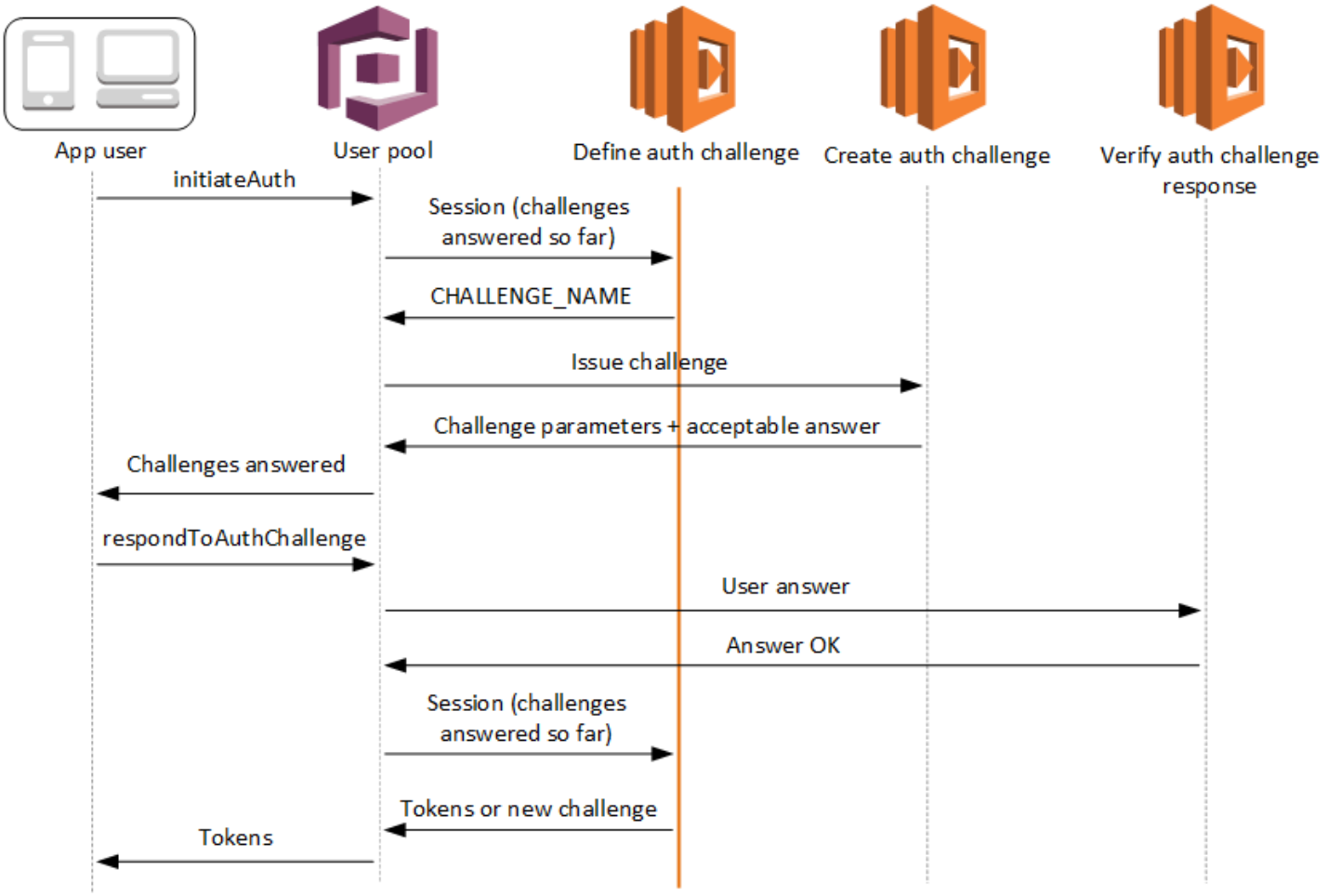
In diesem Ablauf führt ein Benutzer die Authentifizierung durch Beantwortung aufeinanderfolgender Aufforderungen durch, bis die Authentifizierung entweder fehlschlägt oder Token für den Benutzer ausgestellt werden. Diese beiden API-Aufrufe können wiederholt werden, um unterschiedliche Aufforderungen zu integrieren.

> **Note**
>
> Die gehostete Benutzeroberfläche von Amazon Cognito unterstützt keine benutzerdefinierte Authentifizierung mit benutzerdefinierten Authentifizierungs-Challenge-Lambda-Triggern.

Themen

- Lambda-Auslöser für die Definition einer Authentifizierungsaufforderung
- Lambda-Auslöser für die Erstellung einer Authentifizierungsaufforderung
- Lambda-Auslöser für die Verifizierung der Antwort auf eine Authentifizierungsaufforderung

Lambda-Auslöser für die Definition einer Authentifizierungsaufforderung



Authentifizierungsaufforderung definieren

> Amazon Cognito ruft diesen Auslöser auf, um den benutzerdefinierten Authentifizierungsablauf zu initiieren.

Die Anforderung für diesen Lambda-Auslöser enthält `session`. Der `session`-Parameter ist ein Array, das alle Aufforderungen enthält, die dem Benutzer im aktuellen Authentifizierungsprozess präsentiert werden. Die Anfrage enthält auch das entsprechende Ergebnis. Das `session`-Array speichert Aufforderungsdetails (`ChallengeResult`) in chronologischer Reihenfolge. Die Aufforderung `session[0]` stellt die erste Aufforderung dar, die der Benutzer erhält.

Sie können Amazon-Cognito-Benutzerpasswörter überprüfen lassen, bevor es Ihre benutzerdefinierten Herausforderungen ausgibt. Alle Lambda-Trigger, die in der Authentifizierungskategorie Ressourcen- und Anforderungskontingente zugeordnet sind, werden

ausgeführt, wenn Sie eine SRP-Authentifizierung in einem benutzerdefinierten Challenge-Flow durchführen. Es folgt eine Übersicht über den Prozess:

1. Ihre App initiiert die Anmeldung, indem `InitiateAuth` oder `AdminInitiateAuth` mit der `AuthParameters`-Karte aufgerufen werden. Parameter müssen `CHALLENGE_NAME: SRP_A,` und Werte für `SRP_A` und `USERNAME` umfassen.
2. Amazon Cognito ruft Ihren Lambda-Auslöser „Authentifizierungsaufforderung definieren“ mit einer ersten Sitzung auf, die `challengeName: SRP_A` und `challengeResult: true` enthält.
3. Nachdem Sie diese Eingaben empfangen haben, reagiert Ihre Lambda-Funktion mit `challengeName: PASSWORD_VERIFIER`, `issueTokens: false`, `failAuthentication: false`.
4. Wenn die Kennwortverifizierung erfolgreich ist, ruft Amazon Cognito Ihre Lambda-Funktion erneut mit einer neuen Sitzung auf, die `challengeName: PASSWORD_VERIFIER` und `challengeResult: true` enthält.
5. Ihre Lambda-Funktion initiiert Ihre benutzerdefinierten Aufforderungen, indem sie mit `challengeName: CUSTOM_CHALLENGE`, `issueTokens: false` und `failAuthentication: false` antwortet. Wenn Sie Ihren benutzerdefinierten Authentifizierungsablauf nicht mit Passwortüberprüfung starten möchten, können Sie die Anmeldung mit der `AuthParameters`-Karte einschließlich `CHALLENGE_NAME: CUSTOM_CHALLENGE` initiieren.
6. Die Aufforderungsschleife wird so lange ausgeführt, bis alle Aufforderungen beantwortet sind.

Themen

- Definition von Lambda-Auslöserparametern für die Authentifizierungsaufforderung
- Beispiel für „Authentifizierungsaufforderung definieren“

Definition von Lambda-Auslöserparametern für die Authentifizierungsaufforderung

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
```

```
                . . .
        },
        "session": [
            ChallengeResult,
            . . .
        ],
        "clientMetadata": {
            "string": "string",
            . . .
        },
        "userNotFound": boolean
    },
    "response": {
        "challengeName": "string",
        "issueTokens": boolean,
        "failAuthentication": boolean
    }
}
```

Anforderungsparameter für die Definition der Authentifizierungsaufforderung

Wenn Amazon Cognito Ihre Lambda-Funktion aufruft, bietet Amazon Cognito die folgenden Parameter:

userAttributes

Ein oder mehrere Name-Wert-Paare, die Benutzerattribute darstellen.

userNotFound

Ein boolescher Wert, den Amazon Cognito eingibt, wenn `PreventUserExistenceErrors` für Ihren Benutzerpool-Client auf `ENABLED` festgelegt ist. Ein Wert von `true` bedeutet, dass die Benutzer-ID (Benutzername, E-Mail-Adresse usw.) mit keinem vorhandenen Benutzer übereinstimmt. Wenn `PreventUserExistenceErrors` auf `ENABLED` gesetzt ist, informiert der Service die App nicht über nicht existierende Benutzer. Wir empfehlen, dass Ihre Lambda-Funktionen dieselbe Benutzererfahrung beibehalten und die Latenz berücksichtigen. Auf diese Weise kann der Aufrufer abweichendes Verhalten nicht erkennen, wenn der Benutzer vorhanden ist oder nicht vorhanden ist.

Sitzung

Ein Array von `ChallengeResult`-Elementen. Jede enthält die folgenden Elemente:

challengeName

Einer der folgenden Aufforderungstypen: `CUSTOM_CHALLENGE`, `SRP_A`, `PASSWORD_VERIFIER`, `SMS_MFA`, `DEVICE_SRP_AUTH`, `DEVICE_PASSWORD_VERIFIER` oder `ADMIN_NO_SRP_AUTH`.

Wenn Ihre Define Auth-Challenge-Funktion eine `PASSWORD_VERIFIER`-Challenge für einen Benutzer ausgibt, der die Multifaktor-Authentifizierung eingerichtet hat, folgt Amazon Cognito mit einer `SMS_MFA`-Challenge. Fügen Sie in Ihrer Funktion die Behandlung von Eingabeereignissen aus `SMS_MFA`-Challenges hinzu. Sie müssen die `SMS_MFA`-Challenge nicht über Ihre Define-Auth-Challenge-Funktion aufrufen.

> **Important**
>
> Wenn Ihre Funktion feststellt, ob ein Benutzer sich erfolgreich authentifiziert hat und Sie Token ausgeben müssen, überprüfen Sie immer den `challengeName` in Ihrer Define-Auth-Challenge, um sicherzustellen, dass er mit dem erwarteten Wert übereinstimmt.

challengeResult

Setzen Sie diesen Parameter auf `true`, wenn der Benutzer die Aufforderung erfolgreich abgeschlossen hat. Setzen Sie ihn andernfalls auf `false`.

challengeMetadata

Ihr Name für die benutzerdefinierte Aufforderung. Nur verwendet, wenn `challengeName` `CUSTOM_CHALLENGE`entspricht.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion bereitstellen können, die Sie für den Auslöser für die Authentifizierungsaufforderung definieren. Um diese Daten an Ihre Lambda-Funktion zu übergeben, können Sie den `ClientMetadata`-Parameter in den API-Operationen AdminRespondToAuthChallenge und RespondToAuthChallenge verwenden. Die Anforderung, die die Funktion „Authentifizierungsaufforderung definieren“ aufruft, enthält keine Daten, die im ClientMetadata-Parameter in AdminInitiateAuth und InitiateAuth-API-Operationen übergeben wurden.

Antwortparameter für die Definition der Authentifizierungsaufforderung

In der Antwort können Sie die nächste Phase des Authentifizierungsvorgangs zurückgeben.

challengeName

Eine Zeichenfolge, die den Namen der nächsten Aufforderung enthält. Wenn dem Benutzer eine neue Aufforderung angezeigt werden soll, geben Sie den Namen der Aufforderung hier an.

issueTokens

Wenn Sie feststellen, dass der Benutzer den Authentifizierungsaufforderungen ausreichend nachgekommen ist, setzen Sie ihn auf `true`. Wenn der Benutzer den Aufforderungen nicht ausreichend nachgekommen ist, setzen Sie ihn auf `false`.

failAuthentication

Wenn Sie den aktuellen Authentifizierungsprozess beenden möchten, setzen Sie ihn auf `true`. Um den aktuellen Authentifizierungsprozess fortzusetzen, setzen Sie ihn auf `false`.

Beispiel für „Authentifizierungsaufforderung definieren“

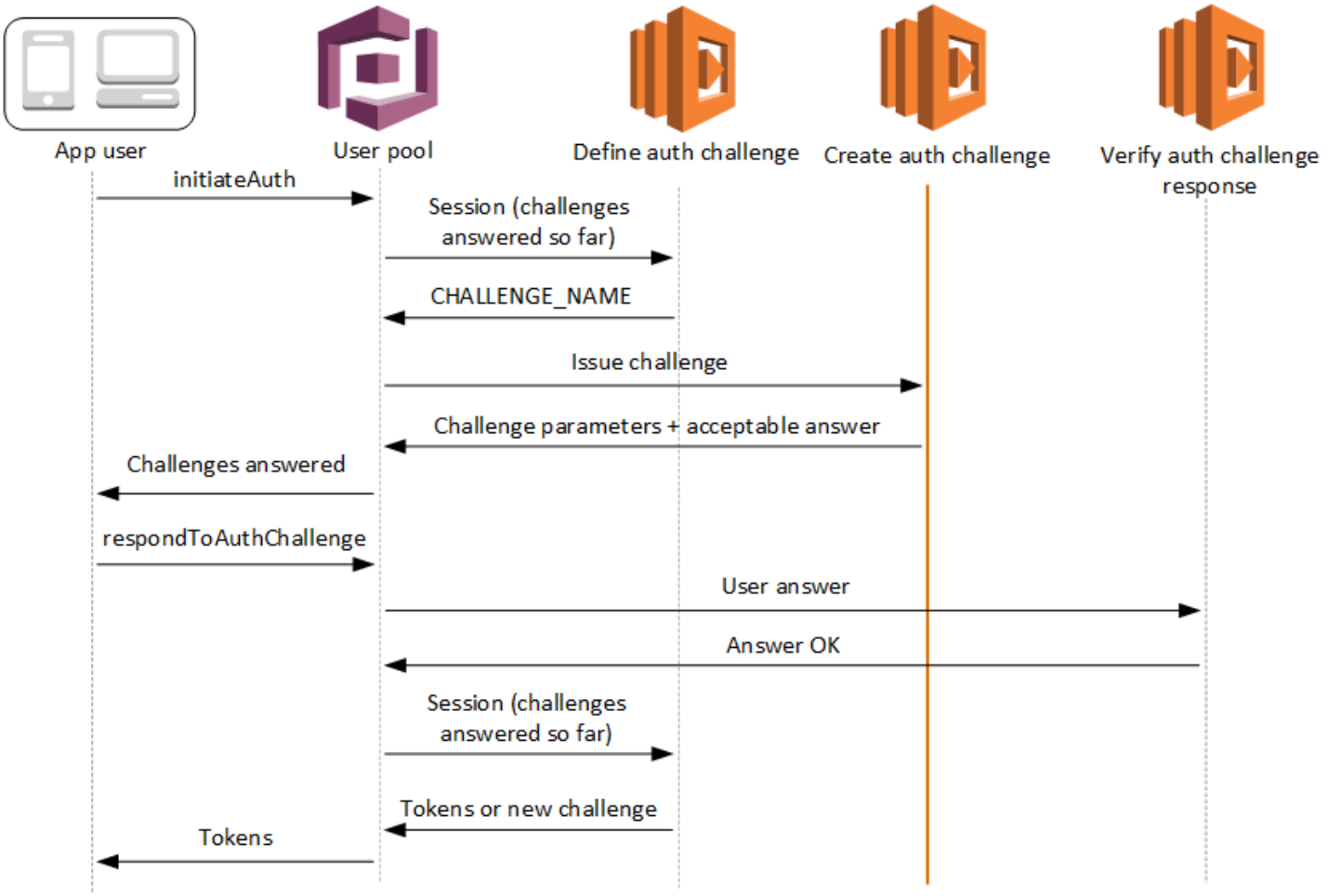
Mit diesem Beispiel werden eine Reihe von Aufforderungen für die Authentifizierung definiert. Token werden nur dann ausgegeben, wenn der Benutzer alle Aufforderungen erfolgreich abgeschlossen hat.

Node.js

```
const handler = async (event) => {
  if (
    event.request.session.length == 1 &&
    event.request.session[0].challengeName == "SRP_A"
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "PASSWORD_VERIFIER";
  } else if (
    event.request.session.length == 2 &&
    event.request.session[1].challengeName == "PASSWORD_VERIFIER" &&
    event.request.session[1].challengeResult == true
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
```

```
    event.response.challengeName = "CUSTOM_CHALLENGE";
  } else if (
    event.request.session.length == 3 &&
    event.request.session[2].challengeName == "CUSTOM_CHALLENGE" &&
    event.request.session[2].challengeResult == true
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "CUSTOM_CHALLENGE";
  } else if (
    event.request.session.length == 4 &&
    event.request.session[3].challengeName == "CUSTOM_CHALLENGE" &&
    event.request.session[3].challengeResult == true
  ) {
    event.response.issueTokens = true;
    event.response.failAuthentication = false;
  } else {
    event.response.issueTokens = false;
    event.response.failAuthentication = true;
  }

  return event;
};

export { handler }
```

Lambda-Auslöser für die Erstellung einer Authentifizierungsaufforderung



Authentifizierungsaufforderung erstellen

> Amazon Cognito aktiviert den Auslöser nach Authentifizierungsaufforderung definieren, falls eine benutzerdefinierte Anforderung als Teil des Auslösers für Authentifizierungsaufforderung definieren festgelegt wurde. Erstellt einen benutzerdefinierten Authentifizierungsablauf.

Dieser Lambda-Auslöser wird aufgerufen, um eine Aufforderung zu erstellen, die dem Benutzer angezeigt wird. Die Anforderung für diesen Lambda-Auslöser umfasst `challengeName` und `session`. `challengeName` ist eine Zeichenfolge und der Name der nächsten Aufforderung für den Benutzer. Der Wert dieses Attributs wird im Lambda-Auslöser „Authentifizierungsaufforderung definieren“ festgelegt.

Die Aufforderungsschleife wird so lange ausgeführt, bis alle Aufforderungen beantwortet werden.

Themen

- Lambda-Auslöserparameter für die Erstellung einer Authentifizierungsaufforderung
- Beispiel für „Authentifizierungsaufforderung erstellen“

Lambda-Auslöserparameter für die Erstellung einer Authentifizierungsaufforderung

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "challengeName": "string",
        "session": [
            ChallengeResult,
            . . .
        ],
        "clientMetadata": {
            "string": "string",
            . . .
        },
        "userNotFound": boolean
    },
    "response": {
        "publicChallengeParameters": {
            "string": "string",
            . . .
        },
        "privateChallengeParameters": {
            "string": "string",
            . . .
        },
        "challengeMetadata": "string"
    }
}
```

Anforderungsparameter für die Erstellung einer Authentifizierungsaufforderung

userAttributes

Ein oder mehrere Name-Wert-Paare, die Benutzerattribute darstellen.

userNotFound

Dieser boolesche Wert wird eingegeben, wenn `PreventUserExistenceErrors` für Ihren Benutzerpool-Client auf `ENABLED` festgelegt ist.

challengeName

Der Name der neuen Aufforderung.

Sitzung

Das session-Element ist ein Array aus `ChallengeResult`-Elementen, die jeweils die folgenden Elemente enthalten:

challengeName

Der Aufforderungstyp: Eins von: `"CUSTOM_CHALLENGE"`, `"PASSWORD_VERIFIER"`, `"SMS_MFA"`, `"DEVICE_SRP_AUTH"`, `"DEVICE_PASSWORD_VERIFIER"` oder von `"ADMIN_NO_SRP_AUTH"`.

challengeResult

Setzen Sie diesen Parameter auf `true`, wenn der Benutzer die Aufforderung erfolgreich abgeschlossen hat. Setzen Sie ihn andernfalls auf `false`.

challengeMetadata

Ihr Name für die benutzerdefinierte Aufforderung. Nur verwendet, wenn `challengeName` `"CUSTOM_CHALLENGE"`entspricht.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion bereitstellen können, die Sie für den Auslöser zum Erstellen einer Authentifizierungsaufforderung angeben. Sie können den ClientMetadata-Parameter in den API-Aktionen AdminRespondToAuthChallenge und RespondToAuthChallenge verwenden, um diese Daten an Ihre Lambda-Funktion zu übergeben. Die Anforderung, die die Funktion „Auth Challenge erstellen“ aufruft, enthält keine Daten, die im ClientMetadata-Parameter in AdminInitiateAuth und InitiateAuth-API-Operationen übergeben wurden.

Antwortparameter für die Erstellung einer Authentifizierungsaufforderung

publicChallengeParameters

Ein oder mehrere Schlüssel-Wert-Paare für die Client-App zur Verwendung in der Aufforderung, die dem Benutzer angezeigt wird. Dieser Parameter sollte alle erforderlichen Informationen enthalten, um dem Benutzer die Aufforderung richtig anzuzeigen.

privateChallengeParameters

Dieser Parameter wird nur vom Lambda-Auslöser „Antwort auf Authentifizierungsaufforderung überprüfen“ verwendet. Dieser Parameter sollte enthalten alle Informationen, die erforderlich ist, um sicherzustellen, dass die Benutzer die Antwort auf die Aufforderung. Anders ausgedrückt, der Parameter `publicChallengeParameters` enthält die Frage, die dem Benutzer angezeigt wird, und `privateChallengeParameters` die gültigen Antworten auf die Frage.

challengeMetadata

Ihr Name für die benutzerdefinierte Aufforderung, sofern es sich um eine benutzerdefinierte Aufforderung handelt.

Beispiel für „Authentifizierungsaufforderung erstellen“

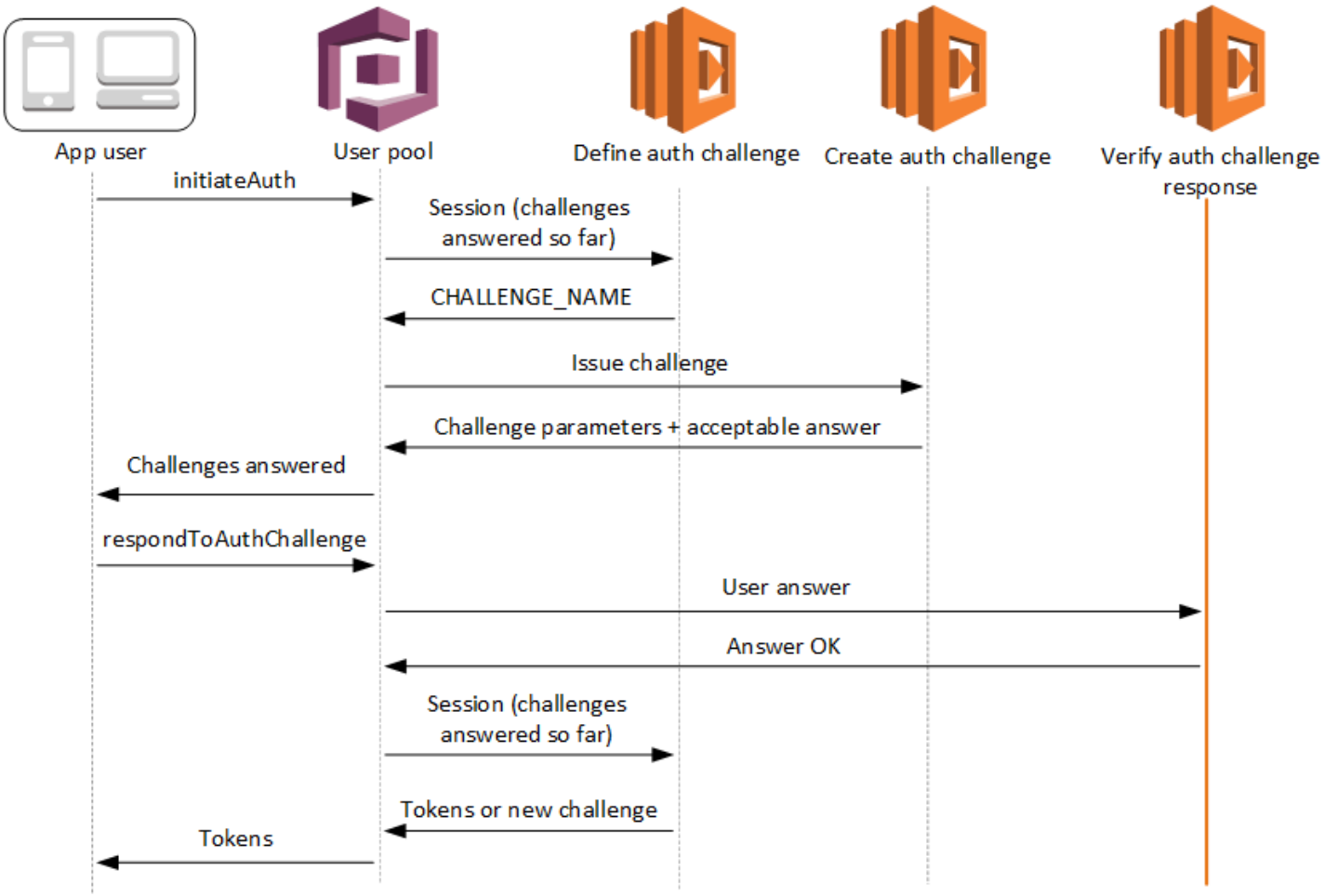
Ein CAPTCHA wird als Aufforderung für den Benutzer erstellt. Die URL für das CAPTCHA-Bild wird den Parametern der öffentlichen Aufforderung als "`captchaUrl`" hinzugefügt. Die erwartete Antwort wird den Parametern der privaten Aufforderung hinzugefügt.

Node.js

```
const handler = async (event) => {
  if (event.request.challengeName !== "CUSTOM_CHALLENGE") {
    return event;
  }

  if (event.request.session.length === 2) {
    event.response.publicChallengeParameters = {};
    event.response.privateChallengeParameters = {};
    event.response.publicChallengeParameters.captchaUrl = "url/123.jpg";
    event.response.privateChallengeParameters.answer = "5";
  }

  if (event.request.session.length === 3) {
    event.response.publicChallengeParameters = {};
```

```
    event.response.privateChallengeParameters = {};
    event.response.publicChallengeParameters.securityQuestion =
      "Who is your favorite team mascot?";
    event.response.privateChallengeParameters.answer = "Peccy";
  }

  return event;
};

export { handler }
```

Lambda-Auslöser für die Verifizierung der Antwort auf eine Authentifizierungsaufforderung



Antwort auf Authentifizierungsaufforderung überprüfen

Amazon Cognito ruft diesen Auslöser auf, um zu überprüfen, ob die Antwort des Benutzers auf eine benutzerdefinierte Authentifizierungsaufforderung gültig ist. Es ist Teil eines benutzerdefinierten Authentifizierungsablaufs für einen Benutzerpool.

Die Anforderung für diesen Auslöser enthält die Parameter `privateChallengeParameters` und `challengeAnswer`. Die `privateChallengeParameters`-Werte werden vom Lambda-Auslöser „Authentifizierungsaufforderung erstellen“ zurückgegeben und enthalten die erwartete Antwort vom Benutzer. Der Parameter `challengeAnswer` enthält die Benutzerantwort auf die Aufforderung.

Die Antwort enthält das `answerCorrect`-Attribut. Wenn der Benutzer die Herausforderung erfolgreich abgeschlossen hat, legt Amazon Cognito den Attributwert auf `true` fest. Wenn der Benutzer die Herausforderung nicht erfolgreich abgeschlossen hat, legt Amazon Cognito den Wert auf `false` fest.

Die Aufforderungsschleife wird so lange ausgeführt, bis die Benutzer alle Aufforderungen beantwortet hat.

Themen

- Lambda-Auslöserparameter für die Überprüfung einer Authentifizierungsaufforderung
- Beispiel für „Antwort auf Authentifizierungsaufforderung überprüfen“

Lambda-Auslöserparameter für die Überprüfung einer Authentifizierungsaufforderung

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "privateChallengeParameters": {
            "string": "string",
            . . .
```

```
        },
        "challengeAnswer": "string",
        "clientMetadata": {
            "string": "string",
            . . .
        },
        "userNotFound": boolean
    },
    "response": {
        "answerCorrect": boolean
    }
}
```

Anforderungsparameter für die Verifizierung einer Authentifizierungsaufforderung

userAttributes

Dieser Parameter enthält ein oder mehrere Name-Wert-Paare, die Benutzerattribute darstellen.

userNotFound

Amazon Cognito gibt diesen booleschen Wert ein, wenn Amazon Cognito `PreventUserExistenceErrors` für Ihren Benutzerpool-Client auf `ENABLED` festgelegt hat.

privateChallengeParameters

Dieser Parameter stammt aus dem Trigger „Authentifizierungsaufforderung erstellen“. Um festzustellen, ob der Benutzer eine Aufforderung erfüllt hat, vergleicht Amazon Cognito die Parameter mit der challengeAnswer eines Benutzers.

Dieser Parameter sollte alle erforderlichen Informationen enthalten, um die Antwort des Benutzerstauf die Aufforderung zu validieren. Diese Informationen beinhalten die Frage, die Amazon Cognito dem Benutzer stellt (`publicChallengeParameters`), sowie die gültigen Antworten auf die Frage (`privateChallengeParameters`). Dieser Parameter wird nur vom Lambda-Auslöser „Antwort auf Authentifizierungsaufforderung überprüfen“ verwendet.

challengeAnswer

Dieser Parameterwert ist die Antwort des Benutzers auf die Aufforderung.

clientMetadata

Dieser Parameter enthält ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion für den Auslöser zur Überprüfung

der Authentifizierungsaufforderung bereitstellen können. Verwenden Sie den ClientMetadata-Parameter in den API-Vorgängen AdminRespondToAuthChallenge und RespondToAuthChallenge, um diese Daten an Ihre Lambda-Funktion zu übergeben. Amazon Cognito enthält keine Daten aus dem ClientMetadata-Parameter in AdminInitiateAuth und InitiateAuth-API-Operationen in der Anforderung, die es an die Funktion zur Überprüfung der Authentifizierungsaufforderung übergibt.

Antwortparameter für die Verifizierung der Authentifizierungsaufforderung

answerCorrect

Wenn der Benutzer die Aufforderung erfolgreich abschließt, legt Amazon Cognito diesen Parameter auf `true` fest. Wenn der Benutzer die Aufforderung nicht erfolgreich abschließt, legt Amazon Cognito den Parameter auf `false` fest.

Beispiel für „Antwort auf Authentifizierungsaufforderung überprüfen“

In diesem Beispiel prüft die Lambda-Funktion, ob die Benutzerantwort auf eine Aufforderung mit der erwarteten Antwort übereinstimmt. Amazon Cognito setzt den Parameter `answerCorrect` auf `true`, wenn die Benutzerantwort mit der erwarteten Antwort übereinstimmt.

Node.js

```
const handler = async (event) => {
  if (
    event.request.privateChallengeParameters.answer ==
    event.request.challengeAnswer
  ) {
    event.response.answerCorrect = true;
  } else {
    event.response.answerCorrect = false;
  }

  return event;
};

export { handler };
```

Lambda-Auslöser für die Vorab-Generierung von Token

Da Amazon Cognito diesen Auslöser aufruft, bevor Token generiert werden, können Sie die Ansprüche in Benutzerpool-Token anpassen. Mit den grundlegenden Features von Version 1 oder dem Auslöserereignis `V1_0` vor der Token-Generierung können Sie das Identitätstoken (ID) anpassen. In Benutzerpools mit aktiven erweiterten Sicherheits-Features können Sie das Ereignis der Version 2 oder das `V2_0`-Auslöserereignis mit Anpassung des Zugriffstokens generieren.

Amazon Cognito sendet ein `V1_0`-Ereignis als Anfrage an Ihre Funktion mit Daten, die in das Identitätstoken geschrieben werden. Ein `V2_0`-Ereignis ist eine einzelne Anfrage mit den Daten, die Amazon Cognito sowohl in die Identitäts- als auch in die Zugriffstoken schreiben würde. Sie müssen Ihre Funktion so aktualisieren, dass sie die neueste Trigger-Version verwendet, und Daten für beide Token in derselben Antwort senden, wenn beide Token angepasst weden sollen.

Dieser Lambda-Auslöser kann einige Ansprüche in Identitäts- und Zugriffstoken hinzufügen, entfernen und ändern, bevor Amazon Cognito sie an Ihre App ausgibt. Um dieses Feature zu verwenden, verknüpfen Sie eine Lambda-Funktion aus der Amazon-Cognito-Benutzerpool-Konsole oder aktualisieren Ihren Benutzerpool `LambdaConfig` über die AWS Command Line Interface (AWS CLI).

Ereignisversionen

Ihr Benutzerpool kann verschiedene Versionen eines Trigger-Ereignisses vor der Token-Generierung an Ihre Lambda-Funktion liefern. Ein `V1_0` Trigger liefert die Parameter für die Änderung von ID-Token. Ein `V2_0` Trigger liefert Parameter für Folgendes.

1. Die Funktionen eines `V1_0` Triggers.
2. Die Möglichkeit, Zugriffstoken anzupassen.
3. Die Fähigkeit, komplexe Datentypen an ID- und Zugriffstoken-Anspruchswerte zu übergeben:
   - String
   - Zahl
   - Boolesch
   - Array aus Zeichenketten, Zahlen, Booleschen Werten oder einer Kombination aus diesen
   - JSON

> **Note**
>
> Im ID-Token können Sie komplexe Objekte mit den Werten von Ansprüchen mit Ausnahme von`phone_number_verified`,`email_verified`, `updated_at` und auffüllen. `address`

Benutzerpools liefern standardmäßig `V1_0` Ereignisse. Um Ihren Benutzerpool für das Senden eines `V2_0` Ereignisses zu konfigurieren, wählen Sie eine Trigger-Event-Version von Basic features + access token customization, wenn Sie Ihren Trigger in der Amazon Cognito Cognito-Konsole konfigurieren. Sie können den Wert von auch `LambdaVersion` in den LambdaConfigParametern einer UpdateUserPool oder CreateUserPool API-Anfrage festlegen. Für die Anpassung des Zugriffstokens mit `V2_0` Ereignissen fallen zusätzliche Kosten an. Weitere Informationen finden Sie unter Amazon Cognito – Preise.

Ausgeschlossene Ansprüche und Bereiche

Amazon Cognito schränkt die Ansprüche und Bereiche ein, die Sie in Zugriffs- und Identitäts-Token hinzufügen, ändern oder unterdrücken können. Wenn Ihre Lambda-Funktion versucht, einen Wert für einen dieser Ansprüche festzulegen, gibt Amazon Cognito ein Token mit dem ursprünglichen Anspruchswert aus, sofern einer in der Anfrage vorhanden war.

Freigegebene Ansprüche

- `acr`
- `amr`
- `at_hash`
- `auth_time`
- `azp`
- `exp`
- `iat`
- `iss`
- `jti`
- `nbf`
- `nonce`
- `origin_jti`
- `sub`

- `token_use`

ID-Token-Ansprüche

- `identities`
- `aud`
- `cognito:username`

Zugriffs-Token-Ansprüche

- `username`
- `client_id`
- `scope`

> **Note**
>
> Sie können die Bereiche in einem Zugriffs-Token mit den Antwortwerten `scopesToAdd` und `scopesToSuppress` ändern, den `scope`-Anspruch können Sie jedoch nicht direkt ändern. Sie können keine Bereiche hinzufügen, die mit `aws.cognito` beginnen, einschließlich des reservierten Bereichs `aws.cognito.signin.user.admin` für Benutzerpools.

- `device_key`
- `event_id`
- `version`

Sie können keine Ansprüche mit den folgenden Präfixen hinzufügen oder überschreiben, Sie können sie jedoch unterdrücken oder verhindern, dass sie im Token erscheinen.

- `dev:`
- `cognito:`

Sie können einen `aud`-Anspruch zu Zugriffs-Token hinzufügen, jedoch muss dessen Wert mit der App-Client-ID der aktuellen Sitzung übereinstimmen. Sie können die Client-ID im Anforderungsereignis von `event.callerContext.clientId` ableiten.

Anpassen des Identitäts-Token

Mit dem Lambda-Trigger vor der Token-Generierung können Sie den Inhalt eines Identitäts-Tokens (ID-Token) aus Ihrem Benutzerpool heraus anpassen. Das ID-Token stellt Benutzerattribute aus einer vertrauenswürdigen Identitätsquelle für die Anmeldung bei einer Web- oder mobilen App bereit. Weitere Informationen zu ID-Token finden Sie unter Verwenden des ID-Tokens.

Der Lambda-Trigger vor der Token-Generierung mit einem ID-Token kann unter anderem wie folgt verwendet werden.

- Eine Änderung an der IAM-Rolle zur Laufzeit vornehmen, die Ihr Benutzer aus einem Identitätspool anfordert.
- Benutzerattribute aus einer externen Quelle hinzufügen.
- Vorhandene Benutzerattributwerte hinzufügen oder ersetzen.
- Die Offenlegung von Benutzerattributen unterdrücken, die aufgrund der autorisierten Bereiche Ihres Benutzers und des Lesezugriffs auf Attribute, die Sie Ihrem App-Client gewährt haben, andernfalls an Ihre App weitergegeben würden.

Anpassen des Zugriffs-Token

Mit dem Lambda-Trigger vor der Token-Generierung können Sie den Inhalt eines Zugriffs-Tokens aus Ihrem Benutzerpool heraus anpassen. Das Zugriffs-Token autorisiert Benutzer, Informationen aus zugriffsgeschützten Ressourcen abzurufen, z. B. aus Token-autorisierten Amazon-Cognito-API-Operationen sowie Drittanbieter-APIs. Sie können zwar Zugriffstoken für die machine-to-machine (M2M-) Autorisierung mit Amazon Cognito mit einer Gewährung von Kundenanmeldedaten generieren, M2M-Anfragen rufen jedoch nicht die Triggerfunktion vor der Token-Generierung auf und können keine benutzerdefinierten Zugriffstoken ausgeben. Weitere Informationen zu Zugriffs-Token finden Sie unter Verwenden des Zugriffstokens.

Der Lambda-Trigger vor der Token-Generierung mit einem Zugriffs-Token kann unter anderem wie folgt verwendet werden.

- Fügen Sie dem `scope`-Anspruch OAuth 2.0-Bereiche hinzu oder unterdrücken Sie diese. Sie können beispielsweise Bereiche zu einem Zugriffs-Token hinzufügen, das aus der API-Authentifizierung von Amazon-Cognito-Benutzerpools resultiert, wodurch nur der Bereich `aws.cognito.signin.user.admin` zugewiesen wird.
- Die Mitgliedschaft eines Benutzers in Benutzerpool-Gruppen ändern.

- Fügen Sie Ansprüche hinzu, die noch nicht in einem Amazon-Cognito-Zugriffstoken enthalten sind.
- Unterdrücken Sie die Offenlegung von Ansprüchen, die andernfalls an Ihre App weitergeleitet würden.

Sie müssen den Benutzerpool so konfigurieren, dass eine aktualisierte Version der Trigger-Anforderung generiert wird um Zugriffsanpassungen in Ihrem Benutzerpool zu unterstützen. Aktualisieren Sie Ihren Benutzerpool wie im folgenden Verfahren gezeigt.

AWS Management Console

So unterstützen Sie die Anpassung von Zugriffs-Token in einem Lambda-Trigger vor der Token-Generierung

1. Navigieren Sie zur Amazon-Cognito-Konsole und wählen Sie User Pools (Benutzerpools) aus.
2. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
3. Falls Sie dies noch nicht getan haben, aktivieren Sie erweiterte Sicherheits-Features auf der Registerkarte App-Integration.
4. Wählen Sie die Registerkarte User pool properties (Benutzerpool-Eigenschaften) und suchen Sie dort nach Lambda-Auslösern.
5. Fügen Sie einen Trigger für die Pre-Token-Generierung hinzu oder bearbeiten Sie ihn.
6. Wählen Sie unter Zuweisen einer Lambda-Funktion eine Lambda-Funktion aus.
7. Wählen Sie eine Auslöserereignis-Version unter Grundlegende Features und Anpassung des Zugriffstokens aus. Diese Einstellung aktualisiert die Anforderungsparameter, die Amazon Cognito an Ihre Funktion sendet, damit sie Felder für die Anpassung von Zugriffs-Token enthalten.

User pools API

So unterstützen Sie die Anpassung von Zugriffs-Token in einem Lambda-Trigger für die Pre-Token-Generierung

Generieren Sie eine oder API-Anfrage CreateUserPool. UpdateUserPool Sie müssen einen Wert für alle Parameter angeben, die nicht auf einen Standardwert festgelegt werden sollen. Weitere Informationen finden Sie unter Aktualisieren der Benutzerpool-Konfiguration.

Nehmen Sie den folgenden Inhalt in den `LambdaVersion`-Parameter Ihrer Anfrage auf. Ein `LambdaVersion` Wert von `V2_0` veranlasst Ihren Benutzerpool, Parameter für die Anpassung von Zugriffs-Token hinzuzufügen. Verwenden Sie einen Lambda-Funktions-ARN mit einer Funktionsversion als den Wert von `LambdaArn`, um eine bestimmte Funktionsversion aufzurufen.

```
"PreTokenGenerationConfig": {
   "LambdaArn": "arn:aws:lambda:us-west-2:123456789012:function:MyFunction",
   "LambdaVersion": "V2_0"
},
```

Themen

- Lambda-Auslöserquellen für die Vorab-Generierung von Token
- Lambda-Auslöserparameter für die Vorab-Generierung von Token
- Beispiel für ein Pre-Token-Auslöserereignis, Version 2: Ansprüche, Bereiche und Gruppen hinzufügen und unterdrücken
- Beispiel für das Ereignis vor der Token-Generierung, Version 2: Fügen Sie Ansprüche mit komplexen Objekten hinzu
- Beispiel für ein Version-Eins-Ereignis vor der Token-Generierung: Hinzufügen eines neuen Anspruchs und Löschen eines vorhandenen Anspruchs
- Beispiel für ein Version-Eins-Ereignis vor der Token-Generierung: Ändern der Gruppenmitgliedschaft des Benutzers

Lambda-Auslöserquellen für die Vorab-Generierung von Token

triggerSource-Wert	Ereignis
`TokenGeneration_HostedAuth`	Wird bei der Authentifizierung durch die Anmeldeseite der gehosteten Amazon-Cognito-Benutzeroberfläche aufgerufen.
`TokenGeneration_Authentication`	Wird aufgerufen, nachdem Benutzer-Authentifizierung-Abläufe abgeschlossen sind.
`TokenGeneration_NewPasswordChallenge`	Wird aufgerufen, nachdem der Benutzer von einem Administrator erstellt wurde. Dieser

triggerSource-Wert	Ereignis
	Ablauf wird aufgerufen, wenn der Benutzer ein temporäres Passwort ändern muss.
`TokenGeneration_AuthenticateDevice`	Wird am Ende der Authentifizierung eines Benutzergeräts aufgerufen.
`TokenGeneration_RefreshTokens`	Wird aufgerufen, wenn ein Benutzer versucht, die Identitäts- und Zugriffs-Token zu aktualisieren.

Lambda-Auslöserparameter für die Vorab-Generierung von Token

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt. Wenn Sie Ihrem Benutzerpool zur Pre-Token-Generierung einen Lambda-Trigger hinzufügen, können Sie eine Trigger-Version auswählen. Diese Version bestimmt, ob Amazon Cognito eine Anfrage mit zusätzlichen Parametern für die Anpassung von Zugriffs-Token an Ihre Lambda-Funktion weitergibt.

Version 1

Mit dem Versions-1-Token können Gruppenmitgliedschaften, IAM-Rollen und neue Ansprüche in ID-Token festgelegt werden.

```
{
    "request": {
        "userAttributes": {"string": "string"},
        "groupConfiguration": {
                "groupsToOverride": [
                    "string",
                    "string"
                ],
                "iamRolesToOverride": [
                    "string",
                    "string"
                ],
                "preferredRole": "string"
        },
        "clientMetadata": {"string": "string"}
    },
```

```
    "response": {
        "claimsOverrideDetails": {
            "claimsToAddOrOverride": {"string": "string"},
            "claimsToSuppress": [
                "string",
                "string"
            ],
            "groupOverrideDetails": {
                "groupsToOverride": [
                    "string",
                    "string"
                ],
                "iamRolesToOverride": [
                    "string",
                    "string"
                ],
                "preferredRole": "string"
            }
        }
    }
}
```

Version 2

Das Version-2-Anforderungsereignis fügt Felder hinzu, mit denen das Zugriffstoken angepasst werden kann. Es fügt auch Unterstützung für komplexe `claimsToOverride` Datentypen im Antwortobjekt hinzu. Ihre Lambda-Funktion kann die folgenden Datentypen im Wert von `claimsToOverride` zurückgeben:

- String
- Zahl
- Boolesch
- Array aus Zeichenketten, Zahlen, Booleschen Werten oder einer Kombination aus diesen
- JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string"
        },
```

```
        "scopes": ["string", "string"],
        "groupConfiguration": {
            "groupsToOverride": ["string", "string"],
            "iamRolesToOverride": ["string", "string"],
            "preferredRole": "string"
        },
        "clientMetadata": {
            "string": "string"
        }
    },
    "response": {
        "claimsAndScopeOverrideDetails": {
            "idTokenGeneration": {
                "claimsToAddOrOverride": {
                    "string": [accepted datatype]
                },
                "claimsToSuppress": ["string", "string"]
            },
            "accessTokenGeneration": {
                "claimsToAddOrOverride": {
                    "string": [accepted datatype]
                },
                "claimsToSuppress": ["string", "string"],
                "scopesToAdd": ["string", "string"],
                "scopesToSuppress": ["string", "string"]
            },
            "groupOverrideDetails": {
                "groupsToOverride": ["string", "string"],
                "iamRolesToOverride": ["string", "string"],
                "preferredRole": "string"
            }
        }
    }
}
```

Anforderungsparameter für die Vorab-Generierung von Token

Name	Beschreibung	Minimale Trigger-Ereignisversion
userAttributes	Die Attribute Ihres Benutzerprofils in Ihrem Benutzerpool.	1
groupConfiguration	Das Eingabeobjekt, das die aktuelle Gruppenkonfiguration enthält. Das Objekt umfasst `groupsToOverride` , `iamRolesToOverride` und `preferredRole` .	1
groupsToOverride	Die Benutzerpoolgruppen, in denen Ihr Benutzer Mitglied ist.	1
iamRolesToÜberschreiben	Sie können eine Benutzerpoolgruppe einer AWS Identity and Access Management (IAM-) Rolle zuordnen. Dieses Element ist eine Liste aller IAM-Rollen aus den Gruppen, in denen Ihr Benutzer Mitglied ist.	1
preferredRole	Sie können eine Priorität für Benutzerpoolgruppen festlegen. Dieses Element enthält den Namen der IAM-Rolle aus der Gruppe mit der höchsten Priorität im `groupsToOverride` -Element.	1
clientMetadata	Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion angeben und für den Auslöser für die Pre-Token-Generierung bereitstellen können. Um diese Daten an Ihre Lambda-Funktion zu übergeben, verwenden Sie den ClientMetadata Parameter in den RespondToAuthChallengeAPI-Operationen AdminRespondToAuthChallengeund. Amazon Cognito bezieht keine Daten aus dem `ClientMetadata` Parameter in AdminInitiateAuthund InitiateAuthAPI-Operationen in die Anfrage ein, die es an die Pre-Token-Generierungsfunktion weitergibt.	1

Name	Beschreibung	Minimale Trigger-Ereignisversion
Bereiche	Die OAuth-2.0-Bereiche Ihres Benutzers. Die in einem Zugriffs-Token enthaltenen Bereiche sind die standardmäßigen und benutzerdefinierten Bereiche des Benutzerpools, die Ihr Benutzer angefordert hat und für deren Ausgabe Sie Ihren App-Client autorisiert haben.	2

Antwortparameter für die Vorab-Generierung von Token

Name	Beschreibung	Minimale Trigger-Ereignisversion
claimsOverrideDetails	Ein Container für alle Elemente in einem `V1_0`-Auslöserereignis.	1
claimsAndScopeOverrideDetails	Ein Container für alle Elemente in einem `V2_0`-Auslöserereignis.	2
idTokenGeneration	Die Ansprüche, die Sie im ID-Token Ihres Benutzers überschreiben, hinzufügen oder unterdrücken möchten. Diese Werte für die Anpassung des übergeordneten ID-Token erscheinen ausschließlich in Ereignissen der Version 2, während die untergeordneten Elemente in Ereignissen der Version 1 erscheinen.	2
accessTokenGeneration	Die Ansprüche, die Sie im Zugriffs-Token Ihres Benutzers überschreiben, hinzufügen oder unterdrücken möchten. Diese Werte für die Anpassung des Zugriffs-Token als übergeordnetes Element erscheinen ausschließlich in Ereignissen der Version 2.	2

Name	Beschreibung	Minimale Trigger-Ereignisversion
claimsToAddOrOverride	Eine Zuordnung von einem oder mehreren Ansprüchen und deren Werte, die Sie hinzufügen oder ändern möchten. Für gruppenbezogene Ansprüche verwenden Sie stattdessen `groupOverrideDetails` . Bei Ereignissen der Version 2 erscheint dieses Element unter `accessTokenGeneration` und `idTokenGeneration` .	1*
claimsToSuppress	Eine Liste der Ansprüche, die Amazon Cognito unterdrücken soll. Wenn Ihre Funktion einen Anspruchswert unterdrückt und ersetzt, unterdrückt Amazon Cognito den Anspruch. Bei Ereignissen der Version 2 erscheint dieses Element unter `accessTokenGeneration` und `idTokenGeneration` .	1

Name	Beschreibung	Minimale Trigger-Ereignisversion
groupOverrideDetails	Das Ausgabeobjekt, das die aktuelle Gruppenkonfiguration enthält. Das Objekt umfasst `groupsToOverride` , `iamRolesToOverride` und `preferredRole` . Ihre Funktion ersetzt das `groupOverrideDetails` -Objekt durch das von Ihnen bereitgestellte Objekt. Wenn Sie ein leeres oder Null-Objekt in der Antwort angeben, löscht Amazon Cognito die Gruppen. Wenn die bestehende Gruppenkonfiguration unverändert beibehalten werden soll, kopieren Sie den Wert des `groupConfiguration` -Objekts der Anforderung in das `groupOverrideDetails` -Objekt der Antwort. Übergeben Sie es dann zurück an den Service. Amazon-Cognito-ID und Zugriffstoken enthalten beide den `cognito:groups` -Anspruch. Ihr `groupOverrideDetails` -Objekt ersetzt den `cognito:groups` -Anspruch in Zugriffs-Token und in ID-Token.	1
scopesToAdd	Eine Liste der OAuth-2.0-Bereiche, die Sie dem `scope`-Anspruch im Zugriffs-Token Ihres Benutzers hinzufügen möchten. Sie können keine Bereichswerte hinzufügen, die ein oder mehrere Leerzeichen enthalten.	2
scopesToSuppress	Eine Liste der OAuth-2.0-Bereiche, die Sie aus dem `scope`-Anspruch im Zugriffs-Token Ihres Benutzers entfernen möchten.	2

* Antwortobjekte auf Ereignisse der Version 1 können Zeichenketten zurückgeben. Antwortobjekte auf Ereignisse der Version 2 können komplexe Objekte zurückgeben.

Beispiel für ein Pre-Token-Auslöserereignis, Version 2: Ansprüche, Bereiche und Gruppen hinzufügen und unterdrücken

In diesem Beispiel werden die folgenden Änderungen an den Token eines Benutzers vorgenommen.

1. Legt ihren `family_name` als `Doe` im ID-Token fest.
2. Verhindert, dass `email`- und `phone_number`-Ansprüche im ID-Token erscheinen.
3. Legt ihren ID-Token-`cognito:roles`-Anspruch auf `"arn:aws:iam::123456789012:role\/sns_callerA","arn:aws:iam::123456789012:role\/sns_callerC","arn:aws:iam::123456789012:role\/sns_callerB"` fest.
4. Legt ihren ID-Token-`cognito:preferred_role`-Anspruch auf `arn:aws:iam::123456789012:role/sns_caller` fest.
5. Fügt die Bereiche `openid`, `email` und `solar-system-data/asteroids.add` zum Zugriffstoken hinzu.
6. Unterdrückt den Bereich `phone_number` und `aws.cognito.signin.user.admin` aus dem Zugriffstoken. Durch das Entfernen von `phone_number` wird das Abrufen der Telefonnummer des Benutzers von `userInfo` verhindert. Durch das Entfernen von `aws.cognito.signin.user.admin` werden API-Anfragen des Benutzers zum Lesen und Ändern des eigenen Profils mit der Amazon-Cognito-Benutzerpool-API verhindert.

   > **Note**
   >
   > Das Entfernen von `phone_number` aus Bereichen verhindert nur dann das Abrufen der Telefonnummer eines Benutzers, wenn die verbleibenden Bereiche im Zugriffstoken `openid` und mindestens einen weiteren Standardbereich enthalten. Weitere Informationen finden Sie unter Grundlegendes zu Bereichen.

7. Legt den ID-Token-`cognito:groups`-Anspruch auf `"new-group-A","new-group-B","new-group-C"` fest.

JavaScript

```
export const handler = function(event, context) {
  event.response = {
    "claimsAndScopeOverrideDetails": {
      "idTokenGeneration": {
        "claimsToAddOrOverride": {
```

```
            "family_name": "Doe"
        },
            "claimsToSuppress": [
              "email",
              "phone_number"
        ]
          },
          "accessTokenGeneration": {
            "scopesToAdd": [
              "openid",
              "email",
              "solar-system-data/asteroids.add"
            ],
            "scopesToSuppress": [
              "phone_number",
              "aws.cognito.signin.user.admin"
            ]
          },
          "groupOverrideDetails": {
            "groupsToOverride": [
        "new-group-A",
        "new-group-B",
        "new-group-C"
      ],
            "iamRolesToOverride": [
              "arn:aws:iam::123456789012:role/new_roleA",
              "arn:aws:iam::123456789012:role/new_roleB",
              "arn:aws:iam::123456789012:role/new_roleC"
            ],
            "preferredRole": "arn:aws:iam::123456789012:role/new_role",
          }
        }
      };
      // Return to Amazon Cognito
      context.done(null, event);
};
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
    "version": "2",
    "triggerSource": "TokenGeneration_Authentication",
    "region": "us-east-1",
    "userPoolId": "us-east-1_EXAMPLE",
    "userName": "JaneDoe",
    "callerContext": {
        "awsSdkVersion": "aws-sdk-unknown-unknown",
        "clientId": "1example23456789"
    },
    "request": {
        "userAttributes": {
            "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "cognito:user_status": "CONFIRMED",
            "email_verified": "true",
            "phone_number_verified": "true",
            "phone_number": "+12065551212",
            "family_name": "Zoe",
            "email": "Jane.Doe@example.com"
        },
        "groupConfiguration": {
            "groupsToOverride": ["group-1", "group-2", "group-3"],
            "iamRolesToOverride": ["arn:aws:iam::123456789012:role/sns_caller1",
 "arn:aws:iam::123456789012:role/sns_caller2", "arn:aws:iam::123456789012:role/
sns_caller3"],
            "preferredRole": ["arn:aws:iam::123456789012:role/sns_caller"]
        },
        "scopes": [
            "aws.cognito.signin.user.admin", "openid", "email", "phone"
        ]
    },
    "response": {
        "claimsAndScopeOverrideDetails": []
    }
}
```

Beispiel für das Ereignis vor der Token-Generierung, Version 2: Fügen Sie Ansprüche mit komplexen Objekten hinzu

In diesem Beispiel werden die folgenden Änderungen an den Token eines Benutzers vorgenommen.

1. Fügt dem ID-Token Ansprüche der Typen Zahl, Zeichenfolge, Boolean und JSON hinzu. Dies ist die einzige Änderung, die durch Trigger-Ereignisse der zweiten Version für das ID-Token verfügbar gemacht wird.
2. Fügt dem Zugriffstoken Ansprüche der Typen Zahl, Zeichenfolge, Boolean und JSON hinzu.
3. Fügt dem Zugriffstoken drei Bereiche hinzu.
4. Unterdrückt die `sub` Ansprüche `email` und in den ID- und Zugriffstoken.
5. Unterdrückt den `aws.cognito.signin.user.admin` Bereich im Zugriffstoken.

JavaScript

```
export const handler = function(event, context) {

    var scopes = ["MyAPI.read", "MyAPI.write", "MyAPI.admin"]
    var claims = {}
    claims["aud"]= event.callerContext.clientId;
    claims["booleanTest"] = false;
    claims["longTest"] = 9223372036854775807;
    claims["exponentTest"] = 1.7976931348623157E308;
    claims["ArrayTest"] = ["test", 9223372036854775807, 1.7976931348623157E308,
 true];
    claims["longStringTest"] = "\{\
        \"first_json_block\": \{\
            \"key_A\": \"value_A\",\
            \"key_B\": \"value_B\"\
        \},\
        \"second_json_block\": \{\
            \"key_C\": \{\
                \"subkey_D\": [\
                    \"value_D\",\
                    \"value_E\"\
                ],\
                \"subkey_F\": \"value_F\"\
            \},\
            \"key_G\": \"value_G\"\
        \}\
    \}";
    claims["jsonTest"] = {
     "first_json_block": {
      "key_A": "value_A",
      "key_B": "value_B"
     },
```

```
     "second_json_block": {
      "key_C": {
       "subkey_D": [
        "value_D",
        "value_E"
       ],
       "subkey_F": "value_F"
      },
      "key_G": "value_G"
     }
    };
    event.response = {
        "claimsAndScopeOverrideDetails": {
            "idTokenGeneration": {
                "claimsToAddOrOverride": claims,
                "claimsToSuppress": ["email","sub"]
            },
            "accessTokenGeneration": {
                "claimsToAddOrOverride": claims,
                "claimsToSuppress": ["email","sub"],
                "scopesToAdd": scopes,
                "scopesToSuppress": ["aws.cognito.signin.user.admin"]
            }
        }
    };
    console.info("EVENT response\n" + JSON.stringify(event, (_, v) => typeof v ===
 'bigint' ? v.toString() : v, 2))
    console.info("EVENT response size\n" + JSON.stringify(event, (_, v) => typeof v
 === 'bigint' ? v.toString() : v).length)
    // Return to Amazon Cognito
    context.done(null, event);
};
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
    "version": "2",
```

```
    "triggerSource": "TokenGeneration_HostedAuth",
    "region": "us-west-2",
    "userPoolId": "us-west-2_EXAMPLE",
    "userName": "JaneDoe",
    "callerContext": {
        "awsSdkVersion": "aws-sdk-unknown-unknown",
        "clientId": "1example23456789"
    },
    "request": {
        "userAttributes": {
            "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "cognito:user_status": "CONFIRMED"
            "email_verified": "true",
            "phone_number_verified": "true",
            "phone_number": "+12065551212",
            "email": "Jane.Doe@example.com"
        },
        "groupConfiguration": {
            "groupsToOverride": ["group-1", "group-2", "group-3"],
            "iamRolesToOverride": ["arn:aws:iam::123456789012:role/sns_caller1"],
            "preferredRole": ["arn:aws:iam::123456789012:role/sns_caller1"]
        },
        "scopes": [
            "aws.cognito.signin.user.admin",
            "phone",
            "openid",
            "profile",
            "email"
        ]
    },
    "response": {
        "claimsAndScopeOverrideDetails": []
    }
}
```

Beispiel für ein Version-Eins-Ereignis vor der Token-Generierung: Hinzufügen eines neuen Anspruchs und Löschen eines vorhandenen Anspruchs

In diesem Beispiel wird ein Trigger-Ereignis der Version 1 mit einer Pre-Token-Generierungs-Lambda-Funktion verwendet, um einen neuen Anspruch hinzuzufügen und einen vorhandenen Anspruch zu unterdrücken.

Node.js

```
const handler = async (event) => {
  event.response = {
    claimsOverrideDetails: {
      claimsToAddOrOverride: {
        my_first_attribute: "first_value",
        my_second_attribute: "second_value",
      },
      claimsToSuppress: ["email"],
    },
  };

  return event;
};

export { handler };
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel: Da das Code-Beispiel nicht alle Anforderungsparameter verarbeitet, können Sie ein Testereignis mit einer leeren Anfrage verwenden. Weitere Informationen zu allgemeinen Anforderungsparametern finden Sie unter Lambda-Auslöserereignis für einen Benutzerpool.

JSON

```
{
  "request": {},
  "response": {}
}
```

Beispiel für ein Version-Eins-Ereignis vor der Token-Generierung: Ändern der Gruppenmitgliedschaft des Benutzers

In diesem Beispiel wird ein Trigger-Ereignis der Version 1 mit einer Pre-Token-Generierungs-Lambda-Funktion verwendet, um die Gruppenmitgliedschaft des Benutzers zu ändern.

Node.js

```
const handler = async (event) => {
  event.response = {
    claimsOverrideDetails: {
      groupOverrideDetails: {
        groupsToOverride: ["group-A", "group-B", "group-C"],
        iamRolesToOverride: [
          "arn:aws:iam::XXXXXXXXXXX:role/sns_callerA",
          "arn:aws:iam::XXXXXXXXX:role/sns_callerB",
          "arn:aws:iam::XXXXXXXXXX:role/sns_callerC",
        ],
        preferredRole: "arn:aws:iam::XXXXXXXXXXX:role/sns_caller",
      },
    },
  };

  return event;
};

export { handler };
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
  "request": {},
  "response": {}
}
```

Lambda-Auslöser für die Benutzermigration.

Wenn ein Benutzer zum Zeitpunkt der Anmeldung mit einem Passwort oder während des Ablaufs bei vergessenem Passwort nicht im Benutzerpool vorhanden ist, ruft Amazon Cognito den Auslöser auf. Nachdem die Lambda-Funktion erfolgreich ausgeführt wurde, erstellt Amazon Cognito den Benutzer

in den Benutzerpool. Weitere Informationen zum Authentifizierungsablauf mit dem Lambda-Auslöser für die Benutzermigration finden Sie unter Importieren von Benutzern in Benutzerpools mit einem Lambda-Auslöser für die Benutzermigration.

Um Benutzer zum Zeitpunkt der Anmeldung oder während des Ablaufs bei vergessenem Passwort aus Ihrem vorhandenen Benutzerverzeichnis in Amazon-Cognito-Benutzerpools zu migrieren, verwenden Sie diese Lambda-Auslöser.

Themen

- Lambda-Auslöserquellen für die Benutzermigration
- Lambda-Auslöserparameter für die Benutzermigration
- Beispiel: Migration eines Benutzers mit einem bestehenden Passwort

Lambda-Auslöserquellen für die Benutzermigration

triggerSource-Wert	Veranstaltung
`UserMigration_Authentication`	Benutzermigration bei der Anmeldung.
`UserMigration_ForgotPassword`	Benutzermigration während des Ablaufs bei vergessenem Passwort.

Lambda-Auslöserparameter für die Benutzermigration

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "userName": "string",
    "request": {
        "password": "string",
        "validationData": {
            "string": "string",
            . . .
        },
        "clientMetadata": {
```

```
            "string": "string",
        . . .
         }
    },
    "response": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "finalUserStatus": "string",
        "messageAction": "string",
        "desiredDeliveryMediums": [ "string", . . .],
        "forceAliasCreation": boolean,
        "enableSMSMFA": boolean
    }
}
```

Anforderungsparameter für die Benutzermigration

userName

Der Benutzername, den der Benutzer bei der Anmeldung eingibt.

password

Das Passwort, den der Benutzer bei der Anmeldung eingibt. Amazon Cognito sendet diesen Wert nicht in einer Anfrage, die durch einen Ablauf bei vergessenem Passwort initiiert wird.

validationData

Ein oder mehrere Schlüssel-Wert-Paare, die die Validierungsdaten in der Anmeldeanforderung des Benutzers enthalten. Um diese Daten an Ihre Lambda-Funktion zu übergeben, können Sie den Parameter ClientMetadata in den API-Aktionen InitiateAuth und AdminInitiateAuth verwenden.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion zum Migrieren von Benutzern bereitstellen können. Um diese Daten an Ihre Lambda-Funktion zu übergeben, können Sie den Parameter ClientMetadata in den API-Aktionen AdminRespondToAuthChallenge und ForgotPassword verwenden.

Antwortparameter für die Benutzermigration

userAttributes

Dies ist ein Pflichtfeld.

Dieses Feld muss ein oder mehrere Name-Wert-Paare enthalten, die Amazon Cognito im Benutzerprofil in Ihrem Benutzerpool speichert und als Benutzerattribute verwendet. Sie können sowohl Standard- als auch benutzerdefinierte Benutzerattribute aufnehmen. Für benutzerdefinierte Attribute muss das Präfix `custom:` angegeben werden, damit sie von Standardattributen unterschieden werden können. Weitere Informationen finden Sie unter Benutzerdefinierte Attribute.

> **Note**
>
> Ein Benutzer benötigt eine verifizierte E-Mail-Adresse oder eine verifizierte Telefonnummer, um seine Passwörter im Ablauf bei vergessenem Passwort zurückzusetzen. Amazon Cognito sendet eine Nachricht mit einem Code zum Zurücksetzen des Passworts an die in den Benutzerattributen angegebene E-Mail-Adresse oder Telefonnummer.

Attribute	Anforderung
Alle Attribute, die beim Erstellen Ihres Benutzerpools als erforderlich gekennzeichnet sind	Wenn erforderliche Attribute für die Migration fehlen, verwendet Amazon Cognito Standardwerte.
`username`	Erforderlich, wenn Sie Ihren Benutzerpool mit Alias-Attributen zusätzlich zum Benutzernamen für die Anmeldung konfiguriert haben und der Benutzer einen gültigen Aliaswert als Benutzernamen eingegeben hat. Dieser Aliaswert kann eine E-Mail-Adresse, der bevorzugte Benutzername oder eine Telefonnummer sein. Wenn die Anforderung und der Benutzerpool die Aliasanforderungen erfüllen, muss die Antwort Ihrer Funktion den erhaltenen `username`-Parameter einem Alias-Attribut zuweisen. Außerdem

Attribute	Anforderung
	muss die Antwort Ihren eigenen Wert dem `username`-Attribut zuweisen. Wenn Ihr Benutzerpool die Bedingungen nicht erfüllt, die für die Zuordnung des empfangenen `username` zu einem Alias erforderlich sind, muss der `username`-Parameter in der Antwort entweder genau mit der Anforderung übereinstimmen oder weggelassen werden. **Note** `username` muss innerhalb des Benutzerpools eindeutig sein.

finalUserStatus

Sie können diesen Parameter auf `CONFIRMED` festlegen, um Ihre Benutzer automatisch zu bestätigen, damit sie sich mit ihren vorherigen Passwörtern anmelden können. Wenn Sie einen Benutzer auf `CONFIRMED` festlegen, müssen sie keine zusätzlichen Maßnahmen ergreifen, bevor sie sich anmelden können. Wenn Sie dieses Attribut nicht auf `CONFIRMED` festlegen, ist es auf `RESET_REQUIRED` eingestellt.

Ein `finalUserStatus` von `RESET_REQUIRED` bedeutet, dass der Benutzer sein Passwort sofort nach der Migration bei der Anmeldung ändern muss und Ihre Client-App die `PasswordResetRequiredException` während des Authentifizierungsflusses erledigen muss.

Note

Amazon Cognito setzt die Kennwortstärkerichtlinie nicht durch, die Sie während der Migration mit einem Lambda-Auslöser für den Benutzerpool konfiguriert haben. Wenn das Kennwort nicht die von Ihnen konfigurierte Passwortrichtlinie erfüllt, akzeptiert Amazon Cognito das Kennwort trotzdem, damit es die Migration des Benutzers fortsetzen kann. Überprüfen Sie die Passwortstärke in Ihrem Code, um Richtlinien für die Passwortstärke durchzusetzen und Passwörter abzulehnen, die nicht der Richtlinie entsprechen. Setzen Sie anschließend finalUserStatus auf `RESET_REQUIRED`, wenn das Passwort nicht der Richtlinie entspricht.

messageAction

Sie können diesen Parameter auf `SUPPRESS` festlegen, um die Begrüßungsnachricht abzulehnen, die Amazon Cognito normalerweise an neue Benutzer sendet. Wenn Ihre Funktion diesen Parameter nicht zurückgibt, sendet Amazon Cognito die Begrüßungsnachricht.

desiredDeliveryMediums

Sie können diesen Parameter auf `EMAIL` festlegen, um die Begrüßungsmeldung per E-Mail zu senden, oder auf `SMS`, um die Begrüßungsnachricht per SMS zu senden. Wenn Ihre Funktion diesen Parameter nicht zurückgibt, sendet Amazon Cognito die Begrüßungsnachricht per SMS.

forceAliasCreation

Wenn Sie diesen Parameter auf `TRUE` setzen und die im Parameter UserAttributes angegebene Telefonnummer oder E-Mail-Adresse bereits als Alias mit einem anderen Benutzer vorhanden ist, migriert der API-Aufruf den Alias vom vorherigen Benutzer zum neu erstellten Benutzer. Der vorherige Benutzer kann sich nicht mehr mit diesem Alias anmelden.

Wenn Sie diesen Parameter auf `FALSE` setzen und der Alias existiert, migriert Amazon Cognito den Benutzer nicht und gibt einen Fehler an die Client-App zurück.

Wenn Sie diesen Parameter nicht zurückgeben, geht Amazon Cognito davon aus, dass der Wert „falsch“ ist.

enableSMSMFA

Legen Sie diesen Parameter auf `true` fest, damit der migrierte Benutzer für die Anmeldung die Multi-Faktor-Authentifizierung (MFA) per SMS-Nachricht abschließen muss. In Ihrem Benutzerpool muss MFA aktiviert sein. Die Attribute des Benutzers in den Anforderungsparametern müssen eine Telefonnummer enthalten, andernfalls schlägt die Migration dieses Benutzers fehl.

Beispiel: Migration eines Benutzers mit einem bestehenden Passwort

Die Lambda-Funktion in diesem Beispiel migriert der Benutzer mit einem bestehenden Passwort und unterdrückt die Begrüßungsmeldung von Amazon Cognito.

Node.js

```
const validUsers = {
  belladonna: { password: "Test123", emailAddress: "bella@example.com" },
};
```

```
// Replace this mock with a call to a real authentication service.
const authenticateUser = (username, password) => {
  if (validUsers[username] && validUsers[username].password === password) {
    return validUsers[username];
  } else {
    return null;
  }
};

const lookupUser = (username) => {
  const user = validUsers[username];

  if (user) {
    return { emailAddress: user.emailAddress };
  } else {
    return null;
  }
};

const handler = async (event) => {
  if (event.triggerSource == "UserMigration_Authentication") {
    // Authenticate the user with your existing user directory service
    const user = authenticateUser(event.userName, event.request.password);
    if (user) {
      event.response.userAttributes = {
        email: user.emailAddress,
        email_verified: "true",
      };
      event.response.finalUserStatus = "CONFIRMED";
      event.response.messageAction = "SUPPRESS";
    }
  } else if (event.triggerSource == "UserMigration_ForgotPassword") {
    // Look up the user in your existing user directory service
    const user = lookupUser(event.userName);
    if (user) {
      event.response.userAttributes = {
        email: user.emailAddress,
        // Required to enable password-reset code to be sent to user
        email_verified: "true",
      };
      event.response.messageAction = "SUPPRESS";
    }
  }
```

```
  return event;
};

export { handler };
```

Lambda-Auslöser für benutzerdefinierte Nachrichten

Amazon Cognito ruft diesen Auslöser vor dem Senden einer E-Mail, einer Nachricht zur Telefonverifizierung oder eines MFA-Codes (Multi-Factor Authentication, Multifaktor-Authentifizierung) auf. Sie können die Nachricht mit Ihrem Auslöser für benutzerdefinierte Nachrichten dynamisch anpassen. Sie können statische benutzerdefinierte Nachrichten können auf der Registerkarte Nachrichtenanpassungen der ursprünglichen Amazon-Cognito-Konsole bearbeiten.

Die Anforderung beinhaltet `codeParameter`. Dies ist eine Zeichenfolge, die als Platzhalter für den von Amazon Cognito dem Benutzerzugestellten Code dient. Fügen Sie die Zeichenfolge `codeParameter` dort im Nachrichtentext ein, an der der Verifizierungscode erscheinen soll. Wenn Amazon Cognito diese Antwort empfängt, ersetzt Amazon Cognito die Zeichenfolge `codeParameter` durch den tatsächlichen Verifizierungscode.

> **Note**
>
> Eine Lambda-Funktion für benutzerdefinierte Nachrichten mit der `CustomMessage_AdminCreateUser`-Auslöserquelle gibt einen Benutzernamen und einen Verifizierungscode zurück. Da ein von einem Administrator erstellter Benutzer sowohl seinen Benutzernamen als auch seinen Code erhalten muss, muss die Antwort der Funktion sowohl `request.usernameParameter` als auch `request.codeParameter` enthalten.

Themen

- Lambda-Auslöserquellen für benutzerdefinierte Nachrichten
- Lambda-Auslöserparameter für benutzerdefinierte Nachrichten
- Benutzerdefinierte Nachricht für das Registrierbeispiel
- Benutzerdefinierte Nachricht für das Beispiel, bei dem der Administrator einen Benutzer anlegt

Lambda-Auslöserquellen für benutzerdefinierte Nachrichten

triggerSource-Wert	Ereignis
`CustomMessage_SignUp`	Benutzerdefinierte Nachricht – Zum Senden des Bestätigungscodes nach der Anmeldung.
`CustomMessage_AdminCreateUser`	Benutzerdefinierte Nachricht – Zum Senden des temporären Passworts an einen neuen Benutzer.
`CustomMessage_ResendCode`	Benutzerdefinierte Nachricht – Zum erneuten Senden des Bestätigungscodes an einen vorhandenen Benutzer.
`CustomMessage_ForgotPassword`	Benutzerdefinierte Nachricht – Zum Senden des Bestätigungscodes für die Anforderung „Passwort vergessen“.
`CustomMessage_UpdateUserAttribute`	Benutzerdefinierte Nachricht – Wenn die E-Mail-Adresse oder die Telefonnummer eines Benutzers geändert wird, sendet dieser Auslöser automatisch einen Verifizierungscode an den Benutzer. Kann nicht für andere Attribute verwendet werden.
`CustomMessage_VerifyUserAttribute`	Benutzerdefinierte Nachricht – Dieser Auslöser sendet einen Verifizierungscode an Benutzer, wenn sie ihn manuell für eine neue E-Mail-Adresse oder Telefonnummer anfordern.
`CustomMessage_Authentication`	Benutzerdefinierte Nachricht – Zum Senden eines MFA-Codes während der Authentifizierung.

Lambda-Auslöserparameter für benutzerdefinierte Nachrichten

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        }
        "codeParameter": "####",
        "usernameParameter": "string",
        "clientMetadata": {
            "string": "string",
            . . .
        }
    },
    "response": {
        "smsMessage": "string",
        "emailMessage": "string",
        "emailSubject": "string"
    }
}
```

Anforderungsparameter für benutzerdefinierte Nachrichten

userAttributes

Ein oder mehrere Name-Wert-Paare, die Benutzerattribute darstellen.

codeParameter

Eine Zeichenfolge, die Sie als Platzhalter für den Verifizierungscode in der benutzerdefinierten Nachricht verwenden.

usernameParameter

Der Benutzername Amazon Cognito bezieht diesen Parameter in Anfragen von Benutzern ein, die von einem Administrator erstellt wurden.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für die Lambda-Funktion bereitstellen können, die Sie für den Auslöser für benutzerdefinierte Nachrichten angeben. Die Anforderung, die eine benutzerdefinierte Nachrichtenfunktion aufruft, enthält keine Daten, die im ClientMetadata Parameter in - AdminInitiateAuth und InitiateAuth-API-Operationen übergeben wurden. Um diese Daten an Ihre Lambda-Funktion zu übergeben, können Sie den ClientMetadata Parameter in den folgenden API-Aktionen verwenden:

- AdminResetUserPassword
- AdminRespondToAuthChallenge
- AdminUpdateUserAttributes
- ForgotPassword
- GetUserAttributeVerificationCode
- ResendConfirmationCode
- SignUp
- UpdateUserAttributes

Antwortparameter für benutzerdefinierte Nachrichten

In der Antwort geben Sie den benutzerdefinierten Text an, der in Nachrichten an Ihre Benutzer enthalten sein soll. Informationen zu den Zeichenfolgeneinschränkungen, die Amazon Cognito auf diese Parameter anwendet, finden Sie unter MessageTemplateType.

smsMessage

Die benutzerdefinierte SMS-Nachricht, die an die Benutzer gesendet wird. Muss den in der Anforderung erhaltenen `codeParameter`-Wert enthalten, den Sie empfangen haben.

emailMessage

Die benutzerdefinierte E-Mail-Nachricht, die an Ihre Benutzer gesendet werden soll. Sie können die HTML-Formatierung im `emailMessage`-Parameter verwenden. Muss den in der Anforderung erhaltenen `codeParameter`-Wert enthalten, den Sie als die Variable `{####}` empfangen haben. Amazon Cognito kann den `emailMessage`-Parameter nur verwenden, wenn das `EmailSendingAccount`-Attribut des Benutzerpools `DEVELOPER` lautet. Wenn das `EmailSendingAccount`-Attribut des Benutzerpools nicht `DEVELOPER` lautet und ein `emailMessage`-Parameter ausgegeben wird, generiert Amazon Cognito einen 400-Fehlercode

`com.amazonaws.cognito.identity.idp.model.InvalidLambdaResponseException`. Wenn Sie den Amazon Simple Email Service (Amazon SES) zum Senden von E-Mail-Nachrichten verwenden möchten, lautet das `EmailSendingAccount`-Attribut eines Benutzerpools `DEVELOPER`. Andernfalls lautet der Wert `COGNITO_DEFAULT`.

emailSubject

Die Betreffzeile für die benutzerdefinierte Nachricht. Sie können den `emailSubject` Parameter nur verwenden, wenn das EmailSendingAccount Attribut des Benutzerpools lautet`DEVELOPER`. Wenn das `EmailSendingAccount`-Attribut des Benutzerpools nicht `DEVELOPER` lautet und Amazon Cognito einen `emailSubject`-Parameter ausgibt, generiert Amazon Cognito einen 400-Fehlercode `com.amazonaws.cognito.identity.idp.model.InvalidLambdaResponseException`. Wenn Sie den Amazon Simple Email Service (Amazon SES) zum Senden von E-Mail-Nachrichten verwenden möchten, lautet das `EmailSendingAccount`-Attribut eines Benutzerpools `DEVELOPER`. Andernfalls lautet der Wert `COGNITO_DEFAULT`.

Benutzerdefinierte Nachricht für das Registrierbeispiel

Diese Lambda-Beispielfunktion passt eine benutzerdefinierte E-Mail oder SMS-Nachricht an, wenn der Service eine App zum Senden eines Verifizierungscodes an den Benutzer benötigt.

Amazon Cognito kann einen Lambda-Auslöser bei verschiedenen Ereignissen aufrufen: nach der Registrierung, beim erneuten Senden eines Verifizierungscodes, bei einem vergessenen Passwort oder bei der Verifizierung eines Benutzerattributs. Die Antwort umfasst Nachrichten sowohl für SMS als auch für E-Mail. Die Nachricht muss den Codeparameter "`####`" enthalten. Dieser Parameter ist der Platzhalter für den Verifizierungscode, den der Benutzer erhält.

Die maximale Länge einer E-Mail-Nachricht beträgt 20.000 UTF-8-Zeichen. Diese Länge beinhaltet den Verifizierungscode. In diesen E-Mail-Nachrichten können Sie HTML-Tags verwenden.

Die maximale Länge von SMS-Nachrichten beträgt 140 UTF-8-Zeichen. Diese Länge beinhaltet den Verifizierungscode.

Node.js

```
const handler = async (event) => {
  if (event.triggerSource === "CustomMessage_SignUp") {
    const message = `Thank you for signing up. Your confirmation code is
 ${event.request.codeParameter}.`;
```

```
    event.response.smsMessage = message;
    event.response.emailMessage = message;
    event.response.emailSubject = "Welcome to the service.";
  }
  return event;
};

export { handler };
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
  "version": 1,
  "triggerSource": "CustomMessage_SignUp/CustomMessage_ResendCode/
CustomMessage_ForgotPassword/CustomMessage_VerifyUserAttribute",
  "region": "<region>",
  "userPoolId": "<userPoolId>",
  "userName": "<userName>",
  "callerContext": {
      "awsSdk": "<calling aws sdk with version>",
      "clientId": "<apps client id>",
      ...
  },
  "request": {
      "userAttributes": {
          "phone_number_verified": false,
          "email_verified": true,
           ...
      },
      "codeParameter": "####"
  },
  "response": {
      "smsMessage": "<custom message to be sent in the message with code parameter>"
      "emailMessage": "<custom message to be sent in the message with code
 parameter>"
      "emailSubject": "<custom email subject>"
  }
```

```
}
```

Benutzerdefinierte Nachricht für das Beispiel, bei dem der Administrator einen Benutzer anlegt

Die Anforderung, die Amazon Cognito an diese Beispiel-Lambda-Funktion für benutzerdefinierte Nachrichten gesendet hat, hat den `triggerSource` Wert `CustomMessage_AdminCreateUser` sowie einen Benutzernamen und ein temporäres Passwort. Die Funktion wird `${event.request.codeParameter}` aus dem temporären Passwort in der Anforderung und `${event.request.usernameParameter}` aus dem Benutzernamen in der Anforderung ausgefüllt.

Ihre benutzerdefinierten Nachrichten müssen die Werte von `codeParameter` und `usernameParameter` in `smsMessage` und `emailMessage` im Antwortobjekt einfügen. In diesem Beispiel schreibt die Funktion dieselbe Nachricht in die Antwortfelder `event.response.smsMessage` und `event.response.emailMessage`.

Die maximale Länge einer E-Mail-Nachricht beträgt 20.000 UTF-8-Zeichen. Diese Länge beinhaltet den Verifizierungscode. Sie können HTML-Tags in diesen E-Mails verwenden. Die maximale Länge von SMS-Nachrichten beträgt 140 UTF-8-Zeichen. Diese Länge beinhaltet den Verifizierungscode.

Die Antwort umfasst Nachrichten sowohl für SMS als auch für E-Mail.

Node.js

```
const handler = async (event) => {
  if (event.triggerSource === "CustomMessage_AdminCreateUser") {
    const message = `Welcome to the service. Your user name is
 ${event.request.usernameParameter}. Your temporary password is
 ${event.request.codeParameter}`;
    event.response.smsMessage = message;
    event.response.emailMessage = message;
    event.response.emailSubject = "Welcome to the service";
  }
  return event;
};

export { handler }
```

Amazon Cognito übergibt Ereignisinformationen an Ihre Lambda-Funktion. Die Funktion gibt dann das gleiche Ereignisobjekt mit allen Änderungen in der Antwort an Amazon Cognito zurück. Sie können in der Lambda-Konsole ein Testereignis mit den für Ihren Lambda-Auslöser relevanten Daten einrichten. Das Folgende ist ein Testereignis für dieses Codebeispiel:

JSON

```
{
  "version": 1,
  "triggerSource": "CustomMessage_AdminCreateUser",
  "region": "<region>",
  "userPoolId": "<userPoolId>",
  "userName": "<userName>",
  "callerContext": {
      "awsSdk": "<calling aws sdk with version>",
      "clientId": "<apps client id>",
      ...
  },
  "request": {
      "userAttributes": {
          "phone_number_verified": false,
          "email_verified": true,
           ...
      },
      "codeParameter": "####",
      "usernameParameter": "username"
  },
  "response": {
      "smsMessage": "<custom message to be sent in the message with code parameter
 and username parameter>"
      "emailMessage": "<custom message to be sent in the message with code parameter
 and username parameter>"
      "emailSubject": "<custom email subject>"
  }
}
```

Benutzerdefinierter Lambda-Auslöser für Sender

Amazon-Cognito-Benutzerpools bieten die Lambda-Auslöser `CustomEmailSender` und `CustomSMSSender`, um E-Mail- und SMS-Benachrichtigungen von Drittanbietern zu aktivieren. Sie können SMS- und E-Mail-Anbieter auswählen, um Benachrichtigungen an Benutzer aus

Ihrem Lambda-Funktionscode zu senden. Wenn Amazon Cognito Benachrichtigungen wie Bestätigungscodes, Verifizierungscodes oder temporäre Passwörter an Benutzer senden muss, aktivieren die Ereignisse Ihre konfigurierten Lambda-Funktionen. Amazon Cognito sendet den Code und die temporären Passwörter (Geheimnisse) an Ihre aktivierten Lambda-Funktionen. Amazon Cognito verschlüsselt diese Geheimnisse mit einem AWS KMS vom Kunden verwalteten Schlüssel und der AWS Encryption SDK. Die AWS Encryption SDK ist eine clientseitige Verschlüsselungsbibliothek, die Ihnen hilft, generische Daten zu verschlüsseln und zu entschlüsseln.

> **Note**
>
> Sie können das AWS CLI oder SDK verwenden, um Ihre Benutzerpools für die Verwendung dieser Lambda-Auslöser zu konfigurieren. Diese Konfigurationen sind über die Amazon-Cognito-Konsole nicht verfügbar.

CustomEmailSender

Amazon Cognito ruft diesen Auslöser auf, um E-Mail-Benachrichtigungen an Benutzer zu senden.

CustomSMSSender

Amazon Cognito ruft diesen Auslöser auf, um SMS-Benachrichtigungen an Benutzer zu senden.

Ressourcen

Die folgenden Ressourcen unterstützen Sie bei der Verwendung des `CustomEmailSender`- und der `CustomSMSSender`-Auslöser.

AWS KMS

AWS KMS ist ein verwalteter Service zum Erstellen und Steuern von AWS KMS-Schlüsseln. Diese Schlüssel verschlüsseln Ihre Daten. Weitere Informationen finden Sie unter Was ist AWS Key Management Service?.

KMS-Schlüssel

Ein KMS-Schlüssel ist eine logische Darstellung eines kryptografischen Schlüssels. Der KMS-Schlüssel enthält Metadaten wie die Schlüssel-ID, das Erstellungsdatum, die Beschreibung und den Schlüsselstatus. Der KMS-Schlüssel enthält auch das zur Ver- und Entschlüsselung von Daten verwendete Schlüsselmaterial. Weitere Informationen finden Sie unter Löschen von AWS-KMS-Schlüsseln.

Symmetrische KMS-Schlüssel

Ein symmetrischer KMS-Schlüssel ist ein 256-Bit-Verschlüsselungsschlüssel, der AWS KMS nicht unverschlüsselt verlässt. Zur Verwendung eines symmetrischen KMS-Schlüssels müssen Sie AWS KMS aufrufen. Amazon Cognito verwendet symmetrische Schlüssel. Derselbe Schlüssel verschlüsselt und entschlüsselt. Weitere Informationen finden Sie unter Symmetrische KMS-Schlüssel.

Benutzerdefinierter Lambda-Auslöser für E-Mail-Sender

Wenn Sie Ihrem Benutzerpool einen benutzerdefinierten E-Mail-Absender-Trigger zuweisen, ruft Amazon Cognito statt des Standardverhaltens eine Lambda-Funktion auf, wenn ein Benutzerereignis das Senden einer E-Mail-Nachricht erfordert. Mit einem benutzerdefinierten Absender-Trigger kann Ihre AWS Lambda-Funktion E-Mail-Benachrichtigungen über eine von Ihnen gewählte Methode und einen Anbieter Ihrer Wahl an Ihre Benutzer senden. Der benutzerdefinierte Code Ihrer Funktion muss alle E-Mail-Nachrichten aus Ihrem Benutzerpool verarbeiten und versenden.

> **Note**
>
> Derzeit können Sie in der Amazon-Cognito-Konsole keine benutzerdefinierten Absender-Trigger zuweisen. Sie können einen Auslöser mit dem `LambdaConfig`-Parameter in einer `CreateUserPool`- oder einer `UpdateUserPool`-API-Anfrage zuordnen.

Führen Sie zur Einrichtung dieses Auslösers die folgenden Schritte aus:

1. Erstellen Sie einen Schlüssel für die symmetrische Verschlüsselung in AWS Key Management Service (AWS KMS). Amazon Cognito generiert Geheimnisse – temporäre Passwörter, Verifizierungs- und Bestätigungscodes – und verwendet diesen KMS-Schlüssel dann zum Verschlüsseln der Geheimnisse. Anschließend können Sie die API-Operation Decrypt in Ihrer Lambda-Funktion verwenden, um die Geheimnisse zu entschlüsseln und im Klartext an den Benutzer zu senden. AWS Encryption SDK ist ein nützliches Tool für AWS KMS-Operationen in Ihrer Funktion.
2. Erstellen Sie eine Lambda-Funktion, die Sie als benutzerdefinierten Sender-Auslöser zuordnen möchten. Gewähren Sie der Lambda-Funktionsrolle `kms:Decrypt`-Berechtigungen für Ihren KMS-Schlüssel.

3. Gewähren Sie dem Amazon-Cognito-Serviceprinzipal `cognito-idp.amazonaws.com`-Zugriff, um die Lambda-Funktion aufzurufen.
4. Schreiben Sie Lambda-Funktionscode, der Ihre Nachrichten an benutzerdefinierte Bereitstellungsmethoden oder Drittanbieter weiterleitet. Um den Verifizierungs- oder Bestätigungscode Ihres Benutzers zuzustellen, dekodieren und entschlüsseln Sie den Wert des `code`-Parameters in der Anforderung mit Base64. Diese Operation erzeugt einen Klartextcode oder ein Passwort, den bzw. das Sie in Ihre Nachricht aufnehmen müssen.
5. Aktualisieren Sie den Benutzerpool, damit er einen benutzerdefinierten Lambda-Trigger des Absenders verwendet. Der IAM-Prinzipal, der einen Benutzerpool mit einem benutzerdefinierten Absender-Trigger aktualisiert oder erstellt, muss über die Berechtigung verfügen, eine Erteilung für Ihren KMS-Schlüssel zu erstellen. Der folgende `LambdaConfig`-Ausschnitt weist benutzerdefinierte SMS- und E-Mail-Absenderfunktionen zu.

```
"LambdaConfig": {
   "KMSKeyID": "arn:aws:kms:us-
east-1:123456789012:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
   "CustomEmailSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
      "LambdaVersion": "V1_0"
   },
   "CustomSMSSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
      "LambdaVersion": "V1_0"
   }
```

Benutzerdefinierte Lambda-Auslöserparameter für SMS-Sender

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "type": "customEmailSenderRequestV1",
        "code": "string",
        "clientMetadata": {
            "string": "string",
             . . .
```

```
            },
        "userAttributes": {
            "string": "string",
            . . .
         }
}
```

Benutzerdefinierte E-Mail-Sender-Anforderungsparameter

type

Die Anforderungsversion. Für ein benutzerdefiniertes E-Mail-Sender-Ereignis ist der Wert dieses Strings immer `customEmailSenderRequestV1`.

Code

Der verschlüsselte Code, den Ihre Funktion entschlüsseln und an Ihren Benutzer senden kann.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für den benutzerdefinierten E-Mail-Sender-Auslöser der Lambda-Funktion bereitstellen können. Sie können den ClientMetadata-Parameter in den API-Aktionen AdminRespondToAuthChallenge und RespondToAuthChallenge verwenden, um diese Daten an Ihre Lambda-Funktion zu übergeben. Amazon Cognito enthält keine Daten aus dem ClientMetadata-Parameter in AdminInitiateAuth und InitiateAuth-API-Operationen in der Anforderung, die es an die Funktion nach der Authentifizierung übergibt.

userAttributes

Ein oder mehrere Schlüssel-Wert-Paare, die Benutzerattribute darstellen.

Benutzerdefinierte E-Mail-Sender-Antwortparameter

Amazon Cognito erwartet keine zusätzlichen Rückgabeinformationen in der benutzerdefinierten E-Mail-Sender-Antwort. Ihre Funktion kann API-Operationen verwenden, um Ihre Ressourcen abzufragen und zu ändern oder Ereignismetadaten in einem externen System aufzuzeichnen.

Aktivieren des benutzerdefinierten Lambda-Auslösers für E-Mail-Sender

Zum Einrichten eines benutzerdefinierten E-Mail-Sender-Auslösers, der eine benutzerdefinierte Logik zum Senden von E-Mail-Nachrichten für Ihren Benutzerpool verwendet, aktivieren Sie

den Auslöser wie folgt. Anhand des folgenden Verfahrens weisen Sie Ihrem Benutzerpool einen benutzerdefinierten E-Mail-Auslöser, einen benutzerdefinierten SMS-Auslöser oder beides zu. Nachdem Sie Ihren benutzerdefinierten E-Mail-Sender-Auslöser hinzugefügt haben, sendet Amazon Cognito immer Benutzerattribute, einschließlich der E-Mail-Adresse, sowie den einmaligen Code an Ihre Lambda-Funktion, wenn andernfalls eine E-Mail mit Amazon Simple Notification Service gesendet worden wäre.

> **Important**
>
> Amazon Cognito maskiert reservierte Zeichen wie < (`<`) und > (`>`) im temporären Passwort Ihres Benutzers im HTML-Format. Diese Zeichen können in temporären Passwörtern, die Amazon Cognito an Ihre benutzerdefinierte E-Mail-Senderfunktion sendet, jedoch nicht in temporären Verifizierungscodes vorkommen. Um temporäre Passwörter senden zu können, muss Ihre Lambda-Funktion die Maskierung dieser Zeichen aufheben, nachdem sie das Passwort entschlüsselt hat und bevor sie die Nachricht an Ihren Benutzer sendet.

1. Erstellen Sie einen Verschlüsselungsschlüssel in AWS KMS. Dieser Schlüssel verschlüsselt temporäre Passwörter und Autorisierungscodes, die Amazon Cognito generiert. Sie können diese Secrets dann in der benutzerdefinierten Lambda-Funktion des Senders entschlüsseln, um sie im Klartext an den Benutzer zu senden.
2. Gewähren Sie dem Amazon-Cognito-Serviceprinzipal `cognito-idp.amazonaws.com` Zugriff, um Codes mit dem KMS-Schlüssel zu verschlüsseln.

   Wenden Sie die folgende ressourcenbasierte Richtlinie auf Ihren KMS-Schlüssel an.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": "cognito-idp.amazonaws.com"
        },
        "Action": "kms:CreateGrant",
        "Resource": "arn:aws:kms:us-
west-2:111222333444:key/1example-2222-3333-4444-999example",
        "Condition": {
            "StringEquals": {
```

```
                "aws:SourceAccount": "111222333444"
            },
            "ArnLike": {
                "aws:SourceArn": "arn:aws:cognito-idp:us-
west-2:111222333444:userpool/us-east-1_EXAMPLE"
            }
        }
    }]
}
```

3. Erstellen Sie eine Lambda-Funktion für den benutzerdefinierten Sender-Auslöser. Amazon Cognito verwendet das AWS-Verschlüsselungs-SDK, um die Secrets, temporären Passwörter und Codes zu verschlüsseln, die die API-Anfragen Ihrer Benutzer autorisieren.

   - Weisen Sie Ihrer Lambda-Funktion eine IAM-Rolle zu, die mindestens über `kms:Decrypt`-Berechtigungen für Ihren KMS-Schlüssel verfügt.

4. Gewähren Sie dem Amazon-Cognito-Serviceprinzipal `cognito-idp.amazonaws.com`-Zugriff, um die Lambda-Funktion aufzurufen.

   Der folgende AWS CLI-Befehl gewährt Amazon Cognito die Berechtigung, Ihre Lambda-Funktion aufzurufen:

   ```
   aws lambda add-permission --function-name lambda_arn --statement-id
    "CognitoLambdaInvokeAccess" --action lambda:InvokeFunction --principal cognito-
   idp.amazonaws.com
   ```

5. Erstellen Sie Ihren Lambda-Funktionscode, um Ihre Nachrichten zu senden. Amazon Cognito verwendet AWS Encryption SDK, um Secrets zu verschlüsseln, bevor sie von Amazon Cognito an die benutzerdefinierte Lambda-Funktion des Senders gesendet werden. Entschlüsseln Sie in Ihrer Funktion das Secret und verarbeiten Sie alle relevanten Metadaten. Senden Sie dann den Code, Ihre eigene benutzerdefinierte Nachricht und die Zieltelefonnummer an die benutzerdefinierte API, die Ihre Nachricht übermittelt.

6. Fügen Sie Ihrer Lambda-Funktion das AWS Encryption SDK hinzu. Weitere Informationen finden Sie unter AWS-Verschlüsselungs-SDK – Programmiersprachen. Aktualisieren Sie das Lambda-Paket, indem Sie die folgenden Schritte ausführen.

   a. Exportieren Sie Ihre Lambda-Funktion als ZIP-Datei in die AWS Management Console.

b. Öffnen Sie Ihre Funktion und fügen Sie das AWS Encryption SDK hinzu. Weitere Informationen und Download-Links finden Sie unter AWS Encryption SDK – Programmiersprachen im Entwicklerhandbuch zum AWS Encryption SDK.

c. Komprimieren Sie Ihre Funktion mit Ihren SDK-Abhängigkeiten und laden Sie die Funktion in Lambda hoch. Weitere Informationen finden Sie unter Bereitstellen von Lambda-Funktionen als ZIP-Dateiarchive im Entwicklerhandbuch zu AWS Lambda.

7. Aktualisieren Sie Ihren Benutzerpool, um benutzerdefinierte Lambda-Auslöser hinzuzufügen. Fügen Sie den Parameter `CustomSMSSender` oder `CustomEmailSender` in die API-Anfrage `UpdateUserPool` ein. Die API-Operation `UpdateUserPool` erfordert alle Parameter Ihres Benutzerpools und die Parameter, die Sie ändern möchten. Wenn Sie nicht alle relevanten Parameter angeben, legt Amazon Cognito die Werte aller fehlenden Parameter auf ihre Standardwerte fest. Schließen Sie entsprechend dem folgenden Beispiel Einträge für alle Lambda-Funktionen ein, die Sie Ihrem Benutzerpool hinzufügen oder im Benutzerpool behalten möchten. Weitere Informationen finden Sie unter Aktualisieren der Benutzerpool-Konfiguration.

```
   #Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
   #user pool configurations.

     --lambda-config "PreSignUp=lambda-arn, \
                      CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                      CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn},
\
                      KMSKeyID=key-id"
```

Wenn Sie einen benutzerdefinierten Lambda-Auslöser für Sender mit der AWS CLI `update-user-pool` entfernen möchten, lassen Sie den Parameter `CustomSMSSender` oder `CustomEmailSender` in `--lambda-config` weg und schließen Sie alle anderen Auslöser ein, die Sie für Ihren Benutzerpool verwenden möchten.

Wenn Sie einen benutzerdefinierten Lambda-Auslöser für Sender mit der API-Anfrage `UpdateUserPool` entfernen möchten, löschen Sie `CustomSMSSender` oder `CustomEmailSender` aus dem Anfragetext, der den Rest der Benutzerpool-Konfiguration enthält.

Codebeispiel

Das folgende Beispiel für Node.js verarbeitet ein E–Mail-Nachrichtenereignis in Ihrer benutzerdefinierten Lambda-Funktion für E-Mail-Sender. In diesem Beispiel wird davon ausgegangen, dass Ihre Funktion zwei Umgebungsvariablen definiert hat.

`KEY_ALIAS`

Der Alias des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln von Benutzercodes verwenden möchten.

`KEY_ARN`

Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln von Benutzercodes verwenden möchten.

```
const AWS = require('aws-sdk');
const b64 = require('base64-js');
const encryptionSdk = require('@aws-crypto/client-node');
//Configure the encryption SDK client with the KMS key from the environment variables.
const { encrypt, decrypt } =
 encryptionSdk.buildClient(encryptionSdk.CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT);
const generatorKeyId = process.env.KEY_ALIAS;
const keyIds = [ process.env.KEY_ARN ];
const keyring = new encryptionSdk.KmsKeyringNode({ generatorKeyId, keyIds })
exports.handler = async (event) => {
 //Decrypt the secret code using encryption SDK.
 let plainTextCode;
 if(event.request.code){
  const { plaintext, messageHeader } = await decrypt(keyring,
 b64.toByteArray(event.request.code));
  plainTextCode = plaintext
 }
 //PlainTextCode now contains the decrypted secret.
 if(event.triggerSource == 'CustomEmailSender_SignUp'){
  //Send an email message to your user via a custom provider.
  //Include the temporary password in the message.
 }
 else if(event.triggerSource == 'CustomEmailSender_ResendCode'){
 }
 else if(event.triggerSource == 'CustomEmailSender_ForgotPassword'){
 }
 else if(event.triggerSource == 'CustomEmailSender_UpdateUserAttribute'){
```

```
 }
 else if(event.triggerSource == 'CustomEmailSender_VerifyUserAttribute'){
 }
 else if(event.triggerSource == 'CustomEmailSender_AdminCreateUser'){
 }
 else if(event.triggerSource == 'CustomEmailSender_AccountTakeOverNotification'){
 }
 return;
};
```

Benutzerdefinierte E-Mail-Sender-Lambda-Auslöser-Quellen

Die folgende Tabelle zeigt die auslösenden Ereignisse für benutzerdefinierte E-Mail-Auslöserquellen in Ihrem Lambda-Code.

`TriggerSource value`	Veranstaltung
`CustomEmailSender_SignUp`	Ein Benutzer meldet sich an und Amazon Cognito sendet eine Willkommensnachricht.
`CustomEmailSender_ForgotPassword`	Ein Benutzer fordert einen Code an, um sein Passwort zurückzusetzen.
`CustomEmailSender_ResendCode`	Ein Benutzer fordert einen Ersatzcode an, um sein Passwort zurückzusetzen.
`CustomEmailSender_UpdateUserAttribute`	Ein Benutzer aktualisiert eine E-Mail-Adresse oder ein Telefonnummernattribut und Amazon Cognito sendet einen Code zur Verifizierung des Attributs.
`CustomEmailSender_VerifyUserAttribute`	Ein Benutzer erstellt eine neue E-Mail-Adresse oder ein Telefonnummernattribut und Amazon Cognito sendet einen Code zur Verifizierung des Attributs.
`CustomEmailSender_AdminCreateUser`	Sie erstellen einen neuen Benutzer in Ihrem Benutzerpool und Amazon Cognito sendet ihm ein temporäres Passwort.

`TriggerSource value`	Veranstaltung
`CustomEmailSender_AccountTakeOverNotification`	Amazon Cognito erkennt einen Versuch, ein Benutzerkonto zu übernehmen, und sendet dem Benutzer eine Benachrichtigung.

Benutzerdefinierter Lambda-Auslöser für SMS-Sender

Wenn Sie Ihrem Benutzerpool einen benutzerdefinierten SMS-Absender-Trigger zuweisen, ruft Amazon Cognito statt des Standardverhaltens eine Lambda-Funktion auf, wenn ein Benutzerereignis das Senden einer SMS-Nachricht erfordert. Mit einem benutzerdefinierten Sender-Auslöser kann Ihre AWS Lambda Funktion SMS-Benachrichtigungen über eine von Ihnen gewählte Methode und einen Anbieter an Ihre Benutzer senden. Der benutzerdefinierte Code Ihrer Funktion muss alle SMS-Nachrichten aus Ihrem Benutzerpool verarbeiten und versenden.

> **Note**
>
> Derzeit können Sie in der Amazon-Cognito-Konsole keine benutzerdefinierten Absender-Trigger zuweisen. Sie können einen Auslöser mit dem `LambdaConfig`-Parameter in einer `CreateUserPool`- oder einer `UpdateUserPool`-API-Anfrage zuordnen.

Führen Sie zur Einrichtung dieses Auslösers die folgenden Schritte aus:

1. Erstellen Sie einen symmetrischen Verschlüsselungsschlüssel in AWS Key Management Service (AWS KMS). Amazon Cognito generiert Geheimnisse – temporäre Passwörter, Verifizierungs- und Bestätigungscodes – und verwendet diesen KMS-Schlüssel dann zum Verschlüsseln der Geheimnisse. Anschließend können Sie die API-Operation Decrypt in Ihrer Lambda-Funktion verwenden, um die Geheimnisse zu entschlüsseln und im Klartext an den Benutzer zu senden. ist AWS Encryption SDK ein nützliches Tool für AWS KMS Operationen in Ihrer Funktion.
2. Erstellen Sie eine Lambda-Funktion, die Sie als benutzerdefinierten Sender-Auslöser zuordnen möchten. Gewähren Sie der Lambda-Funktionsrolle `kms:Decrypt`-Berechtigungen für Ihren KMS-Schlüssel.
3. Gewähren Sie dem Amazon-Cognito-Serviceprinzipal `cognito-idp.amazonaws.com`-Zugriff, um die Lambda-Funktion aufzurufen.

4. Schreiben Sie Lambda-Funktionscode, der Ihre Nachrichten an benutzerdefinierte Bereitstellungsmethoden oder Drittanbieter weiterleitet. Um den Verifizierungs- oder Bestätigungscode Ihres Benutzers zuzustellen, dekodieren und entschlüsseln Sie den Wert des `code`-Parameters in der Anforderung mit Base64. Diese Operation erzeugt einen Klartextcode oder ein Passwort, den bzw. das Sie in Ihre Nachricht aufnehmen müssen.
5. Aktualisieren Sie den Benutzerpool, damit er einen benutzerdefinierten Lambda-Trigger des Absenders verwendet. Der IAM-Prinzipal, der einen Benutzerpool mit einem benutzerdefinierten Absender-Trigger aktualisiert oder erstellt, muss über die Berechtigung verfügen, eine Erteilung für Ihren KMS-Schlüssel zu erstellen. Der folgende `LambdaConfig`-Ausschnitt weist benutzerdefinierte SMS- und E-Mail-Absenderfunktionen zu.

```
"LambdaConfig": {
   "KMSKeyID": "arn:aws:kms:us-
east-1:123456789012:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
   "CustomEmailSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
      "LambdaVersion": "V1_0"
   },
   "CustomSMSSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
      "LambdaVersion": "V1_0"
   }
```

Benutzerdefinierte Lambda-Auslöserparameter für SMS-Sender

Die Anforderung, die Amazon Cognito an diese Lambda-Funktion übergibt, ist eine Kombination der folgenden Parameter und der allgemeinen Parameter, die Amazon Cognito allen Anfragen hinzufügt.

JSON

```
{
    "request": {
        "type": "customSMSSenderRequestV1",
        "code": "string",
        "clientMetadata": {
            "string": "string",
             . . .
            },
        "userAttributes": {
            "string": "string",
```

```
            . . .
        }
}
```

Benutzerdefinierte SMS-Sender-Anforderungsparameter

Typ

Die Anforderungsversion. Für ein benutzerdefiniertes SMS-Sender-Ereignis ist der Wert dieses Strings immer `customSMSSenderRequestV1`.

Code

Der verschlüsselte Code, den Ihre Funktion entschlüsseln und an Ihren Benutzer senden kann.

clientMetadata

Ein oder mehrere Schlüssel-Wert-Paare, die Sie als benutzerdefinierte Eingabe für den benutzerdefinierten SMS-Sender-Auslöser der Lambda-Funktion bereitstellen können. Um diese Daten an Ihre Lambda-Funktion zu übergeben, können Sie den - ClientMetadata Parameter in den - AdminRespondToAuthChallenge und RespondToAuthChallenge-API-Aktionen verwenden. Amazon Cognito schließt keine Daten aus dem - ClientMetadata Parameter in - AdminInitiateAuth und InitiateAuth-API-Operationen in die Anforderung ein, die es an die Funktion nach der Authentifizierung übergibt.

userAttributes

Ein oder mehrere Schlüssel-Wert-Paare, die Benutzerattribute darstellen.

Benutzerdefinierte SMS-Sender-Antwortparameter

Amazon Cognito erwartet keine zusätzlichen Rückgabeinformationen in der Antwort. Ihre Funktion kann API-Operationen verwenden, um Ihre Ressourcen abzufragen und zu ändern oder Ereignismetadaten in einem externen System aufzuzeichnen.

Aktivieren des benutzerdefinierten Lambda-Auslösers für SMS-Sender

Sie können einen benutzerdefinierten SMS-Sender-Auslöser einrichten, der eine benutzerdefinierte Logik zum Senden von SMS-Nachrichten für Ihren Benutzerpool verwendet. Anhand des folgenden Verfahrens weisen Sie Ihrem Benutzerpool einen benutzerdefinierten SMS-Auslöser, einen benutzerdefinierten E-Mail-Auslöser oder beides zu. Nachdem Sie Ihren benutzerdefinierten SMS-Sender-Auslöser hinzugefügt haben, sendet Amazon Cognito immer Benutzerattribute,

einschließlich der Telefonnummer, sowie den einmaligen Code an Ihre Lambda-Funktion, anstelle des Standardverhaltens, das eine SMS-Nachricht mit Amazon Simple Notification Service sendet.

> **Important**
>
> Amazon Cognito maskiert reservierte Zeichen wie < (`<`) und > (`>`) im temporären Passwort Ihres Benutzers im HTML-Format. Diese Zeichen können in temporären Passwörtern, die Amazon Cognito an Ihre benutzerdefinierte E-Mail-Senderfunktion sendet, jedoch nicht in temporären Verifizierungscodes vorkommen. Um temporäre Passwörter senden zu können, muss Ihre Lambda-Funktion die Maskierung dieser Zeichen aufheben, nachdem sie das Passwort entschlüsselt hat und bevor sie die Nachricht an Ihren Benutzer sendet.

1. Erstellen Sie einen Verschlüsselungsschlüssel in AWS KMS. Dieser Schlüssel verschlüsselt temporäre Passwörter und Autorisierungscodes, die Amazon Cognito generiert. Sie können diese Secrets dann in der benutzerdefinierten Lambda-Funktion des Senders entschlüsseln, um sie im Klartext an den Benutzer zu senden.
2. Gewähren Sie dem Amazon-Cognito-Serviceprinzipal `cognito-idp.amazonaws.com` Zugriff, um Codes mit dem KMS-Schlüssel zu verschlüsseln.

   Wenden Sie die folgende ressourcenbasierte Richtlinie auf Ihren KMS-Schlüssel an.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": "cognito-idp.amazonaws.com"
        },
        "Action": "kms:CreateGrant",
        "Resource": "arn:aws:kms:us-
west-2:111222333444:key/1example-2222-3333-4444-999example",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "111222333444"
            },
            "ArnLike": {
                "aws:SourceArn": "arn:aws:cognito-idp:us-
west-2:111222333444:userpool/us-east-1_EXAMPLE"
```

```
            }
        }
    }]
}
```

3. Erstellen Sie eine Lambda-Funktion für den benutzerdefinierten Sender-Auslöser. Amazon Cognito verwendet das AWS -Verschlüsselungs-SDK, um die Secrets, temporären Passwörter und Codes zu verschlüsseln, die die API-Anfragen Ihrer Benutzer autorisieren.

   - Weisen Sie Ihrer Lambda-Funktion eine IAM-Rolle zu, die mindestens über `kms:Decrypt`-Berechtigungen für Ihren KMS-Schlüssel verfügt.

4. Gewähren Sie dem Amazon-Cognito-Serviceprinzipal `cognito-idp.amazonaws.com`-Zugriff, um die Lambda-Funktion aufzurufen.

   Der folgende AWS CLI Befehl erteilt Amazon Cognito die Berechtigung zum Aufrufen Ihrer Lambda-Funktion:

```
        aws lambda add-permission --function-name lambda_arn --statement-id
 "CognitoLambdaInvokeAccess" --action lambda:InvokeFunction --principal cognito-
idp.amazonaws.com
```

5. Erstellen Sie Ihren Lambda-Funktionscode, um Ihre Nachrichten zu senden. Amazon Cognito verwendet AWS Encryption SDK , um Secrets zu verschlüsseln, bevor Amazon Cognito die Secrets an die benutzerdefinierte Lambda-Funktion des Senders sendet. Entschlüsseln Sie in Ihrer Funktion das Secret und verarbeiten Sie alle relevanten Metadaten. Senden Sie dann den Code, Ihre eigene benutzerdefinierte Nachricht und die Zieltelefonnummer an die benutzerdefinierte API, die Ihre Nachricht übermittelt.
6. Fügen Sie die AWS Encryption SDK zu Ihrer Lambda-Funktion hinzu. Weitere Informationen finden Sie unter AWS -Verschlüsselungs-SDK – Programmiersprachen. Aktualisieren Sie das Lambda-Paket, indem Sie die folgenden Schritte ausführen.

   a. Exportieren Sie Ihre Lambda-Funktion als ZIP-Datei in die AWS Management Console.

   b. Öffnen Sie Ihre Funktion und fügen Sie hinzu AWS Encryption SDK. Weitere Informationen und Download-Links finden Sie unter AWS Encryption SDK – Programmiersprachen im Entwicklerhandbuch zum AWS Encryption SDK .

c. Komprimieren Sie Ihre Funktion mit Ihren SDK-Abhängigkeiten und laden Sie die Funktion in Lambda hoch. Weitere Informationen finden Sie unter Bereitstellen von Lambda-Funktionen als ZIP-Dateiarchive im Entwicklerhandbuch zu AWS Lambda .

7. Aktualisieren Sie Ihren Benutzerpool, um benutzerdefinierte Lambda-Auslöser hinzuzufügen. Fügen Sie den Parameter `CustomSMSSender` oder `CustomEmailSender` in die API-Anfrage `UpdateUserPool` ein. Die API-Operation `UpdateUserPool` erfordert alle Parameter Ihres Benutzerpools und die Parameter, die Sie ändern möchten. Wenn Sie nicht alle relevanten Parameter angeben, legt Amazon Cognito die Werte aller fehlenden Parameter auf ihre Standardwerte fest. Schließen Sie entsprechend dem folgenden Beispiel Einträge für alle Lambda-Funktionen ein, die Sie Ihrem Benutzerpool hinzufügen oder im Benutzerpool behalten möchten. Weitere Informationen finden Sie unter Aktualisieren der Benutzerpool-Konfiguration.

```
   #Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
   #user pool configurations.

     --lambda-config "PreSignUp=lambda-arn, \
                      CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                      CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn},
\
                      KMSKeyID=key-id"
```

Um einen benutzerdefinierten Lambda-Auslöser für Sender mit einem zu entfernen`update-user-pool` AWS CLI, lassen Sie den `CustomEmailSender` Parameter `CustomSMSSender` oder in weg und schließen Sie alle anderen Auslöser ein`--lambda-config`, die Sie mit Ihrem Benutzerpool verwenden möchten.

Wenn Sie einen benutzerdefinierten Lambda-Auslöser für Sender mit der API-Anfrage `UpdateUserPool` entfernen möchten, löschen Sie `CustomSMSSender` oder `CustomEmailSender` aus dem Anfragetext, der den Rest der Benutzerpool-Konfiguration enthält.

Codebeispiel

Das folgende Beispiel für Node.js verarbeitet ein SMS-Nachrichtenereignis in Ihrer benutzerdefinierten Lambda-Funktion für SMS-Sender. In diesem Beispiel wird davon ausgegangen, dass Ihre Funktion zwei Umgebungsvariablen definiert hat.

KEY_ALIAS

Der Alias des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln von Benutzercodes verwenden möchten.

KEY_ARN

Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln von Benutzercodes verwenden möchten.

```
const AWS = require('aws-sdk');
const b64 = require('base64-js');
const encryptionSdk = require('@aws-crypto/client-node');
//Configure the encryption SDK client with the KMS key from the environment variables.

const { encrypt, decrypt } =
 encryptionSdk.buildClient(encryptionSdk.CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT);
const generatorKeyId = process.env.KEY_ALIAS;
const keyIds = [ process.env.KEY_ARN ];
const keyring = new encryptionSdk.KmsKeyringNode({ generatorKeyId, keyIds })
exports.handler = async (event) => {
 //Decrypt the secret code using encryption SDK.
 let plainTextCode;
 if(event.request.code){
  const { plaintext, messageHeader } = await decrypt(keyring,
 b64.toByteArray(event.request.code));
  plainTextCode = plaintext
 }
 //PlainTextCode now contains the decrypted secret.
 if(event.triggerSource == 'CustomSMSSender_SignUp'){
  //Send an SMS message to your user via a custom provider.
  //Include the temporary password in the message.
 }
 else if(event.triggerSource == 'CustomSMSSender_ResendCode'){
 }
 else if(event.triggerSource == 'CustomSMSSender_ForgotPassword'){
 }
 else if(event.triggerSource == 'CustomSMSSender_UpdateUserAttribute'){
 }
 else if(event.triggerSource == 'CustomSMSSender_VerifyUserAttribute'){
 }
 else if(event.triggerSource == 'CustomSMSSender_AdminCreateUser'){
 }
```

```
  else if(event.triggerSource == 'CustomSMSSender_AccountTakeOverNotification'){
  }
  return;
};
```

Themen

- Bewerten von SMS-Nachrichtenfunktionen mit einer benutzerdefinierten SMS-Sender-Funktion
- Benutzerdefinierte SMS-Sender-Lambda-Auslöser-Quellen

Bewerten von SMS-Nachrichtenfunktionen mit einer benutzerdefinierten SMS-Sender-Funktion

Eine benutzerdefinierte Lambda-Funktion für SMS-Sender akzeptiert die SMS-Nachrichten, die Ihr Benutzerpool senden würde, und die Funktion stellt den Inhalt basierend auf Ihrer benutzerdefinierten Logik bereit. Amazon Cognito sendet die Benutzerdefinierte Lambda-Auslöserparameter für SMS-Sender an Ihre Funktion. Ihre Funktion kann diese Informationen nach Ihren Wünschen verarbeiten. Sie können den Code beispielsweise an ein Amazon Simple Notification Service (Amazon SNS)-Thema senden. Ein Abonnent eines Amazon-SNS-Themas kann eine SMS-Nachricht, ein HTTPS-Endpunkt oder eine E-Mail-Adresse sein.

Informationen zum Erstellen einer Testumgebung für Amazon Cognito-SMS-Messaging mit einer benutzerdefinierten Lambda-Funktion für SMS-Sender finden Sie unter amazon-cognito-user-pool-development-and-testing-with-sms-redirected-to-email in der aws-samples-Bibliothek auf GitHub. Das Repository enthält AWS CloudFormation Vorlagen, die einen neuen Benutzerpool erstellen oder mit einem bereits vorhandenen Benutzerpool arbeiten können. Diese Vorlagen erstellen Lambda-Funktionen und ein Amazon-SNS-Thema. Die Lambda-Funktion, die die Vorlage als benutzerdefinierten SMS-Sender-Auslöser zuweist, leitet Ihre SMS-Nachrichten an die Abonnenten des Amazon-SNS-Themas um.

Wenn Sie diese Lösung in einem Benutzerpool bereitstellen, werden alle Nachrichten, die Amazon Cognito normalerweise mit SNS-Messaging sendet, mit der Lambda-Funktion stattdessen an eine zentrale E-Mail-Adresse gesendet. Verwenden Sie diese Lösung, um SMS-Nachrichten anzupassen und in der Vorschau anzuzeigen und um die Benutzerpool-Ereignisse zu testen, die dazu führen, dass Amazon Cognito eine SMS-Nachricht sendet. Nachdem Sie Ihre Tests abgeschlossen haben, setzen Sie den CloudFormation Stack zurück oder entfernen Sie die benutzerdefinierte SMS-Sender-Funktionszuweisung aus Ihrem Benutzerpool.

> **Important**
>
> Verwenden Sie nicht die Vorlagen in amazon-cognito-user-pool-development-and-testing-with-sms-redirected-to-email, um eine Produktionsumgebung zu erstellen. Die benutzerdefinierte Lambda-Funktion für SMS-Sender in der Lösung simuliert SMS-Nachrichten, die Lambda-Funktion sendet sie jedoch alle an eine einzige zentrale E-Mail-Adresse. Bevor Sie SMS-Nachrichten in einem Amazon-Cognito-Benutzerpool in der Produktion senden können, müssen Sie die unter Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools angegebenen Anforderungen erfüllen.

Benutzerdefinierte SMS-Sender-Lambda-Auslöser-Quellen

Die folgende Tabelle zeigt das auslösende Ereignis für benutzerdefinierte SMS-Auslöserquellen in Ihrem Lambda-Code.

`TriggerSource value`	Ereignis
`CustomSMSSender_SignUp`	Ein Benutzer meldet sich an und Amazon Cognito sendet eine Willkommensnachricht.
`CustomSMSSender_ForgotPassword`	Ein Benutzer fordert einen Code an, um sein Passwort zurückzusetzen.
`CustomSMSSender_ResendCode`	Ein Benutzer fordert einen neuen Code an, um seine Registrierung zu bestätigen.
`CustomSMSSender_VerifyUserAttribute`	Ein Benutzer erstellt eine neue E-Mail-Adresse oder ein Telefonnummernattribut und Amazon Cognito sendet einen Code zur Verifizierung des Attributs.
`CustomSMSSender_UpdateUserAttribute`	Ein Benutzer aktualisiert eine E-Mail-Adresse oder ein Telefonnummernattribut und Amazon Cognito sendet einen Code zur Verifizierung des Attributs.

TriggerSource value	Ereignis
CustomSMSSender_Authentication	Ein Benutzer, der mit SMS MFA (Multifaktor-Authentifizierung) konfiguriert ist, meldet sich an.
CustomSMSSender_AdminCreateUser	Sie erstellen einen neuen Benutzer in Ihrem Benutzerpool und Amazon Cognito sendet ihm ein temporäres Passwort.

Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools

Amazon-Cognito-Benutzertools sind in Amazon Pinpoint integriert, um Analysen für Amazon-Cognito-Benutzerpools bereitzustellen und die Benutzerdaten für Amazon-Pinpoint-Kampagnen zu ergänzen. Amazon Pinpoint bietet Analysen und gezielte Werbekampagnen, um die Nutzerbindung in mobilen Anwendungen mithilfe von Push-Benachrichtigungen zu unterstützen. Mit der Unterstützung von Amazon-Pinpoint-Analysen in Amazon-Cognito-Benutzerpools können Sie Anmeldungen von Benutzerpools, fehlgeschlagene Authentifizierungen, täglich aktive Benutzer (DAUs) und monatlich aktive Benutzer (MAUs) auf der Amazon-Pinpoint-Konsole nachverfolgen. Sie können Daten für unterschiedliche Datenbereiche oder Attribute weiter aufschlüsseln, wie z. B. Geräteplattform, Gerätestandort und App-Version.

Sie können auch benutzerdefinierte Attribute für Ihre App einrichten. Diese können dann verwendet werden, um Ihre Benutzer auf Amazon Pinpoint zu segmentieren und ihnen gezielte Push-Benachrichtigungen zu senden. Wenn Sie auf der Registerkarte Analytics (Analysen) in der Amazon-Cognito-Konsole Share user attribute data with Amazon Pinpoint (Benutzerattributdaten mit Amazon Pinpoint teilen) auswählen, erstellt Amazon Pinpoint zusätzliche Endpunkte für Benutzer-E-Mail-Adressen und -Telefonnummern.

Wenn Sie Amazon-Pinpoint-Analytics in Ihrem Benutzerpool mit der Amazon-Cognito-Konsole aktivieren, erstellen Sie auch eine serviceverknüpfte Rolle, die Amazon Cognito übernimmt, wenn Amazon Pinpoint eine API-Anfrage für Ihren Benutzerpool stellt. Der IAM-Prinzipal, der Ihre Analytics-Konfiguration hinzufügt, muss über CreateServiceLinkedRole-Berechtigungen verfügen. Die serviceverknüpfte Rolle ist AWSServiceRoleForAmazonCognitoIdp. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon Cognito.

Wenn Sie in der Amazon-Cognito-API eine `AnalyticsConfiguration` auf Ihren App-Client anwenden, können Sie Amazon Pinpoint eine benutzerdefinierte IAM-Rolle und eine externe ID zuweisen, um die Rolle zu übernehmen. Die Rolle muss dem `cognito-idp`-Service-Prinzipal vertrauen, und wenn die Richtlinie zur Rollenvertraulichkeit eine externe ID erfordert, muss diese mit Ihrer `AnalyticsConfiguration` übereinstimmen. Sie müssen der Rolle `cognito-idp:Describe*`-Berechtigungen und die folgenden Berechtigungen für Ihr Amazon-Pinpoint-Projekt gewähren.

- `mobiletargeting:UpdateEndpoint`
- `mobiletargeting:PutEvents`

Verfügbarkeit der Amazon-Cognito- und Amazon-Pinpoint-Regionen

Die folgende Tabelle zeigt die AWS-Region-Mappings zwischen Amazon Cognito und Amazon Pinpoint, die eine der folgenden Bedingungen erfüllen.

- Sie können nur ein Amazon-Pinpoint-Projekt in der Region USA Ost (Nord-Virginia) (us-east-1) verwenden.
- Sie können ein Amazon-Pinpoint-Projekt in derselben Region oder in der Region USA Ost (Nord-Virginia) (us-east-1) verwenden.

Standardmäßig kann Amazon Cognito Analysen nur an ein Amazon-Pinpoint-Projekt in derselben AWS-Region senden. Ausnahmen von dieser Regel sind die Regionen in der folgenden Tabelle und Regionen, in denen Amazon Pinpoint nicht verfügbar ist.

Amazon Pinpoint ist in den folgenden Regionen verfügbar. Amazon-Cognito-Benutzerpools in diesen Regionen unterstützen keine Analysen.

- Europa (Mailand)
- Naher Osten (Bahrain)
- Asien-Pazifik (Osaka)
- Israel (Tel Aviv)
- Afrika (Kapstadt)
- Asien-Pazifik (Jakarta)

Die Tabelle zeigt die Beziehung zwischen der Region, in der Sie Ihren Amazon-Cognito-Benutzerpool erstellt haben und die entsprechenden Region in Amazon Pinpoint. Sie müssen Ihr Amazon-Pinpoint-Projekt in einer verfügbaren Region konfigurieren, um es in Amazon Cognito zu integrieren.

Region mit dem Amazon-Cognito-Benutzerpool	Region für das Amazon-Pinpoint-Projekt
ap-northeast-1	us-east-1
ap-northeast-2	us-east-1
ap-south-1	us-east-1, ap-south-1
ap-southeast-1	us-east-1
ap-southeast-2	us-east-1, ap-southeast-2
ca-central-1	us-east-1
eu-central-1	us-east-1, eu-central-1
eu-west-1	us-east-1, eu-west-1
eu-west-2	us-east-1
us-east-1	us-east-1
us-east-2	us-east-1
us-west-2	us-east-1, us-west-2

Beispiele für Regionsmappings

- Wenn Sie einen Benutzerpool in ap-northest-1 erstellen, können Sie Ihr Amazon-Pinpoint-Projekt in us-east-1 erstellen.
- Wenn Sie einen Benutzerpool in ap-south-1 erstellen, können Sie Ihr Amazon-Pinpoint-Projekt entweder in us-east-1 oder in ap-south-1 erstellen.

> **Note**
>
> Für alle AWS-Regionen mit Ausnahme der in der vorherigen Tabelle genannten Regionen kann Amazon Cognito nur ein Amazon-Pinpoint-Projekt verwenden, das sich in derselben Region wie Ihr Benutzerpool befindet. Wenn Amazon Pinpoint in der Region, in der Sie Ihren Benutzerpool erstellt haben, nicht verfügbar und nicht in der Tabelle aufgeführt ist, unterstützt Amazon Cognito keine Amazon-Pinpoint-Analysen in dieser Region. Ausführliche Informationen zu der AWS-Region finden Sie unter Amazon-Pinpoint-Endpunkte und -Kontingente.

Angeben von Amazon-Pinpoint-Analytics-Einstellungen (AWS Management Console)

Sie können Ihren Amazon-Cognito-Benutzerpool so konfigurieren, dass Analysedaten an Amazon Pinpoint gesendet werden. Amazon Cognito sendet nur für lokale Benutzer Analysedaten an Amazon Pinpoint. Nachdem Sie Ihren Benutzerpool für die Verknüpfung mit einem Amazon-Pinpoint-Projekt konfiguriert haben, müssen Sie `AnalyticsMetadata` in Ihre API-Anforderungen aufnehmen. Weitere Informationen finden Sie unter Integrieren Ihrer App in Amazon Pinpoint.

Einstellung der Analysen

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Sie werden möglicherweise aufgefordert, Ihre AWS-Anmeldeinformationen einzugeben.
2. Wählen Sie User Pools (Benutzerpools) und dann einen vorhandenen Benutzerpool aus der Liste aus.
3. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus.
4. Wählen Sie unter App Clients and Analytics (App-Clients und -Analysen) einen vorhandenen App client name (App-Client-Namen) aus der Liste aus.
5. Wählen Sie unter Pinpoint Analytics (Pinpoint-Analysen) die Option Enable (Aktivieren) aus.
6. Wählen Sie eine Pinpoint-Region aus.
7. Wählen Sie ein Amazon-Pinpoint-Projekt oder Create Amazon Pinpoint project (Amazon-Pinpoint-Projekt erstellen) aus.

   > **Note**
   >
   > Die Amazon-Pinpoint-Projekt-ID ist eine für Ihr Amazon-Pinpoint-Projekt eindeutige 32 Zeichen lange Zeichenfolge. Sie wird in der Amazon-Pinpoint-Konsole aufgelistet.

Sie können mehrere Amazon-Cognito-Apps auf ein einziges Amazon-Pinpoint-Projekt abbilden. Jede Amazon-Cognito-App kann jedoch nur auf ein Amazon-Pinpoint-Projekt abgebildet werden.
In Amazon Pinpoint sollte jedes Projekts eine einzelne Anwendung sein. Wenn ein Spieleentwickler beispielsweise zwei Spiele hat, sollte jedes Spiel ein separates Amazon-Pinpoint-Projekt sein, auch wenn beide Spiele denselben Amazon-Cognito-Benutzerpool verwenden. Weitere Informationen zu Amazon-Pinpoint-Projekten finden Sie unter Erstellen eines Projekts in Amazon Pinpoint.

8. Wählen Sie unter User data sharing (Teilen von Benutzerdaten) Share user data with Amazon Pinpoint (Benutzerdaten mit Amazon Pinpoint teilen) aus, wenn Amazon Cognito E-Mail-Adressen und Telefonnummern an Amazon Pinpoint senden und zusätzliche Endpunkte für Benutzer erstellen soll. Nachdem Ihre Benutzer die E-Mail-Adresse und Telefonnummer verifiziert haben, teilt Amazon Cognito diese nur mit Amazon Pinpoint, wenn sie für das Benutzerkonto verfügbar sind.

   Note

   Ein Endpunkt kennzeichnet auf eindeutige Weise ein Benutzergerät, zu dem Sie mit Amazon Pinpoint Push-Benachrichtigungen senden können. Weitere Informationen zu Endpunkten finden Sie unter Addming endpoints (Hinzufügen von Endpunkten) im Amazon-Pinpoint-Entwicklerhandbuch.

9. Wählen Sie Änderungen speichern.

Angeben von Amazon-Pinpoint-Analytics-Einstellungen (AWS CLI- und AWS-API)

Verwenden Sie die folgenden Befehle für die Einstellungen der Amazon-Pinpoint-Analyse für Ihren Benutzerpool.

Die Analyse-Einstellungen für die vorhandene Client-App Ihres Benutzerpools beim Erstellen der App festlegen

- AWS CLI: `aws cognito-idp create-user-pool-client`
- AWS-API: CreateUserPoolClient

Die Analyse-Einstellungen für die vorhandene Client-App Ihres Benutzerpools aktualisieren

- AWS CLI: `aws cognito-idp update-user-pool-client`
- AWS-API: UpdateUserPoolClient

> **Note**
>
> Amazon Cognito unterstützt Integrationen in Regionen, wenn Sie `ApplicationArn` verwenden

Integrieren Ihrer App in Amazon Pinpoint

Sie können Analysemetadaten in Amazon Pinpoint für lokale Benutzer von Amazon Cognito über die Benutzerpool-API veröffentlichen.

Lokale Benutzer

Benutzer, die sich für ein Konto angemeldet haben oder in Ihrem Benutzerpool erstellt wurden, anstatt sich über einen externen Identitätsanbieter (IDP) anzumelden.

Benutzerpool-API

Die Operationen, die Sie in ein AWS-SDK unter Verwendung einer App mit einer benutzerdefinierten Benutzeroberfläche (UI) integrieren können. Sie können keine Analysemetadaten für verbundene oder lokale Benutzer übergeben, die sich über die gehostete Benutzeroberfläche anmelden. Eine Liste der Benutzerpool-API-Vorgänge finden Sie in der Amazon-Cognito-API-Referenz.

Nachdem Sie Ihren Benutzerpool für die Veröffentlichung in einer Kampagne konfiguriert haben, übergibt Amazon Cognito Metadaten für die folgenden API-Vorgänge an Amazon Pinpoint.

- `AdminInitiateAuth`
- `AdminRespondToAuthChallenge`
- `ConfirmForgotPassword`
- `ConfirmSignUp`
- `ForgotPassword`
- `InitiateAuth`

- `ResendConfirmationCode`
- `RespondToAuthChallenge`
- `SignUp`

Wenn Sie Metadaten über die Sitzung Ihres Benutzers an Ihre Amazon-Pinpoint-Kampagne übergeben möchten, nehmen Sie einen `AnalyticsEndpointId`-Wert in den `AnalyticsMetadata`-Parameter Ihrer API-Anfrage auf. Ein JavaScript-Beispiel finden Sie unter Warum wird meine Benutzerpool-Analytik von Amazon Cognito nicht in meinem Amazon-Pinpoint-Dashboard angezeigt? im AWS-Wissenscenter.

Konfigurieren der Benutzerpool-Analysefunktionen

Mit Amazon-Pinpoint-Analysen können Sie Anmeldungen im Amazon-Cognito-Benutzerpool, Anmeldungen, fehlgeschlagene Authentifizierungen, täglich aktive Benutzer (DAUs) und monatlich aktive Benutzer (MAUs) nachverfolgen. Sie können auch mit AWS Mobile SDK for Android oder AWS Mobile SDK for iOS für Ihre App spezifische Benutzerattribute einrichten. Diese können dann verwendet werden, um Ihre Benutzer in Amazon Pinpoint zu segmentieren und ihnen gezielte Push-Benachrichtigungen zu senden.

Auf der Registerkarte App-Integration unter App-Clients und Analysen können Sie zu einem vorhandenen App-Client navigieren oder einen neuen erstellen. In der Konfiguration Ihres App-Clients können Sie ein Amazon-Pinpoint-Projekt spezifizieren, das Sie mit Ihrer App verwenden möchten. Weitere Informationen finden Sie unter Verwenden von Amazon-Pinpoint-Analytics mit Amazon-Cognito-Benutzerpools.

> **Note**
>
> Amazon Pinpoint ist in mehreren AWS-Regionen in Nordamerika, Europa, Asien und Ozeanien verfügbar. Zu den Amazon-Pinpoint-Regionen gehört die Amazon-Pinpoint-API. Wenn eine Amazon-Pinpoint-Region von Amazon Cognito unterstützt wird, sendet Amazon Cognito Ereignisse an Amazon-Pinpoint-Projekte innerhalb derselben Amazon-Pinpoint-Region. Wenn eine Region nicht von Amazon Pinpoint unterstützt wird, unterstützt Amazon Cognito nur das Senden von Ereignissen in us-east-1. Detaillierte Informationen zu Amazon Pinpoint finden Sie unter Amazon Pinpoint endpoints and quotas (Amazon-Pinpoint-Endpunkte und -Kontingente) und Using Amazon Pinpoint analytics with Amazon Cognito user pools (Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools).

Analysen und Kampagnen hinzufügen

1. Wählen Sie Add analytics and campaigns (Analysen und Kampagnen hinzufügen).
2. Wählen Sie Cognito app client (Cognito-App-Client) aus der Liste aus.
3. Um Ihre Amazon-Cognito-App einem Amazon-Pinpoint-Projekt zuzuordnen, wählen Sie das Amazon-Pinpoint-Projekt aus der Liste aus.

   > **Note**
   >
   > Die Amazon-Pinpoint-Projekt-ID ist eine für Ihr Amazon-Pinpoint-Projekt eindeutige 32 Zeichen lange Zeichenfolge. Sie wird in der Amazon-Pinpoint-Konsole aufgelistet.
   > Sie können mehrere Amazon-Cognito-Apps auf ein einziges Amazon-Pinpoint-Projekt abbilden. Jede Amazon-Cognito-App kann jedoch nur auf ein Amazon-Pinpoint-Projekt abgebildet werden.
   > In Amazon Pinpoint sollte jedes Projekts eine einzelne Anwendung sein. Wenn ein Spieleentwickler beispielsweise zwei Spiele hat, sollte jedes Spiel ein separates Amazon-Pinpoint-Projekt sein, auch wenn beide Spiele denselben Amazon-Cognito-Benutzerpool verwenden.

4. Wählen Sie Benutzerattributdaten mit Amazon Pinpoint teilen aus, wenn Amazon Cognito E-Mail-Adressen und Telefonnummern an Amazon Pinpoint senden soll, um zusätzliche Endpunkte für Benutzer zu erstellen.

   > **Note**
   >
   > Ein Endpunkt kennzeichnet auf eindeutige Weise ein Benutzergerät, an das Sie mit Amazon Pinpoint Push-Benachrichtigungen senden können. Weitere Informationen zu Endpunkten finden Sie unter Adding endpoints to Amazon Pinpoint (Hinzufügen von Endpunkten zu Amazon Pinpoint) im Amazon-Pinpoint-Entwicklerhandbuch.

5. Geben Sie eine IAM-Rolle ein, die Sie bereits erstellt haben, oder wählen Sie Neue Rolle erstellen, um eine neue Rolle auf der IAM-Konsole zu erstellen.
6. Wählen Sie Save Changes.
7. Um zusätzliche App-Zuordnungen anzugeben, wählen Sie Add app mapping (App-Zuordnung hinzufügen).
8. Wählen Sie Save Changes.

Verwalten von Benutzern in Ihrem Benutzerpool

Nachdem Sie einen Benutzerpool erstellt haben, können Sie Benutzerkonten erstellen, bestätigen und verwalten. Mit Amazon-Cognito-Benutzerpool-Gruppen können Sie Ihre Benutzer und ihren Zugriff auf Ressourcen durch die Zuweisung von IAM-Rollen zu Gruppen verwalten.

Sie können vorhandene Benutzer mit einem Lambda-Auslöser für die Benutzermigration in einen Benutzerpool importieren. Dieser Ansatz ermöglicht eine nahtlose Migration von Benutzern aus Ihrem vorhandenen Benutzerverzeichnis zu Benutzerpools, wenn diese sich zum ersten Mal bei Ihrem Benutzerpool anmelden.

Themen

Konfigurieren von Richtlinien für die Benutzererstellung

Ihr Benutzerpool kann Benutzern gestatten, sich zu registrieren, oder Sie können sie als Administrator erstellen. Sie können auch kontrollieren, in welchem Umfang der Verifizierungs- und Bestätigungsprozess nach der Registrierung in den Händen Ihrer Benutzer liegt. Beispielsweise möchten Sie eventuell Anmeldungen auf der Grundlage eines externen Validierungsprozesses überprüfen und akzeptieren. Diese Konfiguration bzw. die Richtlinie für die Benutzererstellung durch Administratoren legt auch die Zeit fest, die vergehen muss, bevor ein Benutzer sein Benutzerkonto nicht mehr bestätigen kann.

Amazon Cognito kann als Customer Identity and Access Management (CIAM)-Plattform für Ihre Software die Anforderungen Ihrer öffentlichen Kunden erfüllen. Ein Benutzerpool, der Registrierungen akzeptiert und über einen App-Client mit oder ohne gehostete Benutzeroberfläche verfügt, erstellt ein

Benutzerprofil für jede Person im Internet, die Ihre öffentlich auffindbare App-Client-ID kennt und eine Registrierung anfordert. Ein registriertes Benutzerprofil kann Zugriffs- und Identitätstoken erhalten und auf Ressourcen zugreifen, die Sie für Ihre App autorisiert haben. Bevor Sie die Registrierung in Ihrem Benutzerpool aktivieren, überprüfen Sie Ihre Optionen und stellen Sie sicher, dass die Konfiguration Ihren Sicherheitsstandards entspricht. Stellen Sie die Option Selbstregistrierung aktivieren und `AllowAdminCreateUserOnly` ein, wie in den folgenden Verfahren beschrieben; gehen Sie dabei vorsichtig vor.

AWS Management Console

Die Registerkarte Anmeldeerfahrung Ihres Benutzerpools und der Schritt Konfigurieren der Anmeldeerfahrung des Assistenten zum Erstellen von Benutzerpools enthalten einige Einstellungen für die Registrierung und administrative Erstellung von Benutzern in Ihrem Benutzerpool.

So konfigurieren Sie die Anmeldeerfahrung

1. Wählen Sie unter Von Cognito unterstützte Überprüfung und Bestätigung aus, ob Sie Cognito erlauben, automatisch Nachrichten zur Überprüfung und Bestätigung zu senden. Wenn diese Einstellung aktiviert ist, sendet Amazon Cognito eine E-Mail- oder SMS-Nachricht an neue Benutzer mit einem Code, den sie Ihrem Benutzerpool vorlegen müssen. Dadurch wird bestätigt, dass sie Eigentümer der E-Mail-Adresse oder Telefonnummer sind, das entsprechende Attribut wird als verifiziert festgelegt und das Benutzerkonto für die Anmeldung bestätigt. Die von Ihnen ausgewählten Attribute zur Überprüfung bestimmen die Zustellungsmethoden und Ziele der Bestätigungsnachrichten.
2. Die Überprüfung von Attributänderungen ist nicht wichtig, wenn Sie Benutzer erstellen, sondern bezieht sich auf die Überprüfung von Attributen. Sie können Benutzern, die ihre Anmeldeattribute geändert, aber noch nicht verifiziert haben, gestatten, sich weiterhin entweder mit ihrem neuen oder ihrem ursprünglichen Attributwert anzumelden. Weitere Informationen finden Sie unter Verifizieren, wenn Benutzer ihre E-Mail-Adresse oder Telefonnummer ändern.
3. Unter Erforderliche Attribute werden die Attribute angezeigt, für die ein Wert angegeben werden muss, bevor sich ein Benutzer registrieren kann oder Sie einen Benutzer erstellen können. Sie können die erforderlichen Attribute nur im Assistenten zum Erstellen eines Benutzerpools festlegen.
4. Benutzerdefinierte Attribute sind wichtig für den Benutzererstellungs- und Anmeldeprozess, da Sie einen Wert für unveränderliche benutzerdefinierte Attribute nur dann festlegen können,

wenn Sie zuvor einen Benutzer erstellt haben. Weitere Informationen zu benutzerdefinierten Attributen finden Sie unter Custom attributes (Benutzerdefinierte Attribute).

5. Wählen Sie unter Selbstregistrierung die Option Selbstregistrierung aktivieren aus, wenn Sie möchten, dass Benutzer mit der nicht authentifizierten `SignUp`-API ein neues Konto erstellen können. Wenn Sie die Selbstregistrierung deaktivieren, können Sie neue Benutzer nur als Administrator, in der Amazon-Cognito-Konsole oder mit AdminCreateUser-API-Anfragen erstellen. In einem Benutzerpool, in dem die Selbstregistrierung nicht aktiv ist, geben SignUp-API-Anfragen `NotAuthorizedException` zurück und auf der gehosteten Benutzeroberfläche wird kein Anmeldelink angezeigt.

Für Benutzerpools, in denen Sie als Administrator Benutzer erstellen möchten, können Sie die Dauer ihrer temporären Passwörter auf der Registerkarte Anmeldeerfahrung unter Von Administratoren festgelegte temporäre Passwörter laufen ab in konfigurieren.

Ein weiteres wichtiges Element bei der Erstellung von Benutzern als Administrator ist die Einladungsnachricht. Wenn Sie einen neuen Benutzer erstellen, sendet Amazon Cognito diesem eine Nachricht mit einem Link zu Ihrer App, damit er sich zum ersten Mal anmelden kann. Passen Sie diese Nachrichtenvorlage auf der Registerkarte Nachrichten unter Nachrichtenvorlagen an.

Sie können vertrauliche App-Clients, in der Regel Webanwendungen, mit einem geheimen Client-Secret konfigurieren, das eine Anmeldung ohne das geheime App-Client-Secret verhindert. Aus Sicherheitsgründen sollten Sie App-Client-Secrets nicht auf öffentlichen App-Clients, die in der Regel mobile Apps sind, verteilen. Sie können App-Clients mit Client-Secrets auf der Registerkarte App-Integration der Amazon Cognito-Konsole erstellen.

Amazon Cognito user pools API

Sie können die Parameter für die Erstellung von Benutzern in einem Benutzerpool in einer CreateUserPool- oder UpdateUserPool-API-Anfrage programmgesteuert festlegen.

Das AdminCreateUserConfig-Element legt Werte für die folgenden Eigenschaften eines Benutzerpools fest.

1. Aktivieren der Self-Service-Anmeldung
2. Die Einladungsnachricht, die Sie an neue vom Administrator erstellte Benutzer senden

Wird das folgende Beispiel zu einem vollständigen API-Anfragetext hinzugefügt, wird ein Benutzerpool mit inaktiver Self-Service-Registrierung und einer einfachen Einladungs-E-Mail eingerichtet.

```
"AdminCreateUserConfig": {
      "AllowAdminCreateUserOnly": true,
      "InviteMessageTemplate": {
         "EmailMessage": "Your username is {username} and temporary password is
 {####}.",
         "EmailSubject": "Welcome to ExampleApp",
         "SMSMessage": "Your username is {username} and temporary password is
 {####}."
      }
   }
```

Die folgenden zusätzlichen Parameter einer CreateUserPool- oder UpdateUserPool-API-Anfrage regeln die Erstellung neuer Benutzer.

AutoVerifiedAttributes

Die Attribute, E-Mail-Adressen oder Telefonnummern, an die Sie automatisch eine Nachricht senden möchten, wenn Sie einen neuen Benutzer registrieren.

Richtlinien

Die Passwortrichtlinie für den Benutzerpool.

Schema

Die benutzerdefinierten Attribute des Benutzerpools. Diese sind wichtig für den Benutzererstellungs- und Anmeldeprozess, da Sie einen Wert für unveränderliche benutzerdefinierte Attribute nur dann festlegen können, wenn Sie zuvor einen Benutzer erstellt haben.

Dieser Parameter legt auch die erforderlichen Attribute für Ihren Benutzerpool fest. Wird der folgende Text in das `Schema`-Element eines vollständigen API-Anforderungstexts eingefügt, wird das `email`-Attribut nach Bedarf festgelegt.

```
{
            "Name": "email",
            "Required": true
}
```

Registrieren und Bestätigen von Benutzerkonten

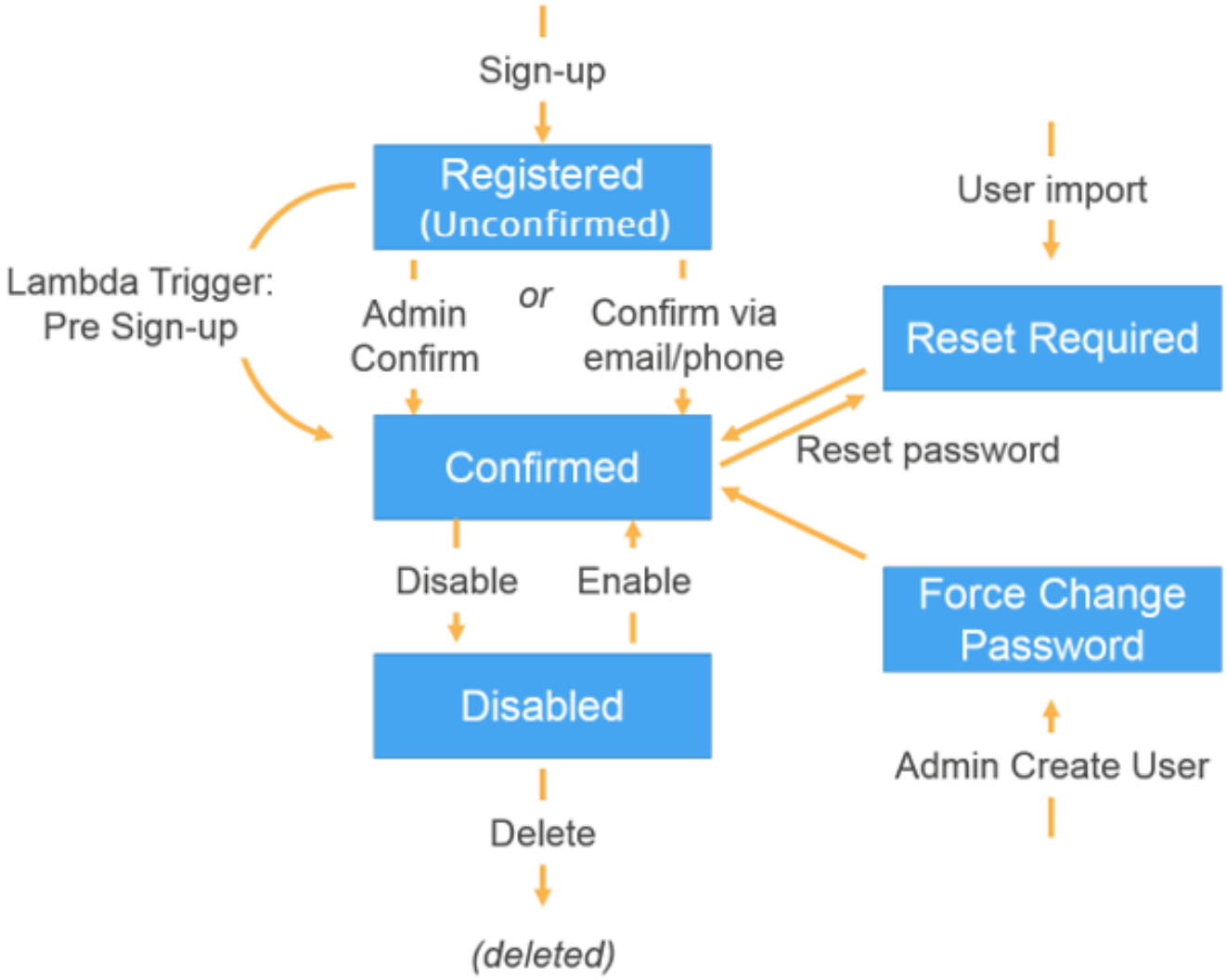
Benutzerkonten können auf eine der folgenden Weisen einem Benutzerpool hinzugefügt werden:

- Der Benutzer meldet sich in der Client-App Ihres Benutzerpools an. Dies kann eine mobile oder webbasierte App sein.
- Sie können das Benutzerkonto in Ihren Benutzerpool importieren. Weitere Informationen finden Sie unter Importieren von Benutzern aus einer CSV-Datei in Benutzerpools.
- Sie können das Benutzerkonto in Ihrem Create a User Pool und den Benutzer einladen, sich anzumelden. Weitere Informationen finden Sie unter Erstellen von Benutzerkonten als Administrator.

Benutzer, die sich selbst registrieren, müssen bestätigt werden, bevor sie sich anmelden können. Importierte und erstellte Benutzer sind bereits bestätigt, aber sie müssen ihr eigenes Passwort erstellen, wenn sie sich das erste Mal anmelden. In den folgenden Abschnitten werden der Bestätigungsprozess und die Verifizierung per E-Mail und Telefon erläutert.

Überblick über die Benutzerkonto-Bestätigung

Das folgende Diagramm illustriert den Bestätigungsprozess:



Ein Benutzerkonto kann einen der folgenden Status aufweisen:

Registriert (Unbestätigt)

Der Benutzer hat sich erfolgreich registriert, kann sich aber nicht anmelden, bis das Benutzerkonto bestätigt wird. Der Benutzer aktiviert ist, aber in diesem Status nicht bestätigt.

Neue Benutzer, die sich selbst registrieren, starten in diesem Status.

Bestätigt

Das Benutzerkonto ist bestätigt, und der Benutzer kann sich anmelden. Wenn ein Benutzer einen Code eingibt oder einem E-Mail-Link folgt, um sein Benutzerkonto zu bestätigen, wird die E-Mail-Adresse oder Telefonnummer automatisch verifiziert. Der Code oder Link ist 24 Stunden gültig.

Wenn das Benutzerkonto vom Administrator oder einem Lambda-Auslöser vor der Anmeldung bestätigt wurde, ist möglicherweise keine verifizierte E-Mail-Adresse oder Telefonnummer mit dem Konto verknüpft.

Zurücksetzen des Passworts erforderlich

Das Benutzerkonto ist bestätigt, aber der Benutzer muss einen Code anfordern und sein Passwort zurücksetzen, bevor er sich anmelden kann.

Benutzerkonten, die von einem Administrator oder Entwickler importiert werden, starten in diesem Status.

Ändern des Passworts erzwingen

Das Benutzerkonto ist bestätigt, und der Benutzer kann sich mit einem temporären Passwort anmelden, aber bei der ersten Anmeldung muss der Benutzer zunächst das Passwort ändern.

Benutzerkonten, die von einem Administrator oder Entwickler erstellt werden, starten in diesem Status.

Disabled

Bevor Sie ein Benutzerkonto löschen können, müssen Sie den Anmeldezugriff für diesen Benutzer deaktivieren.

Überprüfen von Kontaktinformationen bei der Anmeldung

Wenn sich neue Benutzer in Ihrer App anmelden, möchten Sie wahrscheinlich, dass sie mindestens eine Kontaktmethode angeben. Mit den Kontaktinformationen des Benutzers haben Sie zum Beispiel folgende Möglichkeiten:

- Senden Sie ein temporäres Passwort, wenn ein Benutzer sein Passwort zurücksetzt.
- Benachrichtigen der Benutzer, wenn deren persönliche oder finanziellen Daten aktualisiert werden
- Senden von Werbemitteilungen, beispielsweise zu Sonderangeboten oder Rabatten
- Senden von Kontoübersichten oder Zahlungserinnerungen

Für solche Anwendungsfälle müssen Sie Ihre Nachrichten unbedingt an ein Ziel senden, das überprüft wurde. Andernfalls senden Sie Ihre Nachrichten möglicherweise an eine ungültige E-Mail-Adresse oder falsch eingegebene Telefonnummer. Im schlimmsten Fall könnten Sie sogar sensible Daten an böswillige Angreifer übermitteln, die sich als Ihre Benutzer ausgeben.

Um sicherzustellen, dass Sie Nachrichten nur an die richtigen Personen senden, konfigurieren Sie den Amazon-Cognito-Benutzerpool so, dass die Benutzer Folgendes angeben müssen, wenn sie sich anmelden:

a. Eine E-Mail-Adresse oder Telefonnummer.

b. Einen Verifizierungscode, den Amazon Cognito an diese E-Mail-Adresse oder Telefonnummer sendet. Wenn 24 Stunden vergangen sind und der Code oder Link Ihres Benutzers nicht mehr gültig ist, rufen Sie den ResendConfirmationCodeAPI-Vorgang auf, um einen neuen Code oder Link zu generieren und zu senden.

Durch die Angabe des Verifizierungscodes weist ein Benutzer nach, dass er über Zugriff auf das Postfach oder Telefon verfügt, welches den Code erhalten hat. Nachdem der Benutzer den Code angegeben hat, aktualisiert Amazon Cognito die Informationen über den Benutzer im Benutzerpool wie folgt:

- Der Status des Benutzers wird auf festgelegt `CONFIRMED`.
- Die Attribute des Benutzers werden aktualisiert, um anzugeben, dass die E-Mail-Adresse oder Telefonnummer bestätigt wurde.

Sie können diese Informationen in der Amazon-Cognito-Konsole anzeigen. Oder Sie können die `AdminGetUser` API-Operation, den `admin-get-user` Befehl mit dem AWS CLI oder eine entsprechende Aktion in einem der AWS SDKs verwenden.

Wenn ein Benutzer über eine überprüfte Kontaktmethode verfügt, sendet Amazon Cognito dem Benutzer automatisch eine Nachricht, wenn der Benutzer die Zurücksetzung des Passworts anfordert.

Benutzerpool für eine obligatorische Verifizierung der E-Mail-Adresse oder Telefonnummer konfigurieren

Wenn Sie die E-Mail-Adressen und Telefonnummern Ihrer Benutzer überprüfen, stellen Sie sicher, dass Sie Ihre Benutzer kontaktieren können. Gehen Sie wie folgt vor AWS Management Console , um Ihren Benutzerpool so zu konfigurieren, dass Ihre Benutzer ihre E-Mail-Adressen oder Telefonnummern bestätigen müssen.

> **Note**
>
> Wenn Sie noch keinen Benutzerpool in Ihrem Konto eingerichtet haben, finden Sie weitere Informationen unter Erste Schritte mit Benutzerpools.

So konfigurieren Sie den Benutzerpool

1. Navigieren Sie zur Amazon-Cognito-Konsole. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie im Navigationsbereich User Pools (Benutzerpools) aus. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
3. Wählen Sie die Registerkarte Sign-up experience (Anmeldeerlebnis) aus und suchen Sie nach Attribute verification and user account confirmation (Attributverifikation und Benutzerkontobestätigung). Wählen Sie Bearbeiten aus.
4. Wählen Sie unter Von Cognito unterstützte Überprüfung und Bestätigung aus, ob Sie Cognito erlauben, automatisch Nachrichten zur Überprüfung und Bestätigung zu senden. Wenn diese Einstellung aktiviert ist, sendet Amazon Cognito Nachrichten an die von Ihnen ausgewählten Benutzerkontaktattribute, wenn sich ein Benutzer anmeldet oder wenn Sie ein Benutzerprofil erstellen. Um Attribute zu überprüfen und Benutzerprofile für die Anmeldung zu bestätigen, sendet Amazon Cognito einen Code oder Link in Nachrichten an Benutzer. Die Benutzer müssen dann den Code in Ihre Benutzeroberfläche eingeben, damit Ihre App sie in einer `ConfirmSignUp`- oder `AdminConfirmSignUp`-API-Anfrage bestätigen kann.

   > **Note**
   >
   > Sie können Cognito-assisted verification and confirmation (Cognito-unterstütze Verifizierung und Bestätigung) auch deaktivieren und die authentifizierten API-Aktionen oder Lambda-Auslöser verwenden, um Attribute zu verifizieren und Benutzer zu bestätigen.
   > Bei dieser Auswahl sendet Amazon Cognito keine Verifizierungscodes, wenn Benutzer sich anmelden. Aktivieren Sie diese Option, wenn Sie einen benutzerdefinierten Authentifizierungsablauf verwenden, der mindestens eine Kontaktmethode verifiziert, ohne Verifizierungscodes von Amazon Cognito zu nutzen. Sie könnten beispielsweise einen Lambda-Auslöser vor der Anmeldung verwenden, der automatisch E-Mail-Adressen verifiziert, die einer bestimmten Domäne angehören.

Wenn Sie die Kontaktinformationen Ihrer Benutzer nicht verifizieren, können die Benutzer die App möglicherweise nicht verwenden. Beachten Sie, dass die Benutzer die folgenden Aktionen nur mit verifizierten Kontaktinformationen ausführen können:

- Passwörter zurücksetzen – Wenn ein Benutzer eine Option in Ihrer Anwendung auswählt, die die API-Aktion `ForgotPassword` abruft, sendet Amazon Cognito ein temporäres Passwort an die E-Mail-Adresse oder Telefonnummer des Benutzers. Amazon Cognito sendet dieses Kennwort nur, wenn der Benutzer über mindestens eine verifizierte Kontaktmethode verfügt.
- Anmeldung mit einer E-Mail-Adresse oder Telefonnummer als Alias – Wenn Sie den Benutzerpool so konfigurieren, dass diese Alias zulässig sind, kann ein Benutzer sich nur mit einem Alias anmelden, wenn der Alias verifiziert wurde. Weitere Informationen finden Sie unter Anpassen von Anmeldeattributen.

5. Wählen Sie Ihre zu überprüfenden Attribute aus:

SMS-Nachricht senden, Telefonnummer verifizieren

Amazon Cognito sendet einen Verifizierungscode in einer SMS-Nachricht, wenn der Benutzer sich anmeldet. Treffen Sie diese Auswahl, wenn Sie mit Ihren Benutzern normalerweise per SMS kommunizieren. Verifizierte Telefonnummern verwenden Sie beispielsweise zum Senden von Lieferbenachrichtigungen, Terminbestätigungen oder Warnungen. Benutzertelefonnummern sind das verifizierte Attribut, wenn Konten bestätigt werden. Sie müssen zusätzliche Maßnahmen ergreifen, um die E-Mail-Adressen der Benutzer zu überprüfen und mit ihnen zu kommunizieren.

E-Mail-Nachricht senden, E-Mail-Adresse überprüfen

Amazon Cognito sendet einen Verifizierungscode per E-Mail, wenn der Benutzer sich anmeldet. Wählen Sie diese Option aus, wenn Sie mit Ihren Benutzern in der Regel über E-Mail kommunizieren. Sie benötigen verifizierte E-Mail-Adressen zum Beispiel, wenn Sie Abrechnungen, Bestellübersichten oder Sonderangebote senden. Benutzertelefonnummern sind das verifizierte Attribut, wenn Konten bestätigt werden. Sie müssen zusätzliche Maßnahmen ergreifen, um die Telefonnummern der Benutzer zu überprüfen und mit ihnen zu kommunizieren.

SMS-Nachricht senden, wenn die Telefonnummer verfügbar ist, andernfalls E-Mail-Nachricht senden

Wählen Sie diese Option aus, wenn es nicht erforderlich ist, dass alle Benutzer über dieselbe verifizierte Kontaktmethode verfügen. In diesem Fall könnte die Anmeldeseite in Ihrer App die Benutzer dazu auffordern, nur ihre bevorzugte Kontaktmethode zu verifizieren. Beim Übermitteln eines Verifizierungscodes sendet Amazon Cognito den Code an die Kontaktmethode, die in der `SignUp`-Anforderung von Ihrer App bereitgestellt wird. Wenn ein Benutzer eine E-Mail-Adresse und eine Telefonnummer angibt und die App beide Kontaktmethoden in der `SignUp`-Anforderung übermittelt, sendet Amazon Cognito nur einen Verifizierungscode an die Telefonnummer.

Wählen Sie diese Option aus, um festzulegen, dass die Benutzer sowohl eine E-Mail-Adresse als auch eine Telefonnummer verifizieren müssen. Amazon Cognito verifiziert eine Kontaktmethode, wenn sich der Benutzer anmeldet. Die App verifiziert die andere Kontaktmethode, nachdem sich der Benutzer angemeldet hat. Weitere Informationen finden Sie unter Wenn die Benutzer sowohl E-Mail-Adressen als auch Telefonnummern bestätigen müssen.

6. Wählen Sie Save Changes (Änderungen speichern).

Authentifizierungsablauf mit Verifizierung per E-Mail oder Telefon

Wenn der Benutzerpool erfordert, dass die Benutzer ihre Kontaktinformationen verifizieren, muss Ihre Anwendung den folgenden Ablauf unterstützen, wenn sich ein Benutzer anmeldet:

1. Ein Benutzer registriert sich in Ihrer App, indem er einen Benutzernamen, Telefonnummer und/oder E-Mail-Adresse und möglicherweise auch andere Attribute eingibt.
2. Die Amazon-Cognito-Service erhält die Registrierungs-Anforderung von der App. Nachdem verifiziert wurde, dass die Anforderung alle für die Registrierung erforderlichen Attribute enthält, schließt der Service die Registrierung ab und sendet einen Bestätigungscode an die Telefonnummer (per SMS) oder E-Mail-Adresse des Benutzers. Der Code ist 24 Stunden gültig
3. Der Service gibt der App zurück, dass die Registrierung abgeschlossen ist und die Bestätigung des Benutzerkontos noch ansteht. Die Antwort enthält Informationen darüber, wohin der Bestätigungscode gesendet wurde. An diesem Punkt ist das Benutzerkonto in einem unbestätigten Status, und die E-Mail-Adresse und Telefonnummer des Benutzers sind noch nicht verifiziert.

4. Die App kann jetzt den Benutzer dazu auffordern, den Bestätigungscode einzugeben. Der Benutzer muss den Code nicht sofort eingeben. Er kann sich jedoch erst anmelden, nachdem er den Bestätigungscode eingegeben hat.
5. Der Benutzer gibt den Bestätigungscode in die App ein.
6. Die App ruft `ConfirmSignUp` auf und sendet den Code an den Amazon-Cognito-Service, welcher den Code verifiziert. Wenn der Code korrekt ist, wird das Konto des Benutzers bestätigt. Nach erfolgreicher Bestätigung des Benutzerkontos markiert der Amazon-Cognito-Service automatisch das zur Bestätigung verwendete Attribut (E-Mail-Adresse oder Telefonnummer) als verifiziert. Sofern der Wert dieses Attributs nicht geändert wird, muss der Benutzer es nicht erneut verifizieren.
7. An diesem Punkt ist das Benutzerkonto in einem bestätigten Status, und der Benutzer kann sich anmelden.

Wenn die Benutzer sowohl E-Mail-Adressen als auch Telefonnummern bestätigen müssen

Amazon Cognito verifiziert nur eine Kontaktmethode, wenn sich ein Benutzer anmeldet. In Fällen, in denen Amazon Cognito zwischen der Verifizierung einer E-Mail-Adresse oder Telefonnummer wählen muss, verifiziert der Service die Telefonnummer, indem er einen Verifizierungscode per SMS-Nachricht sendet. Wenn Sie beispielsweise den Benutzerpool so konfigurieren, dass Benutzern die Verifizierung über die E-Mail-Adresse oder Telefonnummer gestattet ist und wenn Ihre App beide Attribute bei der Anmeldung übermittelt, verifiziert Amazon Cognito nur die Telefonnummer. Nachdem ein Benutzer seine Telefonnummer verifiziert hat, setzt Amazon Cognito den Status des Benutzers auf `CONFIRMED` und der Benutzer darf sich bei der App anmelden.

Nachdem der Benutzer angemeldet ist, kann die App optional die Verifizierung der während der Anmeldung nicht verifizierten Kontaktmethode bereitstellen. Zur Überprüfung dieser zweiten Methode ruft die App die `VerifyUserAttribute`-API-Aktion auf. Beachten Sie, dass diese Aktion einen `AccessToken`-Parameter erfordert und Amazon Cognito nur authentifizierten Benutzern Zugriff auf Token bietet. Aus diesem Grund können Sie die zweite Kontaktmethode erst verifizieren, nachdem der Benutzer sich angemeldet hat.

Wenn es erforderlich ist, dass die Benutzer sowohl E-Mail-Adressen als auch Telefonnummern verifizieren, führen Sie die folgenden Schritte aus:

1. Konfigurieren Sie den Benutzerpool so, dass Benutzern die Verifizierung über die E-Mail-Adresse oder Telefonnummern gestattet ist.

2. Fordern Sie im Anmeldungsfluss für die App, dass die Benutzer eine E-Mail-Adresse und eine Telefonnummer angeben müssen. Rufen Sie die `SignUp`-API-Aktion auf und stellen Sie die E-Mail-Adresse und Telefonnummer für den `UserAttributes`-Parameter bereit. An diesem Punkt sendet Amazon Cognito einen Verifizierungscode an das Telefon des Benutzers.
3. Zeigen Sie in der App-Oberfläche eine Bestätigungsseite an, auf der der Benutzer den Verifizierungscode eingibt. Bestätigen Sie den Benutzer durch Aufrufen der `ConfirmSignUp`-API-Aktion. An diesem Punkt ist der Status des Benutzers `CONFIRMED`. Die Telefonnummer des Benutzers wurde verifiziert, jedoch nicht die E-Mail-Adresse.
4. Zeigen Sie die Anmeldeseite an und authentifizieren Sie den Benutzer durch Aufrufen der `InitiateAuth`-API-Aktion. Nachdem der Benutzer authentifiziert wurde, gibt Amazon Cognito ein Zugriffstoken an die App zurück.
5. Rufen Sie die `GetUserAttributeVerificationCode`-API-Aktion auf. Geben Sie in der Anforderung die folgenden Parameter an:

   - `AccessToken` – Das von Amazon Cognito bei der Anmeldung des Benutzers zurückgegebene Zugriffstoken.
   - `AttributeName` – Legen Sie `"email"` als Attributwert fest.

   Amazon Cognito sendet einen Verifizierungscode an die E-Mail-Adresse des Benutzers.
6. Zeigen Sie eine Bestätigungsseite an, auf der der Benutzer den Verifizierungscode eingibt. Wenn der Benutzer den Code übermittelt, rufen Sie die `VerifyUserAttribute`-API-Aktion auf. Geben Sie in der Anforderung die folgenden Parameter an:

   - `AccessToken` – Das von Amazon Cognito bei der Anmeldung des Benutzers zurückgegebene Zugriffstoken.
   - `AttributeName` – Legen Sie `"email"` als Attributwert fest.
   - `Code` – Der Verifizierungscode, den der Benutzer eingegeben hat.

   Die E-Mail-Adresse wurde jetzt verifiziert.

Benutzern erlauben, sich in der Anwendung anzumelden, sie aber als Benutzerpool-Administrator bestätigen

Möglicherweise möchten Sie nicht, dass Ihr Benutzerpool automatisch Bestätigungsnachrichten an Ihren Benutzerpool sendet, möchten aber dennoch jedem ermöglichen, sich für ein Konto

anzumelden. Dieses Modell bietet beispielsweise Spielraum für die manuelle Überprüfung neuer Anmeldeanfragen sowie für die Batch-Validierung und Bearbeitung von Anmeldungen. Sie können neue Benutzerkonten in der Amazon Cognito Cognito-Konsole oder mit dem IAM-authentifizierten API-Vorgang bestätigen. AdminConfirmSignUp Sie können Benutzerkonten als Administrator bestätigen, unabhängig davon, ob Ihr Benutzerpool Bestätigungsnachrichten sendet oder nicht.

Mit dieser Technik können Sie die Self-Service-Registrierung eines Benutzers nur bestätigen. Um einen Benutzer zu bestätigen, den Sie als Administrator erstellen, erstellen Sie eine AdminSetUserPasswordAPI-Anfrage mit der Einstellung auf. `Permanent True`

1. Ein Benutzer registriert sich in Ihrer App, indem er einen Benutzernamen, Telefonnummer und/oder E-Mail-Adresse und möglicherweise auch andere Attribute eingibt.
2. Die Amazon-Cognito-Service erhält die Registrierungs-Anforderung von der App. Nachdem Sie verifiziert haben, dass die Anforderung alle für die Anmeldung benötigten Attribute enthält, schließt der Service den Anmeldevorgang ab und gibt an die App zurück, dass die Anmeldung abgeschlossen ist, aber noch nicht bestätigt wurde. An diesem Punkt befindet sich das Benutzerkonto in einem unbestätigten Status. Der Benutzer kann nicht anmelden, bis das Konto bestätigt ist.
3. Bestätigen Sie das Konto des Benutzers. Sie müssen sich bei der API-Anfrage anmelden AWS Management Console oder Ihre API-Anfrage mit AWS Anmeldeinformationen unterschreiben, um das Konto zu bestätigen.

   a. Um einen Benutzer in der Amazon-Cognito-Konsole zu bestätigen, navigieren Sie zur Registerkarte Benutzer, wählen Sie den Benutzer aus, den Sie bestätigen möchten, und wählen Sie im Menü Aktionen die Option Bestätigen aus.

   b. Um einen Benutzer in der AWS API oder CLI zu bestätigen, erstellen Sie eine AdminConfirmSignUpAPI-Anfrage oder admin-confirm-sign-upin der AWS CLI.
4. An diesem Punkt ist das Benutzerkonto in einem bestätigten Status, und der Benutzer kann sich anmelden.

Berechnen von Werten für geheime Hashes

Es ist eine bewährte Methode, Ihrem vertraulichen App-Client einen geheimen Client-Schlüssel zuzuweisen. Wenn Sie Ihrem App-Client einen geheimen Client-Schlüssel zuweisen, müssen API-Anforderungen Ihrer Amazon-Cognito-Benutzerpools einen Hash enthalten, der den geheimen Client-Schlüssel im Anforderungstext enthält. Um zu überprüfen, ob Sie den geheimen Client-Schlüssel für die API-Operationen in den folgenden Listen kennen, verketten Sie den geheimen Client-Schlüssel

mit Ihrer App-Client-ID und dem Benutzernamen Ihres Benutzers und codieren Sie dann diese Zeichenfolge mit Base64.

Wenn Ihre App Benutzer bei einem Client anmeldet, der über einen geheimen Hash verfügt, können Sie den Wert eines beliebigen Benutzerpool-Anmeldeattributs als Benutzernamenselement des geheimen Hashs verwenden. Wenn Ihre App im Rahmen eines Authentifizierungsvorgangs mit `REFRESH_TOKEN_AUTH` neue Token anfordert, hängt der Wert des Benutzernamenselements von Ihren Anmeldeattributen ab. Wenn Ihr Benutzerpool nicht `username` als Anmeldeattribut hat, legen Sie den Wert für den geheimen Hash-Benutzernamen fest, der sich aus dem `sub`-Anspruch des Benutzers aus seinem Zugriffs- oder ID-Token ergibt. Wenn es sich bei `username` um ein Anmeldeattribut handelt, legen Sie den Wert für den geheimen Hash-Benutzernamen aus dem `username`-Anspruch fest.

Die folgenden Amazon-Cognito-Benutzerpool-APIs akzeptieren einen client-secret-Hash-Wert in einem Parameter `SecretHash`.

- ConfirmForgotPassword
- ConfirmSignUp
- ForgotPassword
- ResendConfirmationCode
- SignUp

Darüber hinaus akzeptieren die folgenden APIs einen client-secret-Hash-Wert in einem Parameter `SECRET_HASH`, entweder in Authentifizierungsparametern oder in einer Antwort auf eine Aufforderung.

API-Operation	Übergeordneter Parameter für SECRET_HASH
InitiateAuth	AuthParameters
AdminInitiateAuth	AuthParameters
RespondToAuthChallenge	ChallengeResponses
AdminRespondToAuthChallenge	ChallengeResponses

Der Wert für den geheimen Hash ist ein Base 64-codierter Keyed-Hash Message Authentication Code (HMAC), der anhand des geheimen Schlüssels eines Benutzerpool-Clients und Benutzernamens plus der Client-ID in der Nachricht berechnet wird. Die folgende Pseudocode zeigt, wie dieser Wert berechnet wird. In diesem Pseudocode bedeutet + eine Verkettung, `HMAC_SHA256` stellt eine Funktion dar, die einen HMAC-Wert mit HmacSHA256 produziert, und `Base64` stellt eine Funktion dar, die eine Base-64-codierte Version der Hash-Ausgabe produziert.

```
Base64 ( HMAC_SHA256 ( "Client Secret Key", "Username" + "Client Id" ) )
```

Eine ausführliche Übersicht über die Berechnung und Verwendung des `SecretHash` Parameters finden Sie unter Wie behebe ich die Fehler „Unable to verify secret hash for client“ in meiner Amazon Cognito Cognito-Benutzerpools-API<client-id>? im AWS Knowledge Center.

Sie können die folgenden Codebeispiele in Ihrem serverseitigen Anwendungscode verwenden.

Shell

```
echo -n "[username][app client ID]" | openssl dgst -sha256 -hmac [app client secret]
 -binary | openssl enc -base64
```

Java

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public static String calculateSecretHash(String userPoolClientId, String
 userPoolClientSecret, String userName) {
    final String HMAC_SHA256_ALGORITHM = "HmacSHA256";

    SecretKeySpec signingKey = new SecretKeySpec(
            userPoolClientSecret.getBytes(StandardCharsets.UTF_8),
            HMAC_SHA256_ALGORITHM);
    try {
        Mac mac = Mac.getInstance(HMAC_SHA256_ALGORITHM);
        mac.init(signingKey);
        mac.update(userName.getBytes(StandardCharsets.UTF_8));
        byte[] rawHmac =
 mac.doFinal(userPoolClientId.getBytes(StandardCharsets.UTF_8));
        return Base64.getEncoder().encodeToString(rawHmac);
    } catch (Exception e) {
        throw new RuntimeException("Error while calculating ");
```

```
    }
}
```

Python

```
import sys
import hmac, hashlib, base64
username = sys.argv[1]
app_client_id = sys.argv[2]
key = sys.argv[3]
message = bytes(sys.argv[1]+sys.argv[2],'utf-8')
key = bytes(sys.argv[3],'utf-8')
secret_hash = base64.b64encode(hmac.new(key, message,
 digestmod=hashlib.sha256).digest()).decode()
print("SECRET HASH:",secret_hash)
```

Bestätigen von Benutzerkonten ohne Verifizieren der E-Mail-Adresse oder Telefonnummer

Der Lambda-Auslöser vor der Registrierung kann verwendet werden, um Benutzerkonten bei der Registrierung automatisch zu bestätigen, ohne dass ein Bestätigungscode oder eine Verifizierung der E-Mail-Adresse oder Telefonnummer erforderlich ist. Benutzer, die auf diese Weise bestätigt werden, können sich sofort anmelden, ohne dass Sie einen Code benötigen.

Sie können über diesen Auslöser auch eine E-Mail-Adresse oder Telefonnummer als verifiziert markieren.

> **Note**
>
> Dieser Ansatz ist praktisch, um Benutzern bei den ersten Schritten zu helfen, doch wir empfehlen die automatische Verifizierung von mindestens E-Mail-Adresse oder Telefonnummer. Andernfalls kann der Benutzer nicht wiederhergestellt werden, wenn er sein Passwort vergisst.

Wenn Sie es nicht erfordern, dass der Benutzer bei der Registrierung einen Bestätigungscode erhält und eingibt und Sie E-Mail-Adresse und Telefonnummer nicht im Lambda-Auslöser vor der Registrierung automatisch verifizieren, riskieren Sie, dass Sie keine verifizierte E-Mail-Adresse oder Telefonnummer für das Benutzerkonto haben. Der Benutzer kann die E-Mail-Adresse oder

Telefonnummer zu einem späteren Zeitpunkt verifizieren. Wenn der Benutzer jedoch sein Passwort vergisst und keine verifizierte E-Mail-Adresse oder Telefonnummer hat, wird der Benutzer aus dem Konto gesperrt, da der Ablauf für „Passwort vergessen“ eine verifizierte E-Mail-Adresse oder Telefonnummer erfordert, um einen Verifizierungscode an den Benutzer zu senden.

Verifizieren, wenn Benutzer ihre E-Mail-Adresse oder Telefonnummer ändern

Wenn ein Benutzer seine E-Mail-Adresse oder Telefonnummer in Ihrer App aktualisiert, sendet Amazon Cognito sofort eine Nachricht mit einem Bestätigungscode an einen Benutzer, wenn Sie Ihren Benutzerpool so konfiguriert haben, dass dieses Attribut automatisch verifiziert wird. Der Benutzer muss dann den Code aus der Bestätigungsnachricht an Ihre App weitergeben. Ihre App sendet dann den Code in einer VerifyUserAttributeAPI-Anfrage, um die Überprüfung des neuen Attributwerts abzuschließen.

Wenn Ihr Benutzerpool nicht verlangt, dass Benutzer eine aktualisierte E-Mail-Adresse oder Telefonnummer bestätigen, ändert Amazon Cognito sofort den Wert eines aktualisierten Attributs `email` oder `phone_number` und markiert das Attribut als nicht verifiziert. Mit einer nicht verifizierten E-Mail oder Telefonnummer kann sich Ihr Benutzer nicht anmelden. Sie müssen die Verifizierung des aktualisierten Wertes abschließen, bevor sie dieses Attribut als Anmeldealias verwenden können.

Wenn Ihr Benutzerpool erfordert, dass Benutzer eine aktualisierte E-Mail-Adresse oder Telefonnummer verifizieren, lässt Amazon Cognito das Attribut verifiziert und auf seinen ursprünglichen Wert festgelegt, bis Ihr Benutzer den neuen Attributwert bestätigt. Wenn das Attribut ein Alias für die Anmeldung ist, kann sich Ihr Benutzer mit dem ursprünglichen Attributwert anmelden, bis die Verifizierung das Attribut in den neuen Wert ändert. Weitere Informationen darüber, wie Sie Ihren Benutzerpool so konfigurieren, dass Benutzer aktualisierte Attribute verifizieren, finden Sie unter Konfigurieren der Verifizierung per E-Mail und Telefon.

Sie können einen kundenspezifischen Message-Lambda-Auslöser verwenden, um die Bestätigungsnachricht anzupassen. Weitere Informationen finden Sie unter Lambda-Auslöser für benutzerdefinierte Nachrichten. Wenn die E-Mail-Adresse oder Telefonnummer des Benutzers nicht verifiziert ist, sollte Ihre App den Benutzer darüber informieren, dass er das Attribut verifizieren muss, und eine Schaltfläche oder einen Link bereitstellen, damit Benutzer ihre neue E-Mail-Adresse oder Telefonnummer verifizieren können.

Bestätigungs- und Verifizierungsprozesse für Benutzerkonten, die durch Administratoren und Entwickler erstellt wurden

Benutzerkonten, die von einem Administrator oder Entwickler erstellt werden, haben bereits den Status "Bestätigt", sodass die Benutzer keinen Bestätigungscode eingeben müssen. Die Einladungsnachricht sendet, die der Amazon-Cognito-Service an diese Benutzer sendet, schließt den Benutzernamen und ein temporäres Kennwort ein. Der Benutzer muss das Passwort ändern, bevor er sich anmelden kann. Weitere Informationen finden Sie unter Anpassen von E-Mail- und SMS-Nachrichten in Erstellen von Benutzerkonten als Administrator und dem Custom Message-Auslöser in Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern.

Bestätigungs- und Verifizierungsprozesse für importierte Benutzerkonten

Benutzerkonten, die mithilfe der Benutzerimportfunktion in der CLI oder der AWS Management Console API (siehe Importieren von Benutzern aus einer CSV-Datei in Benutzerpools) erstellt wurden, befinden sich bereits im Status „Bestätigt“, sodass Benutzer keinen Bestätigungscode eingeben müssen. Es wird keine Einladungsnachricht gesendet. Importierte Benutzerkonten verlangen von Benutzern jedoch, zunächst einen Code durch Aufruf der `ForgotPassword`-API anzufordern und dann ein Kennwort mit dem zugestellten Code zu erstellen, indem Sie die `ConfirmForgotPassword` API aufrufen, bevor sie sich anmelden. Weitere Informationen finden Sie unter Von importierten Benutzer verlangen, dass sie ihre Passwörter zurücksetzen.

Die E-Mail-Adresse oder Telefonnummer des Benutzers muss als verifiziert markiert werden, wenn das Benutzerkonto importiert wird, damit beim Anmelden des Benutzers keine Verifizierung verlangt wird.

Senden von E-Mails beim Testen der App

Amazon Cognito sendet E-Mail-Nachrichten an Ihre Benutzer, wenn sie ihre Konten in der Client-App für Ihren Benutzerpool erstellen und verwalten. Wenn Sie den Benutzerpool so konfigurieren, dass die E-Mail-Verifizierung erforderlich ist, sendet Amazon Cognito in folgenden Fällen eine E-Mail:

- Ein Benutzer meldet sich an.
- Ein Benutzer ändert seine E-Mail-Adresse.
- Ein Benutzer führt eine Aktion aus, die die `ForgotPassword`-API-Aktion aufruft.
- Sie erstellen ein Benutzerkonto als Administrator.

Abhängig von der Aktion, die die E-Mail initiiert, enthält die E-Mail einen Verifizierungscode oder ein temporäres Passwort. Ihre Benutzer müssen diese E-Mails empfangen und die Nachricht verstehen. Andernfalls können sie sich u. U. nicht anmelden und Ihre App verwenden.

Um sicherzustellen, dass E-Mails erfolgreich gesendet werden und die Nachricht korrekt ist, testen Sie die Aktionen in der App, die das Senden von E-Mails durch Amazon Cognito initiieren. Beispiel: Über die Anmeldeseite in der App oder mithilfe der `SignUp`-API-Aktion können Sie eine E-Mail initiieren, indem Sie sich mit einer Test-E-Mail-Adresse anmelden. Wenn Sie auf diese Weise einen Test ausführen, beachten Sie Folgendes:

> **Wichtig**
>
> Wenn Sie eine E-Mail-Adresse verwenden, um Aktionen zu testen, die E-Mails von Amazon Cognito initiieren, verwenden Sie keine gefälschte E-Mail-Adresse (ohne Postfach). Verwenden Sie eine echte E-Mail-Adresse, die die E-Mail von Amazon Cognito empfängt, ohne dass eine permanente Unzustellbarkeit auftritt.
> Eine permanente Unzustellbarkeit kommt zustande, wenn Amazon Cognito die E-Mail nicht an das Postfach des Empfängers liefern kann. Wenn das Postfach nicht vorhanden ist, ist das immer der Fall.
> Amazon Cognito begrenzt die Anzahl der E-Mails, die von AWS Konten gesendet werden können, bei denen es ständig zu Hard-Bounces kommt.

Wenn Sie Aktionen testen, die E-Mails initiieren, verwenden Sie eine der folgenden E-Mail-Adressen, um permanente Unzustellbarkeiten zu vermeiden:

- Eine Adresse für ein E-Mail-Konto, das Sie besitzen und für Tests verwenden. Wenn Sie Ihre eigene E-Mail-Adresse verwenden, erhalten Sie die E-Mail, die Amazon Cognito sendet. Mit dieser E-Mail können Sie den Verifizierungscode zum Testen der Anmeldung in Ihrer App nutzen. Wenn Sie die E-Mail-Nachricht für den Benutzerpool angepasst haben, können Sie überprüfen, ob Ihre Anpassungen korrekt dargestellt werden.
- Die Adresse des Postfachsimulators, success@simulator.amazonses.com. Wenn Sie die Simulatoradresse verwenden, sendet Amazon Cognito die E-Mail erfolgreich. Sie können sie aber nicht ansehen. Diese Option ist nützlich, wenn Sie den Verifizierungscode nicht benötigen und die E-Mail-Nachricht nicht überprüfen müssen.
- Die Adresse des Postfachsimulators, der eine beliebige Bezeichnung hinzugefügt wird, zum Beispiel success+user1@simulator.amazonses.com oder success

+user2@simulator.amazonses.com. Amazon Cognito sendet diese Adressen erfolgreich per E-Mail, aber Sie können die gesendeten E-Mails nicht anzeigen. Diese Option ist nützlich, wenn Sie den Anmeldevorgang durch Hinzufügen mehrerer Testbenutzer zum Benutzerpool testen möchten und jeder Testbenutzer über eine eindeutige E-Mail-Adresse verfügt.

Konfigurieren der Verifizierung per E-Mail und Telefon

Auf der Registerkarte Messaging können Sie die Einstellungen für die Verifizierung per E-Mail-Adresse oder Telefonnummer auswählen. Weitere Informationen zur Multi-Faktor-Authentifizierung (MFA) finden Sie unter SMS-MFA.

Amazon Cognito verwendet Amazon SNS zum Senden von SMS-Nachrichten. Wenn Sie noch keine SMS-Nachricht von Amazon Cognito oder einem anderen AWS-Service gesendet haben, platziert Amazon SNS Ihr Konto möglicherweise in der SMS-Sandbox. Wir empfehlen Ihnen, eine Testnachricht an eine verifizierte Telefonnummer zu senden, bevor Sie Ihr Konto aus der Sandbox an die Produktion übergeben. Wenn Sie vorhaben, SMS-Nachrichten an US-Zieltelefonnummern zu senden, müssen Sie außerdem eine Ursprungs- oder Sender-ID von Amazon Pinpoint erhalten. Informationen zum Konfigurieren Ihres Amazon-Cognito-Benutzerpools für SMS-Nachrichten finden Sie unter Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools.

Amazon Cognito kann E-Mail-Adressen und Telefonnummern automatisch verifizieren. Für diese Verifizierung sendet Amazon Cognito einen Verifizierungscode oder einen Verifizierungslink. Bei E-Mail-Adressen kann Amazon Cognito einen Code oder einen Link in einer E-Mail-Nachricht senden. Sie können einen Verifizierungstyp bzw. Code oder Link auswählen, wenn Sie Ihre Vorlage für Verifizierungsnachrichten auf der Registerkarte Messaging der Amazon-Cognito-Konsole bearbeiten. Weitere Informationen finden Sie unter Anpassen von Nachrichten zur E-Mail-Verifizierung.

Bei Telefonnummern sendet Amazon Cognito einen Code in einer SMS-Textnachricht.

Amazon Cognito muss eine Telefonnummer oder E-Mail-Adresse verifizieren, um Benutzer zu bestätigen und ihnen bei der Wiederherstellung vergessener Passwörter zu helfen. Alternativ können Sie Benutzer automatisch mit dem Lambda-Trigger vor der Registrierung bestätigen oder den AdminConfirmSignUpAPI-Vorgang verwenden. Weitere Informationen finden Sie unter Registrieren und Bestätigen von Benutzerkonten.

Der Verifizierungscode oder -link ist 24 Stunden lang gültig.

Wenn Sie die Verifizierung einer E-Mail-Adresse oder Telefonnummer fordern, sendet Amazon Cognito automatisch den Verifizierungscode oder -link, wenn sich ein Benutzer anmeldet. Wenn

im Benutzerpool Benutzerdefinierter Lambda-Auslöser für SMS-Sender oder Benutzerdefinierter Lambda-Auslöser für E-Mail-Sender konfiguriert ist, wird diese Funktion stattdessen aufgerufen.

Hinweise

- SMS-Nachrichten für die Überprüfung von Telefonnummern werden von Amazon SNS separat in Rechnung gestellt. Es fallen keine Gebühren für das Senden von E-Mail-Nachrichten an. Weitere Informationen zu den Amazon-SNS-Preisen erhalten Sie unter Weltweite SMS-Preise. Eine aktuelle Liste von Ländern, in denen SMS-Messaging verfügbar ist, finden Sie in den Informationen zu den unterstützten Regionen und Ländern.
- Wenn Sie Aktionen in Ihrer App testen, die E-Mails von Amazon Cognito generieren, verwenden Sie eine echte E-Mail-Adresse, die Amazon Cognito ohne permanente Unzustellbarkeiten erreichen kann. Weitere Informationen finden Sie unter the section called “Senden von E-Mails beim Testen der App”.
- Der Ablauf zu einem vergessenen Passwort setzt voraus, dass die E-Mail-Adresse des Benutzers oder die Telefonnummer des Benutzers verifiziert wird.

Important

Wenn sich ein Benutzer sowohl mit einer Telefonnummer als auch mit einer E-Mail-Adresse anmeldet und Ihre Benutzerpooleinstellungen die Verifizierung beider Attribute vorschreiben, sendet Amazon Cognito einen Verifizierungscode per SMS-Nachricht an die Telefonnummer. Amazon Cognito hat die E-Mail-Adresse noch nicht verifiziert, daher muss Ihre App anrufen, GetUserum zu erfahren, ob eine E-Mail-Adresse noch bestätigt werden muss. Wenn eine Bestätigung erforderlich ist, muss die App anrufen, GetUserAttributeVerificationCodeum den E-Mail-Bestätigungsprozess einzuleiten. Anschließend muss sie den Bestätigungscode telefonisch einreichen VerifyUserAttribute.

Sie können Ihr Ausgabenkontingent für SMS-Nachrichten für eine AWS-Konto und für einzelne Nachrichten anpassen. Die Limits gelten nur für die Kosten für das Senden von SMS-Nachrichten. Weitere Informationen finden Sie unter Was sind Ausgabequoten auf Konto- und auf Nachrichtenebene und wie funktionieren sie? in den Häufig gestellten Fragen zu Amazon SNS.

Amazon Cognito sendet SMS-Nachrichten mithilfe von Amazon SNS SNS-Ressourcen entweder in dem Land, in AWS-Region dem Sie den Benutzerpool erstellt haben, oder in einer älteren Amazon

SNS SNS-Alternativregion aus der folgenden Tabelle. Die Ausnahme sind Amazon-Cognito-Benutzerpools in der Region Asien-Pazifik (Seoul). Diese Benutzerpools verwenden Ihre Amazon-SNS-Konfiguration in der Region Asien-Pazifik (Tokio). Weitere Informationen finden Sie unter Wählen Sie AWS-Region für Amazon SNS SMS-Nachrichten.

Amazon-Cognito-Region	Veraltete alternative Amazon-SNS-Region
US East (Ohio)	USA Ost (Nord-Virginia)
Asia Pacific (Mumbai)	Asien-Pazifik (Singapur)
Asia Pacific (Seoul)	Asien-Pazifik (Tokio)
Canada (Central)	USA Ost (Nord-Virginia)
Europe (Frankfurt)	Europa (Irland)
Europe (London)	Europa (Irland)

Beispiel:Wenn sich Ihr Amazon-Cognito-Benutzerpool in der Region Asien-Pazifik (Mumbai) befindet und Sie Ihr Ausgabenlimit in ap-southeast-1 erhöht haben, möchten Sie möglicherweise keine separate Erhöhung in ap-southeast-1 beantragen. Stattdessen können Sie Ihre Amazon-SNS-Ressourcen in Asien-Pazifik (Singapur) verwenden.

Verifizieren von Aktualisierungen von E-Mail-Adressen und Telefonnummern

Ein E-Mail-Adress- oder Telefonnummernattribut kann sofort aktiv und nicht verifiziert werden, sobald Ihr Benutzer seinen Wert geändert hat. Amazon Cognito kann auch verlangen, dass Ihr Benutzer den neuen Wert verifiziert, bevor Amazon Cognito das Attribut aktualisiert. Wenn Sie verlangen, dass Ihre Benutzer zuerst den neuen Wert verifizieren, können diese den ursprünglichen Wert für die Anmeldung und den Empfang von Nachrichten verwenden, bis sie den neuen Wert bestätigen.

Wenn Ihre Benutzer ihre E-Mail-Adresse oder Telefonnummer als Anmeldealias in Ihrem Benutzerpool verwenden können, hängt ihr Anmeldename für ein aktualisiertes Attribut davon ab, ob Sie die Verifizierung aktualisierter Attribute verlangen. Wenn Benutzer ein aktualisiertes Attribut bestätigen müssen, kann sich ein Benutzer mit dem ursprünglichen Attributwert anmelden, bis er den neuen Wert verifiziert hat. Wenn Sie nicht verlangen, dass Benutzer ein aktualisiertes Attribut verifizieren, kann sich ein Benutzer weder mit dem neuen noch mit dem ursprünglichen Attributwert anmelden oder Nachrichten empfangen, bis er den neuen Wert bestätigt hat.

Ihr Benutzerpool ermöglicht beispielsweise die Anmeldung mit einem E-Mail-Adressalias und verlangt, dass Benutzer ihre E-Mail-Adresse bei der Aktualisierung verifizieren. Sue, die sich als `sue@example.com` anmeldet, möchte ihre E-Mail-Adresse in `sue2@example.com` ändern, gibt aber versehentlich **s**`sue2@example.com` ein. Sue erhält die Verifizierungs-E-Mail nicht, daher kann sie `ssue2@example.com` nicht bestätigen. Sue meldet sich als `sue@example.com` an und sendet das Formular in Ihrer App erneut, um ihre E-Mail-Adresse in `sue2@example.com` zu ändern. Sie erhält diese E-Mail, stellt Ihrer App den Bestätigungscode zur Verfügung und beginnt, sich als `sue2@example.com` anzumelden.

Wenn Benutzer ein Attribut aktualisieren und Ihr Benutzerpool neue Attributwerte verifiziert

- Benutzer können sich mit dem ursprünglichen Attributwert anmelden, bevor sie den Code bestätigt haben, um den neuen Wert zu verifizieren.
- Sie können sich nur mit dem neuen Attributwert anmelden, nachdem sie den Code bestätigt haben, um den neuen Wert zu verifizieren.
- Wenn Sie `true` in einer AdminUpdateUserAttributesAPI-Anfrage `email_verified` oder `phone_number_verified` auf festlegen, können sie sich anmelden, bevor sie den Code bestätigt haben, den Amazon Cognito an sie gesendet hat.

Wenn Benutzer ein Attribut aktualisieren und Ihr Benutzerpool neue Attributwerte nicht verifiziert

- Benutzer können sich nicht mit dem ursprünglichen Attributwert anmelden oder Nachrichten empfangen.
- Sie können sich nicht mit dem neuen Attributwert anmelden oder Nachrichten nur mit einem Bestätigungscode empfangen, bevor sie den Code bestätigt haben, um den neuen Wert zu verifizieren.
- Wenn Sie `true` in einer AdminUpdateUserAttributesAPI-Anfrage `email_verified` oder `phone_number_verified` auf festlegen, können sie sich anmelden, bevor sie den Code bestätigt haben, den Amazon Cognito an sie gesendet hat.

Verlangen Sie eine Attributverifizierung wie folgt, wenn Benutzer ihre E-Mail-Adresse oder Telefonnummer aktualisieren

1. Melden Sie sich bei der Amazon Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldedaten ein.

2. Wählen Sie im Navigationsbereich erst User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie auf der Registerkarte Sign-up experience (Anmeldeerlebnis) die Option Edit (Bearbeiten) unter Attribute verification and user account confirmation (Attributverifikation und Benutzerkontobestätigung) aus.
4. Wählen Sie Keep original attribute value active when an update is pending (Ursprünglichen Attributwert aktiv lassen, wenn ein Update aussteht) aus.
5. Wählen Sie unter Active attribute values when an update is pending (Aktive Attributwerte, wenn ein Update aussteht) die Attribute aus, die Ihre Benutzer verifizieren müssen, bevor Amazon Cognito den Wert aktualisiert.
6. Wählen Sie Änderungen speichern aus.

Um eine Überprüfung der Attributaktualisierung mit der Amazon Cognito Cognito-API zu verlangen, können Sie den `AttributesRequireVerificationBeforeUpdate` Parameter in einer UpdateUserPoolAnfrage festlegen.

Autorisieren Sie Amazon Cognito zum Senden von SMS in Ihrem Auftrag.

Amazon Cognito benötigt Ihre Zustimmung, um in Ihrem Auftrag SMS-Nachrichten an Benutzer zu senden. Um diese Berechtigung zu erteilen, können Sie eine AWS Identity and Access Management (IAM-) Rolle erstellen. Wählen Sie auf der Registerkarte Messaging der Amazon-Cognito-Konsole unter SMS die Option Bearbeiten aus, um eine Rolle festzulegen.

Konfigurieren der Nachrichten zur SMS- und E-Mail-Verifizierung und zur Einladung von Benutzern

Mit Amazon Cognito können Sie SMS- und E-Mail-Verifizierungsnachrichten sowie Nachrichten zur Einladung von Benutzern anpassen, um die Sicherheit und Benutzererfahrung Ihrer Anwendung zu verbessern. Mit Amazon Cognito können Sie je nach den Anforderungen Ihrer Anwendung zwischen Code-basierten oder Ein-Klick-Link-Verifizierungen wählen. In diesem Thema wird erläutert, wie Sie die Kommunikation über Multi-Faktor-Authentifizierung (MFA) und Verifizierung in der Amazon Cognito-Konsole personalisieren können.

Auf der Registerkarte Messaging unter Nachrichtenvorlagen können Sie Folgendes anpassen:

- Ihre SMS-Nachricht mit Multi-Faktor-Authentifizierung (MFA)
- Ihrer Nachrichten zur SMS- und E-Mail-Verifizierung

- Der Verifizierungstyp für den E-Mail-Code oder -Link
- Ihre Nachrichten zur Einladung von Benutzern
- Absender- und Empfänger-E-Mail-Adressen für E-Mails des Benutzerpools

> **Note**
>
> Die Vorlagen für Nachrichten zur SMS- und E-Mail-Verifizierung werden nur angezeigt, wenn Sie auf der Registerkarte Verfications (Verifizierungen) angegeben haben, dass die Verifizierung von Telefonnummern und E-Mail-Adressen erforderlich ist. Auch die Vorlage der SMS-MFA-Nachricht wird nur angezeigt, wenn die MFA-Einstellung als erforderlich oder optional festgelegt ist.

Themen

Nachrichtenvorlagen

Mithilfe von Nachrichtenvorlagen können Sie über Platzhalter, die durch einen entsprechenden Wert ersetzt werden, Felder in Ihre Nachrichten einfügen.

Vorlagenplatzhalter

Beschreibung	Token
Verifizierungscode	`{####}`
Temporäres Passwort	`{####}`
Benutzername	`{username}`

> **Note**
>
> Sie können den `{username}`-Platzhalter in Nachrichten zur Verifizierung der E-Mail-Adresse nicht verwenden. Sie können den `{username}` Platzhalter in Einladungs-E-Mail-Nachrichten verwenden, die Sie mit der -AdminCreateUserOperation generieren. Diese Einladungs-E-Mail-Nachrichten verwenden zwei Platzhalter: den Benutzernamen als `{username}` und das temporäre Passwort als `{####}`.

Sie können die Platzhalter in der Vorlage für die erweiterte Sicherheit für folgende Zwecke verwenden:

- Einfügen von Details zu einem Ereignis, wie IP-Adresse, Stadt, Land, Anmeldezeitpunkt und Gerätename. Die erweiterten Sicherheitsfunktionen von Amazon Cognito können diese Details analysieren.
- Bestätigen, ob ein Klicklink gültig ist
- Verwenden Sie die Ereignis-ID, das Feedback-Token und den Benutzernamen zum Erstellen eigener Klicklinks.

> **Note**
>
> Sie müssen bereits über eine konfigurierte Domain für Ihren Benutzerpool verfügen, um Klicklinks zu generieren und die Platzhalter `{one-click-link-valid}` und `{one-click-link-invalid}` in erweiterten Sicherheits-E-Mail-Vorlagen verwenden zu können.

Platzhalter in der Vorlage für die erweiterte Sicherheit

Beschreibung	Token
IP-Adresse	`{ip-address}`
Ort	`{city}`
Country (Land)	`{country}`
Anmeldezeit	`{login-time}`

Beschreibung	Token
Gerätename	`{device-name}`
Klicklink ist gültig	`{one-click-link-valid}`
Klicklink ist ungültig	`{one-click-link-invalid}`
Ereignis-ID	`{event-id}`
Feedback-Token	`{feedback-token}`

Anpassen der SMS-Nachricht

> **Note**
>
> In der neuen Amazon-Cognito-Konsolenumgebung können Sie SMS-Nachrichten anpassen.

Sie können die SMS-Nachricht auf der Registerkarte Messaging unter dem Menüpunkt Nachrichtenvorlagen für die Multi-Faktor-Authentifizierung (MFA) anpassen.

> **Important**
>
> Die benutzerdefinierte Nachricht muss den `{####}`-Platzhalter enthalten. Dieser Platzhalter wird mit dem Authentifizierungscode ersetzt, bevor die Nachricht gesendet wird.

Amazon Cognito begrenzt die Länge für SMS-Nachrichten, einschließlich des Authentifizierungscodes, auf maximal 140 UTF-8-Zeichen.

Anpassen von Nachrichten zur SMS-Verifizierung

Sie können die SMS-Nachricht für die Telefonnummer-Überprüfungen durch Bearbeiten der Vorlage unter Do you want to customize your SMS verification messages? (Möchten Sie die E-Mail-Verifizierungsnachrichten anpassen?) anpassen.

> **Important**
>
> Die benutzerdefinierte Nachricht muss den `{####}`-Platzhalter enthalten. Der Platzhalter wird mit dem Verifizierungscode ersetzt, bevor die Nachricht gesendet wird.

Die maximale Nachrichtenlänge beträgt 140 UTF-8-Zeichen, einschließlich des Verifizierungscodes.

Anpassen von Nachrichten zur E-Mail-Verifizierung

Um die E-Mail-Adresse eines Benutzers in Ihrem Benutzerpool mit Amazon Cognito zu überprüfen, können Sie dem Benutzer eine E-Mail-Nachricht mit einem Link senden, den er auswählen kann, oder Sie können ihm einen Code senden, den er eingeben kann.

Wenn Sie den E-Mail-Betreff und den Nachrichteninhalt für Bestätigungsnachrichten von E-Mail-Adressen anpassen möchten, bearbeiten Sie die Vorlage Verifizierungsnachrichten auf der Registerkarte Messaging Ihres Benutzerpools. Sie können einen Verifizierungstyp bzw. Code oder Link auswählen, wenn Sie Ihre Vorlage Verifizierungsnachrichten bearbeiten.

Wenn Sie Code als Verifizierungstyp ausgewählt haben, muss Ihre benutzerdefinierte Nachricht den Platzhalter `{####}` enthalten. Dieser Platzhalter wird mit dem Verifizierungscode ersetzt, wenn Sie die Nachricht senden.

Wenn Sie Link als Verifizierungstyp ausgewählt haben, muss Ihre benutzerdefinierte Nachricht den Platzhalter im Format `{##Verify Your Email##}` enthalten. Sie können die Textzeichenfolge zwischen den Platzhalterzeichen ändern, z. B.: `{##Click here##}`. Dieser Platzhalter wird durch einen Verifizierungslink mit dem Titel Verifizieren Ihrer E-Mail-Adresse ersetzt.

Der Link für eine E-Mail-Bestätigungsnachricht leitet Ihren Benutzer zu einer URL weiter, wie im folgenden Beispiel gezeigt.

```
https://<your user pool domain>/confirmUser/?
client_id=abcdefg12345678&user_name=emailtest&confirmation_code=123456
```

Die maximale Nachrichtenlänge liegt bei 20 000 UTF-8-Zeichen, einschließlich des Verifizierungscodes (falls vorhanden). Sie können in dieser Nachricht HTML-Tags verwenden, um den Inhalt zu formatieren.

Anpassen von Nachrichten zur Einladung von Benutzern

Sie können den Benutzer-Einladungstext personalisieren, den Amazon Cognito neuen Benutzern per SMS oder E-Mail zusendet, indem Sie die Vorlage Einladungsnachricht auf der Registerkarte Messaging bearbeiten.

> **Important**
>
> Die benutzerdefinierte Nachricht muss die Platzhalter `{username}` und `{####}` enthalten. Beim Versenden der Einladungsnachricht ersetzt Amazon Cognito diese Platzhalter durch den Benutzernamen und das Passwort Ihres Benutzers.

Die maximale Länge einer SMS-Nachricht, einschließlich des Verifizierungscodes, beträgt 140 UTF-8-Zeichen. Die maximale Länge einer E-Mail-Nachricht, einschließlich des Verifizierungscodes, beträgt 20 000 UTF-8-Zeichen. Sie können in den E-Mail-Nachrichten HTML-Tags verwenden, um den Inhalt zu formatieren.

Anpassen Ihrer E-Mail-Adresse

Standardmäßig stammen die E-Mail-Nachrichten, die Amazon Cognito an Benutzer in Ihren Benutzerpools sendet, von der Adresse no-reply@verificationemail.com. Sie können personalisierte Absender-E-Mail-Adressen und Empfänger-E-Mail-Adressen angeben, die anstelle von no-reply@verificationemail.com verwendet werden sollen.

So passen Sie die E-Mail-Adressen für FROM und REPLY-TO an

1. Navigieren Sie zur Amazon-Cognito-Konsole und wählen Sie User Pools (Benutzerpools) aus.
2. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen Benutzerpool.
3. Wählen Sie die Registerkarte Messaging aus. Wählen Sie unter Email (E-Mail) die Option Edit (Bearbeiten) aus.
4. Wählen Sie eine SES-Region aus.
5. Wählen Sie eine FROM email address (Absender-E-Mail-Adresse) aus der Liste der E-Mail-Adressen aus, die Sie bei Amazon SES in der ausgewählten SES Region (SES-Region) verifiziert haben. Um eine E-Mail-Adresse aus einer verifizierten Domäne zu verwenden, konfigurieren Sie die E-Mail-Einstellungen im AWS Command Line Interface oder in der AWS-

API. Weitere Informationen finden Sie unter Verifizieren von E-Mail-Adressen und Domänen in Amazon SES im Entwicklerhandbuch für Amazon Simple Email Service.

6. Wählen Sie einen Konfigurationssatz aus der Liste von Konfigurationssätzen in Ihrer ausgewählten SES-Region aus.
7. Geben Sie einen Absendernamen für Ihre E-Mail-Nachrichten im Format `John Stiles <johnstiles@example.com>` ein.
8. Um die Antwort-E-Mail-Adresse anzupassen, geben Sie eine gültige E-Mail-Adresse in das Feld Antwort-E-Mail-Adresse ein.

Autorisieren von Amazon Cognito zum Senden von Amazon-SES-E-Mails in Ihrem Auftrag (über eine benutzerdefinierte Absender-E-Mail-Adresse)

Sie können Amazon Cognito so konfigurieren, dass es E-Mails von einer benutzerdefinierten Absender-E-Mail-Adresse anstelle von seiner Standardadresse sendet. Um eine benutzerdefinierte Adresse zu verwenden, müssen Sie Amazon Cognito die Berechtigung erteilen, E-Mail-Nachrichten von einer von Amazon SES verifizierten Identität zu senden. In den meisten Fällen können Sie die Berechtigung erteilen, indem Sie eine Sendeautorisierungsrichtlinie erstellen. Weitere Informationen finden Sie unter Using sending authorization with Amazon SES (Verwenden der Sendeautorisierung mit Amazon SES) im Entwicklerhandbuch für Amazon Simple Email Service.

Wenn Sie einen Benutzerpool für die Verwendung von Amazon SES für E-Mail-Nachrichten konfigurieren, erstellt Amazon Cognito die `AWSServiceRoleForAmazonCognitoIdpEmailService`-Rolle in Ihrem Konto, um Zugriff auf Amazon SES zu gewähren. Es ist keine Berechtigungsrichtlinie für das Senden von Berechtigungen erforderlich, wenn die servicegebundene `AWSServiceRoleForAmazonCognitoIdpEmailService`-Rolle verwendet wird. Sie müssen nur eine Versandautorisierungsrichtlinie hinzufügen, wenn Sie sowohl die Standard-E-Mail-Funktionalität in Ihrem Benutzerpool verwenden als auch eine verifizierte Amazon-SES-Identität als Absenderadresse.

Weitere Informationen zur serviceverknüpften Rolle, die Amazon Cognito erstellt, finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon Cognito.

Das folgende Beispiel für eine Sendeautorisierungsrichtlinie gewährt Amazon Cognito die eingeschränkte Möglichkeit, eine von Amazon SES verifizierte Identität zu verwenden. Amazon Cognito kann nur dann E-Mail-Nachrichten senden, wenn es diese Funktion für den Benutzerpool in der `aws:SourceArn`-Bedingung als auch für das Konto in der `aws:SourceAccount-`

Bedingung übernimmt. Weitere Beispiele finden Sie unter Beispiele von Amazon-SES-Sendeautorisierungsrichtlinien im Entwicklerhandbuch für Amazon Simple Email Service.

> **Note**
>
> In diesem Beispiel ist der Wert "Sid" eine beliebige Zeichenfolge, die die Anweisung eindeutig identifiziert. Weitere Informationen zur Richtliniensyntax finden Sie unter Amazon-SES-Sendeautorisierungsrichtlinien im Entwicklerhandbuch für Amazon Simple Email Service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "<your SES identity ARN>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "aws:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
}
```

Die Amazon-Cognito-Konsole fügt eine ähnliche Richtlinie für Sie hinzu, wenn Sie im Dropdown-Menü eine Amazon-SES-Identität auswählen. Wenn Sie den Benutzerpool mit der CLI oder der

API konfigurieren, müssen Sie eine Richtlinie an Ihre Amazon-SES-Identität anfügen, die wie das vorherige Beispiel strukturiert ist.

Erstellen von Benutzerkonten als Administrator

Wenn Sie einen eigenen Benutzerpool erstellen, können Sie mithilfe der AWS Management Console, über die AWS Command Line Interface oder mit der Amazon-Cognito-API Benutzer erstellen. Sie können ein Profil für einen neuen Benutzer in einem Create a User Pool und eine Willkommensnachricht mit Anmeldeinformationen an den Benutzer per SMS oder E-Mail senden.

Entwickler und Administratoren können die folgenden Aufgaben ausführen:

- Ein neues Benutzerprofil mithilfe der AWS Management Console oder durch Aufrufen der `AdminCreateUser`-API erstellen.
- Legen Sie Benutzerattributwerte fest.
- Erstellen Sie benutzerdefinierte Attribute.
- Legen Sie den Wert unveränderlicher benutzerdefinierter Attribute in `AdminCreateUser`-API-Anfragen fest. Dieses Feature ist in der Amazon-Cognito-Konsole nicht verfügbar.
- Das temporäre Passwort angeben oder von Amazon Cognito automatisch eines generieren lassen.
- Angeben, ob bereitgestellte E-Mail-Adressen und Telefonnummern für neue Benutzer als verifiziert markiert werden.
- Benutzerdefinierte SMS- und E-Mail-Einladungsnachrichten für neue Benutzer über die AWS Management Console oder einen benutzerdefinierten Message Lambda-Auslöser angeben. Weitere Informationen finden Sie unter Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern.
- Angeben, ob Einladungs-Nachrichten per SMS, E-Mail oder beides gesendet werden.
- Begrüßungsnachricht an einen vorhandenen Benutzer erneut senden, indem die `AdminCreateUser`-API aufgerufen und `RESEND` für den `MessageAction`-Parameter festgelegt wird.

  > **Note**
  >
  > Diese Aktion kann derzeit nicht mithilfe der durchgeführt werden AWS Management Console.

- Das Senden der Einladungsnachricht beim Erstellen des Benutzers unterdrücken.

- Eine Ablaufzeitbeschränkung für das Benutzerkonto (bis zu 90 Tage) angeben.
- Benutzer gestatten, sich anzumelden, oder verlangen, dass neue Benutzer nur durch den Administrator hinzugefügt werden.

Authentifizierungsablauf für Benutzer, die durch Administratoren oder Entwickler erstellt wurden

Der Authentifizierungsablauf für diese Benutzer umfasst den zusätzlichen Schritt des Einsendens des neuen Passwort und Bereitstellen aller fehlenden Werte für erforderliche Attribute. Die Schritte sind nachstehend aufgeführt; die Schritte 5, 6 und 7 gelten nur für diesen Benutzer.

1. Der Benutzer startet die erstmalige Anmeldung, indem er seinen Benutzernamen und sein Passwort absendet.
2. Das SDK ruft auf `InitiateAuth(Username, USER_SRP_AUTH)`.
3. Amazon Cognito gibt die `PASSWORD_VERIFIER`-Herausforderung mit Salt-und-Secret-Block zurück.
4. Das SDK führt die SRP-Berechnungen durch und ruft auf `RespondToAuthChallenge(Username, <SRP variables>, PASSWORD_VERIFIER)`.
5. Amazon Cognito gibt die Aufforderung `NEW_PASSWORD_REQUIRED` zurück. Der Hauptteil dieser Aufforderung umfasst die aktuellen Attribute des Benutzers und alle erforderlichen Attribute in Ihrem Benutzerpool, die derzeit keinen Wert im Benutzerprofil haben. Weitere Informationen finden Sie unter RespondToAuthChallenge.
6. Der Benutzer wird aufgefordert und gibt ein neues Passwort und alle fehlenden Werte für erforderliche Attribute ein.
7. Das SDK ruft auf `RespondToAuthChallenge(Username, <New password>, <User attributes>)`.
8. Wenn der Benutzer einen zweiten Faktor für MFA benötigt, gibt Amazon Cognito die SMS_MFA-Herausforderung zurück und der Code wird übermittelt.
9. Nachdem der Benutzer das Passwort erfolgreich geändert und optional Attribute bereitgestellt oder MFA abgeschlossen hat, wird der Benutzer angemeldet, und Token werden ausgestellt.

Wenn der Benutzer alle Herausforderungen erfüllt hat, markiert der Amazon-Cognito-Service den Benutzer als bestätigt und gibt ID, Zugriff und Aktualisierungs-Token für den Benutzer aus. Weitere Informationen finden Sie unter Verwenden von Token mit Benutzerpools.

Erstellen eines neuen Benutzers in der AWS Management Console

Sie können die Anforderungen an das Benutzerpasswort festlegen, die an Benutzer gesendeten Einladungs- und Verifizierungsnachrichten konfigurieren und neue Benutzer mit der Amazon-Cognito-Konsole hinzufügen.

Festlegen einer Passwortrichtlinie und Aktivieren der Selbstregistrierung

Sie können Einstellungen für minimale Passwortkomplexität konfigurieren und ob Benutzer sich mit öffentlichen APIs in Ihrem Benutzerpool anmelden können.

Konfigurieren einer Passwortrichtlinie

1. Navigieren Sie zur Amazon-Cognito-Konsole und wählen Sie User Pools (Benutzerpools) aus.
2. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
3. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus und suchen Sie nach Password policy (Passwortrichtlinie). Wählen Sie Edit.
4. Wählen Sie für den Passwortrichtlinienmodus Custom (Benutzerdefiniert) aus.
5. Wählen Sie eine Mindestpasswortlänge aus. Beschränkungen für die Passwortlänge finden Sie unter User pools resource quotas (Benutzerpool-Ressourcenkontingente).
6. Wählen Sie eine Passwortkomplexität aus.
7. Wählen Sie aus, wie lange das von Administratoren festgelegte Passwort gültig sein soll.
8. Wählen Sie Save Changes.

Zulassen der Self-Service-Anmeldung

1. Navigieren Sie zur Amazon-Cognito-Konsole und wählen Sie User Pools (Benutzerpools) aus.
2. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
3. Wählen Sie die Registerkarte Sign-up experience (Anmeldungserlebnis) aus und suchen Sie nach Self-service sign-up (Self-Service-Anmeldung). Wählen Sie Edit (Bearbeiten) aus.
4. Wählen Sie aus, ob Sie die Selbstregistrierung aktivieren möchten. Die Selbstregistrierung wird normalerweise mit öffentlichen App-Clients verwendet, die neue Benutzer in Ihrem Benutzerpool registrieren müssen, ohne ein Clientgeheimnis oder AWS Identity and Access Management (IAM)-API-Anmeldeinformationen zu verteilen.

> **Deaktivieren der Selbstregistrierung**
>
> Wenn Sie die Selbstregistrierung nicht aktivieren, müssen neue Benutzer durch administrative API-Aktionen mithilfe von IAM-API-Anmeldeinformationen oder durch Anmeldung bei Verbundanbietern erstellt werden.

5. Wählen Sie Änderungen speichern.

Anpassen von E-Mail- und SMS-Nachrichten

Anpassen von Benutzernachrichten

Sie können die Nachrichten anpassen, die Amazon Cognito an Ihre Benutzer sendet, wenn Sie sie zur Anmeldung einladen, sie sich für ein Benutzerkonto anmelden oder sich anmelden und zur Multi-Faktor-Authentifizierung (MFA) weitergeleitet werden.

> **Note**
>
> Es wird eine Einladungsnachricht gesendet, wenn Sie einen Benutzer in Ihrem Benutzerpool erstellen und ihn einladen, sich anzumelden. Amazon Cognito sendet erste Anmeldeinformationen an die E-Mail-Adresse oder die Telefonnummer des Benutzers.
> Eine Verifizierungsnachricht wird gesendet, wenn sich ein Benutzer für ein Benutzerkonto in Ihrem Benutzerpool anmeldet. Amazon Cognito sendet einen Code an den Benutzer. Wenn der Benutzer Amazon Cognito den Code zur Verfügung stellt, überprüft er seine Kontaktinformationen und bestätigt sein Konto für die Anmeldung. Verifizierungscodes sind 24 Stunden lang gültig.
> Eine MFA-Nachricht wird gesendet, wenn Sie SMS MFA in Ihrem Benutzerpool aktivieren, und ein Benutzer, der SMS MFA konfiguriert hat, sich anmeldet und zu MFA weitergeleitet wird.

1. Navigieren Sie zur Amazon-Cognito-Konsole und wählen Sie User Pools (Benutzerpools) aus.
2. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
3. Wählen Sie die Registerkarte Messaging aus und suchen Sie nach Nachrichtenvorlagen. Wählen Sie Verification messages (Verifizierungsnachrichten),Invitation messages

(Einladungsnachrichten) oder MFA messages (MFA-Nachrichten) aus und klicken Sie anschließend auf Edit (Bearbeiten).

4. Passen Sie die Nachrichten für den ausgewählten Nachrichtentyp an.

> **Note**
>
> Alle Variablen in Nachrichtenvorlagen müssen beim Anpassen der Nachricht enthalten sein. Wenn die Variable, zum Beispiel {####}, nicht enthalten ist, wird Ihr Benutzer nicht über ausreichende Informationen verfügen, um die Nachrichtenaktion abzuschließen. Weitere Informationen finden Sie unter Message templates (Nachrichtenvorlagen).

5. a. Verifizierungsnachrichten

   i. Wählen Sie einen Verifizierungs-Typ für E-Mail-Nachrichten. Eine Codeverifizierung sendet einen numerischen Code, den der Benutzer eingeben muss. Eine Linkverifizierung sendet einen Link, den der Benutzer für die Verifizierung der Kontaktinformationen anklicken kann. Der Text in der Variable für eine Linknachricht wird als Hyperlinktext angezeigt. Beispielsweise wird eine Nachrichtenvorlage mit der Variable {##Click here##} als Click here (Hier klicken) in der E-Mail-Nachricht angezeigt.

   ii. Geben Sie einen E-Mail-Betreff für E-Mail-Nachrichten ein.

   iii. Geben Sie eine benutzerdefinierte E-Mail-Vorlage für E-Mail-Nachrichten ein. Sie können diese Vorlage mit HTML anpassen.

   iv. Geben Sie eine benutzerdefinierte SMS-Vorlage für SMS-Nachrichten ein.

   v. Wählen Sie Save Changes.

   b. Einladungsnachrichten

   i. Geben Sie einen E-Mail-Betreff für E-Mail-Nachrichten ein.

   ii. Geben Sie eine benutzerdefinierte E-Mail-Vorlage für E-Mail-Nachrichten ein. Sie können diese Vorlage mit HTML anpassen.

   iii. Geben Sie eine benutzerdefinierte SMS-Vorlage für SMS-Nachrichten ein.

   iv. Wählen Sie Save Changes.

   c. MFA-Nachrichten

   i. Geben Sie eine benutzerdefinierte SMS-Vorlage für SMS-Nachrichten ein.

   ii. Wählen Sie Save Changes.

Erstellen eines Benutzers

Erstellen eines Benutzers

Sie können über die Amazon-Cognito-Konsole neue Benutzer für Ihren Benutzerpool erstellen. In der Regel können sich Benutzer anmelden, nachdem sie ein Passwort festgelegt haben. Zum Anmelden mit einer E-Mail-Adresse oder einer Telefonnummer muss ein Benutzer das `email`-Attribut verifizieren. Zum Anmelden mit einer Telefonnummer muss der Benutzer das `phone_number`-Attribut verifizieren. Wenn Sie Konten als Administrator bestätigen möchten, können Sie auch die AWS CLI oder die API verwenden oder Benutzerprofile bei einem Verbundidentitätsanbieter erstellen. Weitere Informationen finden Sie in der Amazon-Cognito-API-Referenz.

1. Navigieren Sie zur Amazon-Cognito-Konsole und wählen Sie User Pools (Benutzerpools) aus.
2. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen Benutzerpool.
3. Wählen Sie die Registerkarte Users (Benutzer) und anschließend die Option Create a user (Benutzer erstellen) aus.
4. In den Anforderungen für die Anmeldung im Benutzerpool und die Sicherheit finden Sie Anleitungen zu Passwortanforderungen, verfügbare Methoden zur Kontowiederherstellung und Aliasattribute für Ihren Benutzerpool.
5. Wählen Sie aus, wie Sie eine Invitation message (Einladungsnachricht) senden möchten. Wählen Sie SMS-Nachricht, E-Mail-Nachricht oder beides aus.

   > **Note**
   >
   > Bevor Sie Einladungsnachrichten senden können, müssen Sie einen Absender und die AWS-Region mit Amazon Simple Notification Service und Amazon Simple Email Service auf der Registerkarte Messaging in Ihrem Benutzerpool konfigurieren. Es gelten Empfängernachrichten und Datengebühren. Amazon SES berechnet Ihnen E-Mail-Nachrichten separat und Amazon SNS berechnet Ihnen SMS-Nachrichten separat.

6. Wählen Sie einen Username (Benutzername) für den neuen Benutzer aus.
7. Wählen Sie aus, ob Sie mit Create a password (Passwort erstellen) ein Passwort erstellen oder Amazon Cognito mit Generate a password (Passwort generieren) ein Passwort für den Benutzer generieren lassen möchten. Jedes temporäre Passwort muss der Passwortrichtlinie des Benutzerpools entsprechen.
8. Wählen Sie Create (Erstellen) aus.

9. Wählen Sie die Registerkarte Users (Benutzer) aus und klicken Sie auf den Eintrag User name (Benutzername) für den Benutzer. Fügen Sie User attributes (Benutzerattribute) und Group memberships (Gruppenmitgliedschaften) hinzu und bearbeiten Sie diese. Sehen Sie sich User event history (Ereignisverlauf des Benutzers) an.

Hinzufügen von Gruppen zu einem Benutzerpool

Durch Unterstützung für Gruppen in Amazon-Cognito-Benutzerpools können Sie Gruppen erstellen und verwalten, Benutzer zu Gruppen hinzufügen und aus ihnen entfernen. Verwenden Sie Gruppen, um Sammlungen von Benutzern zu erstellen und deren Berechtigungen zu verwalten oder verschiedene Arten von Benutzern darzustellen. Sie können einer Gruppe eine AWS Identity and Access Management (IAM)-Rolle zuweisen, um die Berechtigungen für Mitglieder einer Gruppe zu definieren.

Sie können Gruppen verwenden, um eine Auswahl von Benutzern in einem Benutzerpool zu erstellen, womit oft die Berechtigungen für diese Benutzer festgelegt werden. Beispielsweise können Sie separate Gruppen für Benutzer, die Leser, Mitwirkende und Redakteure Ihrer Website und Anwendung sind, erstellen. Über die IAM-Rolle, die einer Gruppe zugeordnet ist, können Sie auch verschiedene Berechtigungen für diese verschiedenen Gruppen festlegen, sodass nur Mitwirkende Inhalte in Amazon S3 stellen und nur Herausgeber Inhalte über eine API in Amazon API Gateway veröffentlichen können.

Sie können Gruppen in einem Benutzerpool über die AWS Management Console, die APIs und die CLI erstellen und verwalten. Als Entwickler (mit - AWS Anmeldeinformationen) können Sie die Gruppen für einen Benutzerpool erstellen, lesen, aktualisieren, löschen und auflisten. Sie können auch Benutzer zu Gruppen hinzufügen und aus ihnen entfernen.

Es fallen keine zusätzlichen Kosten für die Nutzung von Gruppen in einem Benutzerpool an. Weitere Informationen finden Sie unter Amazon Cognito – Preise.

Zuweisen von IAM-Rollen zu Gruppen

Sie können Gruppen verwenden, um Berechtigungen für Ihre Ressourcen mithilfe einer IAM-Rolle zu steuern. IAM-Rollen umfassen Vertrauensrichtlinien und Berechtigungsrichtlinien. Die Rollenvertrauensrichtlinie gibt an, wer die Rolle verwenden kann. Die Berechtigungsrichtlinien geben die Aktionen und Ressourcen an, auf die Ihre Gruppenmitglieder zugreifen können. Wenn Sie eine IAM-Rolle erstellen, richten Sie die Rollenvertrauensrichtlinie ein, damit die Gruppenbenutzer die

Rolle übernehmen können. In Rollenberechtigungsrichtlinien geben Sie die Berechtigungen an, die Ihre Gruppe besitzen soll.

Wenn Sie eine Gruppe in Amazon Cognito erstellen, geben Sie eine IAM-Rolle an, indem Sie den ARN der Rolle angeben. Wenn sich Gruppenmitglieder mit Amazon Cognito anmelden, können sie temporäre Anmeldeinformationen aus den Identitätspools erhalten. Ihre Berechtigungen werden anhand der zugeordneten IAM-Rolle bestimmt.

Einzelne Benutzer können sich in mehreren Gruppen befinden. Als Entwickler haben Sie die folgenden Optionen zur automatischen Auswahl der IAM-Rolle, wenn sich ein Benutzer in mehreren Gruppen befindet:

- Sie können für jede Gruppe Rangfolgenwerte zuweisen. Die Gruppe mit der besseren (niedrigeren) Priorität wird ausgewählt und ihre zugehörige IAM-Rolle wird angewendet.
- Ihre App kann auch aus den verfügbaren Rollen wählen, wenn Anmeldeinformationen für einen Benutzer über einen Identitäten-Pool angefordert AWS werden, indem ein Rollen-ARN im -GetCredentialsForIdentity`CustomRoleARN`Parameter angegeben wird. Die angegebene IAM-Rolle muss einer Rolle entsprechen, die für den Benutzer verfügbar ist.

Zuweisen von Prioritätswerten zu Gruppen

Ein Benutzer kann mehreren Gruppen angehören. In den Zugriffs- und ID-Tokens eines Benutzers enthält der `cognito:groups`-Anspruch die Liste aller Gruppen, denen ein Benutzer angehört. Der `cognito:roles`-Anspruch enthält die Liste der Rollen entsprechend den Gruppen.

Da ein Benutzer mehreren Gruppen angehören kann, kann jeder Gruppe eine Priorität zugewiesen werden. Hierbei handelt es sich um eine nicht-negative Zahl, die die Priorität dieser Gruppe im Vergleich zu anderen Gruppen angibt, denen ein Benutzer im Benutzerpool angehört. Null ist die höchste Priorität. Gruppen mit niedrigen Prioritätswerten haben Vorrang vor Gruppen mit höheren Prioritäts- oder NULL-Werten. Wenn ein Benutzer zwei oder mehr Gruppen angehört, ist es die Gruppe mit dem niedrigsten Prioritätswert, deren IAM-Rolle dem `cognito:preferred_role`-Anspruch im Benutzer-ID-Token zugewiesen wird.

Zwei Gruppen können dieselbe Priorität aufweisen. Wenn dies geschieht, hat keine Gruppe Vorrang vor der anderen. Wenn zwei Gruppen mit demselben Prioritätswert den gleichen Rollen-ARN haben, wird diese Rolle im `cognito:preferred_role`-Anspruch in ID-Token für Benutzer in jeder Gruppe verwendet. Wenn die zwei Gruppen unterschiedliche Rollen-ARNs haben, wird der `cognito:preferred_role`-Anspruch nicht in Benutzer-ID-Token festgelegt.

Verwenden von Gruppen zur Steuerung von Berechtigungen mit Amazon API Gateway

Sie können Gruppen in einem Benutzerpool verwenden, um die Berechtigung mit Amazon API Gateway zu steuern. Die Gruppen, bei denen ein Benutzer Mitglied ist, sind sowohl im ID-Token als auch im Zugriffstoken von einem Benutzerpool im `cognito:groups`-Anspruch enthalten. Sie können ID- oder Zugriffstoken mit Anfragen an Amazon API Gateway senden und einen Amazon-Cognito-Benutzerpool-Autorisierer für eine REST-API verwenden. Weitere Informationen finden Sie unter Control access to a REST API using Amazon cognito user pools as authorizer (Zugriff auf eine REST-API mit Amazon-Cognito-Benutzerpools als Autorisierer) im Entwicklerhandbuch von API Gateway.

Sie können den Zugriff auf eine Amazon-API-Gateway-HTTP-API auch mit einem benutzerdefinierten JWT-Autorisierer autorisieren. Weitere Informationen finden Sie unter Controlling access to HTTP APIs with JWT authorizers (Zugriffskontrolle auf HTTP-APIs mit JWT-Autorisierern) im Entwicklerhandbuch von API Gateway.

Einschränkungen für Gruppen

Benutzergruppen unterliegen den folgenden Einschränkungen:

- Die Anzahl der Gruppen, die Sie erstellen können, ist durch die Amazon Cognito-Servicekontingente begrenzt.
- Gruppen können nicht verschachtelt werden.
- Sie können nicht nach Benutzern in einer Gruppe suchen.
- Sie können nicht nach Namen nach Gruppen suchen, aber Sie können Gruppen auflisten.

Erstellen einer neuen Gruppe in der AWS Management Console

Gehen Sie wie folgt vor, um eine neue Gruppe zu erstellen.

Neue Gruppe erstellen

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre - AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Groups (Gruppen) und dann Create group (Gruppe erstellen) aus.

5. Geben Sie auf der Seite Create a group (Ressourcengruppe erstellen) für Group name (Gruppenname) einen Namen für Ihre Gruppe ein.
6. Sie können optional zusätzliche Informationen zu dieser Gruppe mithilfe eines der folgenden Felder angeben:

   - Description (Beschreibung) – Geben Sie Details darüber ein, wofür diese neue Gruppe verwendet wird.
   - Precedence (Priorität) – Amazon Cognito wertet alle Gruppenberechtigungen für einen bestimmten Benutzer aus und wendet sie an, basierend darauf, zu welchen Gruppen der Benutzer gehört und welche Gruppen den niedrigeren Prioritätswert haben. Die Gruppe mit der niedrigeren Priorität wird ausgewählt und ihre zugehörige IAM-Rolle wird angewendet. Weitere Informationen finden Sie unter Zuweisen von Prioritätswerten zu Gruppen.
   - IAM-Rolle – Sie können Ihrer Gruppe eine IAM-Rolle zuweisen, wenn Sie Berechtigungen für Ihre Ressourcen steuern müssen. Wenn Sie einen Benutzerpool in einen Identitäten-Pool integrieren, bestimmt die Einstellung IAM role (IAM-Rolle), welche Rolle im Benutzer-ID-Token zugewiesen wird, wenn der Identitäten-Pool zur Auswahl der Rolle über das Token konfiguriert ist. Weitere Informationen finden Sie unter Zuweisen von IAM-Rollen zu Gruppen.
   - Add users to this group (Dieser Gruppe Benutzer hinzufügen) – Fügen Sie vorhandene Benutzer als Mitglieder dieser Gruppe hinzu, nachdem sie erstellt wurde.
7. Wählen Sie Create (Erstellen) aus, um dies zu bestätigen.

Verwalten von und Suchen nach Benutzerkonten

Nachdem Sie Ihre Benutzerpool erstellt haben, können Sie mithilfe der AWS Management Console, der AWS Command Line Interface oder der Amazon-Cognito-API Benutzer anzeigen und verwalten. In diesem Thema wird beschrieben, wie Sie mithilfe der Benutzer anzeigen und suchen AWS Management Console.

Anzeigen von Benutzerattributen

Gehen Sie wie folgt vor, um Benutzerattribute in der Amazon-Cognito-Konsole anzuzeigen.

Benutzerattribute aufrufen

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS-Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.

3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Users (Benutzer) und wählen Sie dann einen Benutzer in der Liste aus.
5. Auf der Detailseite des Benutzers unter User attributes (Benutzerattribute) können Sie anzeigen, welche Attribute mit dem Benutzer verknüpft sind.

Zurücksetzen des Passworts eines Benutzers.

Gehen Sie wie folgt vor, um ein Benutzerpasswort in der Amazon-Cognito-Konsole zurückzusetzen.

Benutzerpasswort zurücksetzen

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS-Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Users (Benutzer) und wählen Sie dann einen Benutzer in der Liste aus.
5. Wählen Sie auf der Detailseite des Benutzers die Option Actions (Aktionen), Reset password (Passwort zurücksetzen) aus.
6. Überprüfen Sie im Dialog Reset password (Passwort zurücksetzen) die Daten und wählen Sie anschließend Reset (Zurücksetzen) aus.

   Die Aktion bewirkt, dass ein Bestätigungscode sofort an den Benutzer gesendet und das aktuelle Passwort des Benutzers deaktiviert wird, indem der Benutzerstatus auf `RESET_REQUIRED` gesetzt wird. Der Code Reset password (Passwort zurücksetzen) ist für eine Stunde gültig.

Suchen nach Benutzerattributen

Wenn Sie bereits einen Benutzerpool erstellt haben, können Sie im Bereich Users (Benutzer) in der AWS Management Console danach suchen. Sie können auch die Amazon Cognito ListUsers API verwenden, die einen Filter-Parameter akzeptiert.

Sie können nach jedem der folgenden Standardattribute suchen. Benutzerdefinierte Attribute können nicht gesucht werden.

- username (ohne Beachtung der Groß- und Kleinschreibung)

- email
- phone_number
- Name
- given_name
- family_name
- preferred_username
- cognito:user_status (als Status in der Konsole bezeichnet) (ohne Beachtung der Groß- und Kleinschreibung)
- Status (als Enabled in der Konsole bezeichnet) (Groß-/Kleinschreibung beachten)
- sub

> **Note**
>
> Sie können Benutzer auch mit einem clientseitigen Filter auflisten. Der serverseitige Filter entspricht nicht mehr als einem Attribut. Verwenden Sie für die erweiterte Suche einen clientseitigen Filter mit dem `--query`-Parameter der `list-users`-Aktion in der AWS Command Line Interface. Wenn Sie einen clientseitigen Filter verwenden, gibt ListUsers eine paginierte Liste von null oder mehr Benutzern zurück. Sie können mehrere Seiten in einer Zeile ohne Ergebnisse erhalten. Wiederholen Sie die Abfrage mit jedem Paginierungstoken, das zurückgegeben wird, bis Sie einen Null-Paginierungstoken erhalten, und überprüfen Sie dann das kombinierte Ergebnis.
> Weitere Informationen zur serverseitigen und clientseitigen Filterung finden Sie unter AWS CLI-Ergebnisse filtern im AWS Command Line Interface-Benutzerhandbuch.

Suchen nach Benutzern mit der AWS Management Console

Wenn Sie bereits einen Benutzerpool erstellt haben, können Sie im Bereich Users (Benutzer) in der AWS Management Console danach suchen.

AWS Management Console-Suchvorgänge sind immer Präfix-Suchen ("beginnt mit").

In der Amazon-Cognito-Konsole nach einem Benutzer suchen

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Sie werden möglicherweise aufgefordert, Ihre AWS-Anmeldeinformationen einzugeben.

2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Users (Benutzer) aus und geben Sie dann den Benutzernamen des Benutzers in das Suchfeld ein. Beachten Sie, dass bei einigen Attributwerten zwischen Groß- und Kleinschreibung unterschieden wird. (Beispielsweise bei Username (Benutzername).

   Sie können Benutzer auch finden, indem Sie den Suchfilter anpassen, um den Umfang auf andere Benutzereigenschaften zu beschränken, z. B. Email (E-Mail), Phone number (Telefonnummer) oder Last name (Nachname).

Suchen nach Benutzern mit der **`ListUsers`**-API

Um in Ihrer App nach Benutzern zu suchen, verwenden Sie die Amazon-Cognito-ListUsers-API. Diese API verwendet die folgenden Parameter:

- `AttributesToGet`Ein Array von Zeichenfolgen, in dem jede Zeichenfolge der Name eines Benutzerattributs ist, der für jeden Benutzer in den Suchergebnissen zurückgegeben wird. Wenn Sie alle Attribute abrufen möchten, geben Sie keinen `AttributesToGet`-Parameter an oder fordern `AttributesToGet` mit einem Wert der Literalzeichenfolge `null` an.
- `Filter`: Eine Filter-Zeichenfolge in der Form "`AttributeName Filter-Type "AttributeValue"`". Anführungszeichen in der Filterzeichenfolge müssen mit dem umgekehrten Schrägstrich (\) durch Escape-Zeichen geschützt sein. Zum Beispiel `"family_name = \"Reddy \""`. Wenn die Filterzeichenfolge leer ist, gibt `ListUsers` alle Benutzer im Benutzerpool zurück.
  - `AttributeName`: Die Namen des Attributs, nach dem gesucht werden soll. Sie können nur jeweils nach einem Attribut suchen.

    > **Note**
    >
    > Sie können nur nach Standard-Attributen suchen. Benutzerdefinierte Attribute können nicht gesucht werden. Der Grund hierfür ist, dass nur die indizierten Attribute gesucht werden können, und benutzerdefinierte Attribute können nicht indexiert werden.

  - `Filter-Type`: Für eine exakte Übereinstimmung verwenden Sie =, z. B. `given_name = "Jon"`. Für ein Präfix ("beginnt mit"), verwenden Sie ^=, z. B. `given_name ^= "Jon"`.
  - `AttributeValue`: Die Attributwert, der für jeden Benutzer übereinstimmen muss.
- `Limit`: Maximale Anzahl der zurückgegebenen Benutzer.

- `PaginationToken`: Ein Token, um weitere Ergebnisse aus einer vorherigen Suche zu erhalten. Amazon Cognito lässt das Paginierungstoken nach einer Stunde ablaufen.
- `UserPoolId`: Die Benutzerpool-ID für den Benutzerpool, in dem die Suche durchgeführt werden soll.

Alle Suche sind ohne Berücksichtigung von Groß-/Kleinschreibung. Die Suchergebnisse werden nach dem durch die `AttributeName`-Zeichenfolge benannten Attribut in aufsteigender Reihenfolge sortiert.

Beispiele zu Verwendung der **`ListUsers`**-API

Das folgende Beispiel gibt alle Benutzer zurück und enthält alle Attribute.

```
{
    "AttributesToGet": null,
    "Filter": "",
    "Limit": 10,
    "UserPoolId": "us-east-1_samplepool"
}
```

Das folgende Beispiel gibt alle Benutzer zurück, deren Telefonnummer mit"+1312" beginnt, und enthält alle Attribute.

```
{
    "AttributesToGet": null,
    "Filter": "phone_number ^= \"+1312\"",
    "Limit": 10,
    "UserPoolId": "us-east-1_samplepool"
}
```

Das folgende Beispiel gibt die ersten 10 Benutzer zurück, deren Familienname "Reddy" ist. Für jeden Benutzer umfassen die Suchergebnisse den Rufnamen, die Telefonnummer und die E-Mail-Adresse. Wenn es mehr als 10 passender Benutzer im Benutzerpool gibt, enthält die Antwort ein Paginierungs-Token.

```
{
```

```
    "AttributesToGet": [
        "given_name",
        "phone_number",
        "email"
    ],
    "Filter": "family_name = \"Reddy\"",
    "Limit": 10,
    "UserPoolId": "us-east-1_samplepool"
}
```

Wenn das vorherige Beispiel ein Paginierung-Token zurückgibt, gibt das folgende Beispiel die nächsten 10 Benutzer zurück, die der gleichen Filterzeichenfolge entsprechen.

```
{
    "AttributesToGet": [
        "given_name",
        "phone_number",
        "email"
    ],
    "Filter": "family_name = \"Reddy\"",
    "Limit": 10,
    "PaginationToken": "pagination_token_from_previous_search",
    "UserPoolId": "us-east-1_samplepool"
}
```

Wiederherstellen von Benutzerkonten

Mit dem Parameter `AccountRecoverySetting` können Sie anpassen, welche Methode ein Benutzer verwenden kann, um sein Passwort wiederherzustellen, wenn er die API `ForgotPassword` aufruft. `ForgotPassword` sendet einen Wiederherstellungscode an eine verifizierte E-Mail-Adresse oder eine verifizierte Telefonnummer. Der Wiederherstellungscode ist eine Stunde lang gültig. Wenn Sie eine `AccountRecoverySetting` für Ihren Benutzerpool angeben, wählt Amazon Cognito das Code-Bereitstellungsziel basierend auf der von Ihnen festgelegten Priorität aus.

Wenn Sie `AccountRecoverySetting` definieren und ein Benutzer SMS MFA konfiguriert hat, kann SMS nicht als Mechanismus für die Kontowiederherstellung verwendet werden. Die Priorität für diese Einstellung wird bestimmt. Dabei hat 1 die höchste Priorität. Cognito sendet eine Verifizierung nur an eine der angegebenen Methoden.

Beispielsweise ist `admin_only` ein Wert, den der Administrator verwendet, wenn er nicht möchte, dass der Benutzer sein Konto selbst wiederherstellen kann. Der Benutzer müsste sich stattdessen an

den Administrator wenden, damit dieser sein Konto zurücksetzt. Sie können `admin_only` nicht mit einem anderen Mechanismus für die Kontowiederherstellung verwenden.

Wenn Sie `AccountRecoverySetting` nicht angeben, verwendet Amazon Cognito den älteren Mechanismus, um die Methode für die Passwortwiederherstellung zu ermitteln. In diesem Fall verwendet Cognito zuerst eine verifizierte Telefonnummer. Wenn die verifizierte Telefonnummer für den Benutzer nicht gefunden wird, verwendet Cognito als Nächstes eine verifizierte E-Mail-Adresse.

Weitere Informationen zu `AccountRecoverySetting` finden Sie unter CreateUserPool und UpdateUserPool in der API-Referenz zum Amazon Cognito-Identitätsanbieter.

Verhalten bei „Passwort vergessen“

Innerhalb einer Stunde erlauben wir einem Benutzer zwischen 5 und 20 Versuche, einen Passwort-Reset-Code als Teil der Aktionen „forgot-password“ und „confirm-forgot-password“ anzufordern oder einzugeben. Der genaue Wert hängt von den Risikoparametern ab, die den Anforderungen zugeordnet sind. Bitte beachten Sie, dass sich dieses Verhalten ändert.

Importieren von Benutzern in einen Benutzerpool

Es gibt zwei Möglichkeiten, wie Sie Benutzer aus Ihrem vorhandenen Benutzerverzeichnis oder aus einer Benutzerdatenbank in Amazon-Cognito-Benutzerpools importieren oder migrieren können. Unter Verwendung eines Lambda-Auslösers für die Benutzermigration können Sie Benutzer migrieren, wenn sie sich zum ersten Mal mit Amazon Cognito anmelden. Mit diesem Ansatz können Benutzer ihre vorhandenen Passwörter weiterverwenden und müssen sie nach der Migration in Ihren Benutzerpool nicht zurücksetzen. Alternativ können Sie alle Benutzer auf einmal migrieren, indem Sie eine CSV-Datei mit den Benutzerprofilattributen für alle Benutzer hochladen. In den folgenden Abschnitten werden beide Ansätze beschrieben.

Themen

- Importieren von Benutzern in Benutzerpools mit einem Lambda-Auslöser für die Benutzermigration
- Importieren von Benutzern aus einer CSV-Datei in Benutzerpools

Importieren von Benutzern in Benutzerpools mit einem Lambda-Auslöser für die Benutzermigration

Mit diesem Ansatz können Sie Benutzer nahtlos aus Ihrem vorhandenen Benutzerverzeichnis zu Benutzerpools migrieren, wenn sich ein Benutzer zum ersten Mal bei Ihrer App anmeldet

oder ein Zurücksetzen des Passworts anfordert. Fügen Sie Ihrem Benutzerpool eine Lambda-Auslöser für die Benutzermigration.-Funktion hinzu. Daraufhin erhält dieser Metadaten über Benutzer, die sich anzumelden versuchen, und gibt Benutzerprofilinformationen von einer externen Identitätsquelle zurück. Weitere Informationen und Beispiel-Code zu diesem Lambda-Auslöser, einschließlich der Anfrage- und Antwortparameter, finden Sie unter Lambda-Auslöserparameter für die Benutzermigration.

Bevor Sie mit der Migration von Benutzern beginnen, erstellen Sie eine Lambda-Funktion für die Benutzermigration in Ihrem AWS-Konto und legen Sie die Lambda-Funktion als Auslöser der Benutzermigration in Ihrem Benutzerpool fest. Fügen Sie Ihrer Lambda-Funktion eine Autorisierungsrichtlinie hinzu, die nur dem Prinzipal des Amazon-Cognito-Servicekontos `cognito-idp.amazonaws.com` erlaubt, die Lambda-Funktion aufzurufen, und zwar nur im Kontext Ihres eigenen Benutzerpools. Weitere Informationen finden Sie unter Verwenden von ressourcenbasierten Richtlinien für AWS Lambda (Lambda-Funktionsrichtlinien).

Anmeldeprozess

1. Der Benutzer öffnet Ihre App und meldet sich mit der Benutzerpool-API von Amazon Cognito oder über die von Amazon Cognito gehostete Benutzeroberfläche an. Weitere Informationen zur Erleichterung der Anmeldung mit Amazon-Cognito-APIs finden Sie unter Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps.
2. Ihre App sendet den Benutzernamen und das Passwort an Amazon Cognito. Wenn Ihre App über eine benutzerdefinierte Anmelde-Benutzeroberfläche verfügt, die Sie mit einem AWS-SDK entwickelt haben, muss Ihre App InitiateAuth oder AdminInitiateAuth mit dem `USER_PASSWORD_AUTH`- oder dem `ADMIN_USER_PASSWORD_AUTH`-Ablauf verwenden. Wenn Ihre App einen dieser Abläufe verwendet, sendet das SDK das Passwort an den Server.

> **Note**
>
> Bevor Sie einen Auslöser für die Benutzermigration hinzufügen, aktivieren Sie den `USER_PASSWORD_AUTH`- oder den `ADMIN_USER_PASSWORD_AUTH`-Ablauf in den Einstellungen Ihres App-Clients. Sie müssen diese Abläufe anstelle des `USER_SRP_AUTH`-Standardablaufs verwenden. Amazon Cognito muss ein Passwort an Ihre Lambda-Funktion senden, damit es die Authentifizierung Ihres Benutzers im anderen Verzeichnis überprüfen kann. Ein SRP verdeckt das Passwort Ihres Benutzers vor Ihrer Lambda-Funktion.

3. Amazon Cognito prüft, ob der übermittelte Benutzername mit einem Benutzernamen oder Alias im Benutzerpool übereinstimmt. Sie können die E-Mail-Adresse, Telefonnummer oder den bevorzugten Benutzernamen des Benutzers als Alias in Ihrem Benutzerpool festlegen. Wenn der Benutzer nicht existiert, sendet Amazon Cognito Parameter, einschließlich Benutzernamen und Passwort, an Ihre Lambda-Auslöser für die Benutzermigration.-Funktion.
4. Ihre Lambda-Auslöser für die Benutzermigration.-Funktion prüft oder authentifiziert den Benutzer bei Ihrem vorhandenen Benutzerverzeichnis oder Ihrer Benutzerdatenbank. Die Funktion gibt Benutzerattribute zurück, die Amazon Cognito im Benutzerprofil innerhalb des Benutzerpools speichert. Sie können einen `username`-Parameter nur zurückgeben, wenn der übermittelte Benutzername mit einem Aliasattribut übereinstimmt. Wenn Sie möchten, dass Benutzer ihre vorhandenen Passwörter weiterhin verwenden, legen Sie das Attribut `finalUserStatus` in der Lambda-Antwort auf `CONFIRMED` fest. Ihre App muss alle `"response"`-Parameter zurückgeben, die unter Lambda-Auslöserparameter für die Benutzermigration angezeigt werden.

   > **Important**
   >
   > Protokollieren Sie nicht das gesamte Anforderungsereignisobjekt in Ihrem Lambda-Code der Benutzermigration. Dieses Anforderungsereignisobjekt enthält das Passwort des Benutzers. Wenn Sie die Protokolle nicht bereinigen, werden Passwörter in CloudWatch-Protokollen angezeigt.

5. Amazon Cognito erstellt das Benutzerprofil in Ihrem Benutzerpool und gibt Token an Ihren App-Client zurück.
6. Ihre App führt die Token-Aufnahme durch, akzeptiert die Benutzerauthentifizierung und fährt mit dem angeforderten Inhalt fort.

Nachdem Sie Ihre Benutzer migriert haben, verwenden Sie `USER_SRP_AUTH` für die Anmeldung. Das Secure Remote Password (SRP)-Protokoll sendet das Passwort nicht über das Netzwerk und bietet Sicherheitsvorteile gegenüber dem `USER_PASSWORD_AUTH`-Ablauf, den Sie bei der Migration verwendet haben.

Wenn während der Migration Fehler auftreten, wie beispielsweise Probleme mit dem Client-Gerät oder dem Netzwerk, erhält Ihre App Fehlermeldungen von der Benutzerpool-API von Amazon Cognito. In diesem Fall kann Amazon Cognito das Benutzerkonto in Ihrem Benutzerpool möglicherweise nicht erstellen. Der Benutzer sollte dann versuchen, sich erneut anzumelden. Wenn die Anmeldung wiederholt fehlschlägt, versuchen Sie, das Passwort des Benutzers mit dem Ablauf für ein vergessenes Passwort in Ihrer App zurückzusetzen.

Der Ablauf für ein vergessenes Passwort ruft außerdem Ihre Lambda-Auslöser für die Benutzermigration.-Funktion mit einer `UserMigration_ForgotPassword`-Ereignisquelle auf. Da der Benutzer beim Anfordern der Passwortrücksetzung kein Passwort übermittelt, nimmt Amazon Cognito kein Passwort auf für den Fall, dass es an Ihre Lambda-Funktion gesendet wird. Ihre Funktion kann den Benutzer nur in Ihrem vorhandenen Benutzerverzeichnis nachschlagen und Attribute zurückgeben, die dem Benutzerprofil in Ihrem Benutzerpool hinzugefügt werden. Nachdem Ihre Funktion den Aufruf abgeschlossen und ihre Antwort an Amazon Cognito zurückgegeben hat, sendet der Benutzerpool einen Code für die Passwortrücksetzung per E-Mail oder SMS. Fordern Sie den Benutzer in Ihrer App auf, den Bestätigungscode und ein neues Passwort einzugeben, und senden Sie diese Informationen dann in einer ConfirmForgotPassword-API-Anforderung an Amazon Cognito. Sie können auch die integrierten Seiten für den Ablauf für ein vergessenes Passwort in der gehosteten Benutzeroberfläche in Amazon Cognito verwenden.

Importieren von Benutzern aus einer CSV-Datei in Benutzerpools

Sie können Benutzer in einen Amazon-Cognito-Benutzerpool importieren. Die Benutzerinformationen werden aus einer speziell formatierten CSV-Datei importiert. Der Importprozess legt Werte für alle Benutzerattribute fest, mit Ausnahme von password. Der Import des Passworts wird nicht unterstützt, da bewährte Methoden für die Sicherheit verlangen, dass Kennwörter nicht als Klartext verfügbar sind, und wir den Import von Hash-Werten nicht unterstützen. Das bedeutet, dass Ihre Benutzer ihre Passwörter ändern müssen, wenn er sich das erste Mal anmelden. Ihre Benutzer befinden sich also im Zustand RESET_REQUIRED, wenn sie mit dieser Methode importiert werden.

Sie können die Passwörter Ihrer Benutzer mit der API-Anfrage AdminSetUserPassword festlegen, die den Parameter `Permanent` auf `true` setzt.

> **Note**
>
> Das Erstellungsdatum für jeden Benutzer ist eine Zeit, zu der der Benutzer in den Benutzerpool importiert wurde. Erstellungsdatum ist nicht eines der importierten Attribute.

Die grundlegenden Schritte sind:

1. Erstellen Sie eine Rolle für Amazon CloudWatch Logs in der AWS Identity and Access Management-(IAM)-Konsole.
2. Erstellen Sie die Benutzerimport-CSV-Datei.
3. Erstellen Sie den Benutzerimportauftrag und führen Sie ihn aus.

4. Laden Sie die Benutzerimport-CSV-Datei hoch.
5. Starten Sie den Benutzerimportauftrag und führen Sie ihn aus.
6. Verwenden Sie CloudWatch, zum Prüfen des Ereignisprotokolls.
7. Verlangen Sie von den importierten Benutzer, dass sie ihre Passwörter zurücksetzen.

Themen

Erstellen der IAM-Rolle für CloudWatch Logs

Wenn Sie die Amazon-Cognito-CLI oder -API verwenden, müssen Sie eine CloudWatch-IAM-Rolle erstellen. Im Folgenden wird beschrieben, wie Sie eine IAM-Rolle erstellen, mit der Amazon Cognito die Ergebnisse Ihres Importauftrags in CloudWatch Logs schreiben kann.

> **Note**
>
> Wenn Sie einen Importauftrag in der Amazon-Cognito-Konsole erstellen, können Sie gleichzeitig die IAM-Rolle erstellen. Wenn Sie Create a new IAM role (Neue IAM-Rolle erstellen) auswählen, wendet Amazon Cognito automatisch die entsprechende Vertrauensrichtlinie und IAM-Richtlinie auf die Rolle an.

So erstellen Sie die IAM-Rolle für CloudWatch Logs, mit der Sie den Benutzerpool importieren (AWS CLI, API)

1. Melden Sie sich bei der AWS Management Console an, und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Erstellen Sie eine neue IAM-Rolle für einen AWS-Service. Detaillierte Anweisungen finden Sie unter Erstellen einer Rolle für einen AWS-Service im Benutzerhandbuch zu AWS Identity and Access Management.

a. Bei der Auswahl von Use case (Anwendungsfall) für Trusted entity type (Typ der vertrauenswürdigen Entität) können Sie einen beliebigen Service wählen. Amazon Cognito ist derzeit nicht in der Liste der Anwendungsfälle für Services aufgeführt.

b. Wählen Sie im Bildschirm Add permissions (Berechtigungen hinzufügen) die Option Create policy (Richtlinie erstellen) aus und fügen Sie die folgende Richtlinienanweisung ein. Ersetzen Sie *REGION* durch die AWS-Region Ihres Benutzerpools, z. B. `us-east-1`. Ersetzen Sie *ACCOUNT* durch Ihre AWS-Konto-ID, z. B. `111122223333`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:REGION:ACCOUNT:log-group:/aws/cognito/*"
            ]
        }
    ]
}
```

3. Da Sie Amazon Cognito während der Erstellung der Rolle nicht als vertrauenswürdige Entität ausgewählt haben, müssen Sie die Vertrauensstellung der Rolle jetzt manuell bearbeiten. Klicken Sie im Navigationsbereich der IAM-Konsole auf Roles (Rollen) und wählen Sie dann die neu erstellte Rolle aus.
4. Wählen Sie die Registerkarte Trust relationships (Vertrauensstellungen).
5. Wählen Sie Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
6. Fügen Sie die folgende Richtlinienanweisung in das Feld Edit trust policy (Vertrauensrichtlinie bearbeiten) ein und ersetzen Sie dabei den vorhandenen Text:

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

```
                "Effect": "Allow",
                "Principal": {
                    "Service": "cognito-idp.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }
```

7. Wählen Sie Update policy.
8. Notieren Sie den Rollen-ARN. Sie geben den ARN ein, wenn Sie den Importauftrag erstellen.

Erstellen der CSV-Datei für den Benutzerimport

Bevor Sie vorhandene Benutzer in Ihren Benutzerpool importieren können, müssen Sie eine Datei mit durch Kommas getrennten Werten (CSV-Datei) erstellen, die die zu importierenden Benutzer und ihre Attribute enthält. Aus Ihrem Benutzerpool können Sie eine Benutzerimportdatei mit Headern abrufen, die das Attributschema Ihres Benutzerpools widerspiegeln. Anschließend können Sie Benutzerinformationen einfügen, die den Formatierungsanforderungen im Abschnitt Formatieren der CSV-Datei entsprechen.

Herunterladen des CSV-Datei-Headers (Konsole)

Gehen Sie wie folgt vor, um die CSV-Header-Datei herunterzuladen.

So laden Sie den CSV-Datei-Header herunter

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Sie werden möglicherweise aufgefordert, Ihre AWS-Anmeldeinformationen einzugeben.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Users.
5. Wählen Sie im Abschnitt Import users (Benutzer importieren) die Option Create an import job (Importauftrag erstellen) aus.
6. Wählen Sie unter Upload CSV (CSV hochladen) den Link template.csv aus und laden Sie die CSV-Datei herunter.

Herunterladen des CSV-Datei-Headers (AWS CLI)

Um eine Liste der richtigen Header zu erhalten, führen Sie den folgenden CLI-Befehl aus, wobei *USER_POOL_ID* die Befehlszeilen-ID für den Benutzerpool ist, in den Sie Benutzer importieren:

```
aws cognito-idp get-csv-header --user-pool-id "USER_POOL_ID"
```

Beispielantwort:

```
{
    "CSVHeader": [
        "name",
        "given_name",
        "family_name",
        "middle_name",
        "nickname",
        "preferred_username",
        "profile",
        "picture",
        "website",
        "email",
        "email_verified",
        "gender",
        "birthdate",
        "zoneinfo",
        "locale",
        "phone_number",
        "phone_number_verified",
        "address",
        "updated_at",
        "cognito:mfa_enabled",
        "cognito:username"
    ],
    "UserPoolId": "USER_POOL_ID"
}
```

Formatieren der CSV-Datei

Die heruntergeladene CSV-Header-Datei für den Benutzerimport sieht wie die folgende Zeichenfolge aus. Sie enthält auch benutzerdefinierte Attribute, die Sie Ihrem Benutzerpool hinzugefügt haben.

```
cognito:username,name,given_name,family_name,middle_name,nickname,preferred_username,profile,pi
```

Bearbeiten Sie Ihre CSV-Datei, sodass sie diesen Header und die Attributwerte für Ihre Benutzer enthält und gemäß den folgenden Regeln formatiert ist:

> **Note**
>
> Weitere Informationen zu Attributwerten, wie etwas das richtige Format für Telefonnummern, finden Sie unter Attribute für den Benutzerpool.

- Die erste Zeile in der Datei ist die heruntergeladene Kopfzeile mit den Benutzer-Attributnamen.
- Die Reihenfolge der Spalten in der CSV-Datei ist unerheblich.
- Jede Zeile nach der ersten Zeile enthält die Attributwerte für einen Benutzer.
- Alle Spalten in der Kopfzeile müssen vorhanden sein, Sie müssen jedoch nicht in jeder Spalte Werte angeben.
- Die folgenden Attribute sind erforderlich:
  - cognito:username
  - cognito:mfa_enabled
  - email_verified oder phone_number_verified
    - Mindestens eines der automatisch überprüften Attribute muss für jeden Benutzer `true` sein. Ein automatisch verifiziertes Attribut ist eine E-Mail-Adresse oder Telefonnummer, an die Amazon Cognito automatisch einen Code sendet, wenn ein neuer Benutzer Ihrem Benutzerpool beitritt.
    - Der Benutzerpool muss mindestens ein automatisch überprüftes Attribut besitzen, entweder email_verified oder phone_number_verified. Wenn der Benutzerpool keine automatisch überprüften Attribute enthält, wird der Importauftrag nicht gestartet.
    - Wenn der Benutzerpool nur über ein automatisch überprüftes Attribut verfügt, muss dieses Attribut für jeden Benutzer überprüft werden. Wenn der Benutzerpool beispielsweise nur phone_number als automatisch überprüftes Attribut aufweist, muss der Wert phone_number_verified `true` für jeden Benutzer überprüft werden.

> **Note**
>
> Damit Benutzer ihre Passwörter zurücksetzen können, benötigen Sie eine bestätigte E-Mail-Adresse oder Telefonnummer. Amazon Cognito sendet eine Nachricht mit einem Code zum Zurücksetzen des Passworts an die in der CSV-Datei angegebene E-Mail-

Adresse oder Telefonnummer. Die Nachricht wird als SMS an die Telefonnummer gesendet. Weitere Informationen finden Sie unter Überprüfen von Kontaktinformationen bei der Anmeldung.

  - email (wenn email_verified `true` ist)
  - phone_number (wenn phone_number_verified `true` ist)
  - Alle Attribute, die Sie beim Erstellen des Benutzerpools als erforderlich kennzeichnen
- Attributwerte, die Zeichenfolgen sind, sollten nicht in Anführungszeichen gesetzt werden.
- Wenn ein Attributwert ein Komma enthält, müssen Sie einen Backslash (\) vor dem Komma eingeben. Der Grund hierfür ist, dass die Felder in einer CSV-Datei durch Kommas getrennt sind.
- Die Inhalte der CSV-Datei sollten im UTF-8-Format ohne Markierung der Bytereihenfolge vorliegen.
- Das Feld cognito:username ist ein Pflichtfeld und muss in Ihrem Benutzerpool einmalig sein. Es kann eine beliebige Unicode-Zeichenfolge sein. Es kann jedoch keine Leerzeichen oder Tab-Zeichen enthalten.
- Die Werte für birthdate, falls vorhanden, müssen im Format *`mm/dd/yyyy`* vorliegen. Das bedeutet, dass z. B. das Geburtsdatum 1. Februar 1985 als **`02/01/1985`** kodiert werden muss.
- Das Feld cognito:mfa_enabled ist ein Pflichtfeld. Wenn Sie festgelegt haben, dass die Multi-Factor Authentication (MFA) in Ihrem Benutzerpool erforderlich ist, muss das Feld für alle Benutzer `true` sein. Wenn Sie MFA als deaktiviert festgelegt haben, muss dieses Feld für alle Benutzer `false` sein. Wenn Sie MFA als optional festgelegt haben, kann dieses Feld entweder `true` oder `false` sein, aber es darf nicht leer sein.
- Die maximale Zeilenlänge beträgt 16 000 Zeichen.
- Die maximale CSV-Dateigröße ist 100 MB.
- Die maximale Anzahl von Zeilen (Benutzer) in der Datei ist 500 000. Dieser Höchstwert enthält die Kopfzeile nicht.
- Der Feldwert updated_at ist voraussichtlich die Epochenzeit in Sekunden, z. B.: **`1471453471`**.
- Alle führenden bzw. nachgestellten Leerzeichen in einem Attributwert werden getrimmt.

Die folgende Liste ist ein Beispiel für eine CSV-Importdatei für einen Benutzerpool ohne benutzerdefinierte Attribute. Ihr Benutzerpool-Schema kann sich von diesem Beispiel unterscheiden. In diesem Fall müssen Sie Testwerte in der CSV-Vorlage angeben, die Sie aus Ihrem Benutzerpool herunterladen.

```
cognito:username,name,given_name,family_name,middle_name,nickname,preferred_username,profile,pi
```

```
John,,John,Doe,,,,,,,johndoe@example.com,TRUE,,02/01/1985,,,+12345550100,TRUE,123 Any
 Street,,FALSE
Jane,,Jane,Roe,,,,,,,janeroe@example.com,TRUE,,01/01/1985,,,+12345550199,TRUE,100 Main
 Street,,FALSE
```

Erstellen und Ausführen des Amazon-Cognito-Benutzerpool-Importauftrags

In diesem Abschnitt wird beschrieben, wie Sie den Benutzerpool-Importauftrag über die Amazon-Cognito-Konsole und die AWS Command Line Interface (AWS CLI) erstellen und ausführen.

Themen

- Importieren von Benutzern aus einer CSV-Datei (Konsole)
- Importieren von Benutzern (AWS CLI)

Importieren von Benutzern aus einer CSV-Datei (Konsole)

Im folgenden Verfahren wird beschrieben, wie Sie Benutzer aus der CSV-Datei importieren.

So importieren Sie Benutzer aus der CSV-Datei (Konsole)

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Sie werden möglicherweise aufgefordert, Ihre AWS-Anmeldeinformationen einzugeben.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Users.
5. Wählen Sie im Abschnitt Import users (Benutzer importieren) die Option Create an import job (Importauftrag erstellen) aus.
6. Geben Sie auf der Seite Create import job (Importauftrag erstellen) unter Job name einen Auftragsnamen ein.
7. Wählen Sie dann Create a new IAM role (Neue IAM-Rolle erstellen) oder Use an existing IAM role (Vorhandene IAM-Rolle verwenden) aus.

    a. Wenn Sie Create a new IAM role (Neue IAM-Rolle erstellen) ausgewählt haben, geben Sie einen Namen für Ihre neue Rolle ein. Amazon Cognito erstellt automatisch eine Rolle mit den richtigen Berechtigungen und der richtigen Vertrauensstellung. Der IAM-Prinzipal, der den Importauftrag erstellt, muss über die Berechtigungen zum Erstellen von IAM-Rollen verfügen.

   b. Wenn Sie Use an existing IAM role (Vorhandene IAM-Rolle verwenden) ausgewählt haben, wählen Sie eine Rolle aus der Liste unter IAM role selection (Auswahl der IAM-Rolle) aus. Diese Rolle muss über die in Erstellen der IAM-Rolle für CloudWatch Logs beschriebenen Berechtigungen und Vertrauensrichtlinien verfügen.

8. Wählen Sie Create job (Auftrag erstellen) aus, um Ihren Auftrag zu übermitteln, jedoch erst später zu starten. Wählen Sie Create and start job (Auftrag erstellen und starten) aus, um Ihren Auftrag zu übermitteln und sofort zu starten.
9. Wenn Sie Ihren Auftrag erstellt, aber noch nicht gestartet haben, können Sie ihn später starten. Wählen Sie auf der Registerkarte Users (Benutzer) unter Import users (Benutzer importieren) Ihren Importauftrag aus und wählen Sie dann Start (Starten) aus. Sie können auch eine API-Anforderung StartUserImportJob von einem AWS-SDK aus übermitteln.
10. Überwachen Sie den Fortschritt Ihres Benutzerimportauftrags auf der Registerkarte Users (Benutzer) unter Import users (Benutzer importieren). Wenn Ihr Auftrag nicht erfolgreich ist, können Sie den Status-Wert auswählen. Um weitere Informationen zu erhalten, wählen Sie View the CloudWatch logs for more details (CloudWatch-Protokolle anzeigen, um weitere Informationen zu erhalten) aus und überprüfen Sie alle Probleme in der CloudWatch-Logs-Konsole.

Importieren von Benutzern (AWS CLI)

Die folgenden CLI-Befehle sind für den Import von Benutzern in einen Benutzerpool verfügbar:

- `create-user-import-job`
- `get-csv-header`
- `describe-user-import-job`
- `list-user-import-jobs`
- `start-user-import-job`
- `stop-user-import-job`

Um die Liste der Befehlszeilen-Optionen für diese Befehle zu erhalten, verwenden Sie die Befehlszeilen-Option `help`. Beispiel:

```
aws cognito-idp get-csv-header help
```

Erstellen eines Benutzer-Importauftrags

Nachdem Sie Ihre CSV-Datei erstellt haben, erstellen Sie einen Benutzer-Importauftrag, indem Sie den folgenden CLI-Befehl ausführen, wobei *JOB_NAME* der von Ihnen gewählte Name für den Auftrag, *USER_POOL_ID* die Benutzerpool-ID für den Benutzerpool, dem die neuen Benutzer hinzugefügt werden, und *ROLE_ARN* der Rollen-ARN ist, den Sie in Erstellen der IAM-Rolle für CloudWatch Logs erhalten haben:

```
aws cognito-idp create-user-import-job --job-name "JOB_NAME" --user-pool-id
 "USER_POOL_ID" --cloud-watch-logs-role-arn "ROLE_ARN"
```

Die in der Antwort zurückgegebene *PRE_SIGNED_URL* ist 15 Minuten lang gültig. Nach dieser Zeit läuft sie ab, und Sie müssen einen neuen Benutzer-Importauftrag erstellen, um eine neue URL zu erhalten.

Example Beispielantwort:

```
{
    "UserImportJob": {
        "Status": "Created",
        "SkippedUsers": 0,
        "UserPoolId": "USER_POOL_ID",
        "ImportedUsers": 0,
        "JobName": "JOB_NAME",
        "JobId": "JOB_ID",
        "PreSignedUrl": "PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn": "ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957431.965
    }
}
```

Statuswerte für einen Benutzer-Importauftrag

In den Antworten auf Ihre Benutzer-Importbefehle sehen Sie einen der folgenden `Status`-Werte:

- `Created` – Der Auftrag wurde erstellt aber noch nicht gestartet.
- `Pending` – Ein Übergangsstatus. Sie haben den Auftrag gestartet, aber es wurden bislang noch keine Benutzer importiert.
- `InProgress` – Der Auftrag wurde gestartet, und Benutzer werden importiert.

- `Stopping` – Sie haben den Auftrag angehalten, aber der Import von Benutzern wurde noch nicht angehalten.
- `Stopped` – Sie haben den Auftrag angehalten, und es werden keine Benutzer mehr importiert.
- `Succeeded` – Der Schritt wurde erfolgreich ausgeführt.
- `Failed` – Der Auftrag wurde aufgrund eines Fehlers angehalten.
- `Expired` – Sie haben einen Auftrag erstellt, ihn aber nicht innerhalb von 24-48 Stunden gestartet. Alle mit dem Auftrag verbundenen Daten wurden gelöscht, und der Auftrag kann nicht gestartet werden.

Hochladen der CSV-Datei

Verwenden Sie den folgenden `curl`-Befehl zum Hochladen der CSV-Datei mit Ihren Benutzerdaten auf die vorsignierte URL, die Sie aus der Antwort des `create-user-import-job`-Befehls erhalten haben.

```
curl -v -T "PATH_TO_CSV_FILE" -H "x-amz-server-side-encryption:aws:kms"
 "PRE_SIGNED_URL"
```

In der Ausgabe dieses Befehls suchen Sie die Zeichenfolge `"We are completely uploaded and fine"`. Diese Phrase gibt an, dass die Datei erfolgreich hochgeladen wurde.

Beschreiben eines Benutzer-Importauftrags

Um eine Beschreibung Ihres Benutzer-Importauftrag zu erhalten, verwenden Sie den folgenden Befehl, wobei *USER_POOL_ID* Ihre Benutzerpool-ID und *JOB_ID* die Auftrags-ID ist, die beim Erstellen des Benutzer-Importauftrags zurückgegeben wurde.

```
aws cognito-idp describe-user-import-job --user-pool-id "USER_POOL_ID" --job-id
 "JOB_ID"
```

Example Beispielantwort:

```
{
    "UserImportJob": {
        "Status": "Created",
        "SkippedUsers": 0,
        "UserPoolId": "USER_POOL_ID",
```

```
        "ImportedUsers": 0,
        "JobName": "JOB_NAME",
        "JobId": "JOB_ID",
        "PreSignedUrl": "PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn":"ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957431.965
    }
}
```

In der vorherigen Beispielausgabe ist die *PRE_SIGNED_URL* die URL, zu der Sie die CSV-Datei hochgeladen haben. Der *ROLE_ARN* ist der ARN der CloudWatch-Logs-Rolle, den Sie beim Erstellen der Rolle erhalten haben.

Auflisten Ihrer Benutzer-Importaufträge

Zum Auflisten Ihrer Benutzer-Importaufträge verwenden Sie den folgenden Befehl:

```
aws cognito-idp list-user-import-jobs --user-pool-id "USER_POOL_ID" --max-results 2
```

Example Beispielantwort:

```
{
    "UserImportJobs": [
        {
            "Status": "Created",
            "SkippedUsers": 0,
            "UserPoolId": "USER_POOL_ID",
            "ImportedUsers": 0,
            "JobName": "JOB_NAME",
            "JobId": "JOB_ID",
            "PreSignedUrl":"PRE_SIGNED_URL",
            "CloudWatchLogsRoleArn":"ROLE_ARN",
            "FailedUsers": 0,
            "CreationDate": 1470957431.965
        },
        {
            "CompletionDate": 1470954227.701,
            "StartDate": 1470954226.086,
            "Status": "Failed",
            "UserPoolId": "USER_POOL_ID",
            "ImportedUsers": 0,
```

```
            "SkippedUsers": 0,
            "JobName": "JOB_NAME",
            "CompletionMessage": "Too many users have failed or been skipped during the
 import.",
            "JobId": "JOB_ID",
            "PreSignedUrl":"PRE_SIGNED_URL",
            "CloudWatchLogsRoleArn":"ROLE_ARN",
            "FailedUsers": 5,
            "CreationDate": 1470953929.313
        }
    ],
    "PaginationToken": "PAGINATION_TOKEN"
}
```

Aufträge werden in chronologischer Reihenfolge vom zuletzt erstellten bis zum zuerst erstellten aufgelistet. Die Zeichenfolge *PAGINATION_TOKEN* nach dem zweite Auftrag weist darauf hin, dass weitere Ergebnisse für diesen Listenbefehl vorliegen. Um die zusätzlichen Ergebnisse aufzulisten, verwenden Sie die Option `--pagination-token` wie folgt:

```
aws cognito-idp list-user-import-jobs --user-pool-id "USER_POOL_ID" --max-results 10 --
pagination-token "PAGINATION_TOKEN"
```

Starten eines Benutzer-Importauftrags

Zum Starten eines Benutzer-Importauftrags verwenden Sie den folgenden Befehl:

```
aws cognito-idp start-user-import-job --user-pool-id "USER_POOL_ID" --job-id "JOB_ID"
```

Nur ein Importauftrag kann jeweils aktiv sein.

Example Beispielantwort:

```
{
    "UserImportJob": {
        "Status": "Pending",
        "StartDate": 1470957851.483,
        "UserPoolId": "USER_POOL_ID",
        "ImportedUsers": 0,
        "SkippedUsers": 0,
        "JobName": "JOB_NAME",
        "JobId": "JOB_ID",
```

```
        "PreSignedUrl":"PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn": "ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957431.965
    }
}
```

Anhalten eines Benutzer-Importauftrags

Zum Anhalten eines Benutzer-Importauftrags, während dieser ausgeführt wird, verwenden Sie den folgenden Befehl. Nachdem Sie den Auftrag angehalten haben, kann er nicht neu gestartet werden.

```
aws cognito-idp stop-user-import-job --user-pool-id "USER_POOL_ID" --job-id "JOB_ID"
```

Example Beispielantwort:

```
{
    "UserImportJob": {
        "CompletionDate": 1470958050.571,
        "StartDate": 1470958047.797,
        "Status": "Stopped",
        "UserPoolId": "USER_POOL_ID",
        "ImportedUsers": 0,
        "SkippedUsers": 0,
        "JobName": "JOB_NAME",
        "CompletionMessage": "The Import Job was stopped by the developer.",
        "JobId": "JOB_ID",
        "PreSignedUrl":"PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn": "ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957972.387
    }
}
```

Anzeigen der Benutzerpool-Importergebnisse in der CloudWatch-Konsole

Sie können die Ergebnisse Ihres Importauftrags in der Amazon-CloudWatch-Konsole anzeigen.

Themen

- Anzeigen der Ergebnisse
- Interpretieren der Ergebnisse

Anzeigen der Ergebnisse

In den folgenden Schritten wird beschrieben, wie Sie die Benutzerpool-Importergebnisse anzeigen.

So zeigen Sie die Ergebnisse des Benutzerpoolimports an

1. Melden Sie sich bei AWS Management Console an und öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie Logs.
3. Wählen Sie die Protokollgruppe für Ihre Benutzerpool-Importaufträge.
   Der Name der Protokollgruppe liegt in der Form vor `/aws/cognito/userpools/`*`USER_POOL_ID`*`/`*`USER_POOL_NAME`*.
4. Wählen Sie das Protokoll für den Benutzer-Importauftrag, den Sie gerade ausgeführt haben. Der Protokollname liegt in der Form *`JOB_ID`*/*`JOB_NAME`* vor. Die Ergebnisse im Protokoll beziehen sich auf Ihre Benutzer nach Zeilennummer. Es werden keine Benutzerdaten in das Protokoll geschrieben. Für jeden Benutzer wird eine Zeile ähnlich der folgenden angezeigt:

   - `[SUCCEEDED] Line Number 5956 - The import succeeded.`
   - `[SKIPPED] Line Number 5956 - The user already exists.`
   - `[FAILED] Line Number 5956 - The User Record does not set any of the auto verified attributes to true. (Example: email_verified to true).`

Interpretieren der Ergebnisse

Für erfolgreich importierte Benutzer wird der Status auf "PasswordReset" gesetzt.

In den folgenden Fällen wird der Benutzer nicht importiert, der Importauftrag wird jedoch fortgesetzt:

- Keine automatisch überprüften Attribute werden auf gesetzt `true`.
- Die Benutzerdaten entsprechen nicht dem Schema.
- Der Benutzer konnte aufgrund eines internen Fehlers nicht importiert werden.

In den folgenden Fällen schlägt der Importauftrag fehl:

- Die Amazon-CloudWatch-Logs-Rolle kann nicht angenommen werden, sie hat nicht die richtige Zugriffsrichtlinie oder wurde gelöscht.
- Der Benutzerpool wurde gelöscht.

- Amazon Cognito kann die CSV-Datei nicht analysieren.

Von importierten Benutzer verlangen, dass sie ihre Passwörter zurücksetzen

Jeder importierte Benutzer, der sich zum ersten Mal anmeldet und ein Passwort eingibt, wird aufgefordert, ein neues Passwort einzugeben. Das folgende Verfahren beschreibt die Benutzerumgebung in einer benutzerdefinierten App mit lokalen Benutzern, nachdem Sie eine CSV-Datei importiert haben. Wenn sich Ihre Benutzer bei der gehosteten Benutzeroberfläche anmelden, fordert sie Amazon Cognito bei der ersten Anmeldung auf, ein neues Passwort festzulegen.

Von importierten Benutzer verlangen, dass sie ihre Passwörter zurücksetzen

1. Versuchen Sie sich in Ihrer App unbemerkt als der aktuelle Benutzer anzumelden, wobei `InitiateAuth` ein zufallsgeneriertes Passwort angibt.
2. Amazon Cognito gibt eine `NotAuthorizedException` zurück, wenn `PreventUserExistenceErrors` aktiviert ist. Gibt andernfalls `PasswordResetRequiredException` zurück.
3. Ihre App stellt eine API-Anforderung `ForgotPassword` und setzt das Passwort des Benutzers zurück.

    a. Die App übermittelt den Benutzernamen in einer `ForgotPassword`-API-Anforderung.

    b. Amazon Cognito sendet einen Code an die verifizierte E-Mail-Adresse oder Telefonnummer. Das Ziel hängt von den Werten ab, die Sie für `email_verified` und `phone_number_verified` in Ihrer CSV-Datei angegeben haben. In der Antwort auf die Anforderung `ForgotPassword` ist das Ziel des Codes angegeben.

    > **Note**
    >
    > Ihr Benutzerpool muss für die Verifizierung von E-Mail-Adressen oder Telefonnummern konfiguriert sein. Weitere Informationen finden Sie unter Registrieren und Bestätigen von Benutzerkonten.

    c. Ihre App zeigt dem Benutzer eine Nachricht an, damit dieser den Ort überprüft, an den der Code gesendet wurde, und fordert den Benutzer auf, den Code und ein neues Passwort einzugeben.

    d. Der Benutzer gibt den Code und das neue Passwort in der App ein.

e. Die App übermittelt den Code und das neue Passwort in einer API-Anforderung `ConfirmForgotPassword`.

f. Ihre App leitet Ihren Benutzer zur Anmeldung weiter.

Attribute für den Benutzerpool

Attribute sind Informationen, anhand derer Sie einzelne Benutzer ermitteln können, z. B. Name, E-Mail-Adresse und Telefonnummer. Ein neuer Benutzerpool hat einen Satz standardmäßiger Standardattribute. Sie können Ihrer Benutzerpool-Definition auch benutzerdefinierte Attribute in der hinzufügen AWS Management Console. In diesem Thema werden diese Attribute ausführlich beschrieben und Sie erhalten Tipps zum Einrichten des Benutzerpools.

Speichern Sie nicht alle Informationen von Benutzern in Attributen. Speichern Sie beispielsweise Benutzerdaten, die häufig geändert werden (wie Nutzungsstatistiken oder Spielstände) in einem separaten Datenspeicher, z. B. Amazon Cognito Sync oder Amazon DynamoDB.

> **Note**
>
> In einigen Dokumentationen und Standards werden Attribute als Member bezeichnet.

Themen

Standardattribute

Amazon Cognito weist allen Benutzern basierend auf der OpenID-Connect-Spezifikation eine Reihe von Standardattributen zu. Standard- und benutzerdefinierte Attributwerte können standardmäßig aus einer beliebigen Zeichenfolge von bis zu 2048 Zeichen bestehen. Einige Attributwerte unterliegen jedoch Formatierungseinschränkungen.

Die Standardattribute sind:

- address
- birthdate
- email
- family_name
- gender
- given_name
- locale
- middle_name
- name
- nickname
- phone_number
- picture
- preferred_username
- profile
- sub
- updated_at
- website
- zoneinfo

Mit Ausnahme von sub sind Standardattribute standardmäßig für alle Benutzer optional. Um ein Attribut als erforderlich festzulegen, aktivieren Sie bei der Erstellung des Benutzerpools das Kontrollkästchen Required (Erforderlich) neben dem Attribut. Amazon Cognito weist jedem sub-Benutzerattribut einen eindeutigen Benutzerkennungswert zu. Nur die Attribute email und phone_number können verifiziert werden.

> **Note**
>
> Wenn Sie ein Standardattribut als erforderlich markieren, kann ein Benutzer sich erst registrieren, wenn er einen Wert für das Attribut angibt. Um Benutzer zu erstellen und keine Werte für erforderliche Attribute anzugeben, können Administratoren die AdminCreateUserAPI verwenden. Nachdem Sie einen Benutzerpool erstellt haben, können Sie die Markierung eines Attributs als erforderlich und nicht erforderlich nicht mehr ändern.

Details zu Standardattributen und Formateinschränkungen

birthdate

Der Wert muss als gültiges Datum mit 10 Zeichen im Format JJJJ-MM-TT angegeben werden.

email

Benutzer und Administratoren können die Werte für E-Mail-Adressen überprüfen.

Ein Administrator mit den entsprechenden AWS-Konto Berechtigungen kann die E-Mail-Adresse des Benutzers ändern und sie auch als verifiziert markieren. Markieren Sie eine E-Mail-Adresse mit der AdminUpdateUserAttributesAPI oder dem Befehl admin-update-user-attributes AWS Command Line Interface (AWS CLI) als verifiziert. Mit diesem Befehl kann der Administrator das `email_verified`-Attribut in `true` ändern. Sie können einen Benutzer auch auf der Registerkarte Benutzer bearbeiten AWS Management Console , um eine E-Mail-Adresse als verifiziert zu markieren.

Der Wert muss eine gültige E-Mail-Adresszeichenfolge sein, die dem Standard-E-Mail-Format mit dem @-Symbol und der Domäne entspricht und bis zu 2048 Zeichen umfassen kann.

phone_number

Ein Benutzer muss eine Telefonnummer angeben, wenn die SMS-Multi-Faktor-Authentifizierung (MFA) aktiv ist. Weitere Informationen finden Sie unter Hinzufügen der MFA zu einem Benutzerpool.

Benutzer und Administratoren können Telefonnummernwerte überprüfen.

Ein Administrator mit den entsprechenden AWS-Konto Berechtigungen kann die Telefonnummer des Benutzers ändern und sie auch als verifiziert markieren. Markieren Sie eine Telefonnummer mit der AdminUpdateUserAttributesAPI oder dem admin-update-user-attributes AWS CLI Befehl als verifiziert. Mit diesem Befehl kann der Administrator das `phone_number_verified`-Attribut in `true` ändern. Sie können einen Benutzer auch auf der Registerkarte Benutzer bearbeiten AWS Management Console , um eine Telefonnummer als verifiziert zu markieren.

> **Important**
>
> Bei Telefonnummern müssen die folgenden Formatierungsregeln beachtet werden: Eine Telefonnummer muss mit einem Pluszeichen (**+**) beginnen, auf das direkt der Ländercode folgt. Eine Telefonnummer kann nur das Pluszeichen **+** und Ziffern enthalten. Entfernen Sie alle anderen Zeichen, z. B. Klammern, Leerzeichen und Bindestriche

(-) aus einer Telefonnummer, bevor Sie den Wert an den Service übermitteln. Eine Telefonnummer in den Vereinigten Staaten muss beispielsweise das folgende Format haben: **+14325551212**.

preferred_username

Sie können `preferred_username` als erforderlich oder als Alias auswählen, aber nicht beides. Wenn es sich um einen Alias `preferred_username` handelt, können Sie eine Anfrage an den UpdateUserAttributesAPI-Vorgang stellen und den Attributwert hinzufügen, nachdem Sie den Benutzer bestätigt haben.

sub

Indexieren und durchsuchen Sie Ihre Benutzer basierend auf dem `sub`-Attribut. Das `sub`-Attribut ist eine eindeutige Benutzerkennung in jedem Benutzerpool. Benutzer können Attribute wie `phone_number` und `email` ändern. Das `sub`-Attribut hat einen festen Wert. Weitere Informationen zur Suche nach Benutzern finden Sie unter Verwalten von und Suchen nach Benutzerkonten.

Erforderliche Attribute anzeigen

Gehen Sie wie folgt vor, um die erforderlichen Attribute für einen bestimmten Benutzerpool anzuzeigen.

> **Note**
>
> Sie können erforderliche Attribute nicht mehr ändern, nachdem Sie einen Benutzerpool erstellt haben.

Erforderliche Attribute anzeigen

1. Gehen Sie zu Amazon Cognito in der AWS Management Console. Wenn Sie von der Konsole dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Sign-up experience (Anmeldeerlebnis) aus.

5. Zeigen Sie die erforderlichen Attribute Ihres Benutzerpools im Abschnitt Required attributes (Erforderliche Attribute) an.

Benutzernamen und bevorzugter Benutzername

Der Wert `username` ist ein separates Attribut und nicht mit dem Attribut `name` identisch. Jeder Benutzer hat ein `username`-Attribut. Amazon Cognito generiert automatisch einen Benutzernamen für Verbundbenutzer. Sie müssen ein `username`-Attribut angeben, um einen lokalen Benutzer im Amazon-Cognito-Verzeichnis zu erstellen. Nachdem Sie einen Benutzer erstellt haben, können Sie den Wert des `username`-Attributs nicht mehr ändern.

Entwickler können das `preferred_username`-Attribut verwenden, um Benutzern Benutzernamen zuzuteilen, die diese ändern können. Weitere Informationen finden Sie unter Anpassen von Anmeldeattributen.

Wenn für Ihre Anwendung kein Benutzername erforderlich ist, müssen Sie Benutzer nicht zur Angabe eines Benutzernamens auffordern. Ihre App kann im Hintergrund einen eindeutigen Benutzernamen für Benutzer erstellen. Dies kann sich als nützlich erweisen, wenn Sie beispielsweise möchten, dass sich Benutzer mit einer E-Mail-Adresse und einem Passwort registrieren und anmelden. Weitere Informationen finden Sie unter Anpassen von Anmeldeattributen.

Der `username` muss innerhalb eines Benutzerpools eindeutig sein. Ein `username` kann wiederverwendet werden, aber erst, nachdem er gelöscht wurde und nicht mehr in Gebrauch ist. Informationen zu den Zeichenkettenbeschränkungen für die `username` Attribute finden Sie in der Eigenschaft username einer SignUpAPI-Anfrage.

Anpassen von Anmeldeattributen

Beim Erstellen eines Benutzerpools können Sie Benutzernamenattribute einrichten, wenn Sie möchten, dass sich Ihre Benutzer mit einer E-Mail-Adresse oder Telefonnummer als Benutzername registrieren können. Alternativ können Sie Aliasattribute einrichten, um Ihren Benutzern die Wahl zu lassen: Sie können mehrere Attribute angeben, wenn sie sich registrieren, und sich dann mit einem Benutzernamen, einem bevorzugten Benutzernamen, einer E-Mail-Adresse oder einer Telefonnummer anmelden.

> **Important**
>
> Nachdem Sie einen Benutzerpool erstellt haben, können Sie diese Einstellung nicht mehr ändern.

So wählen Sie zwischen Aliasattributen und Benutzernamenattributen

Ihre Anforderung	Aliasattribute	Benutzernamenattribute
Benutzer haben mehrere Anmeldeattribute	Ja[1]	Nein[2]
Benutzer müssen ihre E-Mail-Adresse oder Telefonnummer verifizieren, bevor sie sich damit anmelden können	Ja	Nein
Melden Sie Benutzer mit doppelten E-Mail-Adressen oder Telefonnummern an und vermeiden Sie `UsernameExistsException` Fehler[3]	Ja	Nein
Kann denselben Attributwert für E-Mail-Adresse oder Telefonnummer mehreren Benutzern zuweisen	Ja[4]	Nein

[1] Verfügbare Anmeldeattribute sind der Benutzername, die E-Mail-Adresse, die Telefonnummer und der bevorzugte Benutzername.

[2] Es ist eine Anmeldung mit der E-Mail-Adresse oder der Telefonnummer möglich.

[3] Ihr Benutzerpool generiert keine `UsernameExistsException`-Fehler, wenn sich Benutzer mit möglicherweise doppelten E-Mail-Adressen oder Telefonnummern, aber ohne Benutzernamen registrieren. Dieses Verhalten ist unabhängig von der Option Fehler bei vorhandenen Benutzernamen verhindern, die für Anmelde-, aber nicht für Registrierungsvorgänge gilt.

[3] Nur der letzte Benutzer, der das Attribut verifiziert hat, kann sich damit anmelden.

Option 1: Mehrere Anmeldeattribute (Aliasattribute)

Sie können Aliase aktivieren, wenn Sie Ihren Benutzern die Möglichkeit geben möchten, bei der Anmeldung ihren Benutzernamen oder andere Attributwerte einzugeben. Standardmäßig melden sich Benutzer mit ihrem Benutzernamen und ihrem Passwort an. Der Benutzername ist ein fester Wert, den Benutzer nicht ändern können. Wenn Sie ein Attribut als Alias markieren, können Benutzer anstelle des Benutzernamens dieses Attribut zur Anmeldung verwenden. Die Attribute E-Mail-Adresse, Telefonnummer und bevorzugter Benutzername können als Aliasnamen markiert werden. Wenn Sie beispielsweise die E-Mail-Adresse und die Telefonnummer als Aliasnamen für einen Benutzerpool auswählen, können sich Benutzer in diesem Benutzerpool mit ihrem Benutzernamen, ihrer E-Mail-Adresse oder ihrer Telefonnummer sowie ihrem Passwort anmelden.

Wählen Sie zum Auswählen von Aliasattributen User name (Benutzername) und mindestens eine zusätzliche Anmeldeoption aus, wenn Sie den Benutzerpool erstellen.

> **Note**
>
> Wenn Sie Ihren Benutzerpool ohne Berücksichtigung der Groß- und Kleinschreibung konfigurieren, kann ein Benutzer entweder Klein- oder Großbuchstaben verwenden, um sich mit seinem Alias zu registrieren oder anzumelden. Weitere Informationen finden Sie CreateUserPoolin der Amazon Cognito Cognito-API-Referenz für Benutzerpools.

Wenn Sie die E-Mail-Adresse als Alias auswählen, akzeptiert Amazon Cognito keinen Benutzernamen, der mit einem gültigen E-Mail-Adressformat übereinstimmt. Ebenso wenig akzeptiert Amazon Cognito einen Benutzernamen für diesen Benutzerpool, der einem gültigen Telefonnummernformat entspricht, wenn Sie eine Telefonnummer als Alias auswählen.

> **Note**
>
> Aliaswerte müssen in einem Benutzerpool eindeutig sein. Wenn Sie einen Alias für eine E-Mail-Adresse oder eine Telefonnummer konfigurieren, kann der von Ihnen bereitgestellte Wert nur in einem Konto den bestätigten Status aufweisen. Wenn Ihr Benutzer während der Registrierung eine E-Mail-Adresse oder Telefonnummer als Aliaswert angibt und ein anderer Benutzer diesen Aliaswert bereits verwendet hat, ist die Registrierung erfolgreich. Versucht ein Benutzer allerdings, das Konto mit dieser E-Mail-Adresse (oder

Telefonnummer) zu bestätigen, und gibt den gültigen Code ein, gibt Amazon Cognito einen `AliasExistsException`-Fehler zurück. Der Fehler weist den Benutzer darauf hin, dass bereits ein Konto mit dieser E-Mail-Adresse (oder Telefonnummer) vorhanden ist. An diesem Punkt kann der Benutzer die Erstellung des neuen Kontos abbrechen und stattdessen versuchen, das Passwort für das alte Konto zurückzusetzen. Wenn der Benutzer mit der Erstellung des neuen Kontos fortfährt, muss Ihre App die `ConfirmSignUp`-API mit der Option `forceAliasCreation` aufrufen. `ConfirmSignUp` mit `forceAliasCreation` verschiebt den Alias vom vorherigen Konto in das neu erstellte Konto und markiert das Attribut im vorherigen Konto als nicht verifiziert.

Telefonnummern und E-Mail-Adressen werden für einen Benutzer erst zu aktiven Aliasnamen, nachdem Ihr Benutzer die Telefonnummern und E-Mail-Adressen verifiziert hat. Wir empfehlen, die automatische Verifizierung von E-Mail-Adressen und Telefonnummern zu aktivieren, wenn Sie diese als Aliasnamen verwenden.

Wählen Sie Aliasattribute, um den Fehler `UsernameExistsException` bei den Attributen für E-Mail-Adressen und Telefonnummern zu vermeiden, wenn sich Ihre Benutzer anmelden.

Aktivieren Sie das `preferred_username`-Attribut, damit Ihr Benutzer den Benutzernamen ändern kann, mit dem er sich anmeldet, während sich der `username`-Attributwert nicht ändert. Wenn Sie diese Benutzererfahrung unterstützen möchten, senden Sie den neuen `username`-Wert als `preferred_username` und wählen `preferred_username` als Alias aus. Anschließend können sich Benutzer mit dem neuen Wert anmelden, den sie eingegeben haben. Wenn `preferred_username` als Alias ausgewählt ist, kann Ihr Benutzer den Wert nur dann bereitstellen, wenn er ein Konto bestätigt. Er kann den Wert nicht während der Registrierung bereitstellen.

Wenn sich der Benutzer mit einem Benutzernamen anmeldet, können Sie auswählen, ob er sich mit einem oder mehreren der folgenden Aliasnamen anmelden kann.

- Verifizierte E-Mail-Adresse
- Verifizierte Telefonnummer
- Bevorzugter Benutzername

Nachdem sich der Benutzer registriert hat, kann er diese Aliasnamen ändern.

> **Important**
>
> Wenn Ihr Benutzerpool die Anmeldung mit Aliasen unterstützt und Sie einen Benutzer autorisieren oder suchen möchten, identifizieren Sie Ihren Benutzer nicht anhand seiner Anmeldeattribute. Die Benutzerkennung `sub` mit festem Wert ist der einzige konsistente Bezeichner für die Identität des Benutzers.

Fügen Sie die folgenden Schritte ein, wenn Sie den Benutzerpool erstellen, damit sich Benutzer mit einem Alias anmelden können.

So konfigurieren Sie einen Benutzerpool für die Anmeldung mit einem bevorzugten Benutzernamen

1. Wechseln Sie zu Amazon Cognito in der AWS Management Console. Wenn Sie von der Konsole dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie in der rechten oberen Ecke der Seite Create a user pool (Benutzerpool erstellen) aus, um den Assistenten zur Erstellung von Benutzerpools zu starten.
4. Wählen Sie unter Configure sign-in experience (Anmeldeerlebnis konfigurieren) die Identitätsanbietertypen aus, die Ihrem Benutzerpool zugewiesen werden sollen.
5. Wählen Sie unter Cognito user pool sign-in options (Anmeldeoptionen für Cognito-Benutzerpool) eine beliebige Kombination von Benutzername, E-Mail-Adresse und Telefonnummer aus.
6. Wählen Sie unter Anforderungen an Benutzernamen die Option Anmeldung auch mit bevorzugtem Benutzernamen zulassen aus, damit Benutzer bei der Anmeldung einen alternativen Benutzernamen festlegen können.
7. Wählen Sie Next (Weiter) aus und führen Sie alle Schritte im Assistenten aus.

Option 2: E-Mail-Adresse oder Telefonnummer als Anmeldeattribut (Benutzernamenattribute)

Wenn sich der Benutzer mit einer E-Mail-Adresse oder Telefonnummer als Benutzername registriert, können Sie auswählen, ob er sich nur mit E-Mail-Adressen, nur mit Telefonnummern oder mit beidem registrieren kann.

Wählen Sie zum Auswählen von Benutzernamenattributen nicht Benutzername als Anmeldeoption aus, wenn Sie den Benutzerpool erstellen.

Die E-Mail-Adresse oder Telefonnummer muss eindeutig sein und darf nicht bereits von einem anderen Benutzer verwendet werden. Sie muss nicht bestätigt sein. Nachdem sich der Benutzer mit einer E-Mail-Adresse oder Telefonnummer registriert hat, kann der Benutzer mit dieser E-Mail-Adresse oder Telefonnummer kein neues Konto erstellen. Der Benutzer kann das vorhandene Konto nur wiederverwenden und das Kontopasswort bei Bedarf zurücksetzen. Der Benutzer kann jedoch die E-Mail-Adresse oder Telefonnummer in eine neue E-Mail-Adresse oder Telefonnummer ändern. Wenn die E-Mail-Adresse oder Telefonnummer noch nicht in Gebrauch ist, wird sie zum neuen Benutzernamen.

> **Note**
>
> Wenn sich ein Benutzer mit einer E-Mail-Adresse als Benutzername registriert, kann er den Benutzernamen zu einer anderen E-Mail-Adresse ändern, jedoch nicht zu einer Telefonnummer. Wenn sich Benutzer mit einer Telefonnummer registrieren, können sie den Benutzernamen zu einer anderen Telefonnummer ändern, jedoch nicht zu einer E-Mail-Adresse.

Führen Sie während der Erstellung des Benutzerpools die folgenden Schritte aus, um eine Registrierung und Anmeldung mit einer E-Mail-Adresse oder Telefonnummer einzurichten.

Konfigurieren Sie einen Benutzerpool für die Registrierung und Anmeldung mit einer E-Mail-Adresse oder Telefonnummer wie folgt

1. Wechseln Sie zu Amazon Cognito in der AWS Management Console. Wenn Sie von der Konsole dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie in der rechten oberen Ecke der Seite Create a user pool (Benutzerpool erstellen) aus, um den Assistenten zur Erstellung von Benutzerpools zu starten.
4. Wählen Sie unter Cognito user pool sign-in options (Anmeldeoptionen für Cognito-Benutzerpool) eine beliebige Kombination von Email (E-Mail-Adresse) und Phone number (Telefonnummer) aus, die die Attribute repräsentiert, mit denen sich der Benutzer anmelden kann.
5. Wählen Sie Next (Weiter) aus und führen Sie die restlichen Schritte im Assistenten aus.

> **Note**
>
> Sie müssen die E-Mail-Adresse oder die Telefonnummer nicht als erforderliche Attribute für Ihren Benutzerpool markieren.

So implementieren Sie Option 2 in Ihrer App

1. Rufen Sie die `CreateUserPool`-API auf, um Ihren Benutzerpool zu erstellen. Setzen Sie den Parameter `UserNameAttributes` auf `phone_number`, `email` oder `phone_number | email`.
2. Rufen Sie die `SignUp`-API auf und übergeben Sie eine E-Mail-Adresse oder Telefonnummer an den Parameter `username` der API. Diese API führt folgende Aktionen aus:
   - Wenn die Zeichenfolge `username` ein gültiges E-Mail-Adressformat besitzt, füllt der Benutzerpool das Attribut `email` des Benutzers automatisch mit dem Wert `username` aus.
   - Wenn die Zeichenfolge `username` ein gültiges Telefonnummernformat besitzt, füllt der Benutzerpool das Attribut `phone_number` des Benutzers automatisch mit dem Wert `username` aus.
   - Wenn die `username`-Zeichenfolge kein gültiges E-Mail-Adress- oder Telefonnummernformat besitzt, gibt die `SignUp`-API eine Ausnahme zurück.
   - Die `SignUp`-API generiert eine permanente UUID für Ihren Benutzer und verwendet sie intern als unveränderliches Benutzernamenattribut. Diese UUID hat den gleichen Wert wie der `sub`-Anspruch im Token für die Benutzeridentität.
   - Wenn die Zeichenfolge `username` eine E-Mail-Adresse oder Telefonnummer enthält, die bereits verwendet wird, gibt die `SignUp`-API eine Ausnahme zurück.

Sie können in allen APIs außer der `ListUsers`-API eine E-Mail-Adresse oder Telefonnummer als Alias anstelle des Benutzernamens verwenden. Wenn Sie `ListUsers` aufrufen, können Sie nach dem `email`- oder dem `phone_number`-Attribut suchen. Wenn Sie nach `username` suchen, müssen Sie den tatsächlichen Benutzernamen und keinen Alias angeben.

Custom attributes (Benutzerdefinierte Attribute)

Sie können Ihrem Benutzerpool bis zu 50 benutzerdefinierte Attribute hinzufügen. Für die benutzerdefinierten Attribute können Sie eine Mindest- und/oder Höchstlänge festlegen. Die

maximale Länge eines benutzerdefinierten Attributs darf jedoch nicht mehr als 2048 Zeichen betragen.

Jedes benutzerdefinierte Attribut hat die folgenden Eigenschaften:

- Sie können es als String oder Zahl definieren. Amazon Cognito schreibt benutzerdefinierte Attributwerte nur als Strings in das ID-Token.
- Sie können nicht verlangen, dass Benutzer einen Wert für das Attribut angeben.
- Sie können es nicht mehr entfernen oder ändern, nachdem Sie es dem Benutzerpool hinzugefügt haben.
- Die Zeichenlänge des Attributnamens liegt innerhalb des Grenzwerts, den Amazon Cognito akzeptiert. Weitere Informationen finden Sie unter Kontingente in Amazon Cognito.
- Es kann veränderlich oder unveränderlich sein. Sie können einen Wert nur in ein unveränderliches benutzerdefiniertes Attribut schreiben, wenn Sie einen Benutzer erstellen. Sie können den Wert eines veränderlichen Attributs ändern, wenn Ihr App-Client über Schreibberechtigung für das Attribut verfügt. Weitere Informationen finden Sie unter Attributberechtigungen und -bereiche.

> **Note**
>
> Im Code und in den Regeleinstellungen für Verwenden der rollenbasierten Zugriffskontrolle ist für benutzerdefinierte Attribute zur Unterscheidung von Standardattributen das Präfix `custom:` erforderlich.

Sie können auch Entwicklerattribute hinzufügen, wenn Sie Benutzerpools erstellen, und zwar in der `SchemaAttributes` Eigenschaft von CreateUserPool. Entwicklerattribute verfügen über ein `dev:`-Präfix. Sie können die Entwicklerattribute eines Benutzers nur mit AWS Anmeldeinformationen ändern. Entwicklerattribute sind eine veraltete Funktion, die Amazon Cognito durch Lese- und Schreibberechtigungen für App-Clients ersetzt hat.

Führen Sie die folgenden Schritte aus, um ein neues benutzerdefiniertes Attribut zu erstellen.

Ein benutzerdefiniertes Attribut mithilfe der Konsole hinzufügen

1. Gehen Sie zu Amazon Cognito in der AWS Management Console. Wenn Sie von der Konsole dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.

3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Sign-up experience (Anmeldeerlebnis) und im Abschnitt Custom attributes (Benutzerdefinierte Attribute) Add custom attributes (Benutzerdefinierte Attribute hinzufügen) aus.
5. Geben Sie auf der Seite Add custom attributes (Benutzerdefinierte Attribute hinzufügen) die folgenden Details zu neuen Attributen ein:
   - Geben Sie einen Name (Namen) ein.
   - Wählen Sie als Type (Typ) entweder String (Zeichenfolge) oder Number (Zahl) aus.
   - Geben Sie eine/n Min (minimal) Zeichenfolgelänge oder -Zahlenwert ein.
   - Geben Sie eine/n Max (maximal) Zeichenfolgelänge oder -Zahlenwert ein.
   - Wählen Sie Mutable (Veränderlich) aus, wenn Sie Benutzern die Berechtigung erteilen möchten, den Wert eines benutzerdefinierten Attributs zu ändern, nachdem sie den Anfangswert festgelegt haben.
6. Wählen Sie Save Changes.

Attributberechtigungen und -bereiche

Für jeden App-Client können Sie Lese- und Schreibberechtigungen für jedes Benutzerattribut festlegen. Auf diese Weise können Sie den Zugriff steuern, den jede App benötigt, um jedes Attribut zu lesen und zu ändern, das Sie für Ihre Benutzer speichern. Beispiel: Sie haben ein benutzerdefiniertes Attribut, das angibt, ob ein Benutzer zahlender Kunde ist oder nicht. Ihre Apps können dieses Attribut möglicherweise sehen, es aber nicht direkt ändern. Stattdessen aktualisieren Sie dieses Attribut mit einem Verwaltungstool oder einem Hintergrundprozess. Sie können Berechtigungen für Benutzerattribute über die Amazon-Cognito-Konsole, die Amazon-Cognito-API oder die AWS CLI festlegen. Standardmäßig werden alle neuen benutzerdefinierten Attribute erst verfügbar, wenn Sie Lese- und Schreibberechtigungen dafür einrichten. Wenn Sie einen neuen App-Client erstellen, gewähren Sie Ihrer App standardmäßig Lese- und Schreibberechtigungen für alle Standard- und benutzerdefinierten Attribute. Wenn Sie Ihre App nur auf die Menge an Informationen beschränken möchten, die sie benötigt, weisen Sie Attributen in Ihrer App-Client-Konfiguration bestimmte Berechtigungen zu.

Es hat sich bewährt, Lese- und Schreibberechtigungen für Attribute anzugeben, wenn Sie einen App-Client erstellen. Gewähren Sie Ihrem App-Client Zugriff auf die Mindestanzahl an Benutzerattributen, die Sie für den Betrieb Ihrer Anwendung benötigen.

> **Note**
>
> DescribeUserPoolClientgibt nur Werte für `ReadAttributes` und zurück`WriteAttributes`, wenn Sie andere App-Client-Berechtigungen als die Standardberechtigungen konfigurieren.

So aktualisieren Sie Attributberechtigungen (AWS Management Console)

1. Gehen Sie zu Amazon Cognito in der AWS Management Console. Wenn Sie von der Konsole dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus und klicken Sie anschließend im Abschnitt App clients (App-Clients) auf einen App-Client aus der Liste.
5. Wählen Sie im Abschnitt Attribute read and write permissions (Lese- und Schreibberechtigungen für Attribute) Edit (Bearbeiten) aus.
6. Konfigurieren Sie auf der Seite Edit attribute read and write permissions (Lese- und Schreibberechtigungen für Attribute bearbeiten) Ihre Lese- und Schreibberechtigungen und wählen Sie anschließend Save changes (Änderungen speichern) aus.

Wiederholen Sie diese Schritte für jeden App-Client, der das benutzerdefinierte Attribut verwendet.

Attribute können Sie für jede App als lesbar oder schreibbar markieren. Dies gilt sowohl für Standard- als auch für benutzerdefinierte Attribute. Ihre App kann den Wert von Attributen abrufen, die Sie als lesbar markieren, und den Wert von Attributen festlegen oder ändern, die Sie als schreibbar markieren. Wenn Ihre App versucht, einen Wert für ein Attribut festzulegen, für das sie keine Schreibberechtigung hat, gibt Amazon Cognito `NotAuthorizedException` zurück. GetUserAnfragen enthalten ein Zugriffstoken mit einem App-Client-Anspruch; Amazon Cognito gibt nur Werte für Attribute zurück, die Ihr App-Client lesen kann. Das ID-Token Ihres Benutzers aus einer App enthält nur Ansprüche, die den lesbaren Attributen entsprechen. Alle App-Clients können für den Benutzerpool erforderliche Attribute schreiben. Sie können den Wert eines Attributs in einer API-Anforderung für Amazon-Cognito-Benutzerpools nur festlegen, wenn Sie auch einen Wert für alle erforderlichen Attribute angeben, die noch keinen Wert haben.

Benutzerdefinierte Attribute verfügen über unterschiedliche Funktionen für Lese- und Schreibberechtigungen. Sie können sie für den Benutzerpool als veränderbar oder unveränderlich erstellen und sie für jeden App-Client einzeln als Lese- oder Schreibattribute festlegen.

Ein unveränderliches benutzerdefiniertes Attribut kann einmal während der Benutzererstellung aktualisiert werden. Sie können ein unveränderliches Attribut mit den folgenden Methoden auffüllen.

- `SignUp`: Ein Benutzer meldet sich mit einem App-Client an, der Schreibzugriff auf ein unveränderliches benutzerdefiniertes Attribut hat. Es wird ein Wert für dieses Attribut angegeben.
- Anmeldung mit einem Drittanbieter-IdP: Ein Benutzer meldet sich bei einem App-Client an, der Schreibzugriff auf ein unveränderliches benutzerdefiniertes Attribut hat. Ihre Benutzerpoolkonfiguration für den IdP enthält eine Regel, mit der ein bereitgestellter Antrag einem unveränderlichen Attribut zugeordnet wird.
- `AdminCreateUser`: Sie geben einen Wert für ein unveränderliches Attribut an.

Informationen zu den Bereichen, die Sie Ihren App-Clients zuweisen können, finden Sie unter Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern.

Sie können Attributberechtigungen und -bereiche nach dem Erstellen des Benutzerpools ändern.

Hinzufügen von Benutzerpool-Passwortanforderungen

Starke, komplexe Passwörter sind eine bewährte Sicherheitsmethode für Ihren Benutzerpool. Insbesondere in Anwendungen, die im Internet geöffnet sind, können schwache Passwörter die Anmeldeinformationen Ihrer Benutzer Systemen zugänglich machen, die Passwörter erraten und versuchen, auf Ihre Daten zuzugreifen. Je komplexer ein Passwort ist, desto schwieriger ist es zu erraten. Amazon Cognito verfügt über zusätzliche Tools für Administratoren mit Sicherheitsvorkehrungen, wie erweiterte Sicherheitsfunktionen und AWS WAF Web-ACLs, aber Ihre Passwortrichtlinie ist ein zentrales Element der Sicherheit Ihres Benutzerverzeichnisses.

Passwörter für lokale Benutzer in Amazon-Cognito-Benutzerpools laufen nicht automatisch ab. Als bewährte Methode sollten Sie die Uhrzeit, das Datum und die Metadaten der Zurücksetzen von Benutzerpasswörtern in einem externen System protokollieren. Bei einem externen Passwortaltersprotokoll kann Ihre Anwendung oder ein Lambda-Auslöser das Passwortalter eines Benutzers nachschlagen und nach einem bestimmten Zeitraum eine Zurücksetzung erfordern.

Sie können Ihren Benutzerpool so konfigurieren, dass eine minimale Passwortkomplexität erforderlich ist, die Ihren Sicherheitsstandards entspricht. Komplexe Passwörter haben eine Mindestlänge von mindestens acht Zeichen. Sie enthalten auch eine Mischung aus Groß-, Zahlen- und Sonderzeichen.

So richten Sie eine Benutzerpool-Passwortrichtlinie ein

1. Erstellen Sie einen Benutzerpool und navigieren Sie zum Schritt Konfigurieren der Sicherheitsanforderungen oder greifen Sie auf einen vorhandenen Benutzerpool zu und navigieren Sie zur Registerkarte Anmeldeerfahrung.
2. Navigieren Sie zu Passwortrichtlinie.
3. Wählen Sie einen Passwortrichtlinienmodus aus. Cognito-Standardwerte konfiguriert Ihren Benutzerpool mit den empfohlenen Mindesteinstellungen. Sie können auch eine benutzerdefinierte Passwortrichtlinie wählen.
4. Stellen Sie eine Mindestpasswortlänge ein. Alle Benutzer müssen sich mit einem Passwort registrieren oder erstellt werden, dessen Länge größer oder gleich diesem Wert ist. Sie können diesen Mindestwert bis 99 festlegen, aber Ihre Benutzer können Passwörter mit bis zu 256 Zeichen einrichten.
5. Konfigurieren Sie die Regeln zur Passwortkomplexität unter Passwortanforderungen. Wählen Sie die Zeichentypen – Zahlen, Sonderzeichen, Groß- und Kleinbuchstaben – aus, von denen mindestens einer in jedem Benutzerpasswort verwendet werden muss.

   Sie können mindestens eines der folgenden Zeichen in Passwörtern verwenden. Nachdem Amazon Cognito überprüft hat, ob Passwörter die mindestens erforderlichen Zeichen enthalten, können die Passwörter Ihrer Benutzer zusätzliche Zeichen beliebiger Art bis zur maximalen Passwortlänge enthalten.

   - Lateinische Groß- und Kleinbuchstaben
   - Zahlen
   - Die folgenden Sonderzeichen.

     ```
     ^ $ * . [ ] { } ( ) ? " ! @ # % & / \ , > < ' : ; | _ ~ ` = + -
     ```

   - Nicht am Anfang oder Ende stehende Leerzeichen.
6. Legen Sie einen Wert für Von Administratoren festgelegte temporäre Passwörter laufen ab in fest. Nach Ablauf dieses Zeitraums kann sich ein neuer Benutzer, den Sie in der Amazon-Cognito-Konsole oder mit `AdminCreateUser` erstellt haben, nicht anmelden und kein neues Passwort festlegen. Nachdem er sich mit seinem temporären Passwort angemeldet hat, laufen seine Benutzerkonten nie ab. Um die Passwortdauer in der Amazon Cognito-Benutzerpool-API zu aktualisieren, legen Sie einen Wert für TemporaryPasswordValidityDays in Ihrer -CreateUserPool oder UpdateUserPool-API-Anfrage fest.

- Mit einer der folgenden Methoden setzen Sie den Zugriff für ein abgelaufenes Benutzerkonto zurück.
  - Löschen Sie das Benutzerprofil und erstellen Sie es neu.
  - Legen Sie ein neues permanentes Passwort in einer AdminSetUserPassword API-Anfrage fest.
  - Generieren Sie einen neuen Bestätigungscode in einer AdminResetUserPassword API-Anfrage.

E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools

Bestimmte Ereignisse in der Client-App für Ihren Benutzerpool können dazu führen, dass Amazon Cognito eine E-Mail an Ihre Benutzer sendet. Wenn Sie beispielsweise Ihren Benutzerpool so konfigurieren, dass eine E-Mail-Verifizierung erforderlich ist, erhält der Benutzer eine E-Mail von Amazon Cognito, wenn er in Ihrer App ein neues Konto für sich anmeldet oder sein Passwort zurücksetzt. Abhängig von der Aktion, die die E-Mail initiiert, enthält die E-Mail einen Verifizierungscode oder ein temporäres Passwort.

Für die E-Mail-Zustellung können Sie eine der folgenden Optionen verwenden:

- Die Standard-E-Mail-Konfiguration, die in den Amazon Cognito-Service integriert ist.
- Ihre Amazon Simple Email Service-Konfiguration (Amazon SES)-Konfiguration.

Sie können Ihre Zustelloption ändern, nachdem Sie Ihren Benutzerpool erstellt haben.

Amazon Cognito sendet Ihren Benutzern E-Mail-Nachrichten entweder mit einem Code, den sie eingeben können, oder mit einem URL-Link, den sie auswählen können. Die folgende Tabelle zeigt die Ereignisse, die eine E-Mail-Nachricht generieren können.

Nachrichtenoptionen

Aktivität	API-Operation	Zustelloptionen	Formatierungsoptionen	Anpassbar	Nachrichtenvorlage
Passwort vergessen	ForgotPassword	E-Mail, SMS	Code	Nein	N/A
Einladung	AdminCreateUser	E-Mail, SMS	Code	Ja	Einladungsnachricht
Selbstregistrierung	SignUp	E-Mail, SMS	Code, Link	Ja	Bestätigungsnachricht
Bestätigung der E-Mail-Adresse oder Telefonnummer	UpdateUserAttributes	E-Mail, SMS	Code	Ja	Bestätigungsnachricht
Multi-Faktor-Authentifizierung (MFA)	AdminInitiateAuth, InitiateAuth	SMS	Code	Ja[1]	MFA-Nachricht

[1] Für SMS-Nachrichten.

Amazon SES berechnet Gebühren für E-Mail-Nachrichten. Weitere Informationen finden Sie unter Amazon SES – Preise.

Standard-E-Mail-Konfiguration

Amazon Cognito kann seine Standard-E-Mail-Konfiguration verwenden, um E-Mail-Lieferungen für Sie abzuwickeln. Mit der Standardoption begrenzt Amazon Cognito die Anzahl der pro Tag zugestellten E-Mails an Ihren Benutzerpool. Weitere Informationen zu Service Limits finden Sie unter Kontingente in Amazon Cognito. In den meisten Produktionsumgebungen liegt das Limit der Standard-E-Mail-Funktionalität unter dem erforderlichen Zustellungsvolumen. Wenn Sie das Zustellungsvolumen erhöhen möchten, können Sie Ihre E-Mail-Konfiguration von Amazon SES verwenden.

Wenn Sie die Standardkonfiguration verwenden, verwenden Sie Amazon SES SES-Ressourcen, die von verwaltet werden, AWS um E-Mail-Nachrichten zu senden. Amazon SES fügt E-Mail-Adressen hinzu, die permanente Unzustellbarkeit an eine Unterdrückungsliste auf Kontoebene oder eine globale Unterdrückungsliste zurückgeben. Wenn eine E-Mail-Adresse, die nicht zugestellt werden kann, später zustellbar wird, können Sie nicht kontrollieren, ob sie aus der Unterdrückungsliste entfernt wird, solange Ihr Benutzerpool so konfiguriert ist, dass er die Standardkonfiguration verwendet. Eine E-Mail-Adresse kann auf unbestimmte Zeit auf der Liste mit AWS verwalteter Sperrung verbleiben. Verwenden Sie zum Verwalten nicht zustellbarer E-Mail-Adressen Ihre Amazon-SES-E-Mail-Konfiguration mit einer Unterdrückungsliste auf Kontoebene, wie im nächsten Abschnitt beschrieben.

Wenn Sie die Standard-E-Mail-Konfiguration verwenden, können Sie als Absenderadresse eine der folgenden E-Mail-Adressen verwenden:

- Die Standard-E-Mail-Adresse no-reply@verificationemail.com.
- Eine benutzerdefinierte E-Mail-Adresse. Bevor Sie Ihre eigene E-Mail-Adresse verwenden können, müssen Sie sie mit Amazon SES verifizieren und Amazon Cognito die Berechtigung zur Verwendung dieser Adresse erteilen.

E-Mail-Konfiguration von Amazon SES

Ihre Anwendung erfordert möglicherweise ein höheres Zustellungsvolumen als es die Standardoption zulässt. Um das zulässige Zustellungsvolumen zu erhöhen, verwenden Sie Ihre Amazon SES-Ressourcen mit Ihrem Benutzerpool, um Ihren Benutzern eine E-Mail zu senden. Sie können auch Ihre E-Mail-Versandaktivitäten überwachen, wenn Sie E-Mail-Nachrichten mit Ihrer eigenen Amazon-SES-Konfiguration senden.

Bevor Sie Ihre Konfiguration für Amazon SES verwenden können, müssen Sie eine oder mehrere E-Mail-Adressen oder eine Domäne mit Amazon SES verifizieren. Verwenden Sie als Absender eine verifizierte E-Mail-Adresse oder eine Adresse aus einer verifizierten Domäne, die Sie Ihrem Benutzerpool zuweisen. Wenn Amazon Cognito E-Mails an einen Benutzer sendet, ruft es Amazon SES für Sie auf und verwendet Ihre E-Mail-Adresse.

Wenn Sie Ihre Amazon-SES-Konfiguration verwenden, gelten die folgenden Bedingungen:

- Das E-Mail-Zustellungslimit für Ihren Benutzerpool entspricht dem Zustellungslimit Ihrer Amazon-SES-verifizierten E-Mail-Adresse in Ihrem AWS-Konto.

- Sie können Ihre Nachrichten an nicht zustellbare E-Mail-Adressen mit einer Unterdrückungsliste auf Kontoebene in Amazon SES verwalten, die die globale Unterdrückungsliste außer Kraft setzt. Wenn Sie eine Unterdrückungsliste auf Kontoebene verwenden, wirkt sich die Unzustellbarkeit von E-Mail-Nachrichten auf die Reputation Ihres Kontos als Absender aus. Weitere Informationen finden Sie unter Verwenden der Unterdrückungsliste auf Kontoebene im Entwicklerhandbuch für Amazon Simple Email Service.

Regionen für die E-Mail-Konfiguration von Amazon SES

Für den AWS-Region Ort, an dem Sie einen Benutzerpool erstellen, gilt eine von drei Anforderungen für die Konfiguration von E-Mail-Nachrichten mit Amazon SES. Sie können E-Mail-Nachrichten von Amazon SES in derselben Region wie Ihr Benutzerpool, in mehreren Regionen, einschließlich derselben Region, oder in einer oder mehreren abgelegenen Regionen senden. Um eine optimale Leistung zu erzielen, senden Sie E-Mail-Nachrichten mit einer von Amazon SES verifizierten Identität in derselben Region wie Ihr Benutzerpool, sofern Sie die Möglichkeit dazu haben.

Kategorien von regionalen Anforderungen für von Amazon SES verifizierte Identitäten

Nur in der Region

Ihre Benutzerpools können genauso AWS-Region wie der Benutzerpool E-Mail-Nachrichten mit verifizierten Identitäten senden. In der Standard-E-Mail-Konfiguration ohne benutzerdefinierte `FROM` E-Mail-Adresse verwendet Amazon Cognito eine `no-reply@verificationemail.com` verifizierte Identität in derselben Region.

Abwärtskompatibel

Ihre Benutzerpools können E-Mail-Nachrichten mit verifizierten Identitäten in derselben AWS-Region oder in einer der folgenden alternativen Regionen senden:

- USA Ost (Nord-Virginia)
- USA West (Oregon)
- Europa (Irland)

Diese Funktion unterstützt die Kontinuität von Benutzerpool-Ressourcen, die Sie möglicherweise erstellt haben, um den Amazon Cognito Cognito-Anforderungen zu entsprechen, als der Service gestartet wurde. Benutzerpools aus diesem Zeitraum konnten nur E-Mail-Nachrichten mit verifizierten Identitäten in einer begrenzten Anzahl von versenden. AWS-Regionen In der Standard-E-Mail-Konfiguration ohne benutzerdefinierte `FROM` E-Mail-Adresse verwendet Amazon Cognito eine `no-reply@verificationemail.com` verifizierte Identität in derselben Region.

Alternative Region

Ihre Benutzerpools können E-Mail-Nachrichten mit verifizierten Identitäten in einer Alternative versenden AWS-Region , die sich außerhalb der Benutzerpoolregion befindet. Diese Konfiguration tritt auf, wenn Amazon SES in einer Region, in der Amazon Cognito verfügbar ist, nicht verfügbar ist.

Die Amazon SES SES-Versandautorisierungsrichtlinie für Ihre verifizierte Identität in der alternativen Region muss dem Amazon Cognito-Service Principal der Ursprungsregion vertrauen. Weitere Informationen finden Sie unter So gewähren Sie Berechtigungen zur Verwendung der Standard-E-Mail-Konfiguration.

In einigen dieser Regionen teilt Amazon Cognito E-Mail-Nachrichten für die Standard-E-Mail-Konfiguration von auf zwei alternative Regionen auf. `COGNITO_DEFAULT` In diesen Fällen muss die Amazon SES-Versandautorisierungsrichtlinie für Ihre verifizierte Identität in jeder alternativen Region dem Amazon Cognito-Service Principal der Ursprungsregion vertrauen, um eine benutzerdefinierte `FROM` E-Mail-Adresse verwenden zu können. Weitere Informationen finden Sie unter So gewähren Sie Berechtigungen zur Verwendung der Standard-E-Mail-Konfiguration. Wenn die Amazon SES SES-E-Mail-Konfiguration `DEVELOPER` in diesen Regionen aktiviert ist, müssen Sie eine verifizierte Identität in der ersten aufgelisteten Region verwenden und diese so konfigurieren, dass sie dem Amazon Cognito-Service Principal in der Benutzerpool-Region vertraut. Konfigurieren Sie beispielsweise in einem Benutzerpool im Mittleren Osten (VAE) eine verifizierte Identität in Europa (Frankfurt), die als vertrauenswürdig `cognito-idp.me-central-1.amazonaws.com` eingestuft wird. In der Standard-E-Mail-Konfiguration ohne benutzerdefinierte `FROM` E-Mail-Adresse verwendet Amazon Cognito in jeder Region eine `no-reply@verificationemail.com` verifizierte Identität.

> **Note**
>
> Unter der folgenden Kombination von Bedingungen müssen Sie den `SourceArn` Parameter von EmailConfigurationmit einem Platzhalter im Element Region im Format angeben. `arn:${Partition}:ses:*:${Account}:identity/${IdentityName}` Auf diese Weise kann Ihr Benutzerpool in beiden Fällen E-Mail-Nachrichten mit identischen verifizierten AWS-Konto Identitäten versenden. AWS-Regionen
>
> - Dein EmailSendingAccount ist`COGNITO_DEFAULT`.
> - Sie möchten eine benutzerdefinierte `FROM` Adresse verwenden.

- Ihr Benutzerpool versendet E-Mails in einer alternativen Region.
- Ihr Benutzerpool hat eine zweite [1] alternative Region, die in der folgenden Tabelle der von Amazon SES unterstützten Regionen angegeben ist.

Wenn Sie einen Benutzerpool programmgesteuert erstellen — mit einem AWS SDK, der Amazon Cognito Cognito-API oder CLI AWS CDK, dem oder AWS CloudFormation— sendet Ihr Benutzerpool E-Mail-Nachrichten mit der Amazon SES SES-Identität, die der `SourceArn` Parameter von für Ihren Benutzerpool angibt. EmailConfiguration Die Amazon SES SES-Identität muss einen unterstützten Wert haben AWS-Region. Wenn Ihre `EmailSendingAccount` `COGNITO_DEFAULT` ist und Sie keinen `SourceArn`-Parameter angeben, sendet Amazon Cognito mit den Ressourcen in der Region, in der Sie Ihren Benutzerpool erstellt haben, E-Mail-Nachrichten von `no-reply@verificationemail.com`.

Die folgende Tabelle zeigt, AWS-Regionen wo Sie Amazon SES SES-Identitäten mit Amazon Cognito verwenden können.

Region des Benutzerpools	Option Region	Von Amazon SES unterstützte Regionen
USA Ost (Nord-Virginia)	Abwärtskompatibel	USA West (Oregon), USA Ost (Nord-Virginia), Europa (Irland)
USA Ost (Ohio)	Abwärtskompatibel	USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
USA West (Nordkalifornien)	Nur in der Region	USA West (Nordkalifornien)
USA West (Oregon)	Abwärtskompatibel	USA West (Oregon), USA Ost (Nord-Virginia), Europa (Irland)
Kanada (Zentral)	Abwärtskompatibel	Kanada (Zentral), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)

Region des Benutzerpools	Option Region	Von Amazon SES unterstützte Regionen
Asien-Pazifik (Tokio)	Abwärtskompatibel	Asien-Pazifik (Tokio), USA West (Oregon), USA Ost (Nord-Virginia), Europa (Irland)
Asien-Pazifik (Seoul)	Abwärtskompatibel	Asien-Pazifik (Tokio), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
Asien-Pazifik (Mumbai)	Abwärtskompatibel	Asien-Pazifik (Mumbai), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
Asien-Pazifik (Hyderabad)	Alternative Region	Asien-Pazifik (Mumbai), Asien-Pazifik (Singapur) [1]
Asien-Pazifik (Singapur)	Abwärtskompatibel	Asien-Pazifik (Singapur), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
Asien-Pazifik (Sydney)	Abwärtskompatibel	Asien-Pazifik (Sydney), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
Asien-Pazifik (Osaka)	Nur in der Region	Asien-Pazifik (Osaka)
Asien-Pazifik (Jakarta)	Nur in der Region	Asien-Pazifik (Jakarta)
Asien-Pazifik (Melbourne)	Alternative Region	Asien-Pazifik (Sydney), Asien-Pazifik (Singapur) [1]
Europa (Irland)	Abwärtskompatibel	USA West (Oregon), USA Ost (Nord-Virginia), Europa (Irland)

Region des Benutzerpools	Option Region	Von Amazon SES unterstützte Regionen
Europa (London)	Abwärtskompatibel	Europa (London), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
Europa (Paris)	Nur in der Region	Europa (Paris)
Europa (Frankfurt)	Abwärtskompatibel	Europa (Frankfurt), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
Europa (Zürich)	Alternative Region	Europa (Frankfurt), Europa (London) [1]
Europa (Stockholm)	Nur in der Region	Europa (Stockholm)
Europa (Milan)	Nur in der Region	Europa (Milan)
Europa (Spain)	Alternative Region	Europa (Paris), Europa (Stockholm) [1]
Naher Osten (Bahrain)	Nur in der Region	Naher Osten (Bahrain)
Naher Osten (VAE)	Alternative Region	Europa (Frankfurt), Europa (London) [1]
Südamerika (São Paulo)	Nur in der Region	Südamerika (São Paulo)
Israel (Tel Aviv)	Nur in der Region	Israel (Tel Aviv)
Afrika (Kapstadt)	Nur in der Region	Afrika (Kapstadt)

[1] Wird in Benutzerpools mit der Standard-E-Mail-Konfiguration verwendet. Amazon Cognito verteilt E-Mail-Nachrichten an verifizierte Identitäten mit derselben E-Mail-Adresse in jeder Region. Um eine benutzerdefinierte `FROM` Adresse zu verwenden,

konfigurieren Sie sie `EmailConfiguration` mit einem `SourceArn` Parameter im Format. `arn:${Partition}:ses:*:${Account}:identity/${IdentityName}`

Konfigurieren von E-Mail-Einstellungen für Ihren Benutzerpool

Führen Sie die folgenden Schritte aus, um die E-Mail-Einstellungen für Ihren Benutzerpool zu konfigurieren. Abhängig von den Einstellungen, die Sie verwenden, benötigen Sie möglicherweise IAM-Berechtigungen in Amazon SES, AWS Identity and Access Management (IAM) und Amazon Cognito.

> **Note**
>
> Sie können die Ressourcen, die Sie in den Schritten in AWS-Konten erstellen, nicht teilen. Sie können beispielsweise keinen Benutzerpool in einem Konto konfigurieren und ihn dann mit einer E-Mail-Adresse für Amazon SES in einem anderen Konto verwenden. Wenn Sie Amazon Cognito in mehreren Konten verwenden, müssen Sie diese Schritte für jedes Konto wiederholen.

Schritt 1: Verifizieren Ihrer E-Mail-Adresse oder Domäne mit Amazon SES

Bevor Sie Ihren Benutzerpool konfigurieren, müssen Sie in folgenden Fällen eine oder mehrere Domänen oder E-Mail-Adressen mit Amazon SES verifizieren:

- Sie möchten Ihre eigene E-Mail-Adresse als Absenderadresse verwenden
- Sie möchten für die E-Mail-Zustellung Ihre Amazon-SES-Konfiguration verwenden

Durch Verifizierung Ihrer E-Mail-Adresse oder Domäne bestätigen Sie, dass diese Ihnen gehört, und verhindern somit eine unbefugte Nutzung.

Informationen zum Verifizieren einer E-Mail-Adresse mit Amazon SES finden Sie unter Verifizieren einer E-Mail-Adresse im Entwicklerhandbuch zu Amazon Simple Email Service. Weitere Informationen zum Verifizieren einer Domäne mit Amazon SES finden Sie unter Verifying domains (Domänen verifizieren).

Schritt 2: Verschieben Ihres Kontos aus der Amazon-SES-Sandbox

Lassen Sie diesen Schritt aus, wenn Sie die standardmäßige E-Mail-Konfiguration von Amazon Cognito verwenden.

Wenn Sie Amazon SES zum ersten Mal in einer beliebigen Region verwenden AWS-Region, wird Ihr System AWS-Konto in der Amazon SES SES-Sandbox für diese Region platziert. Amazon SES verwendet die Sandbox zur Betrugs- und Missbrauchsbekämpfung. Wenn die Amazon-SES-Konfiguration Ihre E-Mail-Zustellung übernimmt, müssen Sie Ihr AWS-Konto aus der Sandbox verschieben. Erst dann kann Amazon Cognito E-Mails an Ihre Benutzer senden.

In der Sandbox werden die Anzahl der E-Mails, die Sie versenden können, sowie die Empfängeradressen durch Amazon SES beschränkt. Sie können E-Mails nur an Adressen und Domänen senden, die Sie mit Amazon SES verifiziert haben, oder an Adressen, die dem Amazon-SES-Postfachsimulator zugeordnet sind. Solange Sie in der Sandbox AWS-Konto bleiben, sollten Sie Ihre Amazon SES SES-Konfiguration nicht für Anwendungen verwenden, die sich in der Produktion befinden. Ansonsten könnte Amazon Cognito keine Nachrichten an die E-Mail-Adressen Ihrer Benutzer senden.

Informationen zum Entfernen AWS-Konto aus der Sandbox finden Sie unter Verlassen der Amazon SES SES-Sandbox im Amazon Simple Email Service Developer Guide.

Schritt 3: Erteilen von Berechtigungen für den E-Mail-Versand an Amazon Cognito

Möglicherweise müssen Sie Amazon Cognito bestimmte Berechtigungen erteilen, bevor es Ihren Benutzern E-Mails senden kann. Die Berechtigungen, die Sie gewähren, und das Verfahren, mit dem Sie sie gewähren, hängen davon ab, ob Sie die Standard-E-Mail-Konfiguration oder Ihre Amazon SES SES-Konfiguration verwenden.

So gewähren Sie Berechtigungen zur Verwendung der Standard-E-Mail-Konfiguration

Führen Sie diesen Schritt nur aus, wenn Sie Ihren Benutzerpool auf E-Mail mit Cognito senden konfiguriert oder `EmailSendingAccount` auf `COGNITO_DEFAULT` eingestellt haben.

Mit der Standard-E-Mail-Konfiguration kann Ihr Benutzerpool E-Mail-Nachrichten mit einer der folgenden Adressen senden.

- Die Standardadressen`no-reply@verificationemail.com`.
- Eine benutzerdefinierte Absenderadresse von Ihren verifizierten E-Mail-Adressen oder Domains in Amazon SES.

Wenn Sie eine benutzerdefinierte Adresse verwenden, benötigt Amazon Cognito zusätzliche Berechtigungen, um Benutzern von dieser Adresse aus E-Mails zu senden. Diese Berechtigungen werden durch eine Versandautorisierungsrichtlinie für die Adresse oder Domain in Amazon SES

gewährt. Wenn Sie die Amazon-Cognito-Konsole verwenden, um eine benutzerdefinierte Adresse zu Ihrem Benutzerpool hinzuzufügen, wird die Richtlinie automatisch an die mit Amazon SES verifizierte E-Mail-Adresse angehängt. Wenn Sie Ihren Benutzerpool jedoch außerhalb der Konsole konfigurieren, z. B. mithilfe der AWS CLI oder der Amazon Cognito Cognito-API, müssen Sie die Richtlinie über die Amazon SES SES-Konsole oder die PutIdentityPolicyAPI anhängen.

> **Note**
>
> Sie können eine Absenderadresse in einer verifizierten Domäne nur mit der AWS CLI oder der Amazon-Cognito-API konfigurieren.

Eine Versandautorisierungsrichtlinie ermöglicht oder verweigert den Zugriff basierend auf den Kontoressourcen, die Amazon Cognito zum Aufrufen von Amazon SES verwenden. Weitere Informationen zu ressourcenbasierten Richtlinien finden Sie im IAM-Benutzerhandbuch. Beispiele für ressourcenbasierte Richtlinien finden Sie auch im Entwicklerhandbuch zu Amazon SES.

Example Sendeautorisierungsrichtlinie

Das folgende Beispiel für eine Sendeautorisierungsrichtlinie gewährt Amazon Cognito die eingeschränkte Möglichkeit, eine von Amazon SES verifizierte Identität zu verwenden. Amazon Cognito kann nur dann E-Mail-Nachrichten senden, wenn es diese Funktion für den Benutzerpool in der `aws:SourceArn`-Bedingung als auch für das Konto in der `aws:SourceAccount`-Bedingung übernimmt.

Regions with Amazon SES

Ihre Versandautorisierungsrichtlinie in der Benutzerpoolregion oder alternativen Region muss dem Amazon Cognito-Service Principal das Senden von E-Mail-Nachrichten ermöglichen. Weitere Informationen finden Sie in der Tabelle mit den Regionen. Wenn Ihre Benutzerpool-Region mindestens einem Wert in der Amazon SES SES-Region entspricht, konfigurieren Sie im folgenden Beispiel Ihre Versandautorisierungsrichtlinie mit dem Global Service Principal.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "<your SES identity ARN>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "aws:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
}
```

Opt-in Regions without Amazon SES

Amazon SES ist nicht in allen Opt-ins verfügbar, in AWS-Regionen denen Amazon Cognito verfügbar ist. Der Nahe Osten (VAE) ist ein Beispiel. Dort können nur E-Mails mit verifizierten Identitäten in Europa (Frankfurt) () versendet werden. `eu-central-1` In Benutzerpools mit der Standard-E-Mail-Konfiguration sendet Amazon Cognito auch E-Mail-Nachrichten mit einer verifizierten Identität in jeder von zwei Regionen. Im Fall des Nahen Ostens (VAE) ist die zusätzliche Region Europa (London). Sie müssen die Richtlinien zur Versandautorisierung in beiden Regionen aktualisieren.

Ihre Richtlinien zur Sendeautorisierung in jeder der alternativen Regionen müssen dem Amazon Cognito-Service Principal in der Benutzerpool-Opt-in-Region das Senden von E-Mail-Nachrichten gestatten. Weitere Informationen finden Sie in der Tabelle mit den Regionen. Wenn Ihre Region als Alternative Region markiert ist, konfigurieren Sie Ihre Richtlinien zur Versandautorisierung mit dem regionalen Service Principal wie im folgenden Beispiel. Ersetzen Sie die Beispiel-Regionskennung *`me-central-1`* nach Bedarf durch die erforderliche Region-ID.

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

```
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cognito-idp.me-central-1.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "<your SES identity ARN>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "aws:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
}
```

Weitere Informationen zur Richtliniensyntax finden Sie unter Amazon-SES-Sendeautorisierungsrichtlinien im Entwicklerhandbuch für Amazon Simple Email Service.

Weitere Beispiele finden Sie unter Beispiele von Amazon-SES-Sendeautorisierungsrichtlinien im Entwicklerhandbuch für Amazon Simple Email Service.

Berechtigungen zur Verwendung Ihrer Amazon-SES-Konfiguration erteilen

Wenn Sie Ihren Benutzerpool so konfigurieren, dass Ihre Amazon-SES-Konfiguration verwendet wird, benötigt Amazon Cognito eine zusätzliche Berechtigung, um in Ihrem Namen Amazon SES aufzurufen und E-Mails an Ihre Benutzer zu senden. Diese Autorisierung mit dem IAM-Service erteilt.

Wenn Sie Ihren Benutzerpool mit dieser Option konfigurieren, erstellt Amazon Cognito eine serviceverknüpfte Rolle. Dabei handelt es sich um eine Art der IAM-Rolle in Ihrem AWS-Konto. Diese Rolle enthält die Berechtigungen, mit denen Amazon Cognito auf Amazon SES zugreifen und E-Mails mit Ihrer Adresse senden kann.

Amazon Cognito erstellt Ihre serviceverknüpfte Rolle mit den AWS Anmeldeinformationen der Benutzersitzung, die die Konfiguration festlegt. Die IAM-Berechtigungen dieser Sitzung müssen die

`iam:CreateServiceLinkedRole`-Aktion enthalten. Weitere Informationen zu Berechtigungen in IAM finden Sie unter Zugriffsverwaltung für AWS Ressourcen im IAM-Benutzerhandbuch.

Weitere Informationen zur serviceverknüpften Rolle, die Amazon Cognito erstellt, finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon Cognito.

Schritt 4: Konfigurieren des Benutzerpools

Führen Sie die folgenden Schritte aus, wenn Sie Ihren Benutzerpool folgendermaßen konfigurieren möchten:

- Mit einer benutzerdefinierten Absenderadresse, die als solche angezeigt wird
- Mit einer benutzerdefinierten Antwortadresse, bei der die Nachrichten eingehen, die Ihre Benutzer an die Absenderadresse senden
- Ihre Amazon-SES-Konfiguration

> **Note**
>
> Wenn es sich bei Ihrer verifizierten Identität um eine E-Mail-Adresse handelt, legt Amazon Cognito diese E-Mail-Adresse standardmäßig als Absender- und Empfänger-E-Mail-Adresse fest. Wenn es sich bei Ihrer verifizierten Identität jedoch um eine Domain handelt, müssen Sie einen Wert für die Absender- und die Empfänger-E-Mail-Adresse angeben. Wenn Ihre verifizierte Domain beispiel.com lautet, können Sie no-reply@beispiel.com sowohl als Absender- als auch als Empfänger-E-Mail-Adresse festlegen.

Lassen Sie dieses Verfahren aus, wenn Sie die standardmäßige E-Mail-Konfiguration und -Adresse von Amazon Cognito verwenden möchten.

So konfigurieren Sie den Benutzerpool für die Verwendung einer benutzerdefinierten E-Mail-Adresse

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldedaten ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Messaging aus, suchen Sie nach Email configuration (E-Mail-Konfiguration) und klicken Sie auf Edit (Bearbeiten).

5. Wählen Sie auf der Seite Edit email configuration (E-Mail-Konfiguration bearbeiten) Send email from Amazon SES (E-Mail von Amazon SES senden) oder Send email with Amazon Cognito (E-Mail mit Amazon Cognito senden) aus. Sie können die SES-Region, das Konfigurations-Set und den Absendernamen nur anpassen, wenn Sie Send email from Amazon SES (E-Mail von Amazon SES senden) auswählen.
6. Zum Verwenden einer benutzerdefinierten Absenderadresse folgende Schritte ausführen:
   a. Wählen Sie unter SES-Region die Region mit Ihrer verifizierten E-Mail-Adresse aus.
   b. Wählen Sie unter FROM email address (Absenderadresse) Ihre E-Mail-Adresse aus. Verwenden Sie eine E-Mail-Adresse, die mit Amazon SES verifiziert wurde.
   c. (Optional) Wählen Sie unter Configuration set (Konfigurations-Set) ein Konfigurations-Set für die Verwendung mit Amazon SES aus. Wenn Sie diese Änderung vornehmen und speichern, wird eine serviceverknüpfte Rolle erstellt.
   d. (Optional) Geben Sie unter FROM sender address (Absenderadresse) eine E-Mail-Adresse ein. Sie können nur eine E-Mail-Adresse oder eine E-Mail-Adresse und einen Namen in folgendem Format angeben: `Jane Doe <janedoe@example.com>`.
   e. (Optional) Geben Sie unter REPLY-TO email address (Empfänger-E-Mail-Adresse) die E-Mail-Adresse ein, an die Sie Nachrichten erhalten möchten, die Benutzer an Ihre Absenderadresse senden.
7. Wählen Sie Save Changes.

Verwandte Themen

- Anpassen von Nachrichten zur E-Mail-Verifizierung
- Anpassen von Nachrichten zur Einladung von Benutzern

Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools

Einige Amazon-Cognito-Ereignisse für Ihren Benutzerpool können dazu führen, dass Amazon Cognito SMS-Textnachrichten an Ihre Benutzer sendet. Wenn Sie beispielsweise Ihren Benutzerpool so konfigurieren, dass eine Telefon-Verifizierung erforderlich ist, erhält der Benutzer eine SMS-Textnachricht von Amazon Cognito, wenn er in Ihrer App ein neues Konto für sich anmeldet oder sein Passwort zurücksetzt. Abhängig von der Aktion, die die SMS-Nachricht initiiert, enthält die Nachricht einen Verifizierungscode, ein temporäres Kennwort oder eine Begrüßungsnachricht.

Amazon Cognito verwendet Amazon Simple Notification Service (Amazon SNS) zur Zustellung von SMS-Nachrichten. Wenn Sie zum ersten Mal eine Textnachricht über Amazon Cognito oder Amazon SNS senden, werden Sie von Amazon SNS in eine Sandbox-Umgebung weitergeleitet. In der Sandbox-Umgebung können Sie Ihre Anwendungen auf SMS-Textnachrichten testen. In der Sandbox können Nachrichten nur an verifizierte Telefonnummern gesendet werden.

Amazon SNS berechnet für SMS-Nachrichten Gebühren. Weitere Informationen finden Sie unter Amazon SNS-Preise.

> **Note**
>
> Aufgrund des weltweiten Volumens an unaufgefordertem SMS-Verkehr verhängen einige Regierungen Sperren zwischen Absendern und Empfängern von SMS-Nachrichten. Wenn Sie SMS-Nachrichten für MFA und Benutzeraktualisierungen verwenden, müssen Sie zusätzliche Maßnahmen ergreifen, um sicherzustellen, dass Ihre Nachrichten zugestellt werden. Sie müssen auch die Vorschriften für SMS-Nachrichten in den Ländern, in denen Ihre Benutzer möglicherweise leben, überwachen und Ihre SMS-Nachrichtenkonfiguration auf dem neuesten Stand halten. Weitere Informationen finden Sie unter Mobile Textnachrichten (SMS) im Amazon Simple Notification Service Developer Guide.
> Die Verwendung von SMS-Nachrichten zur Authentifizierung und Überprüfung von Benutzern ist kein bewährtes Sicherheitsverfahren. Telefonnummern können den Besitzer wechseln und stellen möglicherweise keinen zuverlässigen Something You Have-MFA-Faktor für Ihre Benutzer dar. Implementieren Sie stattdessen TOTP MFA in Ihrer App oder mit Ihrem Drittanbieter-IdP. Es ist auch möglich, zusätzliche benutzerdefinierte Authentifizierungsfaktoren mit Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen zu erstellen.

Amazon Cognito sendet Ihren Benutzern SMS-Nachrichten mit einem Code, den diese eingeben können. Die folgende Tabelle zeigt die Ereignisse, die eine SMS-Nachricht generieren können.

Nachrichtenoptionen

Aktivität	API-Operation	Zustelloptionen	Formatierungsoptionen	Anpassbar	Nachrichtenvorlage
Passwort vergessen	ForgotPassword	E-Mail, SMS	Code	Nein	N/A
Einladung	AdminCreateUser	E-Mail, SMS	Code	Ja	Einladungsnachricht
Selbstregistrierung	SignUp	E-Mail, SMS	Code, Link	Ja	Bestätigungsnachricht
Bestätigung der E-Mail-Adresse oder Telefonnummer	UpdateUserAttributes	E-Mail, SMS	Code	Ja	Bestätigungsnachricht
Multi-Faktor-Authentifizierung (MFA)	AdminInitiateAuth, InitiateAuth	SMS, Authentifizierungs-App	Code	Ja[1]	MFA-Nachricht

[1] Für SMS-Nachrichten.

Erstmaliges Einrichten von SMS-Nachrichten in Amazon-Cognito-Benutzerpools

Amazon Cognito verwendet Amazon SNS, um SMS-Nachrichten an Ihre Benutzerpools zu senden. Sie können auch einen Benutzerdefinierter Lambda-Auslöser für SMS-Sender verwenden, um Ihre eigenen Ressourcen zum Senden von SMS-Nachrichten zu verwenden. Wenn Sie Amazon SNS zum ersten Mal für den Versand von SMS-Textnachrichten in einer bestimmten Region einrichten AWS-Region, platziert Amazon SNS Ihre Nachrichten AWS-Konto in der SMS-Sandbox für diese Region. Amazon SNS verwendet die Sandbox, um Betrug und Missbrauch zu verhindern und Compliance-Anforderungen zu erfüllen. Wenn Sie AWS-Konto sich in der Sandbox befinden, legt

Amazon SNS einige Einschränkungen fest. Sie können beispielsweise Textnachrichten an bis zu 10 Telefonnummern senden, die Sie mit Amazon SNS verifiziert haben. Solange Sie in der Sandbox AWS-Konto bleiben, verwenden Sie Ihre Amazon SNS SNS-Konfiguration nicht für Anwendungen, die sich in der Produktion befinden. Wenn Sie sich in der Sandbox befinden, kann Amazon Cognito keine Nachrichten an die Telefonnummern Ihrer Benutzer senden.

So senden Sie SMS-Textnachrichten an Benutzerpool-Benutzer

1. Bereiten Sie eine IAM-Rolle vor, die Amazon Cognito zum Senden von SMS-Nachrichten mit Amazon SNS verwenden kann.
2. Wählen Sie AWS-Region für Amazon SNS SMS-Nachrichten
3. Eine Ursprungsidentität zum Senden von SMS-Nachrichten an US-Telefonnummern anfordern
4. Bestätigen Sie, dass Sie sich in der SMS-Sandbox befinden.
5. Verschieben Ihres Kontos aus der Amazon-SNS-Sandbox
6. Überprüfen Sie die Telefonnummern für Amazon Cognito in Amazon SNS.
7. Die Benutzerpool-Einrichtung in Amazon Cognito abschließen

Bereiten Sie eine IAM-Rolle vor, die Amazon Cognito zum Senden von SMS-Nachrichten mit Amazon SNS verwenden kann.

Wenn Sie eine SMS-Nachricht aus Ihrem Benutzerpool senden, übernimmt Amazon Cognito eine IAM-Rolle in Ihrem Konto. Amazon Cognito verwendet die `sns:Publish`-Berechtigung, die dieser Rolle zugewiesen ist, um SMS-Nachrichten an Ihre Benutzer zu senden. In der Amazon-Cognito-Konsole können Sie auf der Registerkarte Messaging Ihres Benutzerpools unter SMS die Option IAM role selection (Auswahl der IAM-Rolle) festlegen oder diese Auswahl beim Verwenden des Assistenten zum Erstellen des Benutzerpools treffen.

Die folgende IAM-Rollenvertrauensrichtlinie gewährt Benutzerpools von Amazon Cognito eine eingeschränkte Möglichkeit, die Rolle zu übernehmen. Amazon Cognito kann die Rolle nur übernehmen, wenn es dies im Namen des Benutzerpools mit der Bedingung `aws:SourceArn` und im Namen des AWS-Konto mit der Bedingung `aws:SourceAccount` durchführt.

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

```
            "Service": "cognito-idp.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "<your account number>"
            },
            "ArnLike": {
                "aws:SourceArn": "<your user pool ARN>"
            }
        }
    }]
}
```

Sie können einen genauen Benutzerpool-ARN oder ein Platzhalter-ARN im Wert der Bedingung `aws:SourceArn` angeben. Suchen Sie in der AWS Management Console oder mit einer API-Anfrage nach den ARNs Ihrer Benutzerpools. DescribeUserPool

Weitere Informationen zu IAM-Rollen und -Vertrauensrichtlinien finden Sie unter Rollenbegriffe und -Konzepte im AWS Identity and Access Management -Benutzerhandbuch.

Wählen Sie AWS-Region für Amazon SNS SMS-Nachrichten

In einigen Fällen können Sie die Region auswählen AWS-Regionen, die die Amazon SNS SNS-Ressourcen enthält, die Sie für Amazon Cognito-SMS-Nachrichten verwenden möchten. In allen Ländern, in AWS-Region denen Amazon Cognito verfügbar ist, mit Ausnahme von Asien-Pazifik (Seoul), können Sie Amazon SNS SNS-Ressourcen dort verwenden, AWS-Region wo Sie Ihren Benutzerpool erstellt haben. Um Ihre SMS-Nachrichten schneller und zuverlässiger zu gestalten, wenn Sie zwischen verschiedenen Regionen auswählen können, verwenden Sie Amazon SNS-Ressourcen in derselben Region wie Ihr Benutzerpool.

> **Note**
>
> In der können Sie die Region für SMS-Ressourcen erst ändern AWS Management Console, nachdem Sie auf die neue Amazon Cognito Cognito-Konsolenoberfläche umgestellt haben.

Wählen Sie eine Region für SMS-Ressourcen im Schritt Nachrichtenzustellung konfigurieren des Assistenten für neue Benutzerpools aus. Sie können auch Edit (Bearbeiten) unter SMS auf der Registerkarte Messaging eines vorhandenen Benutzerpools auswählen.

Beim Start sendete Amazon Cognito für einige AWS-Regionen SMS-Nachrichten mit Amazon SNS SNS-Ressourcen in einer anderen Region. Um Ihre bevorzugte Region festzulegen, verwenden Sie den `SnsRegion` Parameter des SmsConfigurationTypeObjekts für Ihren Benutzerpool. Wenn Sie eine Amazon-Cognito-Benutzerpool-Ressource in einer Amazon Cognito Region (Amazon-Cognito-Region) aus der folgenden Tabelle programmgesteuert erstellen und keinen `SnsRegion`-Parameter angeben, kann Ihr Benutzerpool SMS-Nachrichten mit Amazon-SNS-Ressourcen in einer Legacy-Amazon SNS Region (Amazon-SNS-Region) senden.

Amazon Cognito Cognito-Benutzerpools im asiatisch-pazifischen Raum (Seoul) AWS-Region müssen Ihre Amazon SNS SNS-Konfiguration in der Region Asien-Pazifik (Tokio) verwenden.

Amazon SNS legt das Ausgabenkontingent für alle neuen Konten auf 1,00 USD pro Monat fest. Möglicherweise haben Sie Ihr Ausgabenlimit in einer AWS-Region , die Sie mit Amazon Cognito verwenden, erhöht. Bevor Sie die AWS-Region für Amazon SNS SMS-Nachrichten ändern, öffnen Sie im AWS Support Center einen Fall zur Erhöhung des Kontingents, um Ihr Limit in der neuen Region zu erhöhen. Weitere Informationen finden Sie unter Anfordern von Erhöhungen Ihres monatlichen SMS-Ausgabenkontingents für Amazon SNS im Entwicklerhandbuch für Amazon Simple Notification Service.

Sie können SMS-Nachrichten für jede Amazon Cognito Region (Amazon-Cognito-Region) in der folgenden Tabelle mit Amazon-SNS-Ressourcen in der entsprechenden Amazon SNS Region (Amazon-SNS-Region) senden.

Amazon-Cognito-Region	Amazon-SNS-Region
USA Ost (Ohio)	USA Ost (Ohio), USA Ost (Nord-Virginia)
Kanada (Zentral)	Kanada (Zentral), USA Ost (Nord-Virginia)
Europa (Frankfurt)	Europa (Frankfurt), Europa (Irland)
Europa (London)	Europa (London), Europa (Irland)
Asien-Pazifik (Seoul)	Asien-Pazifik (Tokio)
USA Ost (Nord-Virginia)	USA Ost (Nord-Virginia)
USA West (Nordkalifornien)	USA West (Nordkalifornien)
USA West (Oregon)	USA West (Oregon)

Amazon-Cognito-Region	Amazon-SNS-Region
Asien-Pazifik (Mumbai)	Asien-Pazifik (Mumbai), Asien-Pazifik (Singapur)
Asien-Pazifik (Hyderabad)	Asien-Pazifik (Hyderabad)
Asien-Pazifik (Singapur)	Asien-Pazifik (Singapur)
Asien-Pazifik (Sydney)	Asien-Pazifik (Sydney)
Asien-Pazifik (Tokio)	Asien-Pazifik (Tokio)
Asien-Pazifik (Jakarta)	Asien-Pazifik (Jakarta)
Asien-Pazifik (Osaka)	Asien-Pazifik (Osaka)
Asien-Pazifik (Melbourne)	Asien-Pazifik (Melbourne)
Europa (Irland)	Europa (Irland)
Europa (Paris)	Europa (Paris)
Europa (Stockholm)	Europa (Stockholm)
Europa (Milan)	Europa (Milan)
Europa (Spain)	Europa (Spain)
Naher Osten (Bahrain)	Naher Osten (Bahrain)
Südamerika (São Paulo)	Südamerika (São Paulo)
Israel (Tel Aviv)	Israel (Tel Aviv)
Afrika (Kapstadt)	Afrika (Kapstadt)
Naher Osten (VAE)	Naher Osten (VAE)
Europa (Zürich)	Europa (Zürich)

Eine Ursprungsidentität zum Senden von SMS-Nachrichten an US-Telefonnummern anfordern

Wenn Sie SMS-Textnachrichten an US-Telefonnummern senden möchten, müssen Sie eine Ursprungsidentität anfordern, unabhängig davon, ob Sie eine SMS-Sandbox-Testumgebung oder eine Produktionsumgebung erstellen.

Ab dem 1. Juni 2021 benötigen US-Anbieter eine Ursprungsidentität, um Nachrichten an US-Telefonnummern zu senden. Wenn Sie noch keine Ursprungsidentität besitzen, müssen Sie eine anfordern. Informationen zum Erhalten einer Ursprungsidentität finden Sie unter Anfordern einer Nummer im Benutzerhandbuch für Amazon Pinpoint.

Wenn Sie in einem der folgenden Länder tätig sind AWS-Regionen, müssen Sie ein AWS Support Ticket öffnen, um eine Identität des Absenders zu erhalten. Detaillierte Anweisungen finden Sie unter Anfordern von Support für SMS-Nachrichten im Entwicklerhandbuch zu Amazon Simple Notification Service.

- USA Ost (Ohio)
- Europa (Stockholm)
- Europa (Paris)
- Europa (Milan)
- Naher Osten (Bahrain)
- Südamerika (São Paulo)
- USA West (Nordkalifornien)

Wenn Sie mehr als eine Absenderidentität in derselben Person haben AWS-Region, wählt Amazon SNS einen Identitätstyp in der folgenden Prioritätsreihenfolge aus: Shortcode, 10DLC, gebührenfreie Nummer. Sie können diese Prioritätsreihenfolge nicht ändern. Weitere Informationen finden Sie unter Häufig gestellte Fragen zu Amazon SNS.

Bestätigen Sie, dass Sie sich in der SMS-Sandbox befinden.

Gehen Sie wie folgt vor, um zu bestätigen, dass Sie sich in der SMS-Sandbox befinden. Wiederholen Sie den Vorgang für jeden AWS-Region , in dem Sie Amazon Cognito Cognito-Produktions-Benutzerpools haben.

Status von SMS-Sandboxen in der Amazon Cognito-Konsole

Aktivität in der SMS-Sandbox bestätigen

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Geben Sie bei Aufforderung Ihre AWS -Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie die Registerkarte Messaging aus.
5. Erweitern Sie im Bereich SMS configuration (SMS-Konfiguration) Move to Amazon SMS production environment (Wechseln zur Amazon-SNS-Produktionsumgebung). Wenn sich Ihr Konto in der SMS-Sandbox befindet, wird die folgende Nachricht angezeigt:

   ```
   You are currently in the SMS Sandbox and cannot send SMS messages to unverified numbers.
   ```

   Wenn diese Nachricht nicht angezeigt wird, hat jemand bereits SMS-Nachrichten in Ihrem Konto eingerichtet. Fahren Sie mit Die Benutzerpool-Einrichtung in Amazon Cognito abschließen fort.
6. Wählen Sie den Link Amazon SNS in der Nachricht. Auf diese Weise wird die Amazon-SNS-Konsole in einer neuen Registerkarte geöffnet.
7. Stellen Sie sicher, dass Sie sich in der Sandbox-Umgebung befinden. Die Konsolenmeldung gibt Ihren Sandbox-Status und AWS-Region wie folgt an:

   ```
   This account is in the SMS sandbox in US East (N. Virginia).
   ```

Verschieben Ihres Kontos aus der Amazon-SNS-Sandbox

Wenn Sie Ihre App testen und nur SMS-Nachrichten an Telefonnummern senden müssen, die Ihre Administratoren überprüfen können, überspringen Sie diesen Schritt.

Um Ihre App in der Produktion zu verwenden, holen Sie Ihr Konto aus der SMS-Sandbox und übergeben Sie es in die Produktion. Nachdem Sie eine Originationsidentität konfiguriert haben AWS-Region , die die Amazon SNS SNS-Ressourcen enthält, die Amazon Cognito verwenden soll, können Sie US-Telefonnummern verifizieren, solange Ihre in der AWS-Konto SMS-Sandbox verbleiben. Wenn Ihre Amazon SNS-Umgebung in Produktion ist, müssen Sie die Benutzertelefonnummern in Amazon SNS nicht verifizieren, um SMS-Nachrichten an Ihre Benutzer zu senden.

Eine Anleitung finden Sie unter Verlassen der SNS-Sandbox im Entwicklerhandbuch zu Amazon Simple Notification Service.

Überprüfen Sie die Telefonnummern für Amazon Cognito in Amazon SNS.

Wenn Sie Ihr Konto aus der SMS-Sandbox verschoben haben, überspringen Sie diesen Schritt.

Wenn Sie sich in der SMS-Sandbox befinden, können Sie Nachrichten an jede Telefonnummer senden, die Sie mit Amazon SNS verifiziert haben.

Gehen Sie wie folgt vor, um eine Telefonnummer zu verifizieren:

1. Fügen Sie eine Sandbox-Zieltelefonnummer im Abschnitt Text messaging (SMS) (Textnachrichten (SMS)) der Amazon-SNS-Konsole hinzu.
2. Fordern Sie eine SMS-Nachricht mit einem Code an die von Ihnen angegebene Telefonnummer an.
3. Geben Sie den Verifizierungscode aus der SMS-Nachricht in der Amazon SNS-Konsole an.

Detaillierte Anweisungen finden Sie unter Hinzufügen und Überprüfen von Telefonnummern in der SMS-Sandbox im Entwicklerhandbuch zu Amazon Simple Notification Service.

> **Note**
>
> Amazon SNS begrenzt die Anzahl der Zieltelefonnummern, die Sie verifizieren können, während Sie sich in der SMS-Sandbox befinden. Informationen dazu finden Sie unter SMS-Sandbox im Entwicklerhandbuch zu Amazon Simple Notification Service.

Die Benutzerpool-Einrichtung in Amazon Cognito abschließen

Kehren Sie zur Browser-Registerkarte zurück, auf der Sie Ihren Benutzerpool erstellt oder bearbeitet haben. Schließen Sie das Verfahren ab. Wenn Sie Ihrem Benutzerpool erfolgreich eine SMS-Konfiguration hinzugefügt haben, sendet Amazon Cognito eine Testnachricht an eine interne Telefonnummer, um zu überprüfen, ob Ihre Konfiguration funktioniert. Amazon SNS berechnet für jede Test-SMS-Nachricht Gebühren.

Verwenden von Token mit Benutzerpools

Authentifizieren Sie mithilfe von Token Benutzer und gewähren Sie Zugriff auf Ressourcen. Bei den Ansprüchen in Tokens handelt es sich um Informationen über Ihren Benutzer. Das ID-Token enthält Angaben zu dessen Identität, wie etwa Benutzername, Familienname und E-Mail-Adresse. Das Zugriffstoken enthält Ansprüche wie `scope`, womit der authentifizierte Benutzer auf APIs von Drittanbietern, Amazon-Cognito-Benutzer-Selfservice-API-Operationen und den UserInfo-Endpunkt zugreifen kann. Sowohl das Zugriffs- als auch das ID-Token enthalten einen `cognito:groups`-Anspruch mit der Gruppenmitgliedschaft Ihres Benutzers in Ihrem Benutzerpool. Weitere Informationen zu Benutzerpoolgruppen finden Sie unter Hinzufügen von Gruppen zu einem Benutzerpool.

Amazon Cognito bietet außerdem auch Refresh-Token, mit denen Sie neue Token abrufen oder vorhandene Token widerrufen können. Aktualisieren Sie ein Token, um eine neue ID und Zugriffstoken abzurufen. Widerrufen Sie ein Token, um den Benutzerzugriff zu widerrufen, der durch Aktualisierungstoken zugelassen wird.

Amazon Cognito gibt Tokens als Base64-kodierte Zeichenfolgen aus. Sie können jede Amazon-Cognito-ID oder jedes Zugriffs-Token von Base64 in Klartext-JSON dekodieren. Amazon-Cognito-Refresh-Token sind verschlüsselt, für Benutzer und Administratoren von Benutzerpools undurchsichtig, und können nur von Ihrem Benutzerpool gelesen werden.

Authentifizieren mit Tokens

Wenn sich ein Benutzer bei Ihrer App anmeldet, überprüft Amazon Cognito die Anmeldeinformationen. Nach erfolgreicher Anmeldung erstellt Amazon Cognito eine Sitzung und gibt ein ID-, Zugriffs- und Refresh-Token für den authentifizierten Benutzer zurück. Mit diesen Token können Sie Ihren Benutzern Zugriff auf nachgelagerte Ressourcen und APIs wie Amazon API Gateway gewähren. Alternativ können Sie diese gegen temporäre AWS -Anmeldeinformationen eintauschen, um auf andere AWS-Services zugreifen zu können.

Speichern von Token

Ihre App muss Token unterschiedlicher Größe speichern können. Die Tokengröße kann sich unter anderem aus Gründen zusätzlicher Ansprüche, Änderungen der Kodierungsalgorithmen und Änderungen der Verschlüsselungsalgorithmen ändern. Wenn Sie den Token-Widerruf in Ihrem Benutzerpool aktivieren, fügt Amazon Cognito JSON Web Tokens zusätzliche Anforderungen hinzu und erhöht deren Größe. Die neuen Ansprüche `origin_jti` und `jti` werden zu Zugriffs- und ID-Token hinzugefügt. Weitere Informationen zum Widerrufen von Token finden Sie unter Widerrufen von Token.

> **Important**
>
> Eine bewährte Methode ist, alle Token während der Übertragung und Speicherung im Kontext Ihrer Anwendung zu sichern. Token können persönlich identifizierende Informationen über Ihre Benutzer und Informationen über das Sicherheitsmodell enthalten, das Sie für Ihren Benutzerpool verwenden.

Anpassen von Token

Sie können die Zugriffs- und ID-Token anpassen, die Amazon Cognito an Ihre App weitergibt. In einer Lambda-Auslöser für die Vorab-Generierung von Token können Sie Token-Ansprüche hinzufügen, ändern und unterdrücken. Der Trigger für Pre-Token-Generierung ist eine Lambda-Funktion, an die Amazon Cognito einen Standardsatz von Ansprüchen sendet. Zu den Ansprüchen gehören OAuth-2.0-Bereiche, die Mitgliedschaft in Benutzerpoolgruppen, Benutzerattribute und mehr. Die Funktion kann dann die Gelegenheit nutzen, Änderungen zur Laufzeit vorzunehmen und die aktualisierten Token-Ansprüche an Amazon Cognito zurückzugeben.

Für die Anpassung des Zugriffs-Token bei Ereignissen der Version 2 fallen zusätzliche Kosten an. Weitere Informationen finden Sie unter Amazon Cognito – Preise.

Themen

- Verwenden des ID-Tokens
- Verwenden des Zugriffstokens
- Verwenden des Aktualisierungs-Tokens
- Widerrufen von Token
- Verifizieren eines JSON-Web-Tokens
- Zwischenspeicherung von Token

Verwenden des ID-Tokens

Das ID-Token ist ein JSON-Web-Token (JWT), das Anforderungen bezüglich der Identität des authentifizierten Benutzers enthält, beispielsweise `name`, `email` und `phone_number`. Sie können diese Identitätsinformationen in Ihrer Anwendung verwenden. Das ID-Token kann ebenfalls verwendet werden, um Benutzer mithilfe Ihrer Ressourcenserver oder Serveranwendungen zu authentifizieren. Sie können auch ein ID-Token außerhalb der Anwendung mit Ihren Web-API-Operationen verwenden. In diesen Fällen müssen Sie die Signatur des ID-Tokens überprüfen, bevor Sie Ansprüchen innerhalb des ID-Tokens vertrauen können. Siehe Verifizieren eines JSON-Web-Tokens.

Sie können den Ablauf des ID-Tokens auf einen beliebigen Wert zwischen 5 Minuten und 1 Tag festlegen. Sie können diesen Wert pro App-Client festlegen.

> **Important**
>
> Wenn sich Ihr Benutzer bei der gehosteten Benutzeroberfläche oder einem Verbundidentitätsanbieter (IdP) anmeldet, setzt Amazon Cognito Sitzungscookies, die 1 Stunde lang gültig sind. Wenn Sie die gehostete Benutzeroberfläche oder den Verbund verwenden und eine Mindestdauer von weniger als 1 Stunde für Ihre Zugriffs- und ID-Token angeben, ist die Sitzung Ihrer Benutzer weiterhin bis zum Ablauf des Cookies gültig. Wenn der Benutzer Token hat, die in der einstündigen Sitzung ablaufen, kann der Benutzer seine Token aktualisieren, ohne sich erneut authentifizieren zu müssen.

ID-Token-Header

Der Header enthält zwei verschiedene Informationen: die Schlüssel-ID (`kid`) und den Algorithmus (`alg`).

```
{
"kid" : "1234example=",
"alg" : "RS256"
}
```

`kid`

Die Schlüssel-ID. Ihr Wert ist ein Hinweis darauf, welcher Schlüssel verwendet wurde, um die JSON-Websignatur (JWS) des Tokens zu sichern. Sie können die Signaturschlüssel-IDs Ihres Benutzerpools am `jwks_uri`-Endpunkt einsehen.

Weitere Informationen über den `kid`-Parameter finden Sie unter dem Schlüssel-ID-(kid)-Header-Parameter.

`alg`

Der kryptografische Algorithmus, mit dem Amazon Cognito das Zugriffstoken sichert. Benutzerpools verwenden einen kryptografischen RS256-Algorithmus, eine RSA-Signatur mit SHA-256.

Weitere Informationen über den `alg`-Parameter finden Sie unter dem Algorithmus-(alg)-Header-Parameter.

Standardnutzlast für ID-Tokens

Dies ist ein Beispiel für eine Payload aus einem ID-Token. Es enthält Ansprüche bezüglich des authentifizierten Benutzers. Weitere Informationen zu OpenID Connect (OIDC) -Standardansprüchen finden Sie in der Liste der OIDC-Standardansprüche. Sie können Ansprüche Ihres eigenen Designs mit einem hinzufügen. Lambda-Auslöser für die Vorab-Generierung von Token

```
<header>.{
    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "cognito:groups": [
        "test-group-a",
        "test-group-b",
        "test-group-c"
    ],
    "email_verified": true,
    "cognito:preferred_role": "arn:aws:iam::111122223333:role/my-test-role",
    "iss": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_example",
    "cognito:username": "my-test-user",
    "middle_name": "Jane",
    "nonce": "abcdefg",
    "origin_jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "cognito:roles": [
        "arn:aws:iam::111122223333:role/my-test-role"
    ],
```

```
    "aud": "xxxxxxxxxxxxexample",
    "identities": [
        {
            "userId": "amzn1.account.EXAMPLE",
            "providerName": "LoginWithAmazon",
            "providerType": "LoginWithAmazon",
            "issuer": null,
            "primary": "true",
            "dateCreated": "1642699117273"
        }
    ],
    "event_id": "64f513be-32db-42b0-b78e-b02127b4f463",
    "token_use": "id",
    "auth_time": 1676312777,
    "exp": 1676316377,
    "iat": 1676312777,
    "jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "email": "my-test-user@example.com"
}
.<token signature>
```

sub

Eine eindeutige ID (UUID) oder ein Betreff für den authentifizierten Benutzer. Möglicherweise ist der Benutzername in Ihrem Benutzerpool nicht eindeutig. Der sub-Anspruch ist der beste Weg, um einen bestimmten Benutzer zu identifizieren.

cognito:groups

Ein Array mit den Namen von Benutzerpoolgruppen, zu denen Ihr Benutzer gehört. Gruppen können eine Kennung sein, die Sie Ihrer App präsentieren, oder sie können eine Anfrage für eine bevorzugte IAM-Rolle aus einem Identitätspool generieren.

cognito:preferred_role

Der ARN der IAM-Rolle, die Sie der Benutzerpoolgruppe mit der höchsten Priorität Ihres Benutzers zugeordnet haben. Weitere Informationen darüber, wie Ihr Benutzerpool diesen Rollenanspruch auswählt, finden Sie unter Zuweisen von Prioritätswerten zu Gruppen.

iss

Der Identitätsanbieter, der das Token ausgegeben hat. Der Anspruch hat das folgende Format:

`https://cognito-idp.<Region>.amazonaws.com/<your user pool ID>`

`cognito:username`

Der Benutzername Ihres Benutzers in Ihrem Benutzerpool.

`nonce`

Der `nonce`-Anspruch stammt von einem gleichnamigen Parameter, den Sie Anforderungen an Ihren OAuth 2.0-Endpunkt `authorize` hinzufügen können. Wenn Sie den Parameter hinzufügen, wird der `nonce`-Anspruch in das ID-Token eingeschlossen, das Amazon Cognito ausgibt, und Sie können ihn zum Schutz vor Wiederholungsangriffen verwenden. Wenn Sie in Ihrer Anfrage keinen `nonce`-Wert angeben, generiert und validiert Amazon Cognito automatisch eine Nonce, wenn Sie sich über eine Drittanbieter-Identität authentifizieren, und fügt sie dann als `nonce`-Anspruch zum ID-Token hinzu. Die Implementierung des `nonce`-Anspruchs in Amazon Cognito basiert auf OIDC-Standards.

`origin_jti`

Eine Token-Widerrufs-ID, die dem Aktualisierungstoken Ihres Benutzers zugeordnet ist. Amazon Cognito verweist auf den `origin_jti` Anspruch, wenn geprüft wird, ob Sie das Token Ihres Benutzers mit der Widerrufen des Endpunkts oder der RevokeTokenAPI-Operation gesperrt haben. Wenn Sie ein Token widerrufen, macht Amazon Cognito alle Zugriffs- und ID-Token mit demselben `origin_jti`-Wert ungültig.

`cognito:roles`

Ein Array mit den Namen der IAM-Rollen, die den Gruppen Ihres Benutzers zugeordnet sind. Jeder Benutzerpoolgruppe kann eine einzelne IAM-Rolle zugeordnet werden. Dieses Array stellt alle IAM-Rollen für die Gruppen Ihrer Benutzer dar, unabhängig von ihrer Rangfolge. Weitere Informationen finden Sie unter Hinzufügen von Gruppen zu einem Benutzerpool.

`aud`

Der Benutzerpool-App-Client, der Ihren Benutzer authentifiziert hat. Amazon Cognito gibt den gleichen Wert im Zugriffstoken-Anspruch `client_id` wieder.

`identities`

Der Inhalt des Benutzerattributs `identities`. Das Attribut enthält Informationen über jedes Profil eines externen Identitätsanbieters, das Sie mit einem Benutzer verknüpft haben, entweder durch Verbundanmeldung oder durch Verknüpfen eines Verbundbenutzers mit einem lokalen Profil. Diese Informationen enthalten ihren Anbieternamen, ihre eindeutige Anbieter-ID und andere Metadaten.

`token_use`

Der vorgesehene Zweck des Tokens. In einem ID-Token ist der Wert `id`.

`auth_time`

Der Authentifizierungszeitpunkt im Unix-Zeitformat, an dem Ihr Benutzer die Authentifizierung abgeschlossen hat.

`exp`

Der Ablaufzeitpunkt im Unix-Zeitformat, an dem das Token Ihres Benutzers abläuft.

`iat`

Der Zeitpunkt, an dem Amazon Cognito das Token Ihres Benutzers ausgegeben hat, im Unix-Zeitformat.

`jti`

Die eindeutige Kennung des JWT.

Das ID-Token kann OIDC-Standardansprüche enthalten, die in OIDC-Standardansprüchen definiert sind. Das ID-Token kann auch benutzerdefinierte Attribute enthalten, die Sie in Ihrem Benutzerpool definieren. Amazon Cognito schreibt benutzerdefinierte Attributwerte unabhängig vom Attributtyp als Strings in das ID-Token.

> **Note**
>
> Den benutzerdefinierten Attributen des Benutzerpools wird immer ein Präfix vorangestellt. `custom:`

ID-Token-Signatur

Die Signatur des ID-Tokens wird basierend auf dem Header und der Nutzlast des JWT-Tokens berechnet. Bevor Sie die Ansprüche in einem von Ihrer App empfangenen ID-Token akzeptieren, überprüfen Sie die Signatur des Tokens. Weitere Informationen finden Sie unter Verifizieren eines JSON-Webtokens. Verifizieren eines JSON-Web-Tokens.

Verwenden des Zugriffstokens

Das Zugriffstoken für Benutzerpools enthält Ansprüche zum authentifizierten Benutzer, eine Liste mit den Gruppen des Benutzers und eine Liste mit Bereichen. Der primäre Zweck des Zugriffstokens ist die Autorisierung von API-Operationen. Ihr Benutzerpool akzeptiert Zugriffstokens, um Self-Service-Operationen für Benutzer zu autorisieren. Beispielsweise können Sie das Zugriffstoken verwenden, um Ihrem Benutzer Zugriff zu erteilen, um Benutzerattribute hinzuzufügen, zu ändern oder zu löschen.

Mit OAuth-2.0-Bereichen in einem Zugriffstoken, das von den benutzerdefinierten Bereichen abgeleitet wird, die Sie Ihrem Benutzerpool hinzufügen, können Sie Ihren Benutzer autorisieren, Informationen von einer API abzurufen. Beispielsweise unterstützt Amazon API Gateway die Autorisierung mit Amazon-Cognito-Zugriffstokens. Sie können einen REST-API-Autorisierer mit Informationen aus Ihrem Benutzerpool befüllen oder Amazon Cognito als JSON-Web-Token-Autorisierer (JWT) für eine HTTP-API verwenden. Um ein Zugriffstoken mit benutzerdefinierten Bereichen zu generieren, müssen Sie es über die öffentlichen Endpunkte Ihres Benutzerpools anfordern.

Das Zugriffstoken Ihres Benutzers ist die Erlaubnis, weitere Informationen zu den Attributen Ihres Benutzers vom UserInfo-Endpunkt anzufordern. Das Zugriffstoken Ihres Benutzers ist auch die Berechtigung, Benutzerattribute zu lesen und zu schreiben. Die Zugriffsebene auf Attribute, die Ihr Zugriffstoken gewährt, hängt von den Berechtigungen ab, die Sie Ihrem App-Client zuweisen, und von den Bereichen, die Sie im Token gewähren.

Das Zugriffstoken ist ein JSON Web Token (JWT). Der Header für das Zugriffstoken hat die gleiche Struktur wie das ID-Token. Amazon Cognito signiert Zugriffstokens mit einem anderen Schlüssel als dem Schlüssel, der ID-Tokens signiert. Der Wert eines Zugriffsschlüssel-ID-Anspruchs (`kid`) entspricht nicht dem Wert des `kid`-Anspruchs in einem ID-Token aus derselben Benutzersitzung. Verifizieren Sie in Ihrem App-Code unabhängig voneinander ID-Tokens und Zugriffstokens. Vertrauen Sie den Ansprüchen in einem Zugriffstoken erst, wenn Sie die Signatur verifiziert haben. Weitere Informationen finden Sie unter Verifizieren eines JSON-Web-Tokens. Sie können den Ablauf des Zugriffstokens auf einen beliebigen Wert zwischen 5 Minuten und 1 Tag festlegen. Sie können diesen Wert pro App-Client festlegen.

> **Important**
>
> Geben Sie für Zugriffs- und ID-Tokens mindestens eine Stunde an, wenn Sie die gehostete Benutzeroberfläche verwenden. Amazon Cognito HostedUI verwendet Cookies, die eine

Stunde lang gültig sind. Wenn Sie mindestens weniger als eine Stunde angeben, erhalten Sie keine niedrigere Ablaufzeit.

Zugriffstoken-Header

Der Header enthält zwei verschiedene Informationen: die Schlüssel-ID (`kid`) und den Algorithmus (`alg`).

```
{
"kid" : "1234example="
"alg" : "RS256",
}
```

`kid`

Die Schlüssel-ID. Ihr Wert ist ein Hinweis darauf, welcher Schlüssel verwendet wurde, um die JSON-Websignatur (JWS) des Tokens zu sichern. Sie können die Signaturschlüssel-IDs Ihres Benutzerpools am `jwks_uri`-Endpunkt einsehen.

Weitere Informationen über den `kid`-Parameter finden Sie unter dem Schlüssel-ID-(kid)-Header-Parameter.

`alg`

Der kryptografische Algorithmus, mit dem Amazon Cognito das Zugriffstoken sichert. Benutzerpools verwenden einen kryptografischen RS256-Algorithmus, eine RSA-Signatur mit SHA-256.

Weitere Informationen über den `alg`-Parameter finden Sie unter dem Algorithmus-(alg)-Header-Parameter.

Standard-Payload für Zugriffstoken

Dies ist eine Beispielnutzlast eines Zugriffstokens. Weitere Informationen finden Sie unter JWT-Ansprüche. Sie können Ansprüche Ihres eigenen Designs mit einem Lambda-Auslöser für die Vorab-Generierung von Token hinzufügen.

```
<header>.
{
```

```
   "sub":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "device_key": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "cognito:groups":[
      "testgroup"
   ],
   "iss":"https://cognito-idp.us-west-2.amazonaws.com/us-west-2_example",
   "version":2,
   "client_id":"xxxxxxxxxxxxexample",
   "origin_jti":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "event_id":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "token_use":"access",
   "scope":"phone openid profile resourceserver.1/appclient2 email",
   "auth_time":1676313851,
   "exp":1676317451,
   "iat":1676313851,
   "jti":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "username":"my-test-user"
}
.<token signature>
```

`sub`

Eine eindeutige ID (UUID) oder ein Betreff für den authentifizierten Benutzer. Möglicherweise ist der Benutzername in Ihrem Benutzerpool nicht eindeutig. Der `sub`-Anspruch ist der beste Weg, um einen bestimmten Benutzer zu identifizieren.

`cognito:groups`

Ein Array mit den Namen von Benutzerpoolgruppen, zu denen Ihr Benutzer gehört.

`iss`

Der Identitätsanbieter, der das Token ausgegeben hat. Der Anspruch hat das folgende Format:

`https://cognito-idp.<Region>.amazonaws.com/<your user pool ID>`

`client_id`

Der Benutzerpool-App-Client, der Ihren Benutzer authentifiziert hat. Amazon Cognito gibt den gleichen Wert im ID-Token-Anspruch `aud` wieder.

`origin_jti`

Eine Token-Widerrufs-ID, die dem Aktualisierungstoken Ihres Benutzers zugeordnet ist. Amazon Cognito verweist auf den `origin_jti` Anspruch, wenn geprüft wird, ob Sie das Token Ihres

Benutzers mit der Widerrufen des Endpunkts oder der RevokeTokenAPI-Operation gesperrt haben. Wenn Sie ein Token widerrufen, macht Amazon Cognito alle Zugriffs- und ID-Token mit demselben `origin_jti`-Wert ungültig.

`token_use`

Der vorgesehene Zweck des Tokens. In einem Zugriffstoken ist der Wert `access`.

`scope`

Eine Liste mit OAuth 2.0-Bereichen, die definieren, welcher Zugriff mit dem Token gewährt wird. Ein Token aus dem Token-Endpunkt kann alle Bereiche enthalten, die Ihr App-Client unterstützt. Ein Token aus der API-Anmeldung in Amazon Cognito enthält nur den Bereich `aws.cognito.signin.user.admin`.

`auth_time`

Der Authentifizierungszeitpunkt im Unix-Zeitformat, an dem Ihr Benutzer die Authentifizierung abgeschlossen hat.

`exp`

Der Ablaufzeitpunkt im Unix-Zeitformat, an dem das Token Ihres Benutzers abläuft.

`iat`

Der Zeitpunkt, an dem Amazon Cognito das Token Ihres Benutzers ausgegeben hat, im Unix-Zeitformat.

`jti`

Die eindeutige Kennung des JWT.

`username`

Der Benutzername Ihres Benutzers in Ihrem Benutzerpool.

Zugriffstoken-Signatur

Die Signatur des Zugriffstokens wird basierend auf dem Header und der Nutzlast des JWT-Tokens berechnet. Bei Verwendung außerhalb einer Anwendung in Ihren Web-APIs müssen Sie diese Signatur vor der Akzeptanz des ID-Tokens immer überprüfen. Weitere Informationen finden Sie unter Verifizieren eines JSON-Web-Tokens.

Verwenden des Aktualisierungs-Tokens

Sie können das Aktualisierungstoken verwenden, um neue ID- und Zugriffstoken abzurufen. Standardmäßig läuft das Aktualisierungstoken 30 Tage, nachdem sich Ihr Anwendungs-Benutzer an Ihrem Benutzerpool angemeldet hat, ab. Wenn Sie eine Anwendung für Ihren Benutzerpool erstellen, können Sie den Ablauf des Aktualisierungstokens der Anwendung auf einen beliebigen Wert zwischen 60 Minuten und 10 Jahren setzen.

Das Mobile SDK for iOS, Mobile SDK for Android, Amplify für iOS, Android und Flutter aktualisieren automatisch Ihre ID und Zugriffstoken, wenn ein gültiges (nicht abgelaufenes) Aktualisierungstoken vorhanden ist. Die ID und Zugriffstoken haben eine verbleibende Mindestgültigkeit von 2 Minuten. Wenn das Aktualisierungstoken abgelaufen ist, muss sich Ihr App-Benutzer neu authentifizieren, indem er sich erneut bei Ihrem Benutzerpool anmeldet. Wenn das Minimum für das Zugriffstoken und das ID-Token auf 5 Minuten festgelegt ist und Sie das SDK verwenden, wird das Aktualisierungstoken kontinuierlich aktualisiert, um auf neue Zugriffs- und ID-Token zuzugreifen. Um das erwartete Verhalten anzuzeigen, legen Sie mindestens 7 Minuten statt 5 Minuten fest.

Das Benutzerkonto selbst ist zeitlich unbegrenzt, solange sich der Benutzer mindestens einmal vor Ablauf des Zeitraums von `UnusedAccountValidityDays` für neue Konten angemeldet hat.

Abrufen neuer Zugriffs- und Identitätstoken mit einem Aktualisierungstoken

Verwenden Sie die API oder die gehostete Benutzeroberfläche, um die Authentifizierung für Aktualisierungstoken zu initiieren.

Um das Aktualisierungstoken zu verwenden, um neue IDs und Zugriffstoken mit der Benutzerpools-API abzurufen, verwenden Sie die AdminInitiateAuthInitiateAuthAPI-Operationen oder. Übergeben Sie `REFRESH_TOKEN_AUTH` für den `AuthFlow`-Parameter. Übergeben Sie das Aktualisierungstoken des Benutzers in der `AuthFlow`-Eigenschaft `AuthParameters` als Wert von `"REFRESH_TOKEN"`. Amazon Cognito gibt neue ID- und Zugriffstoken zurück, nachdem Ihre API-Anfrage alle Herausforderungen bestanden hat.

> **Note**
>
> Um die Benutzerpools-API von Amazon Cognito zum Aktualisieren von Token für Benutzer einer gehosteten Benutzeroberfläche zu verwenden, generieren Sie eine `InitiateAuth`-Anfrage.

Sie können Aktualisierungstoken auch an den Token-Endpunkt in einem Benutzerpool übermitteln, in dem Sie eine Domain konfiguriert haben. Fügen Sie in den Anfragetext den `grant_type`-Wert `refresh_token` und den `refresh_token`-Wert des Aktualisierungstokens des Benutzers ein.

Widerrufen von Aktualisierungstokens

Sie können Aktualisierungstokens widerrufen, die einem Benutzer gehören. Weitere Informationen über Token finden Sie unter Widerrufen von Token.

> **Note**
>
> Durch das Widerrufen des Aktualisierungstokens werden alle ID- und Zugriffstoken widerrufen, die Amazon Cognito aus Aktualisierungsanfragen mit diesem Token ausgestellt hat.

Durch Verwendung der API-Operationen `GlobalSignOut` und `AdminUserGlobalSignOut` können Benutzer sich bei allen Geräten abmelden, bei denen sie aktuell angemeldet sind, wenn Sie alle Token eines Benutzers widerrufen. Die Abmeldung eines Benutzers hat folgende Auswirkungen.

- Mit dem Aktualisierungstoken des Benutzers können keine neuen Token für den Benutzer abgerufen werden.
- Mit dem Zugriffstoken des Benutzers können keine über Token autorisierte API-Anforderungen gesendet werden.
- Der Benutzer muss sich erneut authentifizieren, um neue Tokens zu erhalten. Da Sitzungscookies für gehostete Benutzeroberflächen nicht automatisch ablaufen, können Benutzer sich mit einem Sitzungscookie erneut authentifizieren, ohne die Anmeldeinformationen noch einmal eingeben zu müssen. Nachdem Sie die Benutzer der gehosteten Benutzeroberfläche abgemeldet haben, leiten Sie sie an den Logout-Endpunkt weiter, damit Amazon Cognito das Sitzungscookie löscht.

Mit Aktualisierungstoken können Sie Benutzersitzungen in Ihrer App für eine lange Zeit aufrechterhalten. Im Laufe der Zeit möchten Benutzer möglicherweise die Autorisierung für einige Geräte, auf denen sie sich angemeldet haben, aufheben und ihre Sitzung kontinuierlich aktualisieren. Wenn Sie einen Benutzer von einem einzelnen Gerät abmelden möchten, widerrufen Sie sein Aktualisierungstoken. Wenn sich Ihr Benutzer von allen authentifizierten Sitzungen abmelden möchte, generieren Sie eine GlobalSignOutAPI-Anfrage. Ihre App kann dem Benutzer eine Auswahl wie Von

allen Geräten abmelden bieten. `GlobalSignOut` akzeptiert das gültige – unveränderte, nicht abgelaufene, nicht widerrufene – Zugriffstoken eines Benutzers. Da diese API über Token autorisiert ist, können Benutzer sie nicht verwenden, um die Abmeldung für andere Benutzer zu initiieren.

Sie können jedoch eine AdminUserGlobalSignOutAPI-Anfrage generieren, die Sie mit Ihren AWS Anmeldeinformationen autorisieren, um jeden Benutzer von all seinen Geräten abzumelden. Die Administratoranwendung muss diesen API-Vorgang mit AWS Entwickleranmeldedaten aufrufen und die Benutzerpool-ID und den Benutzernamen des Benutzers als Parameter übergeben. Die `AdminUserGlobalSignOut`-API kann alle Benutzer vom Benutzerpool abmelden.

Weitere Informationen zu Anfragen, die Sie entweder mit AWS Anmeldeinformationen oder dem Zugriffstoken eines Benutzers autorisieren können, finden Sie unterAuthentifizierte und nicht authentifizierte API-Operationen der Amazon-Cognito-Benutzerpools.

Widerrufen von Token

Sie können ein Aktualisierungstoken für einen Benutzer mithilfe der AWS API widerrufen. Wenn Sie ein Aktualisierungstoken widerrufen, werden alle Zugriffstoken, die zuvor von diesem Aktualisierungstoken ausgegeben wurden, ungültig. Die anderen Aktualisierungstoken, die an den Benutzer ausgegeben wurden, sind nicht betroffen.

> **Note**
>
> JWT-Token sind eigenständig mit einer Signatur und einer Ablaufzeit, die beim Erstellen des Token zugewiesen wurde. Widerrufene Token können nicht mit Amazon-Cognito-API-Aufrufen verwendet werden, die ein Token erfordern. Widerrufene Token sind jedoch weiterhin gültig, wenn sie mit einer beliebigen JWT-Bibliothek verifiziert werden, die die Signatur und den Ablauf des Tokens verifiziert.

Sie können ein Aktualisierungstoken für einen Benutzerpoolclient mit aktiviertem Tokenwiderruf widerrufen. Wenn Sie einen neuen Benutzerpool-Client erstellen, ist der Tokenwiderruf standardmäßig aktiviert.

Tokenwiderruf aktivieren

Bevor Sie ein Token für einen vorhandenen Benutzerpool-Client widerrufen können, müssen Sie den Tokenwiderruf aktivieren. Sie können den Token-Widerruf für bestehende Benutzerpool-Clients mithilfe der AWS CLI oder der AWS API aktivieren. Rufen Sie dazu den `aws cognito-`

`idp describe-user-pool-client` CLI-Befehl oder die `DescribeUserPoolClient` API-Operation auf, um die aktuellen Einstellungen von Ihrem App-Client abzurufen. Dann rufen Sie den `aws cognito-idp update-user-pool-client` CLI-Befehl oder die `UpdateUserPoolClient` API-Operation auf. Fügen Sie die aktuellen Einstellungen von Ihrem App-Client hinzu und setzen Sie den Parameter `EnableTokenRevocation` auf `true`.

Wenn Sie einen neuen Benutzerpool-Client mithilfe der AWS Management Console, der AWS CLI oder der AWS API erstellen, ist der Token-Widerruf standardmäßig aktiviert.

Nachdem Sie den Token-Widerruf aktiviert haben, werden neue Anforderungen in den JSON-Web-Tokens von Amazon Cognito hinzugefügt. Die `origin_jti`- und `jti`-Ansprüche werden zu Zugriffs- und ID-Token hinzugefügt. Diese Ansprüche erhöhen die Größe des Anwendungsclient-Zugriffs und ID-Tokens.

Um einen App-Client mit aktiviertem Token-Widerruf zu erstellen oder zu ändern, fügen Sie den folgenden Parameter in Ihre CreateUserPoolClientoder Ihre UpdateUserPoolClientAPI-Anfrage ein.

```
"EnableTokenRevocation": true
```

Widerrufen eines Token

Sie können ein Aktualisierungstoken mithilfe einer RevokeTokenAPI-Anfrage widerrufen, beispielsweise mit dem `aws cognito-idp revoke-token` CLI-Befehl. Sie können Token auch mit dem Widerrufen des Endpunkts widerrufen. Dieser Endpunkt ist verfügbar, nachdem Sie Ihrem Benutzerpool eine Domäne hinzugefügt haben. Sie können den Widerrufsendpunkt entweder auf einer von Amazon Cognito gehosteten Domäne oder auf Ihrer eigenen benutzerdefinierten Domäne verwenden.

> **Note**
>
> Ihre Widerrufsanforderung für ein Aktualisierungstoken muss dieselbe Client-ID beinhalten, die zum Abrufen des Tokens verwendet wurde.

Es folgt ein Beispiel für eine `RevokeToken`-API-Anforderung.

```
{
   "ClientId": "1example23456789",
   "ClientSecret": "abcdef123456789ghijklexample",
   "Token": "eyJjdHkiOiJKV1QiEXAMPLE"
```

```
}
```

Es folgt ein Beispiel für eine cURL-Anforderung an den Endpunkt `/oauth2/revoke` eines Benutzerpools mit einer benutzerdefinierten Domain.

```
curl --location 'auth.mydomain.com/oauth2/revoke' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--header 'Authorization: Basic Base64Encode(client_id:client_secret)' \
--data-urlencode 'token=abcdef123456789ghijklexample' \
--data-urlencode 'client_id=1example23456789'
```

Die Operation `RevokeToken` und der Endpunkt `/oauth2/revoke` erfordern keine zusätzliche Autorisierung, es sei denn, Ihr App-Client verfügt über einen geheimen Client-Schlüssel.

Verifizieren eines JSON-Web-Tokens

Diese Schritte beschreiben die Verifizierung eines Benutzerpool-JSON-Web-Tokens (JWT).

Themen

- Voraussetzungen
- Bestätigen Sie Token mit aws-jwt-verify
- Tokens verstehen und überprüfen

Voraussetzungen

Ihre Bibliothek, Ihr SDK oder Ihr Software-Framework erledigt möglicherweise bereits Aufgaben in diesem Abschnitt. AWS SDKs bieten Tools für die Handhabung und Verwaltung von Amazon Cognito Cognito-Benutzerpool-Tokens in Ihrer App. AWS Amplify beinhaltet Funktionen zum Abrufen und Aktualisieren von Amazon Cognito Cognito-Token.

Weitere Informationen finden Sie auf den folgenden Seiten.

- Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps
- Codebeispiele für Amazon Cognito Identity Provider mit AWS SDKs
- Erweiterte Workflows im Amplify Dev Center

Viele Bibliotheken sind zum Decodieren und Verifizieren eines JSON Web Token (JWT) verfügbar. Wenn Sie Tokens für die serverseitige API-Verarbeitung manuell verarbeiten müssen, oder wenn

Sie andere Programmiersprachen verwenden, können diese Bibliotheken hilfreich sein. Weitere Informationen finden Sie in der Liste der OpenID Foundation mit Bibliotheken für die Arbeit mit JWT-Token.

Bestätigen Sie Token mit aws-jwt-verify

AWS Empfiehlt der aws-jwt-verifyBibliothek in einer App von Node.js, die Parameter im Token zu validieren, das Ihr Benutzer an Ihre App übergibt. Mit `aws-jwt-verify` können Sie einen `CognitoJwtVerifier` mit den Anspruchswerten auffüllen, die Sie für einen oder mehrere Benutzerpools überprüfen möchten. Zu den Werten, die überprüft werden können, gehören, dass

- die Zugangs- oder ID-Tokens nicht falsch formatiert oder abgelaufen sind und eine gültige Signatur haben;
- die Zugriffstokens von den korrekten Benutzerpools und App-Clients kommen;
- die Zugriffstokenansprüche die korrekten OAuth-2.0-Bereiche enthalten;
- die Schlüssel, die Ihre Zugangs- und ID-Tokens signiert haben, mit einem `kid`-Signaturschlüssel aus der JWKS-URI Ihrer Benutzerpools übereinstimmen.

  Die JWKS-URI enthält öffentliche Informationen über den privaten Schlüssel, mit dem das Token Ihres Benutzers signiert wurde. Die JWKS-URI für Ihren Benutzerpool finden Sie unter `https://cognito-idp.<Region>.amazonaws.com/<userPoolId>/.well-known/jwks.json`.

Weitere Informationen und Beispielcode, den Sie in einer Node.js -App oder einem AWS Lambda Authorizer verwenden können, finden Sie aws-jwt-verifyunter GitHub.

Tokens verstehen und überprüfen

Bevor Sie die Token-Inspektion in Ihre App integrieren, sollten Sie sich überlegen, wie Amazon Cognito JWTs zusammenstellt. Rufen Sie Beispiel-Tokens aus Ihrem Benutzerpool ab. Dekodieren und untersuchen Sie diese im Detail, um ihre Eigenschaften zu verstehen, und legen Sie fest, was Sie wann überprüfen möchten. So können Sie beispielsweise in einem Szenario die Gruppenmitgliedschaft und in einem anderen Bereiche überprüfen.

In den folgenden Abschnitten wird ein Prozess beschrieben, mit dem Sie Amazon Cognito JWTs bei der Vorbereitung Ihrer App manuell überprüfen können.

Bestätigen der Struktur des JWT

Ein JSON-Web-Token (JWT) enthält drei Abschnitte mit einem `.` (Punkt)-Zeichen dazwischen.

Header

Die Schlüssel-ID, `kid`, und der RSA-Algorithmus, `alg`, mit denen Amazon Cognito das Token signiert hat. Amazon Cognito signiert Tokens mit einem `alg` von `RS256`.

Nutzlast

Token-Ansprüche. In einem ID-Token enthalten die Ansprüche Benutzerattribute und Informationen über den Benutzerpool, `iss`, und den App-Client, `aud`. In einem Zugriffstoken umfasst die Payload Bereiche, Gruppenmitgliedschaft, Ihren Benutzerpool als `iss` und Ihren App-Client als `client_id`.

Signatur

Die Signatur ist nicht base64-dekodierbar wie der Header und die Payload. Es handelt sich um eine RSA256-ID, die aus einem Signaturschlüssel und Parametern abgeleitet ist, die Sie auf Ihrer JWKS-URI sehen können.

Der Header und die Payload sind base64-kodiertes JSON. Sie können sie anhand der Zeichen `eyJ` am Anfang erkennen, die zum Startzeichen `{` dekodiert werden. Wenn Ihr Benutzer Ihrer App ein base64-kodiertes JWT präsentiert und dies nicht das Format `[JSON Header].[JSON Payload].[Signature]` hat, ist es kein gültiges Amazon-Cognito-Token und Sie können es verwerfen.

Überprüfen des JWT

Die JWT-Signatur ist eine gehashte Kombination aus Header und Nutzlast. Amazon Cognito generiert zwei Paare RSA-Kryptoschlüssel für jeden Benutzerpool. Ein privater Schlüssel signiert Zugriffstokens und der andere signiert ID-Tokens.

Verifizieren der Signatur eines JWT-Tokens

1. Dekodieren Sie das ID-Token.

   Die OpenID Foundation pflegt auch eine Liste mit Bibliotheken für die Arbeit mit JWT-Token.

   Sie können es auch verwenden, AWS Lambda um Benutzerpool-JWTs zu dekodieren. Weitere Informationen finden Sie unter Amazon Cognito JWT-Token dekodieren und verifizieren mithilfe von. AWS Lambda
2. Vergleichen Sie die lokale Schlüssel-ID (`kid`) mit der öffentlichen `kid`.

a. Laden Sie den entsprechenden JWK (JSON Web Key) für Ihren Benutzerpool herunter und speichern Sie ihn. Er ist als Teil eines JWKS (JSON Web Key Set) verfügbar. Sie können ihn finden, indem Sie die folgende `jwks_uri`-URI für Ihre Umgebung konstruieren:

```
https://cognito-idp.<Region>.amazonaws.com/<userPoolId>/.well-known/jwks.json
```

Weitere Informationen zu JWK- und JWK-Sets finden Sie unter JSON Web Key (JWK).

> **Note**
>
> Amazon Cognito rotiert möglicherweise die Signaturschlüssel in Ihrem Benutzerpool. Es hat sich bewährt, öffentliche Schlüssel in Ihrer App zwischenzuspeichern, indem Sie `kid` als Cache-Schlüssel verwenden und den Cache regelmäßig aktualisieren. Vergleichen Sie die `kid` in den Tokens, die Ihre App erhält, mit Ihrem Cache. Wenn Sie ein Token mit dem korrekten Aussteller, aber einer anderen `kid` erhalten, hat Amazon Cognito möglicherweise den Signaturschlüssel rotiert. Aktualisieren Sie den Cache von Ihrem Benutzerpool–`jwks_uri`-Endpunkt aus.

Dies ist eine `jwks.json`-Beispieldatei:

```
{
 "keys": [{
  "kid": "1234example=",
  "alg": "RS256",
  "kty": "RSA",
  "e": "AQAB",
  "n": "1234567890",
  "use": "sig"
 }, {
  "kid": "5678example=",
  "alg": "RS256",
  "kty": "RSA",
  "e": "AQAB",
  "n": "987654321",
  "use": "sig"
 }]
}
```

Schlüssel-ID (**`kid`**)

Der `kid`-Parameter ist ein Hinweis darauf, welcher Schlüssel verwendet wurde, um die JSON-Websignatur (JWS) des Tokens zu sichern.

Algorithmus (**`alg`**)

Der `alg`-Header-Parameter stellt den kryptografischen Algorithmus dar, mit dem das ID-Token gesichert wird. Benutzerpools verwenden einen kryptografischen RS256-Algorithmus, eine RSA-Signatur mit SHA-256. Weitere Informationen zu RSA finden Sie unter RSA-Kryptografie.

Schlüsseltyp (**`kty`**)

Der `kty`-Parameter identifiziert die kryptografischen Algorithmus-Familie, die mit dem Schlüssel verwendet wird, z. B. „RSA“ in diesem Beispiel.

RSA-Exponent (**`e`**)

Der e-Parameter enthält den Exponentenwert für den öffentlichen RSA-Schlüssel. Er wird als Base64urlUInt-kodierten Wert dargestellt.

RSA-Modulo (**`n`**)

Der n-Parameter enthält den Modulo-Wert für den öffentlichen RSA-Schlüssel. Er wird als Base64urlUInt-kodierten Wert dargestellt.

Verwenden von (**`use`**)

Der `use`-Parameter beschreibt die beabsichtigte Verwendung des öffentlichen Schlüssels. In diesem Beispiel stellt der `use`-Wert `sig` die Signatur dar.

b. Durchsuchen Sie den öffentlichen JSON-Web-Schlüssel nach einer `kid` die mit der `kid` Ihres JWT übereinstimmt.

3. Verwenden Sie eine JWT-Bibliothek, um die Signatur des Ausstellers mit der Signatur im Token zu vergleichen. Die Signatur des Emittenten leitet sich aus dem öffentlichen Schlüssel (den RSA-Modulen `"n"`) des `kid` in jwks.json ab, der mit dem Token `kid` übereinstimmt. Möglicherweise müssen Sie den JWK zuerst in das PEM-Format konvertieren. Das folgende Beispiel benutzt das JWT und das JWK und verwendet die Node.js-Bibliothek, jsonwebtoken, um die JWT-Signatur zu überprüfen:

Node.js

```
var jwt = require('jsonwebtoken');
var jwkToPem = require('jwk-to-pem');
var pem = jwkToPem(jwk);
jwt.verify(token, pem, { algorithms: ['RS256'] }, function(err, decodedToken) {
});
```

Überprüfen der Ansprüche

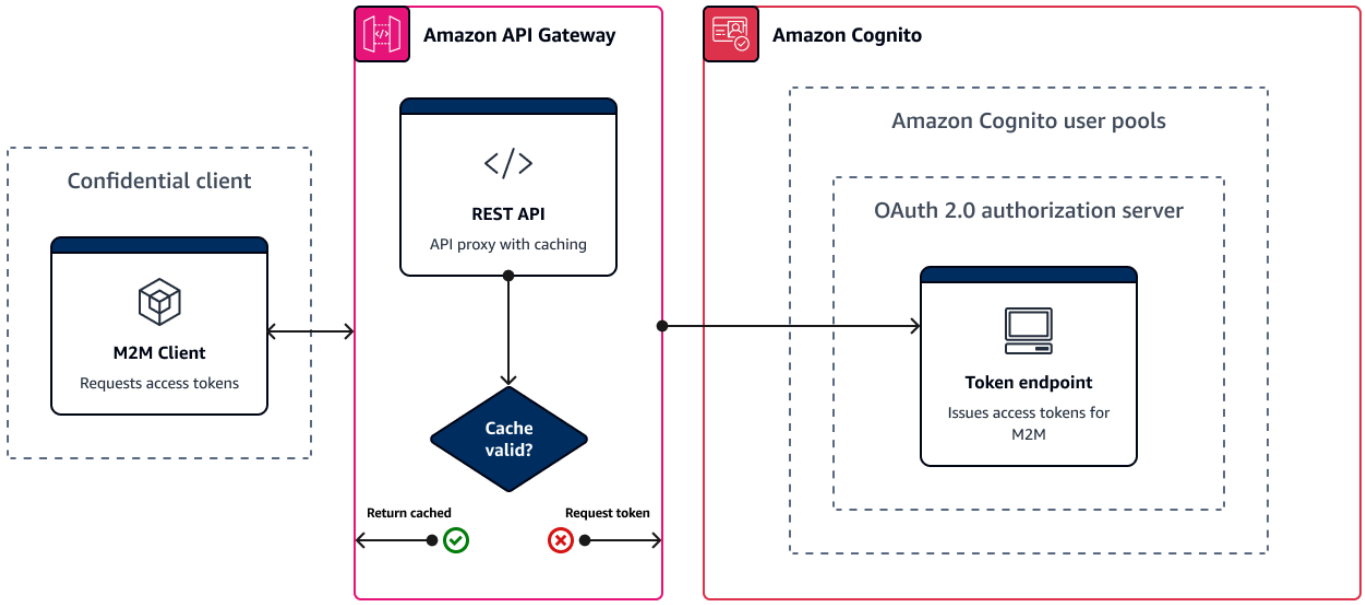
Überprüfen der JWT-Ansprüche

1. Stellen Sie mit einer der folgenden Methoden sicher, dass das Token nicht abgelaufen ist.
   a. Dekodieren Sie das Token und vergleichen Sie den `exp`-Anspruch mit der aktuellen Uhrzeit.
   b. Wenn Ihr Zugriffstoken einen `aws.cognito.signin.user.admin` Anspruch enthält, senden Sie eine Anfrage an eine API wie. GetUser API-Anfragen, die Sie mit einem Zugriffs-Token autorisieren, geben einen Fehler zurück, wenn Ihr Token abgelaufen ist.
   c. Präsentieren Sie Ihr Zugriffs-Token in einer Anfrage an den UserInfo-Endpunkt. Ihre Anfrage gibt einen Fehler zurück, wenn Ihr Token abgelaufen ist.
2. Der `aud`-Anspruch in einem ID-Token und der `client_id`-Anspruch in einem Zugriffstoken müssen mit der App-Client-ID übereinstimmen, die im Amazon-Cognito-Benutzerpool erstellt wurde.
3. Der Aussteller (`iss`)-Anspruch muss mit Ihrem Benutzerpool übereinstimmen. Ein in der `us-east-1`-Region erstellter Benutzerpool hat den folgenden `iss`-Wert:

   `https://cognito-idp.us-east-1.amazonaws.com/<userpoolID>`.
4. Prüfen Sie den `token_use`-Anspruch.
   - Wenn Sie nur das Zugriffstoken in den Web-API-Operationen akzeptieren, muss der Wert lauten `access`.
   - Verwenden Sie nur das ID-Token, muss der Wert sein `id`.
   - Wenn Sie ID- und Zugriffstoken verwenden, muss der `token_use`-Anspruch `id` oder `access` sein.

Jetzt können Sie den Ansprüchen innerhalb des Tokens vertrauen.

Zwischenspeicherung von Token



Ihre App muss jedes Mal, wenn Sie ein neues JSON Web Token (JWT) abrufen möchten, eine der folgenden Anforderungen erfolgreich abschließen.

- Fordern Sie Kundenanmeldeinformationen oder eine Autorisierungscode-Erteilung von Token-Endpunkt an.
- Fordern Sie eine implizite Erteilung von Ihrer gehosteten Benutzeroberfläche an.
- Authentifizieren Sie einen lokalen Benutzer in einer Amazon Cognito Cognito-API-Anfrage wie. InitiateAuth

Sie können Ihren Benutzerpool so konfigurieren, dass Tokens in Minuten, Stunden oder Tagen ablaufen. Zur Sicherstellung der Leistung und Verfügbarkeit Ihrer App verwenden Sie Amazon-Cognito-Token, bis sie ablaufen, und rufen Sie erst dann neue Token ab. Eine Zwischenspeicherungslösung, die Sie für Ihre App erstellen, hält Token verfügbar und verhindert die Ablehnung von Anfragen durch Amazon Cognito, wenn Ihre Anforderungsrate zu hoch ist. Eine clientseitige App muss Token in einem Speichercache ablegen. Eine serverseitige App kann einen verschlüsselten Cache-Mechanismus zum Speichern von Token hinzufügen.

Wenn Ihr Benutzerpool ein hohes Volumen an Benutzern oder machine-to-machine Aktivitäten generiert, stoßen Sie möglicherweise auf die Beschränkungen, die Amazon Cognito für

die Anzahl der Token-Anfragen festlegt, die Sie stellen können. Wenn Sie die Anzahl der Anfragen, die Sie an Amazon-Cognito-Endpunkte stellen, reduzieren möchten, können Sie Authentifizierungsdaten entweder sicher speichern und wiederverwenden oder exponentielle Backoffs und Wiederholungsversuche implementieren.

Authentifizierungsdaten stammen aus zwei Klassen von Endpunkten. OAuth-2.0-Endpunkte von Amazon Cognito umfassen den Tokenendpunkt, der Client-Anmeldeinformationen und Autorisierungscode-Anforderungen der gehosteten Benutzeroberfläche bereitstellt. Service-Endpunkte beantworten Benutzerpool-API-Anfragen wie `InitiateAuth` und `RespondToAuthChallenge`. Jede Art von Anfrage hat eigene Grenzen. Weitere Informationen zu Limits finden Sie unter Kontingente in Amazon Cognito.

Zwischenspeichern von machine-to-machine Zugriffstoken mit Amazon API Gateway

Mit dem Zwischenspeichern von API-Gateway-Token kann Ihre App bei Ereignissen abskalieren, die das Standardkontingent für die Anforderungsrate von OAuth-Endpunkte von Amazon Cognito überschreiten.

Sie können die Zugriffstoken zwischenspeichern, sodass Ihre App nur dann ein neues Zugriffstoken anfordert, wenn ein zwischengespeichertes Token abgelaufen ist. Andernfalls gibt Ihr Caching-Endpunkt ein Token aus dem Cache zurück. Dadurch wird ein zusätzlicher Aufruf eines Amazon-Cognito-API-Endpunkts verhindert. Wenn Sie Amazon API Gateway als Proxy für Token-Endpunkt verwenden, reagiert Ihre API auf die meisten Anfragen, die andernfalls zu Ihrem Anforderungskontingent beitragen würden, und vermeidet erfolglose Anfragen aufgrund der Kontingentbegrenzung.

Die folgende API-Gateway-basierte Lösung bietet eine Low-Code-/No-Code-Implementierung von Token-Caching mit niedriger Latenz. API-Gateway-APIs werden während der Übertragung und optional im Ruhezustand verschlüsselt. Ein API-Gateway-Cache ist ideal für die Gewährung von OAuth 2.0-Client-Anmeldeinformationen, eine häufig umfangreiche Art der Gewährung, die Zugriffstoken für Autorisierungs- und Microservice-Sitzungen generiert. machine-to-machine In einem Fall wie einem Anstieg des Datenverkehrs, der dazu führt, dass Ihre Microservices horizontal skaliert werden, kann es passieren, dass viele Systeme dieselben Client-Anmeldeinformationen verwenden, und zwar in einem Umfang, der die AWS Anforderungsratenbegrenzung Ihres Benutzerpools oder App-Clients überschreitet. Zur Erhaltung der Verfügbarkeit von Apps und einer geringen Latenz hat sich eine Caching-Lösung in solchen Szenarien als Methode bewährt.

In dieser Lösung definieren Sie einen Cache in Ihrer API, um ein separates Zugriffstoken für jede Kombination aus OAuth-Bereichen und App-Client zu speichern, die Sie in Ihrer App anfordern

möchten. Wenn Ihre App eine Anfrage stellt, die mit dem Cache-Schlüssel übereinstimmt, antwortet Ihre API mit einem Zugriffstoken, das Amazon Cognito aufgrund der ersten Anfrage ausgegeben hat, die mit dem Cache-Schlüssel übereinstimmt. Wenn die Dauer Ihres Cache-Schlüssels abläuft, leitet Ihre API die Anforderung an Ihren Tokenendpunkt weiter und speichert ein neues Zugriffstoken im Cache.

> **Note**
>
> Die Dauer Ihres Cache-Schlüssels muss kürzer sein als die Zugriffstokendauer Ihres App-Clients.

Der Cache-Schlüssel ist eine Kombination der OAuth-Bereiche, die Sie im URL-Parameter `scope` und im `Authorization`-Header der Anfrage anfordern. Der `Authorization`-Header enthält die App-Client-ID und den geheimen Client-Schlüssel. Sie müssen keine zusätzliche Logik in Ihrer App implementieren, um diese Lösung zu verwenden. Sie müssen Ihre Konfiguration nur aktualisieren, um den Pfad zu Ihrem Tokenendpunkt des Benutzerpools zu ändern.

Sie können Token-Caching auch mit for Redis implementieren. ElastiCache Für die differenzierte Steuerung mit AWS Identity and Access Management (IAM)-Richtlinien erwägen Sie die Verwendung eines Amazon-DynamoDB-Caches.

> **Note**
>
> Das Zwischenspeichern im API Gateway ist mit zusätzlichen Kosten verbunden. Weitere Informationen finden Sie unter Preise.

So richten Sie einen Caching-Proxy mit API Gateway ein

1. Öffnen Sie die API-Gateway-Konsole und erstellen Sie eine REST-API.
2. Erstellen Sie eine POST-Methode in Resources (Ressourcen).
   a. Wählen Sie HTTP als Integration type (Integrationstyp) aus.
   b. Wählen Sie Use HTTP proxy integration (HTTP-Proxy-Integration verwenden) aus.
   c. Geben Sie als Endpoint URL (Endpunkt-URL) `https://<your user pool domain>/oauth2/token` ein.
3. Konfigurieren Sie den Cache-Schlüssel in Resources (Ressourcen).

a. Bearbeiten Sie Method request (Methodenanforderung) Ihrer POST-Methode.

b. Legen Sie den `scope`-Parameter und den `Authorization`-Header als Caching-Schlüssel fest.

   i. Fügen Sie unter URL query string parameters (URL-Abfragezeichenfolgenparameter) eine Abfragezeichenfolge hinzu und wählen Sie Caching (Zwischenspeichern) für die `scope`-Zeichenfolge aus.

   ii. Fügen Sie unter HTTP request headers (HTTP-Anforderungsheader) einen Header hinzu und wählen Sie Caching für den `Authorization`-Header aus.

4. Konfigurieren Sie das Caching unter Stages (Phasen).

   a. Wählen Sie die Phase aus, die Sie ändern möchten.

   b. Wählen Sie unter Settings (Einstellungen) die Option Enable API cache (API-Cache aktivieren) aus.

   c. Wählen Sie einen Wert für Cache capacity (Cache-Kapazität) aus.

   d. Wählen Sie einen Cache time-to-live (TTL) von mindestens 3600 Sekunden.

   e. Deaktivieren Sie das Kontrollkästchen Autorisierung erforderlich.

5. Notieren Sie sich unter Stages (Phasen) die Invoke URL (Aufruf-URL).

6. Aktualisieren Sie Ihre App auf POST-Tokenanfragen an die Invoke URL (Aufruf-URL) Ihrer API anstatt an den `/oauth2/token`-Endpunkt Ihres Benutzerpools.

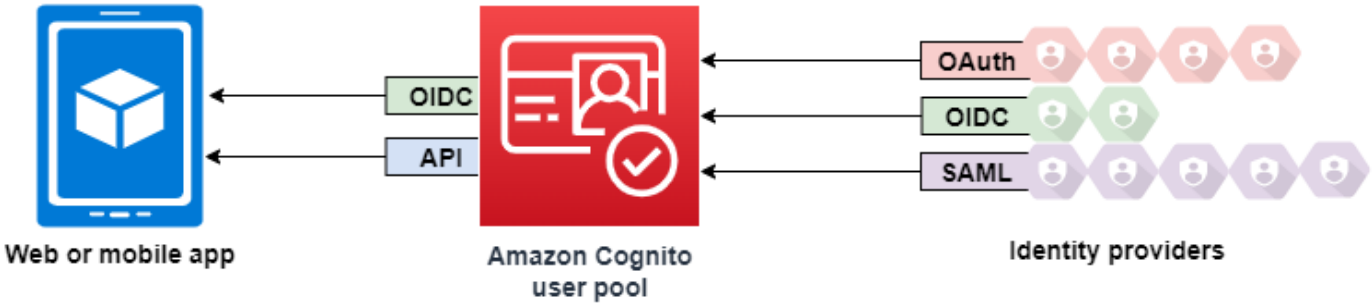
Zugriff auf Ressourcen nach einer erfolgreichen Benutzerpool-Authentifizierung

Ihre App-Benutzer können sich entweder direkt über einen Benutzerpool anmelden oder sich über einen externen Identitätsanbieter (IdP) zusammenschließen. Der Benutzerpool verwaltet den Aufwand für die Verwaltung der Token, die bei der Anmeldung in sozialen Netzwerken über Facebook, Google, Amazon und Apple sowie von OpenID Connect (OIDC) und SAML zurückgegeben werden. IdPs Weitere Informationen finden Sie unter Verwenden von Token mit Benutzerpools.

Nach einer erfolgreichen Authentifizierung erhält Ihre App Benutzerpool-Token von Amazon Cognito. Sie können Benutzerpool-Token verwenden, um:

- Rufen Sie AWS Anmeldeinformationen ab, die Anfragen nach Anwendungsressourcen AWS-Services wie Amazon DynamoDB und Amazon S3 autorisieren.
- Stellen Sie einen temporären, widerruflichen Authentifizierungsnachweis bereit.
- Füllen Sie Identitätsdaten in ein Benutzerprofil in Ihrer App ein.
- Autorisieren Sie Änderungen am Profil des angemeldeten Benutzers im Benutzerpoolverzeichnis.
- Autorisieren Sie Anfragen nach Benutzerinformationen mit einem Zugriffstoken.
- Autorisieren Sie Anfragen an Daten, die sich hinter zugriffsgeschützten externen APIs befinden, mit Zugriffstoken.
- Autorisieren Sie den Zugriff auf Anwendungsressourcen, die auf dem Client oder Server gespeichert sind, mit Amazon Verified Permissions.

Weitere Informationen finden Sie unter Ablauf der Authentifizierung in Benutzerpools und Verwenden von Token mit Benutzerpools.



Themen

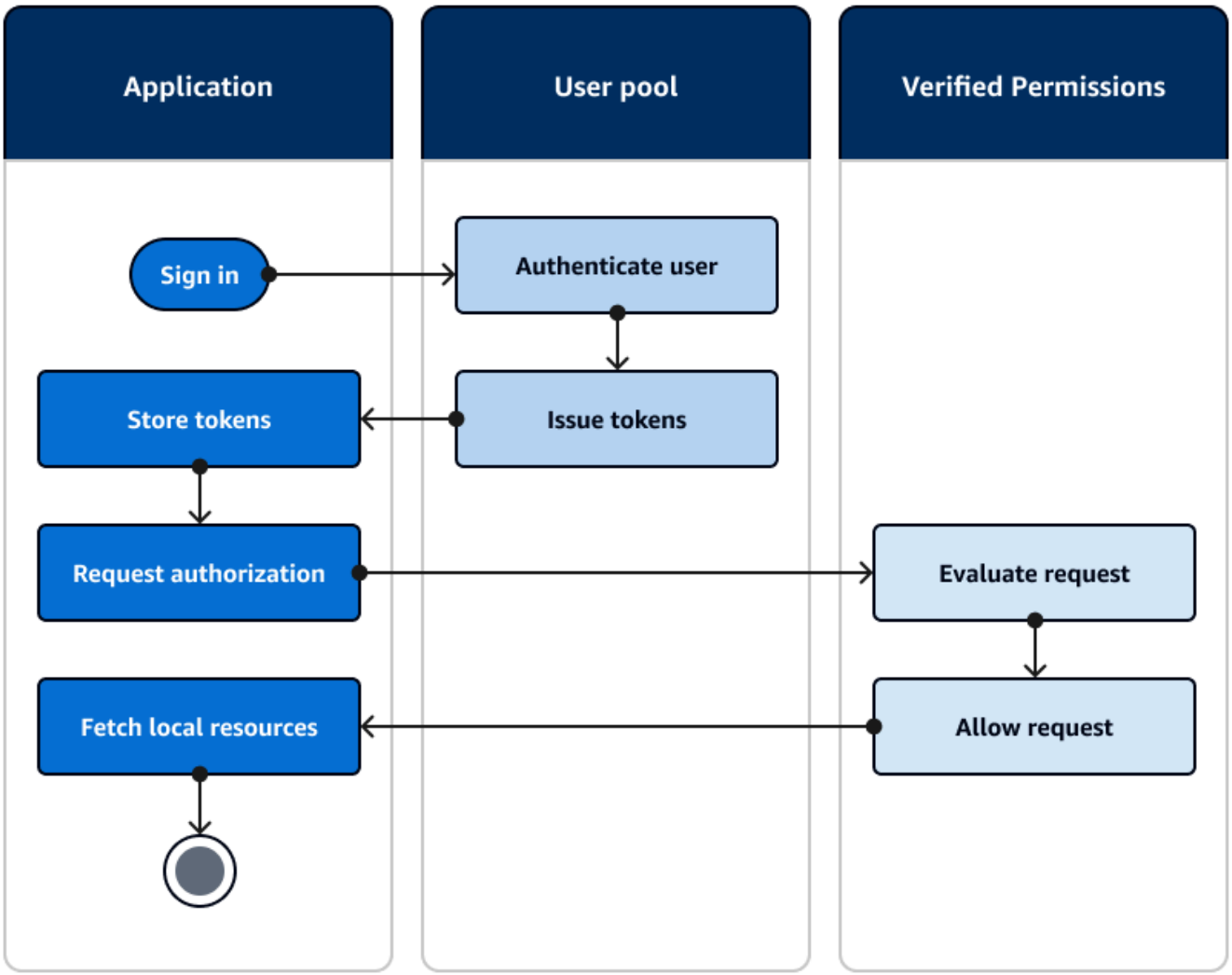
- Autorisieren des Zugriffs auf Client- oder Serverressourcen mit von Amazon Verified Permissions
- Zugriff auf Ressourcen mit API Gateway nach der Anmeldung
- Zugriff AWS-Services über einen Identitätspool nach der Anmeldung

Autorisieren des Zugriffs auf Client- oder Serverressourcen mit von Amazon Verified Permissions

Ihre App kann die Token von einem angemeldeten Benutzer an Amazon Verified Permissions weitergeben. Verified Permissions ist ein skalierbarer, detaillierter Service zur Verwaltung und Autorisierung von Berechtigungen für benutzerdefinierte Anwendungen, die Sie erstellt haben. Ein

Amazon Cognito Cognito-Benutzerpool kann eine Identitätsquelle für einen Richtlinienspeicher für verifizierte Berechtigungen sein. Verified Permissions trifft Autorisierungsentscheidungen für angeforderte Aktionen und Ressourcen, z. B. `GetPhoto` für`premium_badge.png`, vom Principal und seinen Attributen in Benutzerpool-Token.

Das folgende Diagramm zeigt, wie Ihre Anwendung das Token eines Benutzers in einer Autorisierungsanfrage an Verified Permissions übergeben kann.



Erste Schritte mit Amazon Verified Permissions

Nachdem Sie Ihren Benutzerpool mit Verified Permissions integriert haben, erhalten Sie eine zentrale Quelle für detaillierte Autorisierung für all Ihre Amazon Cognito-Apps. Dadurch entfällt die Notwendigkeit einer detaillierten Sicherheitslogik, die Sie sonst zwischen all Ihren Apps

programmieren und replizieren müssten. Weitere Informationen zur Autorisierung mit verifizierten Berechtigungen finden Sie unter. Autorisierung mit Amazon Verified Permissions

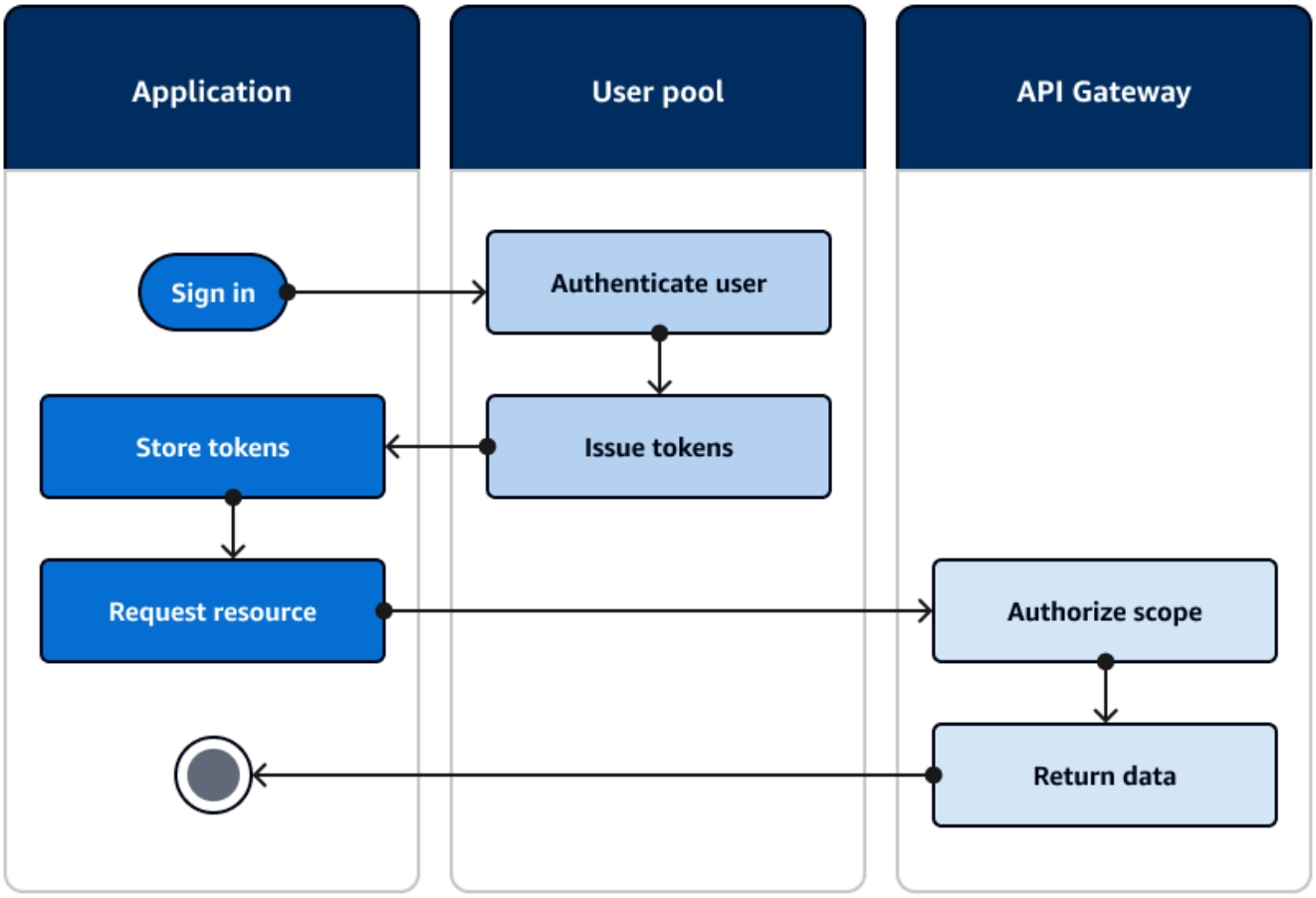
Für Autorisierungsanfragen mit verifizierten Berechtigungen sind AWS Anmeldeinformationen erforderlich. Sie können einige der folgenden Techniken implementieren, um Anmeldeinformationen sicher auf Autorisierungsanfragen anzuwenden.

- Betreiben Sie eine Webanwendung, die Geheimnisse im Server-Backend speichern kann.
- Besorgen Sie sich authentifizierte Anmeldeinformationen für den Identitätspool.
- Proxyieren Sie Benutzeranfragen über eine access-token-authorized API und fügen Sie der Anfrage AWS Anmeldeinformationen hinzu.

Zugriff auf Ressourcen mit API Gateway nach der Anmeldung

Amazon Cognito Cognito-Benutzerpools-Token werden häufig verwendet, um Anfragen an eine API Gateway zu autorisieren. Die OAuth 2.0-Bereiche in Zugriffstoken können eine Methode und einen Pfad autorisieren, z. B. für. `HTTP GET /app_assets` ID-Token können als generische Authentifizierung für eine API dienen und Benutzerattribute an den Back-End-Dienst übergeben. API Gateway bietet zusätzliche benutzerdefinierte Autorisierungsoptionen wie JWT-Autorisierer für HTTP-APIs und Lambda-Autorisierer, die eine detailliertere Logik anwenden können.

Das folgende Diagramm zeigt eine Anwendung, die Zugriff auf eine REST-API mit den OAuth 2.0-Bereichen in einem Zugriffstoken erhält.



Ihre App muss die Token aus authentifizierten Sitzungen sammeln und sie als Inhaber-Token zu einem `Authorization` Header in der Anfrage hinzufügen. Konfigurieren Sie den Authorizer, den Sie für die API, den Pfad und die Methode zur Auswertung von Token-Inhalten konfiguriert haben. API Gateway gibt nur Daten zurück, wenn die Anfrage den Bedingungen entspricht, die Sie für Ihren Autorisierer eingerichtet haben.

Einige mögliche Methoden, mit denen die API Gateway API den Zugriff von einer Anwendung aus genehmigen kann, sind:

- Das Zugriffstoken enthält den richtigen OAuth 2.0-Bereich. Der Amazon Cognito Cognito-Benutzerpool-Autorisierer für eine REST-API ist eine gängige Implementierung mit einer niedrigen Eintrittsbarriere. Sie können auch den Hauptteil, die Abfragezeichenfolgenparameter und die Header einer Anfrage an diesen Autorisierungstyp auswerten.

- Das ID-Token ist gültig und nicht abgelaufen. Wenn Sie ein ID-Token an einen Amazon Cognito Cognito-Autorisierer übergeben, können Sie eine zusätzliche Validierung des ID-Token-Inhalts auf Ihrem Anwendungsserver durchführen.
- Eine Gruppe, ein Anspruch, ein Attribut oder eine Rolle in einem Zugriffs- oder ID-Token erfüllt die Anforderungen, die Sie in einer Lambda-Funktion definieren. Ein Lambda-Autorisierer analysiert das Token im Anforderungsheader und wertet es für eine Autorisierungsentscheidung aus. Sie können in Ihrer Funktion eine benutzerdefinierte Logik erstellen oder eine API-Anfrage an Amazon Verified Permissions stellen.

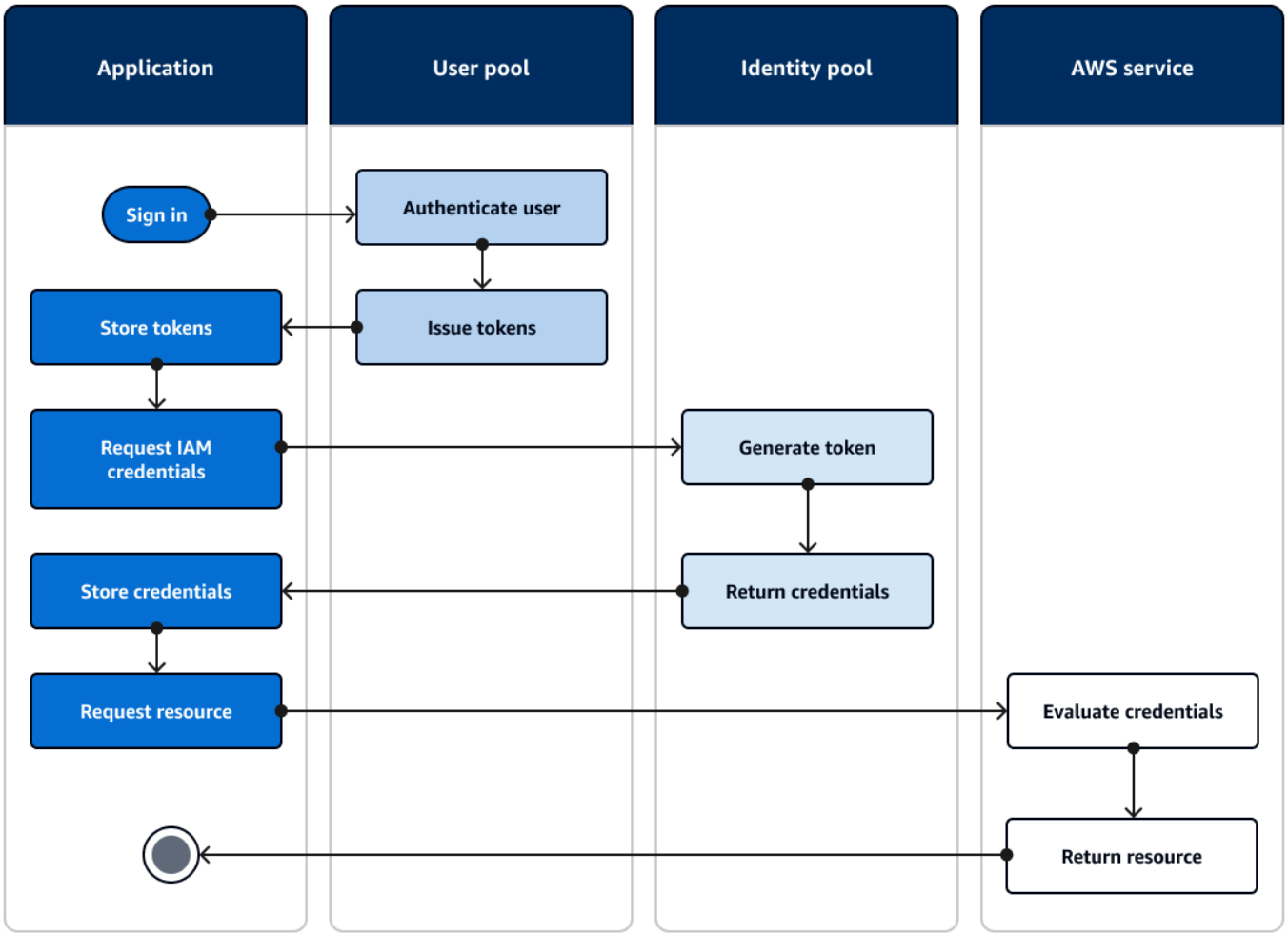
Sie können Anfragen an eine AWS AppSync GraphQL-API auch mit Tokens aus einem Benutzerpool autorisieren.

Zugriff AWS-Services über einen Identitätspool nach der Anmeldung

Nachdem sich Ihre Benutzer mit einem Benutzerpool angemeldet haben, können sie AWS-Services mit temporären API-Anmeldeinformationen, die von einem Identitätspool ausgestellt wurden, darauf zugreifen.

Ihre Web- oder Mobil-App erhält Token aus einem Benutzerpool. Wenn Sie Ihren Benutzerpool als Identitätsanbieter für Ihren Identitätspool konfigurieren, tauscht der Identitätspool Token gegen temporäre AWS Anmeldeinformationen aus. Diese Anmeldeinformationen können auf IAM-Rollen und deren Richtlinien beschränkt werden, die Benutzern Zugriff auf eine begrenzte Anzahl von Ressourcen gewähren. AWS Weitere Informationen finden Sie unter Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf.

Das folgende Diagramm zeigt, wie sich eine Anwendung mit einem Benutzerpool anmeldet, Anmeldeinformationen für den Identitätspool abruft und ein Asset von einem anfordert. AWS-Service



Sie können die Anmeldeinformationen für den Identitätspool verwenden, um:

- Stellen Sie detaillierte Autorisierungsanfragen an Amazon Verified Permissions mit den eigenen Anmeldeinformationen Ihres Benutzers.
- Stellen Sie eine Connect zu einer Amazon API Gateway Gateway-REST-API oder einer AWS AppSync GraphQL-API her, die Verbindungen mit IAM autorisiert.
- Stellen Sie eine Connect zu einem Datenbank-Backend wie Amazon DynamoDB oder Amazon RDS her, das Verbindungen mit IAM autorisiert.
- Rufen Sie Anwendungsressourcen aus einem Amazon S3 S3-Bucket ab.
- Initiieren Sie eine Sitzung mit einem WorkSpaces virtuellen Amazon-Desktop.

Identitätspools funktionieren nicht ausschließlich innerhalb einer authentifizierten Sitzung mit einem Benutzerpool. Sie akzeptieren auch die Authentifizierung direkt von externen Identitätsanbietern und können Anmeldeinformationen für nicht authentifizierte Gastbenutzer generieren.

Weitere Informationen zur Verwendung von Identitätspools zusammen mit Benutzerpoolgruppen zur Steuerung des Zugriffs auf Ihre AWS Ressourcen finden Sie unter Hinzufügen von Gruppen zu einem Benutzerpool und. Verwenden der rollenbasierten Zugriffskontrolle Weitere Informationen zu Identitätspools und AWS Identity and Access Management finden Sie auch unterIdentitäten-Pool-Konzepte.

Einrichten eines Benutzerpools mit AWS Management Console

Erstellen Sie einen Amazon-Cognito-Benutzerpool und notieren Sie die Benutzerpool-ID und App-Client-ID für jede Ihrer Client-Apps. Weitere Informationen zum Erstellen von Benutzerpools finden Sie unter Erste Schritte mit Benutzerpools.

Einrichtung eines Identitätspools mit AWS Management Console

Das folgende Verfahren beschreibt, wie Sie den verwenden AWS Management Console , um einen Identitätspool in einen oder mehrere Benutzerpools und Client-Apps zu integrieren.

So fügen Sie einen Identitätsanbieter (IdP) zu Amazon-Cognito-Benutzerpools hinzu:

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie einen Amazon-Cognito-Benutzerpool.
5. Geben Sie eine Benutzerpool-ID und eine App-Client-ID ein.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

    a. Sie können Benutzern dieses IdP die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder Sie können eine Rolle mit Regeln auswählen. Mit einem IdP für Amazon-Cognito-Benutzerpools können Sie auch eine Rolle mit preferred_role in Token auswählen. Weitere Informationen

zur `cognito:preferred_role`-Anforderung finden Sie unter Zuweisen von Prioritätswerten zu Gruppen.

i. Wenn Sie „Rolle mit Regeln auswählen“ ausgewählt haben, geben Sie den Quellanspruch aus Ihrer Benutzerauthentifizierung, den Operator, den Sie verwenden möchten, um den Anspruch mit der Regel zu vergleichen, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

ii. Wenn Sie „Rolle mit bevorzugtem Rollenanspruch auswählen“ in Tokens ausgewählt haben, stellt Amazon Cognito Anmeldeinformationen für die Rolle im Anspruch Ihres Benutzers aus. `cognito:preferred_role` Wenn kein Anspruch für eine bevorzugte Rolle vorliegt, stellt Amazon Cognito Anmeldeinformationen auf der Grundlage Ihrer Rollenauflösung aus.

b. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   - Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.
   - Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.
   - Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Integration eines Benutzerpools in einen Identitäten-Pool

Nachdem Ihr App-Benutzer authentifiziert wurde, fügen Sie der Anmeldezuweisung im Anmeldeinformationsanbieter das Identitäts-Token des Benutzers hinzu. Der Anbietername ist von Ihrer Amazon-Cognito-Benutzerpool-ID abhängig. Er hat die folgende Struktur:

```
cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>
```

Sie können den Wert für aus der Benutzerpool-ID ableiten. <region> Wenn die Benutzerpool-ID beispielsweise lautetus-east-1_EXAMPLE1, dann <region>ist us-east-1 die. Wenn die Benutzerpool-ID lautetus-west-2_EXAMPLE2, dann <region>ist dieus-west-2.

JavaScript

```
var cognitoUser = userPool.getCurrentUser();

if (cognitoUser != null) {
 cognitoUser.getSession(function(err, result) {
  if (result) {
   console.log('You are now logged in.');

   // Add the User's Id Token to the Cognito credentials login map.
   AWS.config.credentials = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'YOUR_IDENTITY_POOL_ID',
    Logins: {
     'cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>':
 result.getIdToken().getJwtToken()
    }
   });
  }
 });
}
```

Android

```
cognitoUser.getSessionInBackground(new AuthenticationHandler() {
 @Override
 public void onSuccess(CognitoUserSession session) {
  String idToken = session.getIdToken().getJWTToken();

  Map<String, String> logins = new HashMap<String, String>();
  logins.put("cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>",
 session.getIdToken().getJWTToken());
  credentialsProvider.setLogins(logins);
 }

});
```

iOS - objective-C

```
AWSServiceConfiguration *serviceConfiguration = [[AWSServiceConfiguration alloc]
 initWithRegion:AWSRegionUSEast1 credentialsProvider:nil];
AWSCognitoIdentityUserPoolConfiguration *userPoolConfiguration =
 [[AWSCognitoIdentityUserPoolConfiguration alloc] initWithClientId:@"YOUR_CLIENT_ID"
  clientSecret:@"YOUR_CLIENT_SECRET" poolId:@"YOUR_USER_POOL_ID"];
[AWSCognitoIdentityUserPool
 registerCognitoIdentityUserPoolWithConfiguration:serviceConfiguration
 userPoolConfiguration:userPoolConfiguration forKey:@"UserPool"];
AWSCognitoIdentityUserPool *pool = [AWSCognitoIdentityUserPool
 CognitoIdentityUserPoolForKey:@"UserPool"];
AWSCognitoCredentialsProvider *credentialsProvider = [[AWSCognitoCredentialsProvider
 alloc] initWithRegionType:AWSRegionUSEast1 identityPoolId:@"YOUR_IDENTITY_POOL_ID"
 identityProviderManager:pool];
```

iOS - swift

```
let serviceConfiguration = AWSServiceConfiguration(region: .USEast1,
 credentialsProvider: nil)
let userPoolConfiguration = AWSCognitoIdentityUserPoolConfiguration(clientId:
 "YOUR_CLIENT_ID", clientSecret: "YOUR_CLIENT_SECRET", poolId: "YOUR_USER_POOL_ID")
AWSCognitoIdentityUserPool.registerCognitoIdentityUserPoolWithConfiguration(serviceConfigura
 userPoolConfiguration: userPoolConfiguration, forKey: "UserPool")
let pool = AWSCognitoIdentityUserPool(forKey: "UserPool")
let credentialsProvider = AWSCognitoCredentialsProvider(regionType: .USEast1,
 identityPoolId: "YOUR_IDENTITY_POOL_ID", identityProviderManager:pool)
```

Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools

Sie können die Multifaktor-Authentifizierung (MFA) für einen Benutzerpool verwenden, um die Identität Ihrer Benutzer zu schützen. MFA fügt einen zweiten Authentifizierungsfaktor hinzu, damit Ihr Benutzerpool nicht nur auf dem Benutzernamen und dem Passwort basiert. Sie können entweder SMS-Textnachrichten oder zeitgesteuerte Einmalpasswörter (TOTP) als zweite Faktoren für die Anmeldung Ihrer Benutzer verwenden. Sie können auch die adaptive Authentifizierung mit ihrem risikobasierten Modell verwenden, um zu prognostizieren, wann Sie möglicherweise einen anderen Authentifizierungsfaktor benötigen. Zu den erweiterten Sicherheitsfunktionen des Benutzerpools gehören die adaptive Authentifizierung und der Schutz vor gefährdeten Anmeldeinformationen.

Themen

Hinzufügen der MFA zu einem Benutzerpool

Die Multi-Faktor-Authentifizierung (MFA) erhöht die Sicherheit für Ihre App. Sie fügt den Authentifizierungsfaktor Etwas, das Sie haben zu dem Faktor Etwas, das Sie kennen oder wissen hinzu, der aus dem Benutzernamen und einem Passwort besteht. Sie können SMS-Textnachrichten oder zeitgesteuerte Einmalpasswörter (TOTP) als zweite Faktoren für die Anmeldung Ihrer Benutzer verwenden.

> **Note**
>
> Wenn sich ein neuer Benutzer zum ersten Mal bei Ihrer App anmeldet, gibt Amazon Cognito OAuth 2.0-Token aus, auch wenn Ihr Benutzerpool MFA erfordert. Der zweite Authentifizierungsfaktor bei der Erstanmeldung Ihres Benutzer ist die Bestätigung der Verifizierungsnachricht, die Amazon Cognito an ihn sendet. Wenn Ihr Benutzerpool MFA erfordert, fordert Amazon Cognito Ihren Benutzer auf, einen zusätzlichen Anmeldefaktor zu registrieren, der nach der Erstanmeldung bei jedem weiteren Anmeldeversuch verwendet werden soll.

Mit der adaptiven Authentifizierung können Sie Ihren Benutzerpool so konfigurieren, dass ein zweiter Faktor gefordert wird, wenn ein erhöhtes Risiko besteht. Weitere Informationen darüber, wie Sie Ihrem Benutzerpool eine adaptive Authentifizierung hinzufügen, finden Sie unter Hinzufügen erweiterter Sicherheit zu einem Benutzerpool..

Wenn Sie die MFA für einen Benutzerpool auf `required` festlegen, müssen alle Benutzer MFA abschließen, um sich anzumelden. Jeder Benutzer muss mindestens einen MFA-Faktor wie SMS oder TOTP einrichten, um sich anzumelden. Wenn Sie MFA auf `required` festlegen, müssen

Sie die MFA-Einschließung in das Benutzeronboarding einschließen, damit Ihr Benutzerpool die Anmeldung ermöglicht.

Wenn Sie SMS als MFA-Faktor aktivieren, können Sie die Benutzer auffordern, Telefonnummern anzugeben und sie bei der Anmeldung zu verifizieren. Wenn MFA auf `required` eingestellt ist und nur SMS als Faktor unterstützt wird, müssen die Benutzer eine Telefonnummer angeben. Benutzer ohne Telefonnummern benötigen Ihren Support, um ihrem Profil eine Telefonnummer hinzuzufügen, bevor sie sich anmelden können. Sie können nicht verifizierte Telefonnummern für SMS-MFA verwenden. Diese Nummern erhalten den Status „Bestätigt“, nachdem die MFA erfolgreich war.

Wenn Sie MFA als erforderlich festgelegt und SMS sowie TOTP als unterstützte Verifizierungsmethoden aktiviert haben, fordert Amazon Cognito neue Benutzer ohne Telefonnummern auf, TOTP MFA einzurichten. Wenn Sie MFA als erforderlich festgelegt haben und die einzige aktivierte MFA-Methode TOTP ist, fordert Amazon Cognito alle neuen Benutzer auf, TOTP MFA bei der zweiten Anmeldung einzurichten. Amazon Cognito generiert eine Aufforderung zur Einrichtung von TOTP-MFA als Reaktion auf API-Operationen InitiateAuth. AdminInitiateAuth

Die gehostete Benutzeroberfläche fordert Benutzer auf, MFA einzurichten, wenn Sie MFA als erforderlich festlegen. Wenn Sie MFA in Ihrem Benutzerpool als optional festlegen, erhalten die Benutzer der gehosteten Benutzeroberfläche keine Aufforderung. Wenn Sie mit optionaler MFA arbeiten möchten, müssen Sie in Ihrer App eine Oberfläche erstellen, die Ihre Benutzer auffordert, auszuwählen, dass sie MFA einrichten möchten, und die sie dann durch die API-Eingaben führt, um ihren zusätzlichen Anmeldefaktor zu verifizieren.

Nach fünf erfolglosen Versuchen, einen MFA-Code zu präsentieren, beginnt Amazon Cognito mit dem unter Ablauf der Authentifizierung in Benutzerpools beschriebenen exponentiellen Timeout-Kontosperrungsprozess.

Themen

Voraussetzungen

Berücksichtigen Sie Folgendes, bevor Sie MFA einrichten:

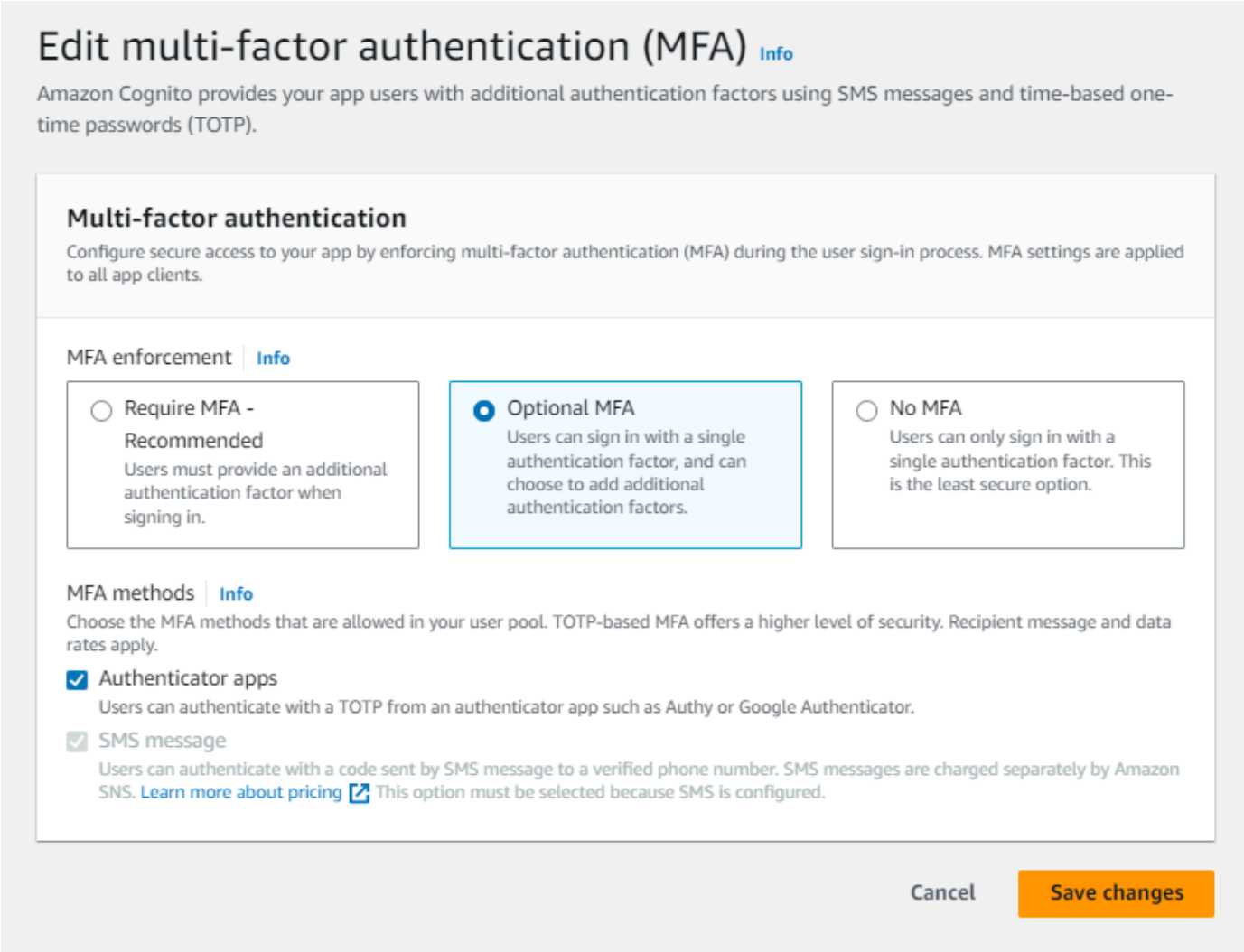
- Wenn Sie MFA in Ihrem Benutzerpool aktivieren und SMS-Testnachricht als zweiten Faktor auswählen, können Sie SMS-Nachrichten an ein Telefonnummernattribut senden, das Sie in Amazon Cognito nicht verifiziert haben. Wenn Ihr Benutzer SMS MFA abgeschlossen hat, stellt Amazon Cognito sein `phone_number_verified`-Attribut auf `true` ein.
- Wenn sich Ihr Konto in der SMS-Sandbox befindet AWS-Region , die die Amazon Simple Notification Service (Amazon SNS) -Ressourcen für Ihren Benutzerpool enthält, müssen Sie die Telefonnummern in Amazon SNS überprüfen, bevor Sie eine SMS-Nachricht senden können. Weitere Informationen finden Sie unter Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools.
- Erweiterte Sicherheitsfunktionen erfordern, dass Sie MFA aktivieren und in der Amazon-Cognito-Benutzerpool-Konsole als optional festgelegen. Weitere Informationen finden Sie unter Hinzufügen erweiterter Sicherheit zu einem Benutzerpool..

Konfigurieren der Multi-Faktor-Authentifizierung

Sie können MFA in der Amazon-Cognito-Konsole konfigurieren

Konfigurieren der MFA in der Amazon-Cognito-Konsole

1. Melden Sie sich bei der Amazon Cognito-Konsole an.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte Sign-in experience (Anmeldeerlebnis) aus. Suchen Sie nach Multi-factor authentication (Multi-Faktor-Authentifizierung) und wählen Sie Edit (Bearbeiten) aus.
5. Wählen Sie die Methode MFA enforcement (Durchsetzung von MFA), die Sie für Ihren Benutzerpool verwenden möchten.



a. MFA erforderlich. Alle Benutzer in Ihrem Benutzerpool müssen sich mit einem zusätzlichen SMS-Code oder zeitgesteuerten Einmalpasswörtern (TOTP) anmelden.

b. Optional MFA (Optionale MFA) – Sie können Ihren Benutzern die Möglichkeit geben, einen zusätzlichen Anmeldefaktor zu registrieren, und Benutzern ohne konfigurierten MFA trotzdem erlauben, sich anzumelden. Wählen Sie diese Option, wenn Sie die adaptive Authentifizierung verwenden. Weitere Informationen zur adaptiven Authentifizierung finden Sie unter Hinzufügen erweiterter Sicherheit zu einem Benutzerpool..

c. Kein MFA. Ihre Benutzer können keinen zusätzlichen Anmeldefaktor registrieren.

6. Wählen Sie die MFA-Methoden aus, die Sie in Ihrer App unterstützen. Sie können SMS-Nachrichten oder Authentifizierungs-Apps, die TOTPs generieren, als zweiten Faktor festlegen. Wir empfehlen Ihnen, TOTP-basierte MFA zu implementieren, damit die Kontowiederherstellung SMS-Nachrichten verwenden kann.

7. Wenn Sie SMS-Nachrichten als zweiten Faktor verwenden und Sie keine IAM-Rolle für die Verwendung mit Amazon Simple Notification Service (Amazon SNS) für SMS-Nachrichten konfiguriert haben, können Sie eine in der Konsole erstellen. Suchen Sie in der Registerkarte Messaging für Ihren Benutzerpool SMS und wählen Sie Bearbeiten aus. Sie können auch eine vorhandene Rolle verwenden, mit der Amazon Cognito SMS-Nachrichten für Sie an Ihre Benutzer senden kann. Weitere Informationen finden Sie unter IAM-Rollen.
8. Wählen Sie Save Changes.

SMS-MFA

Wenn sich ein Benutzer mit aktivierter MFA anmeldet, gibt er zuerst seinen Benutzernamen und sein Passwort ein. Die Client-App erhält eine `getMFA`-Antwort, die angibt, wohin der Autorisierungscode gesendet wurde. Die Client-App muss dem Benutzer mitteilen, wo er den Code findet (z. B. die Telefonnummer, an die der Code gesendet wurde). Als nächstes stellt es ein Formular zur Eingabe des Codes bereit. Schließlich sendet die Client-App den Code zum Abschluss des Anmeldeprozesses. Das Ziel wird verborgen und verbirgt alle bis auf die letzten vier Ziffern der Telefonnummer. Wenn eine App die Amazon-Cognito-gehostete Benutzeroberfläche verwendet, wird eine Seite angezeigt, auf der der Benutzer den MFA-Code eingeben kann.

Der SMS-Autorisierungscode ist für Authentication flow session duration (Dauer der Authentifizierungsablaufsitzung) gültig, die Sie für Ihren App-Client festgelegt haben.

Legen Sie die Dauer einer Authentifizierungsablaufsitzung in der Amazon-Cognito-Konsole auf der Registerkarte App integration (App-Integration) fest, wenn Sie Ihren App-Client unter App clients and analytics (App-Clients und -Analysen) ändern. Sie können die Dauer der Authentifizierungsablaufsitzung auch in einer `CreateUserPoolClient`- oder `UpdateUserPoolClient`-API-Anforderung festlegen. Weitere Informationen finden Sie unter Ablauf der Authentifizierung in Benutzerpools.

Wenn Benutzer keinen Zugriff mehr auf das Gerät haben, an das die SMS-MFA-Codes gesendet werden, müssen sie sich an den Kundenservice wenden. Ein Administrator mit den erforderlichen AWS-Konto Berechtigungen kann die Telefonnummer des Benutzers ändern, jedoch nur über die AWS CLI oder die API.

Durchlaufen Benutzer den SMS-MFA-Ablauf erfolgreich, werden ihre Telefonnummern ebenfalls als bestätigt gekennzeichnet.

> **Note**
>
> SMS für MFA wird separat in Rechnung gestellt. (Es fallen keine Gebühren für das Senden von Verifizierungscodes an E-Mail-Adressen an.) Weitere Informationen zu den Amazon-SNS-Preisen erhalten Sie unter Weltweite SMS-Preise. Eine aktuelle Liste von Ländern, in denen SMS-Messaging verfügbar ist, finden Sie in den Informationen zu den unterstützten Regionen und Ländern.

> **Important**
>
> Um sicherzustellen, dass SMS-Nachrichten zum Verifizieren von Telefonnummern und für die SMS-MFA gesendet werden, müssen Sie ein erhöhtes Ausgabenlimit bei Amazon SNS anfordern.
> Amazon Cognito verwendet Amazon SNS zum Senden von SMS-Nachrichten an Benutzer. Die Anzahl der von Amazon SNS übermittelten SMS-Nachrichten unterliegt den Ausgabenlimits. Ausgabenlimits können für ein AWS Konto und für einzelne Nachrichten festgelegt werden. Die Limits gelten nur für die Kosten für den Versand von SMS-Nachrichten.
> Das standardmäßige Ausgabenlimit pro Konto (sofern nicht angegeben) liegt bei 1,00 USD pro Monat. Wenn Sie das Limit erhöhen möchten, reichen Sie im AWS Support Center einen Fall zur Erhöhung des SNS-Limits ein. Geben Sie unter New limit value (Neuer Limit-Wert) das gewünschte monatliche Ausgabenlimit ein. Geben Sie im Feld Use Case Description (Beschreibung des Anwendungsfalls) an, dass Sie eine Erhöhung des monatlichen Ausgabenlimits für SMS wünschen.

Weitere Informationen darüber, wie Sie Ihrem Benutzerpool eine MFA hinzufügen, finden Sie unter Hinzufügen der MFA zu einem Benutzerpool. Weitere Informationen zu SMS-Nachrichten mit Amazon SNS in Ihrem Benutzerpool finden Sie unterEinstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools.

TOTP-Software-Token-MFA

Wenn Sie das TOTP-Software-Token-MFA in Ihrem Benutzerpool einrichten, meldet sich Ihr Benutzer mit einem Benutzernamen und einem Kennwort an und schließt die Authentifizierung dann mit einem TOTP ab. Nachdem Ihr Benutzer einen Benutzernamen und ein Passwort festgelegt und verifiziert

hat, kann er ein TOTP-Software-Token für MFA aktivieren. Wenn Ihre App die von Amazon Cognito gehostete Benutzeroberfläche zur Anmeldung von Benutzern verwendet, gibt Ihr Benutzer seinen Benutzernamen und sein Passwort ein und sendet dann das TOTP-Passwort auf einer zusätzlichen Anmeldeseite.

Sie können TOTP-MFA für Ihren Benutzerpool in der Amazon-Cognito-Konsole aktivieren oder die Amazon-Cognito-API-Operationen verwenden. Auf Benutzerpoolebene können Sie aufrufen, um MFA SetUserPoolMfaConfigzu konfigurieren und TOTP MFA zu aktivieren.

> **Note**
>
> Wenn Sie die TOTP-Software-Token-MFA für den Benutzerpool nicht aktiviert haben, kann Amazon Cognito das Token nicht verwenden, um Benutzer zuzuordnen und zu verifizieren. In diesem Fall erhalten Benutzer eine `SoftwareTokenMFANotFoundException` Ausnahme mit der Beschreibung `Software Token MFA has not been enabled by the userPool`. Wenn Sie das Software-Token-MFA später für den Benutzerpool deaktivieren, können Benutzer, die zuvor ein TOTP-Token zugeordnet und verifiziert haben, es weiterhin für MFA verwenden.

TOTP wird für Benutzer in mehreren Schritten konfiguriert. Der Benutzer erhält dabei einen geheimen Code, den er durch Eingabe eines einmaligen Passworts bestätigt. Danach können Sie TOTP-MFA für den Benutzer aktivieren oder TOTP als bevorzugte MFA-Methode für den Benutzer festlegen.

Wenn Sie Ihren Benutzerpool so konfigurieren, dass TOTP-MFA erforderlich ist und sich Ihre Benutzer auf der gehosteten Benutzeroberfläche bei Ihrer App anmelden, automatisiert Amazon Cognito den Benutzerprozess. Amazon Cognito fordert Ihre Benutzer auf, eine MFA-Methode auszuwählen, zeigt einen QR-Code an, um ihre Authentifizierungs-App einzurichten, und überprüft ihre MFA-Registrierung. In Benutzerpools, in denen Sie Benutzern erlaubt haben, zwischen SMS- und TOTP-MFA zu wählen, bietet Amazon Cognito Ihren Benutzern verschiedene Methoden zur Auswahl an. Weitere Informationen zur Anmeldungsumgebung der gehosteten Benutzeroberfläche finden Sie unter So registrieren Sie sich für ein neues Konto bei der von Amazon Cognito gehosteten Benutzeroberfläche.

> **Important**
>
> Wenn einem Benutzerpool eine AWS WAF Web-ACL zugeordnet ist und eine Regel in Ihrer Web-ACL ein CAPTCHA enthält, kann dies zu einem nicht behebbaren Fehler bei

der TOTP-Registrierung für gehostete Benutzeroberflächen führen. Informationen zum Erstellen einer Regel mit einer CAPTCHA-Aktion, die sich nicht auf TOTP der gehosteten Benutzeroberfläche auswirkt, finden Sie unter Konfiguration Ihrer AWS WAF Web-ACL für Hosted UI TOTP MFA. Weitere Informationen zu AWS WAF Web-ACLs und Amazon Cognito finden Sie unter. Eine AWS WAF Web-ACL einem Benutzerpool zuordnen

Informationen zum Implementieren der TOTP-MFA in einer benutzerdefinierten Benutzeroberfläche, in der Sie die Amazon-Cognito-API verwenden, finden Sie unter Konfigurieren der MFA für einen Benutzer in der Amazon-Cognito-Benutzerpool-API.

Informationen darüber, wie Sie Ihrem Benutzerpool eine MFA hinzufügen, finden Sie unter Hinzufügen der MFA zu einem Benutzerpool.

TOTP MFA Überlegungen und Einschränkungen

1. Amazon Cognito unterstützt Softwaretoken-MFA über eine Authentifizierungs-App, die TOTP-Codes generiert. Amazon Cognito unterstützt kein hardwarebasiertes MFA.
2. Wenn Ihr Benutzerpool TOTP für einen Benutzer benötigt, der ihn nicht konfiguriert hat, erhält Ihr Benutzer ein einmaliges Zugriffstoken, mit dem Ihre App TOTP MFA für den Benutzer aktivieren kann. Nachfolgende Anmeldeversuche schlagen fehl, bis Ihr Benutzer einen zusätzlichen TOTP-Anmeldefaktor registriert hat.
   - Ein Benutzer, der sich in Ihrem Benutzerpool mit der `SignUp` API-Operation oder über die gehostete Benutzeroberfläche anmeldet, erhält nach Abschluss der Anmeldung einmalige Token.
   - Nachdem Sie einen Benutzer erstellt und der Benutzer sein ursprüngliches Passwort festgelegt hat, gibt Amazon Cognito einmalige Token von der gehosteten Benutzeroberfläche an den Benutzer aus. Wenn Sie ein dauerhaftes Passwort für den Benutzer festlegen, gibt Amazon Cognito bei der erstmaligen Anmeldung des Benutzers einmalige Token aus.
   - Amazon Cognito gibt keine einmaligen Token an einen vom Administrator erstellten Benutzer aus, der sich mit den InitiateAuthoder API-Vorgängen anmeldet. AdminInitiateAuth Nachdem es Ihrem Benutzer in der Aufforderung gelungen ist, ein Anfangspasswort festzulegen, oder wenn Sie ein dauerhaftes Passwort für den Benutzer festlegen, fordert Amazon Cognito den Benutzer sofort auf, MFA einzurichten.
3. Wenn ein Benutzer in einem Benutzerpool, der MFA benötigt, bereits ein einmaliges Zugriffstoken erhalten, jedoch noch nicht TOTP MFA eingerichtet hat, kann sich der Benutzer erst mit der gehosteten Benutzeroberfläche anmelden, nachdem er MFA eingerichtet hat. Anstelle des

Zugriffstokens können Sie den `session` Antwortwert aus einer Anfrage InitiateAuthoder AdminInitiateAuthin `MFA_SETUP` einer Anfrage verwenden. AssociateSoftwareToken

4. Wenn Ihre Benutzer TOTP eingerichtet haben, können sie es auch dann für MFA verwenden, wenn Sie TOTP für den Benutzerpool später deaktivieren.
5. Amazon Cognito akzeptiert ausschließlich TOTPs von Authenticator-Apps, die Codes mit der SHA-1-Hash-Funktion generieren. Mit SHA-256-Hashing generierte Codes geben einen `Code mismatch`-Fehler zurück.

Konfigurieren der MFA für einen Benutzer in der Amazon-Cognito-Benutzerpool-API

Wenn sich ein Benutzer zum ersten Mal anmeldet, verwendet Ihre App das einmalige Zugriffstoken zur Generierung des privaten TOTP-Schlüssels und präsentiert ihn Ihrem Benutzer in Text- oder QR-Codeformat. Ihr Benutzer konfiguriert seine Authentifizierungs-App und stellt ein TOTP für nachfolgende Anmeldeversuche bereit. Ihre App oder die gehostete Benutzeroberfläche präsentiert Amazon Cognito in Antworten auf MFA-Aufforderungen.

Themen

- Zuordnen des TOTP-Software-Tokens
- Bestätigen des TOTP-Token
- Melden Sie sich mit TOTP-MFA an
- Entfernen des TOTP-Tokens

Zuordnen des TOTP-Software-Tokens

Um das TOTP-Token zu verknüpfen, senden Sie dem Benutzer einen geheimen Code, den er mit einem einmaligen Passwort validieren muss. Token werden in drei Schritten verknüpft.

1. Wenn Ihr Benutzer das TOTP-Softwaretoken MFA auswählt, rufen Sie an, AssociateSoftwareTokenum einen eindeutigen generierten gemeinsamen geheimen Schlüsselcode für das Benutzerkonto zurückzugeben. Sie können die Autorisierung entweder AssociateSoftwareToken mit einem Zugriffstoken oder einer Sitzungszeichenfolge durchführen.
2. Ihre App präsentiert dem Benutzer den privaten Schlüssel oder einen QR-Code, den Sie anhand des privaten Schlüssels generieren. Ihr Benutzer muss den Schlüssel in eine TOTP-Generierungsanwendung wie Google Authenticator eingeben. Sie können Folgendes libqrencode verwenden, um einen QR-Code zu generieren.

3. Ihr Benutzer gibt den Schlüssel ein oder scannt den QR-Code in eine Authentifizierungs-App wie Google Authenticator. Daraufhin beginnt die App mit der Generierung von Codes.

Bestätigen des TOTP-Token

Als Nächstes bestätigen Sie das TOTP-Token. Fordern Sie Beispielcodes von Ihrem Benutzer an und stellen Sie sie dem Amazon Cognito-Service zur Verfügung, um zu bestätigen, dass der Benutzer wie folgt erfolgreich TOTP-Codes generiert.

1. Ihre App fordert Ihren Benutzer zur Eingabe eines Codes auf, um zu zeigen, dass er seine Authentifizierungs-App ordnungsgemäß eingerichtet hat.
2. Die Authentifizierungs-App des Benutzers zeigt ein temporäres Passwort an. Die Authentifizierungs-App erstellt das Passwort auf Basis des geheimen Schlüssels, den Sie dem Benutzer zur Verfügung gestellt haben.
3. Ihr Benutzer gibt sein temporäres Passwort ein. Ihre App übergibt das temporäre Passwort an Amazon Cognito in einer `VerifySoftwareToken`-API-Anforderung.
4. Amazon Cognito hat den mit dem Benutzer verknüpften geheimen Schlüssel beibehalten, und generiert ein TOTP und vergleicht ihn mit dem von Ihrem Benutzer bereitgestellten TOTP. Wenn sie übereinstimmen, gibt `VerifySoftwareToken` eine `SUCCESS`-Antwort zurück.
5. Amazon Cognito verknüpft den TOTP-Faktor mit dem Benutzer.
6. Wenn die `VerifySoftwareToken`-Vorgang eine `ERROR` Antwort ausgibt, stellen Sie sicher, dass die Uhr des Benutzers richtig eingestellt ist und dass er die maximale Zahl erneuter Versuche nicht überschritten hat. Amazon Cognito akzeptiert TOTP-Token innerhalb von 30 Sekunden vor oder nach dem Versuch, um kleinere Taktversätze zu berücksichtigen. Wenn Sie das Problem behoben haben, versuchen Sie den VerifySoftwareToken Vorgang erneut.

Melden Sie sich mit TOTP-MFA an

Zu diesem Zeitpunkt meldet sich Ihr Benutzer mit dem zeitbasierten Einmalpasswort an. Der Prozess läuft folgendermaßen ab:

1. Benutzer geben ihren Benutzernamen und ihr Passwort ein, um sich bei Ihrer Client-App anzumelden.
2. Die TOTP-MFA-Eingabe wird angezeigt und der Benutzer wird von der Anwendung aufgefordert, ein temporäres Passwort einzugeben.
3. Der Benutzer erhält ein temporäres Passwort von einer zugeordneten TOTP-Generierung-App.

4. Der Benutzer gibt den TOTP-Code in der Client-App ein. Die Anwendung sendet eine Bestätigungsanforderung an den Amazon-Cognito-Service. Bei jeder Anmeldung RespondToAuthChallengesollte aufgerufen werden, um eine Antwort auf die neue TOTP-Authentifizierungsherausforderung zu erhalten.
5. Wenn das Token von Amazon Cognito bestätigt wird, ist die Anmeldung erfolgreich abgeschlossen und der Benutzer fährt mit dem Authentifizierungsablauf fort.

Entfernen des TOTP-Tokens

Schließlich sollte Ihre App dem Benutzer erlauben, die TOTP-Konfiguration zu deaktivieren. Derzeit können Sie das TOTP-Softwaretoken eines Benutzers nicht löschen. Wenn Sie das Softwaretoken Ihres Benutzers ersetzen möchten, verknüpfen und verifizieren Sie ein neues Softwaretoken. Um TOTP-MFA für einen Benutzer zu deaktivieren, rufen Sie SetUserMFAPreference auf, um Ihren Benutzer so zu ändern, dass er kein MFA oder nur SMS-MFA verwendet.

1. Erstellen Sie in Ihrer App eine Schnittstelle für Benutzer, die die MFA zurücksetzen möchten. Fordern Sie einen Benutzer in dieser Schnittstelle zur Eingabe des Passworts auf.
2. Wenn Amazon Cognito eine TOTP-MFA-Anfrage zurückgibt, aktualisieren Sie die MFA-Präferenz Ihres Benutzers mit MFAPreference. SetUser
3. Teilen Sie dem Benutzer in Ihrer App mit, dass MFA deaktiviert wurde, und fordern Sie ihn auf, sich erneut anzumelden.

Konfiguration Ihrer AWS WAF Web-ACL für Hosted UI TOTP MFA

Wenn einem Benutzerpool eine AWS WAF Web-ACL zugeordnet ist und eine Regel in Ihrer Web-ACL ein CAPTCHA enthält, kann dies zu einem nicht behebbaren Fehler bei der TOTP-Registrierung für gehostete Benutzeroberflächen führen. AWS WAF CAPTCHA-Regeln wirken sich auf diese Weise nur auf TOTP MFA in der gehosteten Benutzeroberfläche aus. Die SMS-MFA ist nicht betroffen.

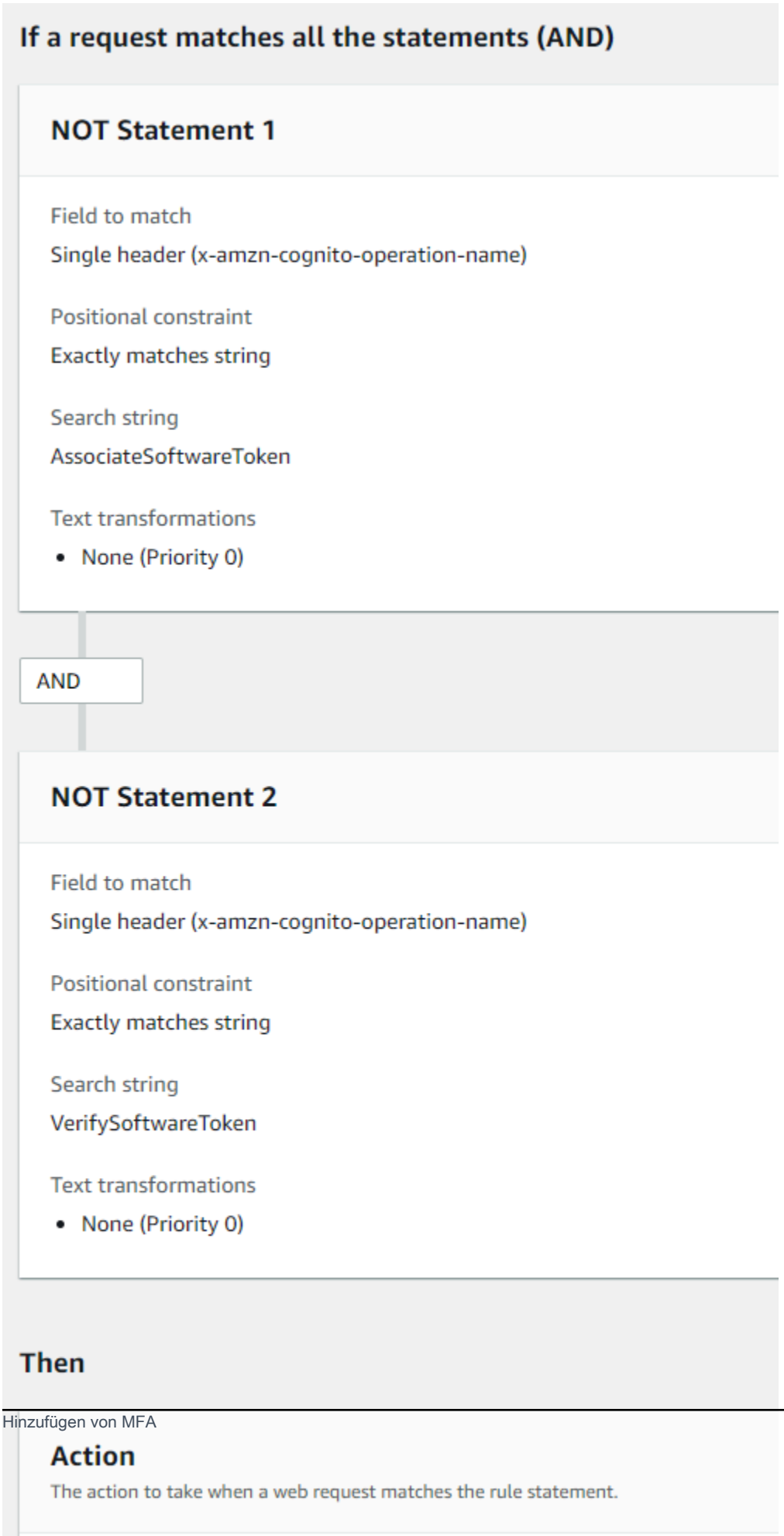
Amazon Cognito zeigt den folgenden Fehler an, wenn Ihre CAPTCHA-Regel es einem Benutzer nicht erlaubt, die TOTP-MFA-Einrichtung abzuschließen.

Anfrage aufgrund des WAF-Captchas nicht zulässig.

Dieser Fehler tritt auf, wenn AWS WAF Sie als Antwort auf VerifySoftwareTokenAPI-Anfragen, die Ihr Benutzerpool im Hintergrund stellt, zur AssociateSoftwareTokenEingabe eines CAPTCHA aufgefordert werden. Zum Erstellen einer Regel, die eine CAPTCHA-Aktion hat und sich nicht

auf den TOTP-Algorithmus der gehosteten Benutzeroberfläche auswirkt, schließen Sie die `x-amzn-cognito-operation-name`-Header-Werte `AssociateSoftwareToken` und `VerifySoftwareToken` aus der CAPTCHA-Aktion in Ihrer Regel aus.

Der folgende Screenshot zeigt eine AWS WAF Beispielregel, die eine CAPTCHA-Aktion auf alle Anfragen anwendet, die keinen Header-Wert von oder haben. `x-amzn-cognito-operation-name` `AssociateSoftwareToken` `VerifySoftwareToken`


If a request matches all the statements (AND)
NOT Statement 1
Field to match
Single header (x-amzn-cognito-operation-name)
Positional constraint
Exactly matches string
Search string
AssociateSoftwareToken
Text transformations
None (Priority 0)
AND
NOT Statement 2
Field to match
Single header (x-amzn-cognito-operation-name)
Positional constraint
Exactly matches string
Search string
VerifySoftwareToken
Text transformations
None (Priority 0)
Then
Action
The action to take when a web request matches the rule statement.

Weitere Informationen zu AWS WAF Web-ACLs und Amazon Cognito finden Sie unter. Eine AWS WAF Web-ACL einem Benutzerpool zuordnen

Hinzufügen erweiterter Sicherheit zu einem Benutzerpool.

Nachdem Sie einen Benutzerpool erstellt haben, haben Sie Zugriff auf Advanced security (Erweiterte Sicherheit) auf der Navigationsleiste in der Amazon-Cognito-Konsole. Sie können die erweiterten Sicherheitsfunktionen für den Benutzerpool aktivieren und die Aktionen anpassen, die als Reaktion auf verschiedenen Risiken ausgeführt werden. Sie können auch den Prüfmodus verwenden, um Metriken zu erkannten Risiken zu erfassen, ohne Sicherheitsminderungen anzuwenden. Im Auditmodus veröffentlichen die erweiterten Sicherheitsfunktionen Metriken auf Amazon CloudWatch. Sie können erweiterte Sicherheitsmetriken einsehen, nachdem Amazon Cognito sein erstes erweitertes Sicherheitsereignis generiert hat. Siehe Anzeigen erweiterter Sicherheitsmetriken.

Zu den erweiterten Sicherheitsfunktionen des Benutzerpools gehören die Erkennung kompromittierter Anmeldeinformationen und die adaptive Authentifizierung.

Kompromittierte Anmeldeinformationen

Benutzer verwenden Passwörter für mehrere Benutzerkonten erneut. Die Funktion für kompromittierte Anmeldeinformationen von Amazon Cognito kompiliert Daten aus öffentlich zugänglichen Benutzernamen und Passwörtern und vergleicht die Anmeldeinformationen Ihrer Benutzer mit Listen von offengelegten Anmeldeinformationen. Die Erkennung kompromittierter Anmeldeinformationen sucht auch nach häufig erratenen Passwörtern.

Sie können die Benutzeraktionen auswählen, die eine Überprüfung auf kompromittierte Anmeldeinformationen veranlassen, sowie die Aktion, die Amazon Cognito als Reaktion darauf durchführen soll. Bei Anmelde-, Registrierungs- und Kennwortänderungsereignissen kann Amazon Cognito die Anmeldung blockieren oder die Anmeldung zulassen. In beiden Fällen generiert Amazon Cognito ein Benutzeraktivitätsprotokoll, in dem Sie weitere Informationen über das Ereignis finden.

Adaptive Authentifizierung

Amazon Cognito kann Standort- und Geräteinformationen aus Anmeldeanfragen Ihrer Benutzer überprüfen und automatisch reagieren, um die Benutzerkonten in Ihrem Benutzerpool vor verdächtigen Aktivitäten zu schützen.

Wenn Sie die erweiterte Sicherheit aktivieren, weist Amazon Cognito der Benutzeraktivität eine Risikobewertung zu. Sie können eine automatische Reaktion auf verdächtige Aktivitäten

zuweisen: Sie können MFA verlangen, die Anmeldung blockieren oder lediglich die Aktivitätsdetails und die Risikobewertung protokollieren. Sie können Ihren Benutzer auch automatisch per E-Mail über die verdächtige Aktivität informieren, so dass er sein Passwort zurücksetzen oder andere selbstgesteuerte Maßnahmen ergreifen kann.

Anpassen von Zugriffs-Token

Wenn Sie erweiterte Sicherheitsfunktionen aktivieren, können Sie Ihren Benutzerpool so konfigurieren, dass Antworten auf ein Lambda-Trigger-Ereignis der Version 2 akzeptiert werden. Mit Version 2 können Sie Bereiche und andere Ansprüche in Zugriffs-Token anpassen. So erzielen Sie mehr Flexibilität für Benutzerauthentisierungen. Weitere Informationen finden Sie unter Anpassen des Zugriffs-Token.

Themen

Überlegungen und Einschränkungen

- Für die erweiterten Sicherheitsfunktionen von Amazon Cognito fallen zusätzliche Gebühren an. Sehen Sie sich die Preisseite von Amazon Cognito an.
- Amazon Cognito unterstützt die adaptive Authentifizierung und die Erkennung kompromittierter Anmeldeinformationen mit den folgenden Standardauthentifizierungsabläufen:`USER_PASSWORD_AUTH`, und. `ADMIN_USER_PASSWORD_AUTH` `USER_SRP_AUTH` Sie können keine erweiterte Sicherheit mit einem `CUSTOM_AUTH`-Fluss und Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen, oder mit föderierter Anmeldung verwenden.
- Mit den erweiterten Sicherheitsfunktionen von Amazon Cognito im Modus Vollständige Funktion können Sie für die IP-Adresse die Ausnahmen Immer blockieren und Immer zulassen erstellen. Einer Sitzung von einer IP-Adresse auf der Ausnahmeliste Immer blockieren wird

durch adaptive Authentifizierung keine Risikostufe zugewiesen und kann sich nicht bei Ihrem Benutzerpool anmelden.

- Blockierte Anfragen von IP-Adressen auf einer Ausnahmeliste Immer blockieren in Ihrem Benutzerpool trägt zu den Anforderungsratenkontingenten für Ihre Benutzerpools bei. Erweiterte Sicherheitsfunktionen von Amazon Cognito verhindern keine DDoS-Angriffe (Distributed Denial of Service). Um Schutzmaßnahmen gegen volumetrische Angriffe in Ihren Benutzerpools zu implementieren, fügen Sie Web-ACLs hinzu. AWS WAF Weitere Informationen finden Sie unter Eine AWS WAF Web-ACL einem Benutzerpool zuordnen.
- Die Gewährung von Kundenanmeldedaten ist für die machine-to-machine (M2M-) Autorisierung ohne Verbindung zu Benutzerkonten vorgesehen. Die erweiterten Sicherheits-Features überwachen nur Benutzerkonten und Passwörter in Ihrem Benutzerpool. Um Sicherheitsfunktionen in Ihre M2M-Aktivität zu integrieren, sollten Sie die Funktionen AWS WAF zur Überwachung von Anforderungsraten und Inhalten in Betracht ziehen. Weitere Informationen finden Sie unter Eine AWS WAF Web-ACL einem Benutzerpool zuordnen.

Voraussetzungen

Bevor Sie beginnen, muss Folgendes sichergestellt sein:

- Einen Benutzerpool mit einem App-Client. Weitere Informationen finden Sie unter Erste Schritte mit Benutzerpools.
- Setzen Sie die Multi-Faktor-Authentifizierung (MFA) in der Amazon-Cognito-Konsole auf Optional, um die risikobasierte adaptive Authentifizierung zu verwenden. Weitere Informationen finden Sie unter Hinzufügen der MFA zu einem Benutzerpool.
- Wenn Sie E-Mail-Benachrichtigungen verwenden, öffnen Sie die Amazon-SES-Konsole, um die E-Mail-Adresse oder Domäne zu konfigurieren und zu verifizieren, die für Ihre E-Mail-Nachrichten verwendet werden soll. Weitere Informationen über Amazon SES finden Sie unter Verifizieren von Identitäten in Amazon SES.

Konfiguration der erweiterten Sicherheitsfunktionen

Sie können erweiterte Sicherheitsfunktionen von Amazon Cognito in der AWS Management Console konfigurieren.

Konfigurieren der erweiterten Sicherheit für einen Benutzerpool

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldedaten ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus. Suchen Sie nach Advanced security (Erweiterte Sicherheit) und wählen Sie Enable (Aktivieren) aus. Wenn Sie die erweiterte Sicherheit bereits aktiviert haben, wählen Sie Edit (Bearbeiten) aus.
5. Wählen Sie Full function (Vollständige Funktion) aus, um erweiterte Sicherheitsreaktionen für kompromittierte Anmeldeinformationen und adaptive Authentifizierung zu konfigurieren. Wählen Sie Nur Audit aus, um Informationen zu sammeln und Benutzerpooldaten an diese zu senden CloudWatch. Die Preise für erweiterte Sicherheit gelten für die Modi Audit only (Nur prüfen) und Full function (Vollständige Funktion). Weitere Informationen finden Sie unter Amazon Cognito – Preise.

   Wir empfehlen, die erweiterten Sicherheitsfunktionen zwei Wochen im Auditmodus zu betreiben, bevor Sie Maßnahmen ergreifen. In dieser Zeit kann Amazon Cognito die Nutzungsmuster Ihrer App-Benutzer lernen.
6. Wenn Sie Audit only (Nur prüfen) ausgewählt haben, wählen Sie Save changes (Änderungen speichern) aus. Wenn Sie Full function (Vollständige Funktion) ausgewählt haben:

   a. Wählen Sie aus, ob Sie eine benutzerdefinierte Aktion ausführen oder Cognito-Standardfunktionen verwenden möchten, um auf möglicherweise kompromittierte Anmeldeinformationen zu reagieren. Cognito-Standardfunktionen:

      i. Erkennen kompromittierter Anmeldeinformationen bei Anmeldung, Registrierung und Passwortänderung.

      ii. Reagieren Sie auf kompromittierte Anmeldeinformationen mit der Aktion Block sign-in (Anmeldung blockieren).

   b. Wenn Sie Custom (benutzerdefinierte) Aktionen für Compromised credentials (kompromittierte Anmeldeinformationen) ausgewählt haben, wählen Sie die Benutzerpool-Aktionen aus, die Amazon Cognito für die Event detection (Ereigniserkennung) verwendet, sowie die Compromised credentials responses (Antworten auf kompromittierte Anmeldeinformationen), die Amazon Cognito auslösen soll. Sie können bei möglicherweise

kompromittierten Anmeldeinformationen entweder die Anmeldung blockieren oder die Anmeldung zulassen.

c. Wählen Sie unter Adaptive authentication (Adaptive Authentifizierung) aus, wie Sie auf schädliche Anmeldeversuche reagieren möchten. Wählen Sie aus, ob Sie eine benutzerdefinierte Aktion ausführen oder Cognito-Standardfunktionen verwenden möchten, um auf verdächtige schädliche Aktivitäten zu reagieren. Wenn Sie Cognito defaults (Cognito-Standardfunktionen) auswählen, blockiert Amazon Cognito die Anmeldung auf allen Risikostufen und benachrichtigt den Benutzer nicht.

d. Wenn Sie benutzerdefinierte Aktionen für Adaptive authentication (Adaptive Authentifizierung) ausgewählt haben, wählen Sie die Aktion Automatic risk repsonse (Automatische Reaktion auf Risiken) aus. Amazon Cognito wird diese verwenden, um je nach Schweregrad auf erkannte Risiken zu reagieren. Wenn Sie einer Risikostufe eine Reaktion zuweisen, können Sie einem höheren Risiko keine weniger restriktive Reaktion zuweisen. Sie können den Risikostufen folgende Reaktionen zuweisen:

   i. Allow sign-in (Anmeldung zulassen) – Ergreifen Sie keine vorbeugenden Maßnahmen.

   ii. Optional MFA (Optionale MFA) – Wenn der Benutzer MFA konfiguriert hat, verlangt Amazon Cognito, dass der Benutzer bei der Anmeldung immer einen zusätzlichen SMS- oder zeitgesteuerten Einmalpasswort (TOTP)-Faktor bereitstellt. Wenn der Benutzer keine MFA konfiguriert hat, kann er sich weiterhin normal anmelden.

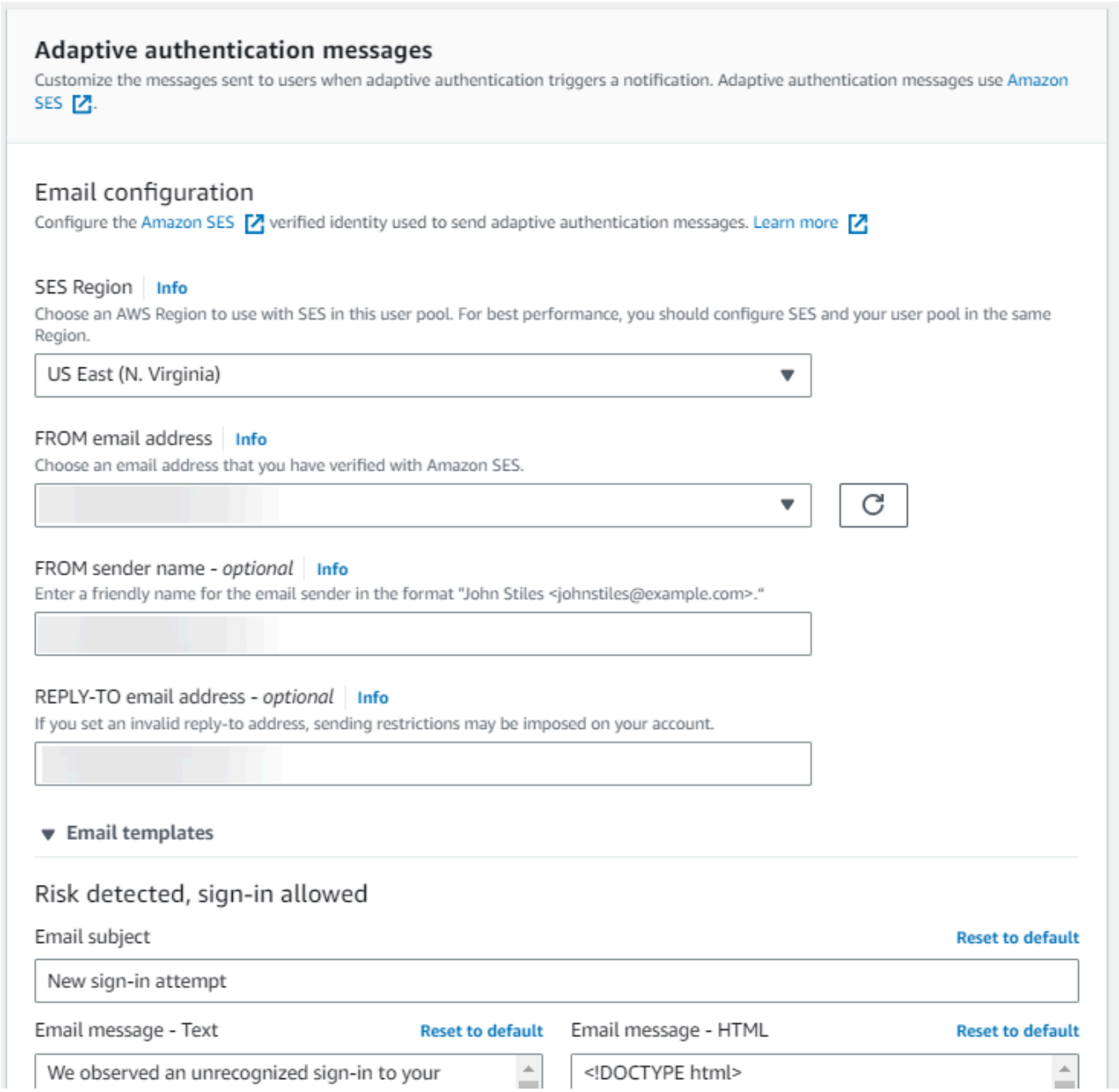
   iii. Optionale MFA – Wenn der Benutzer MFA konfiguriert hat, verlangt Amazon Cognito, dass der Benutzer bei der Anmeldung immer einen zusätzlichen SMS- oder TOTP-Faktor bereitstellt. Wenn der Benutzer keine MFA konfiguriert hat, fordert Amazon Cognito ihn auf, MFA einzurichten. Bevor Sie automatisch MFA für Ihre Benutzer anfordern, konfigurieren Sie in Ihrer App einen Mechanismus, um Telefonnummern für SMS-MFA zu erfassen oder Authentifizierungs-Apps für TOTP MFA zu registrieren.

   iv. Block sign-in (Anmeldung blockieren) – Verhindern Sie, dass sich der Benutzer anmeldet.

   v. Notify user (Benutzer benachrichtigen) – Senden Sie dem Benutzer eine E-Mail-Nachricht mit Informationen über das Risiko, das Amazon Cognito festgestellt hat, und Ihre Reaktion auf das Risiko. Sie können E-Mail-Nachrichtenvorlagen für von Ihnen gesendete Nachrichten anpassen.

7. Wenn Sie im vorherigen Schritt Notify user (Benutzer benachrichtigen) ausgewählt haben, können Sie Ihre Einstellungen für die E-Mail-Zustellung und E-Mail-Nachrichtenvorlagen für die adaptive Authentifizierung anpassen.

a. Wählen Sie unter Email configuration (E-Mail-Konfiguration) die SES Region (SES-Region), die FROM email address (Absender-E-Mail-Adresse), die FROM sender name (Absendernamen) und die REPLY-TO email address (Empfänger-E-Mail-Adresse) aus, die Sie für die adaptive Authentifizierung verwenden möchten. Weitere Informationen zur Integration Ihrer Benutzerpool-E-Mail-Nachrichten in Amazon Simple Email Service finden Sie unter Email settings for Amazon Cognito user pools (E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools).



b. Wählen Sie Email templates (E-Mail-Vorlagen) aus, um Benachrichtigungen der adaptiven Authentifizierung für HTML- und Klartext-Versionen von E-Mail-Nachrichten

anzupassen. Weitere Informationen über E-Mail-Nachrichtenvorlagen finden Sie unter Nachrichtenvorlagen.

8. Erweitern Sie IP address exceptions (IP-Adressausnahmen), um eine Immer zulassen- oder eine Immer blockieren-Liste für IPv4- oder IPv6-Adressbereiche zu erstellen, die immer zugelassen oder blockiert werden – unabhängig von der fortgeschrittenen Sicherheitsrisikobewertung. Geben Sie die IP-Adressbereiche in CIDR-Notation an (z. B. 192.168.100.0/24).
9. Wählen Sie Save Changes.

Überprüfung auf kompromittierte Anmeldeinformationen

Amazon Cognito kann erkennen, wenn Benutzername und Passwort eines Benutzers an anderer Stelle kompromittiert wurden. Dies kann der Fall sein, wenn Benutzer Anmeldeinformationen auf mehr als einer Website verwenden, oder wenn sie einfach zu erratende Passwörter auswählen. Amazon Cognito überprüft lokale Benutzer, die sich mit Benutzernamen und Passwort anmelden, in der gehosteten Benutzeroberfläche und mit der Amazon-Cognito-API. Ein lokaler Benutzer existiert ausschließlich in Ihrem Benutzerpool-Verzeichnis, ohne dass ein Verbund über einen externen IdP besteht.

Unter Advanced security (Erweiterte Sicherheit) auf der Registerkarte App integration (App-Integration) der Amazon-Cognito-Konsole können Sie Compromised credentials (Kompromittierte Anmeldeinformationen) konfigurieren. Konfigurieren Sie Event detection (Ereigniserkennung), um die Benutzerereignisse auszuwählen, die Sie auf kompromittierte Anmeldeinformationen überwachen möchten. Konfigurieren Sie Compromised credentials responses (Antworten auf kompromittierte Anmeldeinformationen), um zu entscheiden, ob der Benutzer bei Erkennung kompromittierter Anmeldeinformationen zugelassen oder blockiert werden soll. Amazon Cognito kann bei der Registrierung, Anmeldung und bei Passwortänderungen auf kompromittierte Anmeldeinformationen prüfen.

Wenn Sie Anmeldung zulassen wählen, können Sie Amazon CloudWatch Logs überprüfen, um die Bewertungen zu überwachen, die Amazon Cognito zu Benutzerereignissen vornimmt. Weitere Informationen finden Sie unter Anzeigen erweiterter Sicherheitsmetriken. Wenn Sie Block sign-in (Anmeldung blockieren) auswählen, verhindert Amazon Cognito die Anmeldung von Benutzern, die kompromittierte Anmeldeinformationen verwenden. Wenn Amazon Cognito die Anmeldung für einen Benutzer blockiert, wird der `UserStatus` des Benutzers auf `RESET_REQUIRED` festgelegt. Ein Benutzer mit dem Status `RESET_REQUIRED` muss sein Passwort ändern, bevor er sich erneut anmelden kann.

> **Note**
>
> Derzeit führt Amazon Cognito bei der Anmeldung mit SRP (Secure Remote Password) keine Prüfung auf kompromittierte Anmeldeinformationen durch. SRP sendet bei der Anmeldung einen Hash-Passwortnachweis. Amazon Cognito hat keinen internen Zugriff auf Passwörter und kann daher nur ein Passwort auswerten, das Ihr Kunde ihm im Klartext übergibt. Amazon Cognito überprüft Anmeldungen, die die API mit Flow und die AdminInitiateAuthInitiateAuthAPI mit `ADMIN_USER_PASSWORD_AUTH` `USER_PASSWORD_AUTH` Flow verwenden, auf kompromittierte Anmeldeinformationen.

Weitere Informationen darüber, wie Sie Ihrem Benutzerpool einen Schutz gegen nicht mehr zuverlässige Anmeldeinformationen hinzufügen, finden Sie unter Hinzufügen erweiterter Sicherheit zu einem Benutzerpool..

Verwendung der adaptiven Authentifizierung

Mit der adaptiven Authentifizierung können Sie Ihren Benutzerpool so konfigurieren, dass verdächtige Anmeldungen blockiert werden, oder dass ein zweiter Faktor gefordert wird, wenn ein erhöhtes Risiko besteht. Bei jedem Anmeldeversuch generiert Amazon Cognito eine Risikobewertung, die angibt, wie wahrscheinlich es ist, dass die Anmeldeanforderung von einer kompromittierten Quelle stammt. Diese Risikobewertung basiert auf Faktoren, zu denen Geräte- und Benutzerinformationen gehören. Die adaptive Authentifizierung kann die Multi-Faktor-Authentifizierung (MFA) für einen Benutzer in Ihrem Benutzerpool aktivieren oder erfordern, wenn Amazon Cognito ein Risiko in einer Benutzersitzung erkennt und der Benutzer noch keine MFA-Methode ausgewählt hat. Wenn Sie MFA für einen Benutzer aktivieren, wird dieser immer aufgefordert, während der Authentifizierung einen zweiten Faktor bereitzustellen oder einzurichten, unabhängig davon, wie Sie die adaptive Authentifizierung konfiguriert haben. Aus Sicht Ihrer Benutzer bietet Ihre App Unterstützung bei der Einrichtung von MFA, und optional verhindert Amazon Cognito, dass sie sich erneut anmelden, bevor sie einen zusätzlichen Faktor konfiguriert haben.

Amazon Cognito veröffentlicht Anmeldeversuche, deren Risikolevel und fehlgeschlagene Anfragen an Amazon. CloudWatch Weitere Informationen finden Sie unter Anzeigen erweiterter Sicherheitsmetriken.

Weitere Informationen darüber, wie Sie Ihrem Benutzerpool eine adaptive Authentifizierung hinzufügen, finden Sie unter Hinzufügen erweiterter Sicherheit zu einem Benutzerpool..

Themen

- Übersicht über die adaptive Authentifizierung
- Hinzufügen von Benutzergeräte- und Sitzungsdaten zu API-Anforderungen
- Anzeige des Ereignisverlaufs des Benutzers
- Bereitstellung von Ereignisfeedback
- Senden von Benachrichtigungsmeldungen

Übersicht über die adaptive Authentifizierung

Auf der Seite Erweiterte Sicherheit auf der Registerkarte App-Integration der Amazon-Cognito-Konsole können Sie Einstellungen für die adaptive Authentifizierung festlegen, darunter auch, welche Aktionen für unterschiedliche Risikoebenen ausgeführt werden und welche Benachrichtigungen an Benutzer gesendet werden. Sie können allen Ihren App-Clients eine globale erweiterte Sicherheitskonfiguration zuweisen, jedoch eine Konfiguration auf Client-Ebene auf einzelne App-Clients anwenden.

Die adaptive Authentifizierung von Amazon Cognito weist jeder Benutzersitzung eine der folgenden Risikostufen zu: Hoch, Mittel, Niedrig oder Kein Risiko.

Wägen Sie Ihre Optionen sorgfältig ab, wenn Sie Ihre Enforcement method (Durchsetzungsmethode) von Audit-only (Nur Audit) zu Full-fuction (Vollfunktionsfähig) ändern. Die automatischen Antworten, die Sie auf Risikoniveaus anwenden, beeinflussen das Risikoniveau, das Amazon Cognito nachfolgenden Benutzersitzungen mit denselben Merkmalen zuweist. Wenn Sie sich beispielsweise dafür entschieden haben, keine Maßnahme zu ergreifen oder Allow (Zulassen) für Nutzersitzungen zu wählen, die Amazon Cognito zunächst als risikoreich einstuft, geht Amazon Cognito davon aus, dass ähnliche Sitzungen ebenfalls ein geringeres Risiko aufweisen.

Sie können für jede Risikoebene folgende Optionen festlegen:

Option	Action
Sobald Sie die Details auf dieser Seite überprüft haben, klicken Sie auf	Benutzer können sich ohne zusätzlichen Faktor anmelden.
Optionale MFA	Benutzer mit einem konfigurierten zweiten Faktor müssen eine zweite Faktoraufforderung abschließen, um sich anmelden zu können. Als zweite Faktoren stehen eine Telefonnu

Option	Action
	mmer für SMS und ein TOTP-Softwaretoken zur Verfügung. Benutzer ohne einen zweiten konfigurierten Faktor können sich mit nur einem Satz von Anmeldeinformationen anmelden.
MFA erforderlich	Benutzer mit einem konfigurierten zweiten Faktor müssen eine zweite Faktoraufforderung abschließen, um sich anmelden zu können. Amazon Cognito blockiert die Anmeldung für Benutzer, die keinen zweiten Faktor konfiguriert haben.
Blockieren	Amazon Cognito blockiert alle Anmeldeve rsuche auf der festgelegten Risikoebene.

> **Note**
>
> Sie müssen Telefonnummern nicht verifizieren, um sie für SMS als zweiter Authentifizierungsfaktor verwenden zu können.

Hinzufügen von Benutzergeräte- und Sitzungsdaten zu API-Anforderungen

Sie können Informationen über die Sitzung Ihres Benutzers sammeln und an die erweiterte Sicherheit von Amazon Cognito weitergeben, wenn Sie die API verwenden, um ihn zu registrieren, anzumelden und sein Passwort zurückzusetzen. Zu diesen Informationen gehören die IP-Adresse des Benutzers und eine eindeutige Gerätekennung.

Möglicherweise verfügen Sie über ein zwischengelagertes Netzwerkgerät zwischen Ihren Benutzern und Amazon Cognito, wie einen Proxy-Service oder einen Anwendungsserver. Sie können die Kontextdaten der Benutzer sammeln und an Amazon Cognito weitergeben, damit die adaptive Authentifizierung Ihr Risiko basierend auf den Eigenschaften des Benutzerendpunkts anstelle Ihres Servers oder Proxys berechnet. Wenn Ihre clientseitige App API-Operationen von Amazon Cognito direkt aufruft, zeichnet die adaptive Authentifizierung automatisch die Quell-IP-Adresse auf. Es werden jedoch keine anderen Geräteinformationen wie `user-agent` aufgezeichnet, es sei denn, Sie erfassen auch eine Geräteidentifikation.

Generieren Sie diese Daten mit der Amazon Cognito Context-Datenerfassungsbibliothek und senden Sie sie mit den Parametern ContextDataund UserContextDataan Amazon Cognito Advanced Security. Die Bibliothek zur Erfassung von Kontextdaten ist in den AWS SDKs enthalten. Weitere Informationen finden Sie unter Integration von Amazon Cognito mit Web- und mobilen Apps. Sie können `ContextData` senden, wenn Sie erweiterte Sicherheitsfunktionen in Ihrem Benutzerpool aktiviert haben. Weitere Informationen finden Sie unter Konfiguration der erweiterten Sicherheitsfunktionen.

Wenn Sie die folgenden von Amazon Cognito authentifizierten API-Operationen von Ihrem Anwendungsserver aus aufrufen, übergeben Sie die IP des Geräts des Benutzers im Parameter `ContextData`. Übergeben Sie außerdem Ihren Servernamen, Serverpfad und verschlüsselte Geräteidentifikationsdaten.

- AdminInitiateAuth
- AdminRespondToAuthChallenge

Wenn Sie nicht authentifizierte API-Operationen von Amazon Cognito aufrufen, können Sie `UserContextData` an die erweiterten Sicherheitsfunktionen von Amazon Cognito senden. Diese Daten enthalten eine Geräteidentifikation im Parameter `EncodedData`. Sie können auch einen Parameter `IpAddress` in den `UserContextData` übergeben, wenn Sie die folgenden Bedingungen erfüllen:

- Sie haben die erweiterten Sicherheitsfunktionen in Ihrem Benutzerpool aktiviert. Weitere Informationen finden Sie unter Konfiguration der erweiterten Sicherheitsfunktionen.
- Ihr App-Client verfügt über einen Client-Schlüssel. Weitere Informationen finden Sie unter Konfigurieren eines Benutzerpool-App-Clients.
- Sie haben Accept additional user context data (Zusätzliche Benutzerkontextdaten akzeptieren) in Ihrem App-Client aktiviert. Weitere Informationen finden Sie unter Akzeptieren zusätzlicher Benutzerkontextdaten (AWS Management Console).

Ihre App kann den Parameter `UserContextData` mit verschlüsselten Geräteidentifikationsdaten und der IP-Adresse des Geräts des Benutzers in den folgenden nicht authentifizierten API-Operationen von Amazon Cognito auffüllen.

- InitiateAuth
- RespondToAuthChallenge

- SignUp
- ConfirmSignUp
- ForgotPassword
- ConfirmForgotPassword
- ResendConfirmationCode

Akzeptieren zusätzlicher Benutzerkontextdaten (AWS Management Console)

Ihr Benutzerpool akzeptiert eine IP-Adresse in einem Parameter `UserContextData` nach dem Aktivieren der Funktion Accept additional user context data (Zusätzliche Benutzerkontextdaten akzeptieren). Sie müssen diese Funktion in folgenden Fällen nicht aktivieren:

- Ihre Benutzer melden sich nur mit authentifizierten API-Vorgängen wie an AdminInitiateAuth , und Sie verwenden den `ContextData` Parameter.
- Ihre nicht authentifizierten API-Operationen sollen nur eine Geräteidentifikation, aber keine IP-Adresse, an erweiterte Sicherheitsfunktionen von Amazon Cognito senden.

Aktualisieren Sie Ihren App-Client wie folgt in der Amazon-Cognito-Konsole, um ihn um Unterstützung für zusätzliche Benutzerkontextdaten zu erweitern.

1. Melden Sie sich bei der Amazon Cognito-Konsole an.
2. Wählen Sie im Navigationsbereich erst Manage your User Pools (Eigene Benutzerpools verwalten) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte App integration (Anwendungsintegration) aus.
4. Wählen oder erstellen Sie einen App-Client unter App clients and analytics (App-Clients und Analysen). Weitere Informationen finden Sie unter Konfigurieren eines Benutzerpool-App-Clients.
5. Wählen Sie Edit (Bearbeiten) aus dem Container App client information (App-Client-Informationen) aus.
6. Wählen Sie in Advanced authentication settings (Erweiterte Authentifizierungseinstellungen) für Ihren App-Client Accept additional user context data (Zusätzliche Benutzerkontextdaten akzeptieren) aus.
7. Wählen Sie Änderungen speichern aus.

Um Ihren App-Client so zu konfigurieren, dass er Benutzerkontextdaten in der Amazon Cognito Cognito-API akzeptiert, setzen Sie `EnablePropagateAdditionalUserContextData` ihn `true` in einer CreateUserPoolClientUpdateUserPoolClientOder-Anfrage auf. Weitere Informationen zum Aktivieren der erweiterten Sicherheit von Ihrer Web- oder mobilen App aus finden Sie unter Aktivieren der erweiterten Benutzerpool-Sicherheit über Ihre App. Erfassen Sie Benutzerkontextdaten von der Client-Seite, wenn Ihre App Amazon Cognito von Ihrem Server aufruft. Im Folgenden finden Sie ein Beispiel, das die JavaScript SDK-Methode `getData` verwendet.

```
var encodedData =
 AmazonCognitoAdvancedSecurityData.getData(username, userPoolId, clientId);
```

Wenn Sie Ihre App zur Verwendung der adaptiven Authentifizierung entwerfen, empfehlen wir Ihnen, das neueste Amazon Cognito SDK in Ihre App zu integrieren. Die neueste Version des SDK sammelt Daten zur Geräteidentifizierung wie Geräte-ID, Modell und Zeitzone. Weitere Informationen über Amazon-Cognito-SDKs finden Sie unter Installieren eines Benutzerpool-SDK. Die erweiterte Sicherheit von Amazon Cognito speichert und weist nur Ereignissen, die Ihre App im richtigen Format übermittelt, eine Risikobewertung zu. Wenn Amazon Cognito eine Fehlermeldung zurückgibt, überprüfen Sie, ob Ihre Anforderung einen gültigen geheimen Hash enthält und ob es sich beim Parameter `IPaddress` um eine gültige IPv4- oder IPv6-Adresse handelt.

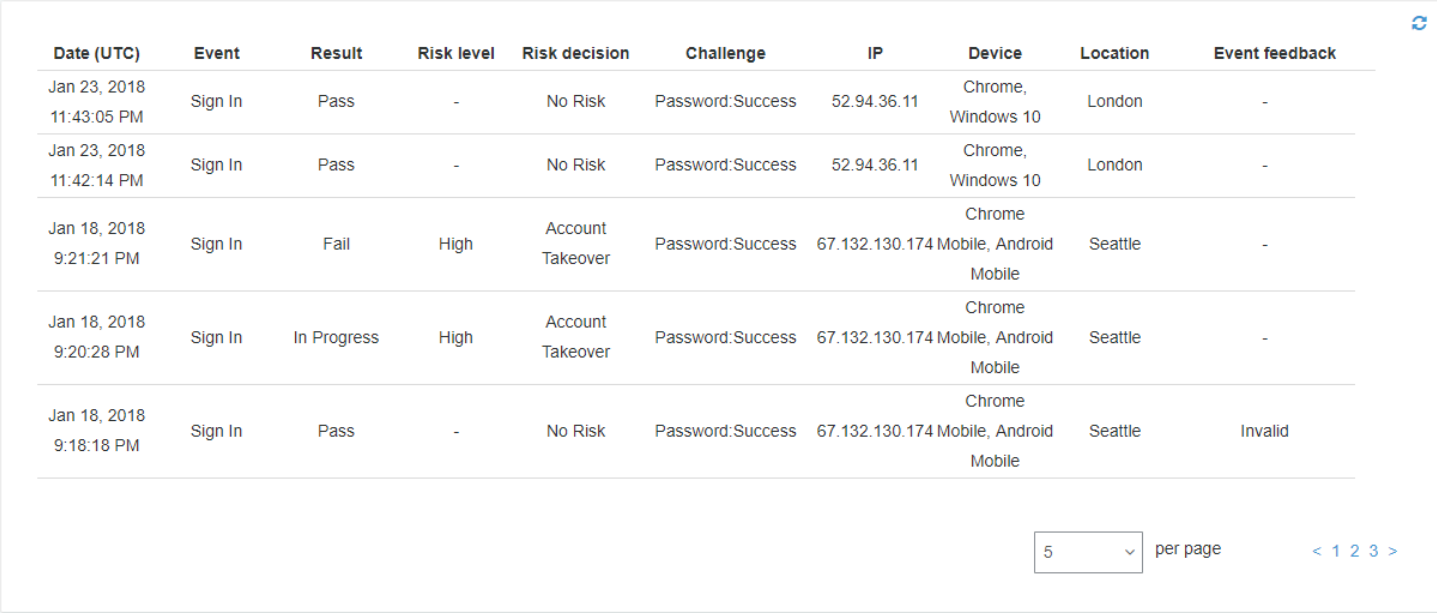
`ContextData`- und **`UserContextData`**-Ressourcen

- AWS Amplify SDK for Android: GetUserContextData
- AWS Amplify SDK for iOS: userContextData
- JavaScript: amazon-cognito-advanced-security-data.min.js

Anzeige des Ereignisverlaufs des Benutzers

> **Note**
>
> In der neuen Amazon-Cognito-Konsole können Sie den Benutzerereignisverlauf auf der Registerkarte Users (Benutzer) anzeigen.

Sie können einen Benutzer in der Amazon-Cognito-Konsole auf der Registerkarte Benutzer auswählen, um den Anmeldungsverlauf für einen Benutzer anzuzeigen. Der Ereignisverlauf für Benutzer wird von Amazon Cognito für zwei Jahre aufbewahrt.



Date (UTC)	Event	Result	Risk level	Risk decision	Challenge	IP	Device	Location	Event feedback
Jan 23, 2018 11:43:05 PM	Sign In	Pass	-	No Risk	Password:Success	52.94.36.11	Chrome, Windows 10	London	-
Jan 23, 2018 11:42:14 PM	Sign In	Pass	-	No Risk	Password:Success	52.94.36.11	Chrome, Windows 10	London	-
Jan 18, 2018 9:21:21 PM	Sign In	Fail	High	Account Takeover	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	-
Jan 18, 2018 9:20:28 PM	Sign In	In Progress	High	Account Takeover	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	-
Jan 18, 2018 9:18:18 PM	Sign In	Pass	-	No Risk	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	Invalid

5 per page < 1 2 3 >

Jedes Anmeldeereignis hat eine Ereignis-ID. Das Ereignis enthält auch entsprechende Kontextdaten wie dem Standort, Gerätdetails sowie den Risikoerkennungsergebnissen. Sie können den Benutzerereignisverlauf mit der Amazon Cognito Cognito-API-Operation AdminListUserAuthEventsoder mit AWS Command Line Interface (AWS CLI) mit admin-list-user-auth-events abfragen.

Sie können die Ereignis-ID auch mit dem Token abgleichen, das Amazon Cognito zum Zeitpunkt der Aufzeichnung des Ereignisses ausgegeben hat. Die ID- und Zugriffstoken enthalten diese Ereignis-ID in ihrer Nutzlast. Außerdem gleicht Amazon Cognito die Verwendung von Aktualisierungstoken mit der ursprünglichen Ereignis-ID ab. Sie können die ursprüngliche Ereignis-ID bis zur Ereignis-ID des Anmeldeereignisses zurückverfolgen, das zur Ausgabe des Amazon-Cognito-Tokens führte. Sie können die Nutzung eines Tokens in Ihrem System auf ein bestimmtes Authentifizierungsereignis zurückführen. Weitere Informationen finden Sie unter Verwenden von Token mit Benutzerpools.

Bereitstellung von Ereignisfeedback

Ereignisfeedback wirkt sich in Echtzeit auf die Risikobewertung aus und verbessert langfristig den Risikobewertungsalgorithmus. Sie können Feedback zur Gültigkeit von Anmeldeversuchen über die Amazon-Cognito-Konsole und -API-Operationen bereitstellen.

> **Note**
>
> Ihr Ereignisfeedback beeinflusst das Risikoniveau, das Amazon Cognito nachfolgenden Benutzersitzungen mit denselben Merkmalen zuweist.

Wählen Sie in der Amazon-Cognito-Konsole auf der Registerkarte Users (Benutzer) einen Benutzer aus und wählen Sie Provide event feedback (Ereignisfeedback bereitstellen) aus. Sie können die Veranstaltungsdetails überprüfen und Set as valid (Als gültig festlegen) oder Set as invalid (Als ungültig festlegen).

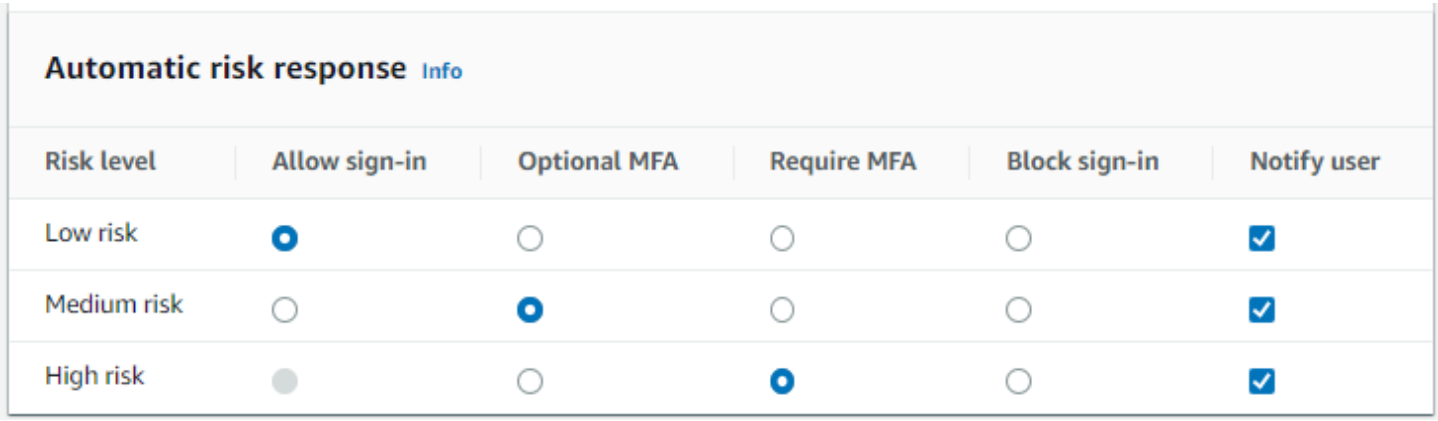
Die Konsole listet den Anmeldeverlauf auf der Registerkarte Benutzer und Gruppen auf. Wenn Sie einen Eintrag auswählen, können Sie das Ereignis als gültig oder ungültig markieren. Sie können Feedback auch über den API-Vorgang AdminUpdateAuthEventFeedbackfür den Benutzerpool und über den AWS CLI Befehl admin-update-auth-event -feedback geben.

Wenn Sie in der Amazon-Cognito-Konsole die Option Set as valid (Als gültig festlegen) auswählen oder für `valid` in der API einen Wert von `FeedbackValue` angeben, teilen Sie Amazon Cognito mit, dass Sie einer Benutzersitzung vertrauen, für die Amazon Cognito ein gewisses Risiko bewertet hat. Wenn Sie in der Amazon-Cognito-Konsole die Option Set as invalid (Als ungültig festlegen) auswählen oder für `invalid` in der API einen Wert von `FeedbackValue` angeben, teilen Sie Amazon Cognito mit, dass Sie einer Benutzersitzung nicht vertrauen, oder dass Sie nicht glauben, dass Amazon Cognito ein ausreichend hohes Risikoniveau bewertet hat.

Senden von Benachrichtigungsmeldungen

Mit erweitertem Sicherheitsschutz kann Amazon Cognito Ihre Benutzer über risikobehaftete Anmeldeversuche informieren. Amazon Cognito kann Benutzer auch zur Auswahl von Links auffordern, um anzugeben, ob die Anmeldung gültig oder ungültig war. Amazon Cognito verwendet dieses Feedback, um die Genauigkeit der Risikoerkennung für Ihren Benutzerpool zu verbessern.

Wählen Sie im Abschnitt Automatic risk response (Automatische Reaktion auf Risiken) die Option Notify Users (Benutzer benachrichtigen) für die Fälle mit niedrigem, mittlerem und hohem Risiko.



Amazon Cognito sendet E-Mail-Benachrichtigungen an Ihre Benutzer, unabhängig davon, ob sie ihre E-Mail-Adresse bestätigt haben.

Sie können E-Mail-Benachrichtigungen anpassen und sowohl Klartextversionen als auch HTML-Versionen bereitstellen. Um Ihre E-Mail-Benachrichtigungen anzupassen, öffnen Sie Email templates (E-Mail-Vorlagen) unter Adaptive authentication messages (Adaptive Authentifizierungsnachrichten) in Ihrer erweiterten Sicherheitskonfiguration. Weitere Informationen über E-Mail-Vorlagen finden Sie unter Nachrichtenvorlagen.

Anzeigen erweiterter Sicherheitsmetriken

Amazon Cognito veröffentlicht Metriken für erweiterte Sicherheitsfunktionen für Ihr Konto bei Amazon CloudWatch. Die erweiterten Sicherheitsmetriken von Amazon Cognito sind nach Risikoebene und Anforderungsebene gruppiert.

Um Metriken in der CloudWatch Konsole anzuzeigen

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metriken aus.
3. Wählen Sie Amazon Cognito.
4. Wählen Sie eine Gruppe von aggregierten Metriken wie By Risk Classification (Nach Risikoeinstufung) aus.
5. Auf der Registerkarte All metrics (Alle Metriken) werden alle Metriken für diese Auswahl angezeigt. Sie haben die folgenden Möglichkeiten:

   - Um die Tabelle sortieren, verwenden Sie die Spaltenüberschrift.
   - Um eine Metrik grafisch darzustellen, müssen Sie das Kontrollkästchen neben der Metrik aktivieren. Um alle Metriken auszuwählen, aktivieren Sie das Kontrollkästchen in der Kopfzeile der Tabelle.
   - Um nach Ressource zu filtern, müssen Sie zunächst die Ressourcen-ID und dann die Option Add to search (Zu Suche hinzufügen) wählen.
   - Um nach Metrik zu filtern, müssen Sie den Metriknamen und anschließend Add to search (Zu Suche hinzufügen) wählen.

Metrik	Beschreibung	Metrikdimensionen
CompromisedCredentialRisk	Anforderungen, bei denen Amazon Cognito kompromit	Vorgang: Die Arten des Vorgangs `PasswordChange` , `SignIn` oder

Metrik	Beschreibung	Metrikdimensionen
	tierte Anmeldeinformationen erkannt hat	`SignUp` sind die einzigen Dimensionen. UserPoolId: Die Kennung des Benutzerpools. RiskLevel: hoch (Standard), mittel oder niedrig.
AccountTakeoverRisk	Anforderungen, bei denen Amazon Cognito ein Risiko zur Kontoübernahme erkannt hat.	Vorgang: Die Arten des Vorgangs `PasswordChange` , `SignIn` oder `SignUp` sind die einzigen Dimensionen. UserPoolId: Die Kennung des Benutzerpools. RiskLevel: hoch, mittel oder niedrig.
OverrideBlock	Anforderungen, die von Amazon Cognito aufgrund der vom Entwickler vorgenomm enen Konfiguration geblockt hat	Vorgang: Die Arten des Vorgangs `PasswordChange` , `SignIn` oder `SignUp` sind die einzigen Dimensionen. UserPoolId: Die Kennung des Benutzerpools. RiskLevel: hoch, mittel oder niedrig.

Metrik	Beschreibung	Metrikdimensionen
Risk	Anforderungen, die von Amazon Cognito als riskant gekennzeichnet wurden	Vorgang: Der Typ des Vorgangs, z. B. `PasswordChange` , `SignIn` oder `SignUp`. UserPoolId: Die Kennung des Benutzerpools.
NoRisk	Anforderungen, die von Amazon Cognito als nicht riskant eingestuft wurden.	Vorgang: Der Typ des Vorgangs, z. B. `PasswordChange` , `SignIn` oder `SignUp`. UserPoolId: Die Kennung des Benutzerpools.

Amazon Cognito bietet Ihnen zwei vordefinierte Gruppen von Metriken, in CloudWatch denen Sie sofort analysieren können. By Risk Classification (Nach Risikoeinstufung) legt die Granularität der Risikoebene für Anforderungen fest, die Amazon Cognito als riskant eingestuft. By Request Classification (Nach Anforderungseinstufung) spiegelt Metriken wider, die nach Anforderungsebene aggregiert sind.

Aggregierte Metrikgruppen	Beschreibung
By Risk Classification	Anforderungen, die von Amazon Cognito als riskant eingestuft wurden.
By Request Classification	Metriken, die nach Anforderung aggregiert wurden

Aktivieren der erweiterten Benutzerpool-Sicherheit über Ihre App

Nach der Konfiguration der erweiterten Sicherheitsfunktionen für Ihren Benutzerpool müssen Sie sie in Ihrer Web- oder mobilen App aktivieren.

Verwenden Sie erweiterte Sicherheit mit JavaScript

1. Fügen Sie das Amazon Cognito Identity SDK für JavaScript zu Ihrer App hinzu.
2. Stellen Sie in CognitoUserPool.js `AdvancedSecurityDataCollectionFlag` auf `true` ein. Setzen Sie `UserPoolId` auf Ihre Benutzerpool-ID.
3. Fügen Sie diesen Quellverweis zur JavaScript Datei Ihrer App hinzu. Ersetze ihn `<region>` durch einen AWS-Region aus der folgenden Liste: `us-east-1us-east-2,us-west-2,,eu-west-1,eu-west-2`, oder`eu-central-1`.

```
<script src="https://amazon-cognito-assets.<region>.amazoncognito.com/amazon-cognito-advanced-security-data.min.js"></script>
```

Verwenden der erweiterten Sicherheit mit Android

1. Erstellen Sie Ihre App mit AWS Amplify für Android. Weitere Informationen finden Sie unter Projekteinrichtung im AWS Amplify Dev Center.
2. Fügen Sie mit `userContextDataProvider` Benutzer- und Geräteinformationen in Ihre Authentifizierungsanfragen ein.

Informationen zum Hinzufügen von Benutzerkontextdaten im alten Android-SDK finden Sie unter aws-android-sdkcognito-identityprovider-asf.

Verwenden der erweiterten Sicherheit mit iOS

1. Erstellen Sie Ihre App mit AWS Amplify für Swift oder Flutter. Weitere Informationen finden Sie unter Swift-Projekteinrichtung und Flutter-Projekteinrichtung im AWS Amplify Dev Center.
2. Nehmen Sie Benutzer- und Geräteinformationen in Ihre Authentifizierungsanfragen auf. Ein Beispiel zur Verwendung mit der InitiateAuthAPI-Operation finden Sie unter `userContextData` InitiateAuthInput+amplify.Swift on. GitHub

Hinweise zum Hinzufügen von Benutzerkontextdaten im älteren iOS-SDK finden Sie unterAWSCognitoIdentityProviderASF.

Eine AWS WAF Web-ACL einem Benutzerpool zuordnen

AWS WAF ist eine Firewall für Webanwendungen. Mit einer AWS WAF Web-Zugriffskontrollliste (Web ACL) können Sie Ihren Benutzerpool vor unerwünschten Anfragen an Ihre gehostete

Benutzeroberfläche und Amazon Cognito API-Serviceendpunkte schützen. Eine Web-ACL ermöglicht eine differenziertere Kontrolle aller HTTPS-Webanforderungen, auf die Ihr Benutzerpool reagiert. Weitere Informationen zu AWS WAF Web-ACLs finden Sie unter Verwalten und Verwenden einer Web-Zugriffskontrollliste (Web-ACL) im AWS WAF Entwicklerhandbuch.

Wenn Sie eine AWS WAF Web-ACL mit einem Benutzerpool verknüpft haben, leitet Amazon Cognito ausgewählte nicht vertrauliche Header und Inhalte von Anfragen Ihrer Benutzer an weiter. AWS WAF AWS WAF untersucht den Inhalt der Anfrage, vergleicht sie mit den Regeln, die Sie in Ihrer Web-ACL angegeben haben, und gibt eine Antwort an Amazon Cognito zurück.

Wissenswertes über AWS WAF Web-ACLs und Amazon Cognito

- Anfragen, die von blockiert wurden, werden für AWS WAF keinen Anfragetyp auf das Kontingent für die Anforderungsrate angerechnet. Der AWS WAF Handler wird vor den Drosselungsprozeduren auf API-Ebene aufgerufen.
- Wenn Sie eine Web-ACL erstellen, dauert es ein wenig, bis die Web-ACL vollständig weitergegeben wurde und für Amazon Cognito verfügbar ist. Die Übertragungszeit kann zwischen einigen Sekunden und mehreren Minuten liegen. AWS WAF gibt a zurück `WAFUnavailableEntityException`, wenn Sie versuchen, eine Web-ACL zuzuordnen, bevor sie vollständig weitergegeben wurde.
- Sie können einem Benutzerpool jeweils eine Web-ACL zuordnen.
- Ihre Anfrage könnte zu einer Nutzlast führen, die die Grenzwerte für das übersteigt, was AWS WAF überprüfen kann. Unter Behandlung übergroßer Anforderungskomponenten im AWS WAF Entwicklerhandbuch erfahren Sie, wie Sie konfigurieren können, wie mit übergroßen Anfragen von Amazon Cognito AWS WAF umgegangen wird.
- Sie können eine Web-ACL, die AWS WAF Fraud Control Account Takeover Prevention (ATP) verwendet, nicht mit einem Amazon Cognito Cognito-Benutzerpool verknüpfen. Sie implementieren die ATP-Funktion, wenn Sie die verwaltete Regelgruppe `AWS-AWSManagedRulesATPRuleSet` hinzufügen. Stellen Sie sicher, dass Ihre Web-ACL diese verwaltete Regelgruppe nicht verwendet, bevor Sie sie einem Benutzerpool zuordnen.
- Wenn Sie eine AWS WAF Web-ACL mit einem Benutzerpool verknüpft haben und eine Regel in Ihrer Web-ACL ein CAPTCHA enthält, kann dies zu einem nicht behebbaren Fehler bei der TOTP-Registrierung für gehostete Benutzeroberflächen führen. Informationen zum Erstellen einer Regel mit einer CAPTCHA-Aktion, die sich nicht auf TOTP der gehosteten Benutzeroberfläche auswirkt, finden Sie unter Konfiguration Ihrer AWS WAF Web-ACL für Hosted UI TOTP MFA.

AWS WAF untersucht Anfragen an die folgenden Endpunkte.

Gehostete Benutzeroberfläche

Anforderungen aller Endpunkte in der Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI.

Öffentliche API-Operationen

Anfragen von Ihrer App an die Amazon Cognito Cognito-API, die keine AWS Anmeldeinformationen zur Autorisierung verwenden. Dazu gehören API-Operationen wie InitiateAuthRespondToAuthChallenge, und. GetUser Die API-Operationen, die in den Geltungsbereich von fallen, erfordern AWS WAF keine Authentifizierung mit AWS Anmeldeinformationen. Sie sind entweder nicht authentifiziert oder mit einer Sitzungszeichenfolge bzw. einem Zugriffstoken autorisiert. Weitere Informationen finden Sie unter Authentifizierte und nicht authentifizierte API-Operationen der Amazon-Cognito-Benutzerpools.

Sie können die Regeln in Ihrer Web-ACL mit den Regelaktionen Count (Zählen), Allow (Zulassen) und Block (Blockieren) konfigurieren oder ein CAPTCHA als Antwort auf eine Anfrage anzeigen, die mit einer Regel übereinstimmt. Weitere Informationen finden Sie unter AWS WAF -Regeln im Entwicklerhandbuch zu AWS WAF . Abhängig von der Regelaktion können Sie die Antwort anpassen, die Amazon Cognito an Ihre Benutzer zurückgibt.

Important

Ihre Optionen zum Anpassen der Antwort im Fehlerfall hängen von der Art und Weise ab, wie Sie eine API-Anforderung vornehmen.

- Sie können den Fehlercode und den Antworttext von Anforderungen der gehosteten Benutzeroberfläche anpassen. Sie können nur in der gehosteten Benutzeroberfläche ein CAPTCHA anzeigen, das Ihr Besucher lösen muss.
- Für Anforderungen, die Sie mit der Benutzerpool-API von Amazon Cognito vornehmen, können Sie den Antworttext einer Anforderung anpassen, die eine Blockieren-Antwort erhält. Sie können auch einen benutzerdefinierten Fehlercode im Bereich 400 bis 499 angeben.
- Die SDKs AWS Command Line Interface (AWS CLI) und die AWS SDKs geben bei Anfragen, die eine Block - oder CAPTCHA-Antwort erzeugen, einen `ForbiddenException` Fehler zurück.

Zuordnen einer Web-ACL zu Ihrem Benutzerpool

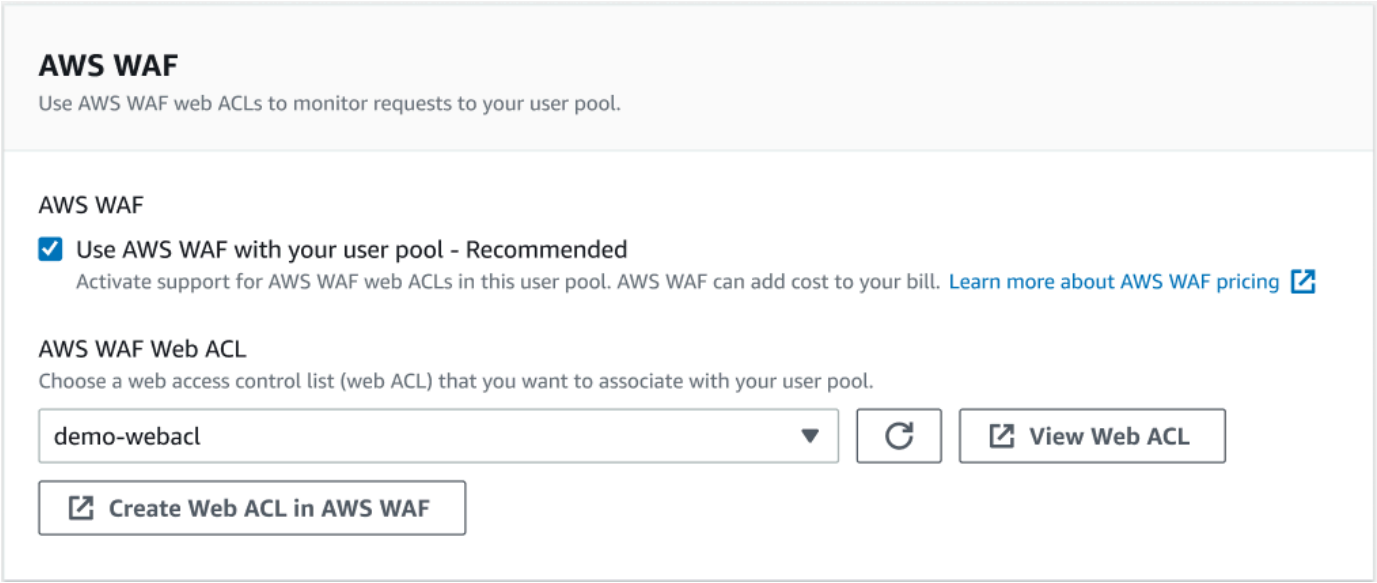
Um mit einer Web-ACL in Ihrem Benutzerpool arbeiten zu können, muss Ihr AWS Identity and Access Management (IAM-) Principal über die folgenden Amazon Cognito Cognito-Berechtigungen verfügen. Informationen zu AWS WAF Berechtigungen finden Sie unter AWS WAF API-Berechtigungen im AWS WAF Entwicklerhandbuch.

- `cognito-idp:AssociateWebACL`
- `cognito-idp:DisassociateWebACL`
- `cognito-idp:GetWebACLForResource`
- `cognito-idp:ListResourcesForWebACL`

Auch wenn Sie IAM-Berechtigungen erteilen müssen, sind die aufgelisteten Aktionen nur für Berechtigungen bestimmt und entsprechen keiner API-Operation.

Um sie AWS WAF für Ihren Benutzerpool zu aktivieren und eine Web-ACL zuzuordnen

1. Melden Sie sich bei der Amazon Cognito-Konsole an.
2. Wählen Sie im Navigationsbereich erst User Pools (Benutzerpools) aus und anschließend den Benutzerpool, den Sie bearbeiten möchten.
3. Wählen Sie die Registerkarte User pool properties (Benutzerpool-Eigenschaften).
4. Wählen Sie Edit (Bearbeiten) neben AWS WAF.
5. Wählen Sie AWS WAFunter AWS WAF Mit Ihrem Benutzerpool verwenden aus.



6. Wählen Sie eine AWS WAF Web-ACL, die Sie bereits erstellt haben, oder wählen Sie Web-ACL erstellen in, AWS WAF um eine Web-ACL in einer neuen AWS WAF Sitzung in der zu erstellen AWS Management Console.
7. Wählen Sie Änderungen speichern aus.

Verwenden Sie ACL aus der API, um Ihrem Benutzerpool im AWS Command Line Interface oder einem SDK programmgesteuert eine AssociateWebWeb-ACL zuzuordnen. AWS WAF Amazon Cognito verfügt nicht über eine separaten API-Operation zum Zuordnen einer Web-ACL.

Web-ACLs testen und protokollieren AWS WAF

Wenn Sie eine Regelaktion in Ihrer Web-ACL auf Count setzen, wird die Anfrage einer Anzahl von Anfragen AWS WAF hinzugefügt, die der Regel entsprechen. Setzen Sie zum Testen einer Web-ACL mit Ihrem Benutzerpool die Regelaktionen auf Count (Zählen) und berücksichtigen Sie das Volumen der Anforderungen, die mit den einzelnen Regeln übereinstimmen. Wenn beispielsweise eine Regel, die Sie auf die Aktion Block (Blockieren) setzen möchten, mit einer großen Anzahl von Anforderungen übereinstimmt, die Sie als normalen Benutzerverkehr einstufen. müssen Sie Ihre Regel möglicherweise neu konfigurieren. Weitere Informationen finden Sie im AWS WAF Entwicklerhandbuch unter Testen und Optimieren Ihres AWS WAF Schutzes.

Sie können auch so konfigurieren AWS WAF , dass Anforderungsheader in einer Amazon CloudWatch Logs-Protokollgruppe, einem Amazon Simple Storage Service (Amazon S3) -Bucket oder einer Amazon Data Firehose protokolliert werden. Sie können die Amazon-Cognito-Anforderungen, die Sie mit der Benutzerpool-API vornehmen, anhand von `x-amzn-cognito-client-id` und `x-amzn-cognito-operation-name` identifizieren. Anforderungen der gehosteten Benutzeroberfläche umfassen nur den Header `x-amzn-cognito-client-id`. Weitere Informationen finden Sie unter Protokollieren des Web-ACL-Datenverkehrs im Entwicklerhandbuch zu AWS WAF .

AWS WAF Web-ACLs unterliegen nicht den Preisen für die erweiterten Sicherheitsfunktionen von Amazon Cognito. Die Sicherheitsfunktionen von AWS WAF ergänzen die erweiterten Sicherheitsfunktionen von Amazon Cognito. Sie können beide Funktionen in einem Benutzerpool aktivieren. AWS WAF stellt die Prüfung von Benutzerpool-Anfragen separat in Rechnung. Weitere Informationen finden Sie unter AWS WAF -Preisgestaltung.

Die Protokollierung von AWS WAF Anforderungsdaten unterliegt einer zusätzlichen Abrechnung durch den Dienst, für den Sie Ihre Protokolle verwenden. Weitere Informationen finden Sie unter

Preise für die Protokollierung von Web-ACL-Datenverkehrsinformationen im Entwicklerhandbuch zu AWS WAF .

Berücksichtigung der Groß-/Kleinschreibung im Benutzerpool

Amazon Cognito Cognito-Benutzerpools, die Sie in der erstellen, AWS Management Console unterscheiden standardmäßig nicht zwischen Groß- und Kleinschreibung. Wenn ein Benutzerpool Groß- und Kleinschreibung nicht beachtet, beziehen Sie sich benutzer@beispiel.com und Benutzer@beispiel.com auf denselben Benutzer. Wenn für Benutzernamen in einem Benutzerpool die Groß- und Kleinschreibung nicht beachtet wird, gilt dasselbe für die `preferred_username`- und `email`-Attribute.

Um die Einstellungen für die Groß- und Kleinschreibung des Benutzerpools zu berücksichtigen, identifizieren Sie Benutzer in Ihrem App-Code basierend auf einem alternativen Benutzerattribut. Da die Groß-/Kleinschreibung eines Benutzernamens, eines bevorzugten Benutzernamens oder eines E-Mail-Attributs variieren kann, verweisen Sie stattdessen auf das `sub`-Attribut. Sie können auch ein unveränderliches benutzerdefiniertes Attribut in Ihrem Benutzerpool erstellen und dem Attribut in jedem neuen Benutzerprofil Ihren eigenen eindeutigen Identifikationswert zuweisen. Beim Erstellen eines Benutzers können Sie einen Wert in das unveränderliche, benutzerdefinierte Attribut schreiben, das Sie erstellt haben.

> **Note**
>
> Unabhängig von den Einstellungen für die Groß- und Kleinschreibung Ihres Benutzerpools erfordert Amazon Cognito, dass ein Verbundbenutzer eines SAML- oder OIDC-Identitätsanbieters (IDP) eine eindeutige und Groß- und Kleinschreibung berücksichtigende `NameId` oder einen `sub`-Anspruch übergibt. Weitere Informationen zur Berücksichtigung von Groß- und Kleinschreibung bei Unique Identifier und zu SAML IdPs finden Sie unter. Verwenden der SP-initiierten SAML-Anmeldung

Erstellen eines Benutzerpools mit Berücksichtigung der Groß-/Kleinschreibung

Wenn Sie Ressourcen mit den Operationen AWS Command Line Interface (AWS CLI) und API-Operationen wie erstellen CreateUserPool, müssen Sie den booleschen `CaseSensitive` Parameter auf setzen. `false` Diese Einstellung erstellt einen Benutzerpool, in dem die Groß- und Kleinschreibung nicht berücksichtigt wird. Wenn Sie keinen Wert angeben, verwendet `CaseSensitive` standardmäßig `true`. Dies ist das Gegenteil des Standardverhaltens für

Benutzerpools, die Sie im AWS Management Console erstellen. Vor dem 12. Februar 2020 wurde bei Benutzerpools unabhängig von der Plattform standardmäßig Groß- und Kleinschreibung beachtet.

Sie können die Einstellungen für die Berücksichtigung von Groß- und Kleinschreibung für jeden Benutzerpool in Ihrem Konto auf der Registerkarte „Anmeldevorgang“ AWS Management Console oder im DescribeUserPoolAPI-Vorgang überprüfen.

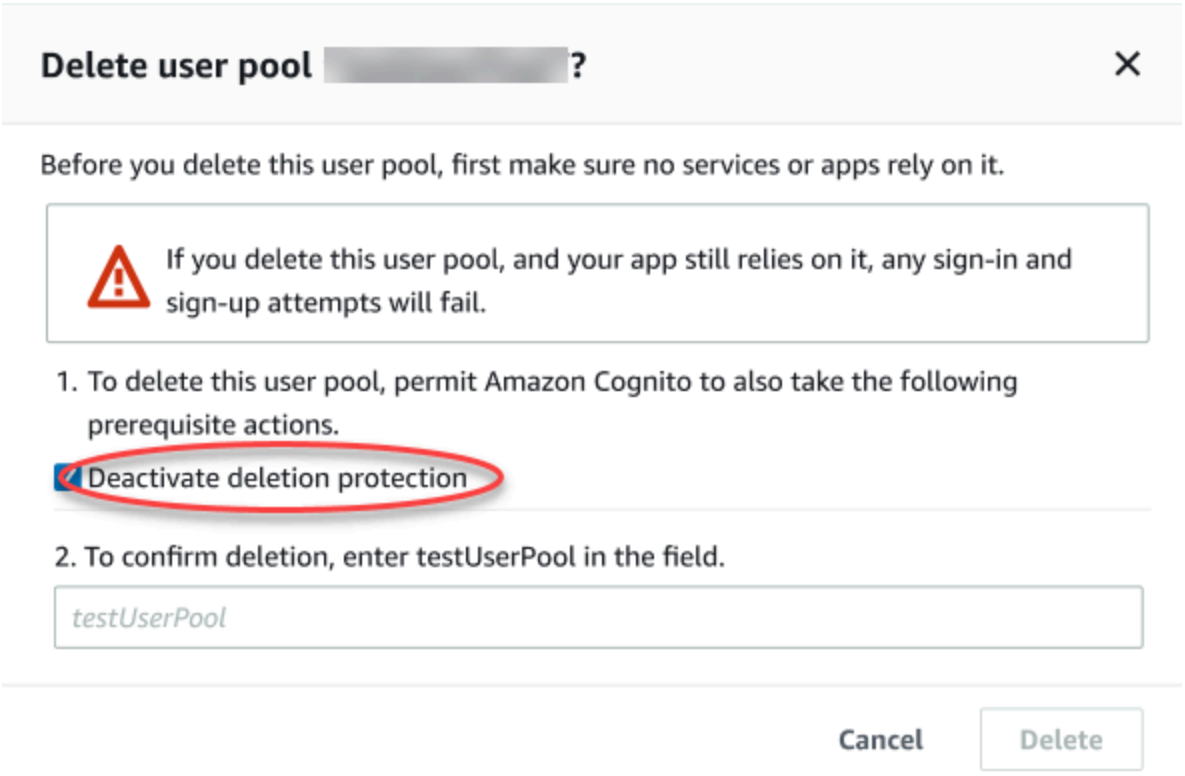
Migration zu einem neuen Benutzerpool

Aufgrund potenzieller Konflikte zwischen Benutzerprofilen können Sie einen Amazon-Cognito-Benutzerpool nicht von „Groß- und Kleinschreibung wird berücksichtigt“ auf „Groß- und Kleinschreibung wird nicht berücksichtigt“ ändern. Migrieren Sie stattdessen Ihre Benutzer in einen neuen Benutzerpool. Sie müssen Migrationscode erstellen, um fallbezogene Konflikte zu lösen. Dieser Code muss entweder einen eindeutigen neuen Benutzer zurückgeben oder den Anmeldeversuch ablehnen, wenn er einen Konflikt erkennt. Weisen Sie in einem neuen Benutzerpool, in dem die Groß-/Kleinschreibung nicht beachtet wird, ein Lambda-Auslöser für die Benutzermigration. zu. Mit dieser AWS Lambda Funktion können Benutzer im neuen Benutzerpool ohne Berücksichtigung der Groß- und Kleinschreibung erstellt werden. Die Lambda-Funktion findet und dupliziert den Benutzer aus dem Benutzerpool mit Beachtung der Groß- und Kleinschreibung, wenn es dem Benutzer nicht gelingt, sich mit dem Benutzerpool ohne Berücksichtigung der Groß- und Kleinschreibung anzumelden. Sie können auch einen Lambda-Trigger für die Migration von Benutzern bei ForgotPasswordEreignissen aktivieren. Amazon Cognito übergibt Benutzerinformationen und Ereignismetadaten aus der Anmeldung- oder Passwort-Wiederherstellungsaktion an Ihre Lambda-Funktion. Sie können Ereignisdaten verwenden, um Konflikte zwischen Benutzernamen und E-Mail-Adressen zu verwalten, wenn Ihre Funktion den neuen Benutzer in Ihrem Benutzerpool ohne Berücksichtigung der Groß- und Kleinschreibung erstellt. Diese Konflikte bestehen zwischen Benutzernamen und E-Mail-Adressen, die in einem Benutzerpool ohne Berücksichtigung der Groß- und Kleinschreibung eindeutig wären, aber in einem Benutzerpool mit Beachtung der Groß- und Kleinschreibung identisch sind.

Weitere Informationen zur Verwendung eines Lambda-Triggers für die Migration von Benutzern zwischen Amazon Cognito Cognito-Benutzerpools finden Sie unter Migrieren von Benutzern zu Amazon Cognito Cognito-Benutzerpools im Blog. AWS

Löschschutz für Benutzerpools

Aktivieren Sie den Löschschutz, damit Ihre Administratoren den Benutzerpool nicht versehentlich löschen. Wenn der Löschschutz aktiviert ist, müssen Sie vor dem Löschen des Benutzerpools bestätigen, dass er wirklich gelöscht werden soll. Wenn Sie einen Benutzerpool in der AWS Management Console löschen, können Sie gleichzeitig den Löschschutz deaktivieren. Wenn Sie die Aufforderung zur Deaktivierung des Löschschutzes akzeptieren und bestätigen, dass der Benutzerpool gelöscht werden soll, wie in der folgenden Abbildung dargestellt, löscht Amazon Cognito den Benutzerpool.



Wenn Sie einen Benutzerpool mit einer Amazon-Cognito-API-Anfrage löschen möchten, müssen Sie zunächst in einer UpdateUserPool-Anforderung `DeletionProtection` auf `Inactive` setzen. Wenn Sie den Löschschutz nicht deaktivieren, gibt Amazon Cognito die Fehlermeldung `InvalidParameterException` zurück. Nachdem Sie den Löschschutz deaktiviert haben, können Sie den Benutzerpool in einer DeleteUserPool-Anforderung löschen.

Amazon Cognito aktiviert Deletion protection (Löschschutz) standardmäßig, wenn Sie einen neuen Benutzerpool in der AWS Management Console erstellen. Wenn Sie einen Benutzerpool mit der API `CreateUserPool` erstellen, ist der Löschschutz standardmäßig inaktiv. Wenn Sie diese Funktion in Benutzerpools verwenden möchten, die Sie mit der AWS CLI oder einem AWS-SDK erstellen, legen Sie für den Parameter `DeletionProtection` `True` fest.

Sie können den Status des Löschschutzes im Container Deletion protection (Löschschutz) auf der Registerkarte User pool settings (Benutzerpool-Einstellungen) der Amazon-Cognito-Konsole aktivieren oder deaktivieren.

So konfigurieren Sie Löschschutz

1. Melden Sie sich bei der Amazon-Cognito-Konsole an. Sie werden möglicherweise aufgefordert, Ihre AWS-Anmeldeinformationen einzugeben.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte User pool settings (Benutzerpool-Einstellungen) aus. Suchen Sie Deletion Protection (Löschschutz) und wählen Sie Activate (Aktivieren) oder Deactivate (Deaktivieren) aus.
5. Bestätigen Sie Ihre Auswahl im nächsten Dialogfenster.

Verwalten von Reaktionen auf Fehler bei vorhandenen Benutzern

Amazon Cognito unterstützt die Anpassung von Fehlerantworten, die von Benutzerpools zurückgegeben werden. Benutzerdefinierte Fehlerantworten sind für die Erstellung und Authentifizierung von Benutzern, die Passwortwiederherstellung und Bestätigung verfügbar.

Verwenden Sie die `PreventUserExistenceErrors`-Einstellung eines Benutzerpool-App-Clients, um Fehler im Zusammenhang mit der Benutzerexistenz zu aktivieren oder zu deaktivieren. Wenn Sie einen neuen App-Client mit der Amazon Cognito Cognito-Benutzerpools-API erstellen`LEGACY`, `PreventUserExistenceErrors` ist diese standardmäßig oder deaktiviert. In der Amazon Cognito Cognito-Konsole `PreventUserExistenceErrors` ist die Option Benutzerexistenzfehler verhindern — eine Einstellung von `ENABLED` für — standardmäßig ausgewählt. Gehen Sie wie folgt vor, um Ihre `PreventUserExistenceErrors` Konfiguration zu aktualisieren:

- Ändern Sie den Wert `PreventUserExistenceErrors` zwischen `ENABLED` und `LEGACY` in einer UpdateUserPoolClientAPI-Anfrage.
- Bearbeiten Sie Ihren App-Client in der Amazon Cognito Cognito-Konsole und ändern Sie den Status von Prevent user existence errors zwischen selected (`ENABLED`) und deselected (`LEGACY`).

Wenn diese Eigenschaft den Wert von hat`LEGACY`, gibt Ihr App-Client eine `UserNotFoundException` Fehlerantwort zurück, wenn ein Benutzer versucht, sich mit einem Benutzernamen anzumelden, der in Ihrem Benutzerpool nicht existiert.

Wenn diese Eigenschaft den Wert von hat`ENABLED`, gibt Ihr App-Client die Nichtexistenz eines Benutzerkontos in Ihrem Benutzerpool nicht mit einem `UserNotFoundException` Fehler an. Eine `PreventUserExistenceErrors` Konfiguration von `ENABLED` hat die folgenden Auswirkungen:

- Amazon Cognito antwortet mit unspezifischen Informationen auf API-Anfragen, bei denen die Antwort andernfalls offenlegen könnte, dass ein gültiger Benutzer existiert.
- Die Amazon Cognito Cognito-APIs „Anmeldung“ und „Passwort vergessen“ geben eine generische Antwort auf einen Authentifizierungsfehler zurück. Die Fehlerantwort gibt an, dass der Benutzername oder das Passwort falsch ist.
- Die Amazon Cognito Cognito-APIs zur Kontobestätigung und Kennwortwiederherstellung geben eine Antwort zurück, die darauf hinweist, dass ein Code an ein simuliertes Übermittlungsmedium gesendet wurde, und nicht eine teilweise Darstellung der Kontaktinformationen eines Benutzers.

Die folgenden Informationen beschreiben das Verhalten von Benutzerpool-Vorgängen, wenn diese Option auf eingestellt `PreventUserExistenceErrors` ist. `ENABLED`

Vorgänge zur Authentifizierung und Benutzererstellung

Sie können Fehlerantworten sowohl bei der Benutzername-Password- als auch bei der Secure Remote Password (SRP) -Authentifizierung konfigurieren. Sie können auch die Fehler, die bei der benutzerdefinierten Authentifizierung zurückgegeben werden, anpassen. Die folgenden APIs führen diese Authentifizierungsvorgänge durch:

- `AdminInitiateAuth`
- `AdminRespondToAuthChallenge`
- `InitiateAuth`
- `RespondToAuthChallenge`

In der folgenden Liste sehen Sie, wie Sie Fehlerantworten in Operationen zum Authentifizieren von Benutzern anpassen können.

Authentifizierung mit Benutzername und Passwort

Um einen Benutzer mit `ADMIN_USER_PASSWORD_AUTH` und `USER_PASSWORD_AUTH` anzumelden, geben Sie den Benutzernamen und das Passwort in einer `AdminInitiateAuth`- oder `InitiateAuth`-API-Anforderung ein. Amazon Cognito gibt einen generischen `NotAuthorizedException`-Fehler zurück, wenn der Benutzername oder das Passwort falsch ist.

Secure-Remote-Password-(SRP)-basierte Authentifizierung

Um einen Benutzer mit `USER_SRP_AUTH` anzumelden, geben Sie den Benutzernamen und einen `SRP_A`-Parameter in einer `AdminInitiateAuth`- oder `InitiateAuth`-API-Anforderung ein. Als Reaktion darauf kehrt Amazon Cognito zurück `SRP_B` und salzt für den Benutzer. Wenn ein Benutzer nicht gefunden wird, gibt Amazon Cognito im ersten Schritt eine simulierte Antwort zurück, wie in RFC 5054 beschrieben. Amazon Cognito gibt denselben „Salt“ und eine interne Benutzer-ID im Format Universally Unique Identifier (UUID) für dieselbe Kombination aus Benutzername und Benutzerpool zurück. Wenn Sie eine `RespondToAuthChallenge`API-Anforderung mit einem Passwortnachweis senden, gibt Amazon Cognito einen generischen `NotAuthorizedException`-Fehler zurück, wenn der Benutzername oder das Passwort falsch ist.

> **Note**
>
> Sie können eine generische Antwort mit Authentifizierung von Benutzername und Passwort simulieren, wenn Sie verifizierungsbasierte Aliasattribute verwenden und der unveränderliche Benutzername nicht als UUID formatiert ist.

Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen

Wenn Sie den Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen verwenden und Fehlerantworten aktivieren, gibt `LambdaChallenge` einen booleschen Parameter namens `UserNotFound` zurück. Dann wird er in der Anfrage von `DefineAuthChallenge`-, `VerifyAuthChallenge`- und `CreateAuthChallenge`-Lambda-Auslösern übergeben. Sie können diesen Auslöser verwenden, um benutzerdefinierte Autorisierungsherausforderungen für einen nicht vorhandenen Benutzer zu simulieren. Wenn Sie den Lambda–Auslöser der Vorauthentifizierung für einen nicht vorhandenen Benutzer aufrufen, gibt Amazon Cognito `UserNotFound` zurück.

In der folgenden Liste sehen Sie, wie Sie Fehlerantworten in Operationen zum Erstellen von Benutzern anpassen können.

SignUp

Der `SignUp` Vorgang kehrt immer zurück`UsernameExistsException`, wenn ein Benutzername bereits vergeben ist. Wenn Sie nicht möchten, dass Amazon Cognito bei der Registrierung von Benutzern in Ihrer App einen Fehler `UsernameExistsException` für E-Mail-Adressen und Telefonnummern zurückgibt, verwenden Sie verifizierungsbasierte Aliasattribute. Weitere Informationen zu Aliassen finden Sie unter Anpassen von Anmeldeattributen.

Ein Beispiel dafür, wie Amazon Cognito die Verwendung von API-Anforderungen `SignUp` zur Erkennung von Benutzern in Ihrem Benutzerpool verhindern kann, finden Sie unter Vermeiden von Fehlern `UsernameExistsException` in Bezug auf E-Mail-Adressen und Telefonnummern bei der Registrierung.

Importierte Benutzer

Wenn `PreventUserExistenceErrors` aktiviert ist, wird während der Authentifizierung importierter Benutzer ein allgemeiner `NotAuthorizedException`-Fehler zurückgegeben, der angibt, dass entweder der Benutzername oder das Kennwort falsch war, anstatt `PasswordResetRequiredException` zurückzugeben. Weitere Informationen finden Sie unter Erfordern, dass importierte Benutzer ihre Kennwörter zurücksetzen.

Lambda-Auslöser für die Benutzermigration.

Amazon Cognito gibt eine simulierte Antwort für nicht vorhandene Benutzer zurück, wenn im ursprünglichen Ereigniskontext vom Lambda-Auslöser eine leere Antwort festgelegt wurde. Weitere Informationen finden Sie unter Lambda-Auslöser für die Benutzermigration.

Vermeiden von Fehlern **`UsernameExistsException`** in Bezug auf E-Mail-Adressen und Telefonnummern bei der Registrierung

Das folgende Beispiel zeigt, wie Sie bei der Konfiguration von Aliasattributen in Ihrem Benutzerpool verhindern können, dass doppelte E-Mail-Adressen und Telefonnummern als Antwort auf die API-Anforderung `SignUp` den Fehler `UsernameExistsException` generieren. Sie müssen Ihren Benutzerpool mit einer E-Mail-Adresse oder Telefonnummer als Aliasattribut erstellt haben. Weitere Informationen finden Sie im Abschnitt Anpassen von Anmeldeattributen unter Attribute für den Benutzerpool.

1. Jie meldet sich mit einem neuen Benutzernamen an und gibt auch die E-Mail-Adresse `jie@example.com` an. Amazon Cognito sendet einen Code an seine E-Mail-Adresse.

   Beispiel für einen AWS CLI Befehl

   ```
   aws cognito-idp sign-up --client-id 1234567890abcdef0 --username jie --password
    PASSWORD --user-attributes Name="email",Value="jie@example.com"
   ```

   Beispielantwort

   ```
   {
       "UserConfirmed": false,
       "UserSub": "<subId>",
       "CodeDeliveryDetails": {
           "AttributeName": "email",
           "Destination": "j****@e****",
           "DeliveryMedium": "EMAIL"
       }
   }
   ```

2. Jie gibt den Code an, der ihm zur Bestätigung der E-Mail-Adresse zugesendet wurde. Damit ist seine Registrierung als Benutzer abgeschlossen.

   Beispiel für einen AWS CLI Befehl

   ```
   aws cognito-idp confirm-sign-up --client-id 1234567890abcdef0 --username=jie --
   confirmation-code xxxxxx
   ```

3. Shirley registriert ein neues Benutzerkonto und gibt die E-Mail-Adresse `jie@example.com` an. Amazon Cognito gibt keinen Fehler `UsernameExistsException` zurück und sendet einen Bestätigungscode an Jies E-Mail-Adresse.

   Beispiel für einen AWS CLI Befehl

   ```
   aws cognito-idp sign-up --client-id 1234567890abcdef0 --username shirley --password
    PASSWORD --user-attributes Name="email",Value="jie@example.com"
   ```

   Beispielantwort

   ```
   {
       "UserConfirmed": false,
   ```

```
    "UserSub": "<new subId>",
    "CodeDeliveryDetails": {
        "AttributeName": "email",
        "Destination": "j****@e****",
        "DeliveryMedium": "EMAIL"
    }
}
```

4. In einem anderen Szenario ist Shirley Eigentümerin von `jie@example.com`. Shirley ruft den Code ab, den Amazon Cognito an Jies E-Mail-Adresse gesendet hat, und versucht, das Konto zu bestätigen.

   Beispiel für einen AWS CLI Befehl

   ```
   aws cognito-idp confirm-sign-up --client-id 1234567890abcdef0 --username=shirley --
   confirmation-code xxxxxx
   ```

   Beispielantwort

   ```
   An error occurred (AliasExistsException) when calling the ConfirmSignUp operation: An
    account with the email already exists.
   ```

Amazon Cognito gibt keinen Fehler auf Shirleys Anforderung `aws cognito-idp sign-up` zurück, obwohl `jie@example.com` einem vorhandenen Benutzer zugewiesen ist. Shirley muss nachweisen, dass die E-Mail-Adresse ihr gehört, bevor Amazon Cognito eine Fehlerantwort zurückgibt. In einem Benutzerpool mit Aliasattributen verhindert dieses Verhalten die Verwendung der öffentlichen `SignUp`-API, um zu überprüfen, ob ein Benutzer mit einer bestimmten E-Mail-Adresse oder Telefonnummer existiert.

Dieses Verhalten unterscheidet sich von der Antwort, die Amazon Cognito auf eine `SignUp`-Anforderung mit einem vorhandenen Benutzernamen zurückgibt, wie im folgenden Beispiel gezeigt wird. Shirley erfährt zwar aus dieser Antwort, dass ein Benutzer mit dem Benutzernamen `jie` bereits vorhanden ist, sie erfährt jedoch nichts über die mit dem Benutzer verknüpften E-Mail-Adressen oder Telefonnummern.

CLI-Beispielbefehl

```
aws cognito-idp sign-up --client-id 1example23456789 --username jie --password PASSWORD
      --user-attributes Name="email",Value="shirley@example.com"
```

Beispielantwort

```
An error occurred (UsernameExistsException) when calling the SignUp operation: User
 already exists
```

Operationen zum Zurücksetzen des Passworts

Amazon Cognito gibt die folgenden Antworten auf Vorgänge zum Zurücksetzen von Benutzerpasswörtern zurück, wenn Sie Fehler bei vorhandenen Benutzern verhindern.

ForgotPassword

Wenn ein Benutzer nicht gefunden wird, deaktiviert ist oder keinen verifizierten Übermittlungsmechanismus zum Wiederherstellen seines Kennworts hat, gibt Amazon Cognito `CodeDeliveryDetails` mit einem simulierten Bereitstellungsmedium für einen Benutzer zurück. Das simulierte Bereitstellungsmedium wird durch das Eingabe-Benutzernamensformat und die Verifizierungseinstellungen des Benutzerpools bestimmt.

ConfirmForgotPassword

Amazon Cognito gibt den `CodeMismatchException`-Fehler für Benutzer zurück, die nicht vorhanden oder deaktiviert sind. Wenn bei der Verwendung von `ForgotPassword` kein Code angefordert wird, gibt Amazon Cognito den `ExpiredCodeException`-Fehler zurück.

Bestätigungsoperationen

Amazon Cognito gibt die folgenden Antworten auf Vorgänge zum Bestätigen und Verifizieren von Benutzern zurück, wenn Sie Fehler bei vorhandenen Benutzern verhindern.

ResendConfirmationCode

Amazon Cognito gibt `CodeDeliveryDetails` für einen deaktivierten oder einen nicht vorhandenen Benutzer zurück. Amazon Cognito sendet einen Bestätigungscode an die E-Mail- oder Telefonnummer des bestehenden Benutzers.

ConfirmSignUp

`ExpiredCodeException` gibt zurück, ob ein Code abgelaufen ist. Amazon Cognito gibt `NotAuthorizedException` zurück, wenn ein Benutzer nicht autorisiert ist. Wenn der Code nicht den Erwartungen des Servers entspricht, gibt Amazon Cognito `CodeMismatchException` zurück.

Amazon-Cognito-Identitätspools

Ein Amazon-Cognito-Identitätspool ist ein Verzeichnis von Verbundidentitäten, die Sie gegen AWS -Anmeldeinformationen austauschen können. Identitätspools generieren temporäre AWS Anmeldeinformationen für die Benutzer Ihrer App, unabhängig davon, ob sie sich angemeldet haben oder Sie sie noch nicht identifiziert haben. Mit AWS Identity and Access Management (IAM-) Rollen und Richtlinien können Sie die Berechtigungsstufe wählen, die Sie Ihren Benutzern gewähren möchten. Benutzer können als Gäste beginnen und Ressourcen abrufen, die Sie in AWS-Services speichern. Anschließend können sie sich bei einem externen Identitätsanbieter anmelden, um den Zugriff auf Ressourcen freizuschalten, die Sie registrierten Mitgliedern zur Verfügung stellen. Bei dem externen Identitätsanbieter kann es sich um einen (sozialen) OAuth-2.0-Anbieter für Verbraucher wie Apple oder Google, um einen benutzerdefinierten SAML- oder OIDC-Identitätsanbieter oder um ein benutzerdefiniertes Authentifizierungsschema, auch Entwickleranbieter genannt, nach Ihrem eigenen Design handeln.

Eigenschaften von Amazon-Cognito-Identitätspools

Unterschreiben Sie Anfragen für AWS-Services

> Signieren Sie API-Anfragen AWS-Services an Amazon Simple Storage Service (Amazon S3) und Amazon DynamoDB. Analysieren Sie Benutzeraktivitäten mit Diensten wie Amazon Pinpoint und Amazon CloudWatch.

Filtern von Anforderungen mit ressourcenbasierten Richtlinien

> Nutzen Sie granulare Kontrolle über den Benutzerzugriff auf Ihre Ressourcen aus. Verwandeln Sie Benutzeranforderungen in IAM-Sitzungs-Tags und erstellen Sie IAM-Richtlinien, die bestimmten Untergruppen Ihrer Benutzer Zugriff auf Ressourcen gewähren.

Gastzugang zuweisen

> Für Ihre Benutzer, die sich noch nicht angemeldet haben, konfigurieren Sie Ihren Identitätspool so, dass AWS -Anmeldeinformationen mit einem engen Zugriffsumfang generiert werden. Authentifizieren Sie Benutzer über einen Single-Sign-On-Anbieter, um ihnen den Zugriff zu erleichtern.

Zuweisen von IAM-Rollen auf der Grundlage von Benutzereigenschaften

> Weisen Sie allen Ihren authentifizierten Benutzern eine einzige IAM-Rolle zu oder wählen Sie die Rolle basierend auf den Anforderungen der einzelnen Benutzer aus.

Akzeptieren mehrerer Identitätsanbieter

Tauschen Sie eine ID oder ein Zugriffstoken, ein Benutzerpool-Token, eine SAML-Assertion oder ein OAuth-Token eines sozialen Anbieters gegen Anmeldeinformationen ein. AWS

Validieren Ihrer eigenen Identitäten

Führen Sie Ihre eigene Benutzervalidierung durch und verwenden Sie Ihre AWS Entwickleranmeldedaten, um Anmeldeinformationen für Ihre Benutzer auszustellen.

Möglicherweise haben Sie bereits einen Amazon-Cognito-Benutzerpool, der Authentifizierungs- und Autorisierungsdienste für Ihre App bereitstellt. Sie können Ihren Benutzerpool als Identitätsanbieter (IdP) für Ihren Identitätspool einrichten. In diesem Fall können sich Ihre Benutzer über Ihren Benutzerpool authentifizieren IdPs, ihre Ansprüche in einem gemeinsamen OIDC-Identitätstoken zusammenfassen und dieses Token gegen Anmeldeinformationen eintauschen. AWS Ihr Benutzer kann dann seine Anmeldeinformationen in einer signierten Anfrage an Ihre AWS-Services weitergeben.

Sie können auch authentifizierte Anforderungen von einem Ihrer Identitätsanbieter direkt Ihrem Identitätspool präsentieren. Amazon Cognito passt Benutzeransprüche von SAML-, OAuth- und OIDC-Anbietern in eine API-Anfrage für kurzfristige Anmeldeinformationen an. AssumeRoleWithWebIdentity

Amazon-Cognito-Benutzerpools fungieren wie OIDC-Identitätsanbieter für Ihre SSO-fähigen Apps. Identitätspools fungieren als AWS-Identitätsanbieter für jede App mit Ressourcenabhängigkeiten, die am besten mit IAM-Autorisierung funktionieren.

Amazon-Cognito-Identitätspools unterstützen die folgenden Identitätsanbieter:

- Öffentliche Anbieter: Login with Amazon als Identitätspools (IdP) einrichten, Facebook als Identitätspools einrichten (IdP), Google als Identitätspool-IdP einrichten, Mit Apple anmelden als Identitätspool-IdP einrichten, Twitter.
- Amazon-Cognito-Benutzerpools
- Einrichtung eines OIDC-Anbieters als Identitätspool-IdP
- Einrichtung eines SAML-Anbieters als Identitätspool-IdP
- Entwicklerauthentifizierte Identitäten (Identitätspools)

Weitere Informationen zur regionalen Verfügbarkeit von Amazon-Cognito-Identitäten-Pools finden Sie unter Regionale Verfügbarkeit von AWS -Services.

Weitere Informationen zu Amazon-Cognito-Identitätspools finden Sie in den folgenden Themen.

Themen

- Verwenden von Identitätspools (Verbundidentitäten)
- Identitäten-Pool-Konzepte
- Bewährte Sicherheitsmethoden für Amazon Cognito Cognito-Identitätspools
- Verwenden von Attributen für Zugriffskontrolle
- Verwenden der rollenbasierten Zugriffskontrolle
- Abrufen von Anmeldeinformationen
- Zugreifen auf Dienste AWS
- Externe Identitätsanbieter von Identitäten-Pools
- Entwicklerauthentifizierte Identitäten (Identitätspools)
- Wechseln von nicht authentifizierten Benutzern zu authentifizierten Benutzern (Identitäten-Pools)

Verwenden von Identitätspools (Verbundidentitäten)

Amazon Cognito Cognito-Identitätspools bieten temporäre AWS Anmeldeinformationen für Benutzer, die Gäste sind (nicht authentifiziert), und für Benutzer, die authentifiziert wurden und ein Token erhalten haben. Ein Identitäten-Pool ist eine Speicher von Benutzer-Identitätsdaten, die speziell für Ihr Konto gelten.

So erstellen Sie einen neuen Identitäten-Pool in der Konsole

1. Melden Sie sich bei der Amazon-Cognito-Konsole an und wählen Sie Identitätspools aus.
2. Wählen Sie Identitätspool erstellen.
3. Wählen Sie unter Identitätspool-Vertrauen konfigurieren aus, ob Sie Ihren Identitätspool für authentifizierten Zugriff, Gastzugriff oder beides einrichten möchten.

    - Wenn Sie Authentifizierter Zugriff ausgewählt haben, wählen Sie einen oder mehrere Identitätstypen aus, die Sie als Quelle für authentifizierte Identitäten in Ihrem Identitätspool festlegen möchten. Wenn Sie einen benutzerdefinierten Entwickleranbieter konfigurieren,

können Sie diesen nicht ändern oder löschen, nachdem Sie Ihren Identitätspool erstellt haben.

4. Wählen Sie unter Berechtigungen konfigurieren eine Standard-IAM-Rolle für authentifizierte Benutzer oder Gastbenutzer in Ihrem Identitätspool aus.

   a. Wählen Sie Neue IAM-Rolle erstellen, wenn Sie möchten, dass Amazon Cognito für Sie eine neue Rolle mit grundlegenden Berechtigungen und einer Vertrauensbeziehung zu Ihrem Identitätspool erstellt. Geben Sie einen IAM-Rollen-Namen ein, um Ihre neue Rolle zu identifizieren, zum Beispiel `myidentitypool_authenticatedrole`. Wählen Sie Richtliniendokument anzeigen aus, um die Berechtigungen zu überprüfen, die Amazon Cognito Ihrer neuen IAM-Rolle zuweist.

   b. Sie können sich dafür entscheiden, eine bestehende IAM-Rolle zu verwenden, wenn Sie bereits eine Rolle in Ihrer haben AWS-Konto , die Sie verwenden möchten. Sie müssen Ihre IAM-Rollen-Vertrauensrichtlinie so konfigurieren, dass sie `cognito-identity.amazonaws.com` beinhaltet. Konfigurieren Sie Ihre Rollen-Vertrauensrichtlinie so, dass Amazon Cognito die Rolle nur übernehmen kann, wenn nachgewiesen wird, dass die Anforderung von einem authentifizierten Benutzer in Ihrem spezifischen Identitätspool stammt. Weitere Informationen finden Sie unter Vertrauensstellungen und Berechtigungen für Rollen.

5. Geben Sie in Connect Identity Providers die Details der Identitätsanbieter (IdPs) ein, die Sie unter Identitätspool-Trust konfigurieren ausgewählt haben. Möglicherweise werden Sie aufgefordert, OAuth-App-Client-Informationen anzugeben, einen Amazon-Cognito-Benutzerpool auszuwählen, einen IAM-IdP auszuwählen oder eine benutzerdefinierte ID für einen Entwickleranbieter einzugeben.

   a. Wählen Sie die Rolleneinstellungen für jeden IdP aus. Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln auswählen. Bei einem Amazon-Cognito-Benutzerpool-IdP können Sie auch eine Rolle mit preferred_role in Token auswählen. Weitere Informationen zur `cognito:preferred_role`-Anforderung finden Sie unter Zuweisen von Prioritätswerten zu Gruppen.

      i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die

Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

b. Sie können Attribute für die Zugriffskontrolle für jeden IdP separat konfigurieren. Attribute für die Zugriffskontrolle ordnen Benutzeranforderungen den Prinzipal-Tags zu, die Amazon Cognito auf die temporäre Sitzung anwendet. Sie können IAM-Richtlinien erstellen, um den Benutzerzugriff anhand der Tags zu filtern, die Sie auf die jeweilige Sitzung anwenden.

i. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

ii. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

iii. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

6. Geben Sie unter Eigenschaften konfigurieren unter Identitätspool-Name einen Namen ein.
7. Wählen Sie unter Standardauthentifizierung (klassische Authentifizierung) aus, ob Sie den Standardablauf aktivieren möchten. Wenn der Basisablauf aktiv ist, können Sie die Rollenauswahl, die Sie für Sie getroffen haben, umgehen IdPs und AssumeRoleWithWebIdentitydirekt anrufen. Weitere Informationen finden Sie unter Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf.
8. Wählen Sie unter Tags die Option Tag hinzufügen aus, wenn Sie Tags auf Ihren Identitätspool anwenden möchten.
9. Bestätigen Sie unter Überprüfen und erstellen die Auswahl, die Sie für Ihren neuen Identitätspool getroffen haben. Wählen Sie Bearbeiten, um zum Assistenten zurückzukehren und Einstellungen zu ändern. Wählen Sie danach Identitätspool erstellen aus.

IAM-Rollen von Benutzern

Eine IAM-Rolle definiert die Berechtigungen für Ihre Benutzer für den Zugriff auf AWS Ressourcen, wie z. Amazon Cognito Sync Benutzer Ihrer Anwendung übernehmen die Rollen, die Sie erstellen. Sie können unterschiedliche Rollen für authentifizierte und nicht authentifizierte Benutzer angeben. Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen.

Authentifizierte und nicht authentifizierte Identitäten

Amazon-Cognito-Identitäten-Pools unterstützen authentifizierte und nicht authentifizierte Identitäten. Authentifizierte Identitäten gehören zu Benutzer, die von einem unterstützten Identitätsanbieter authentifiziert werden. Nicht authentifizierte Identitäten gehören in der Regel Gastbenutzern.

- Informationen zum Konfigurieren authentifizierter Identitäten mit einem öffentlichen Anmeldeanbieter finden Sie unter Externe Identitätsanbieter von Identitäten-Pools.
- Informationen zum Konfigurieren Ihres eigenen Backend-Authentifizierungsprozesses finden Sie unter Entwicklerauthentifizierte Identitäten (Identitätspools).

Gastzugang aktivieren oder deaktivieren

Amazon Cognito Identity Pools Gastzugriff (nicht authentifizierte Identitäten) bietet eine eindeutige Kennung und AWS Anmeldeinformationen für Benutzer, die sich nicht bei einem Identitätsanbieter authentifizieren. Wenn Ihre Anwendung Benutzer zulässt, die sich nicht anmelden, können Sie den Zugriff für nicht authentifizierte Identitäten aktivieren. Weitere Informationen hierzu finden Sie unter Erste Schritte mit Amazon Cognito Cognito-Identitätspools.

So aktualisieren Sie den Gastzugriff in einem Identitätspool

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Suchen Sie Gastzugang. In einem Identitätspool, der derzeit keinen Gastzugriff unterstützt, ist der Status Inaktiv.

    a. Wenn der Gastzugriff aktiv ist und Sie ihn deaktivieren möchten, wählen Sie Deaktivieren aus.

    b. Wenn der Gastzugang inaktiv ist und Sie ihn aktivieren möchten, wählen Sie Bearbeiten.

        - Wählen Sie eine Standard-IAM-Rolle für Gastbenutzer in Ihrem Identitätspool.

            A. Wählen Sie Neue IAM-Rolle erstellen, wenn Sie möchten, dass Amazon Cognito für Sie eine neue Rolle mit grundlegenden Berechtigungen und einer Vertrauensbeziehung zu Ihrem Identitätspool erstellt. Geben Sie einen IAM-Rollen-Namen ein, um Ihre neue Rolle zu identifizieren, zum Beispiel `myidentitypool_authenticatedrole`. Wählen Sie

Richtliniendokument anzeigen aus, um die Berechtigungen zu überprüfen, die Amazon Cognito Ihrer neuen IAM-Rolle zuweist.

B. Sie können sich dafür entscheiden, eine bestehende IAM-Rolle zu verwenden, wenn Sie bereits eine Rolle in Ihrer AWS-Konto haben, die Sie verwenden möchten. Sie müssen Ihre IAM-Rollen-Vertrauensrichtlinie so konfigurieren, dass sie `cognito-identity.amazonaws.com` beinhaltet. Konfigurieren Sie Ihre Rollen-Vertrauensrichtlinie so, dass Amazon Cognito die Rolle nur übernehmen kann, wenn nachgewiesen wird, dass die Anforderung von einem authentifizierten Benutzer in Ihrem spezifischen Identitätspool stammt. Weitere Informationen finden Sie unter Vertrauensstellungen und Berechtigungen für Rollen.

C. Wählen Sie Änderungen speichern aus.

D. Um den Gastzugriff zu aktivieren, wählen Sie auf der Registerkarte Benutzerzugriff die Option Aktivieren aus.

Ändern der mit einem Identitätstyp verknüpften Rolle

Jede Identität im Identitäten-Pool ist entweder authentifiziert oder nicht authentifiziert. Authentifizierte Identitäten gehören Benutzern, die von einem öffentlichen Anmeldeanbieter (Amazon-Cognito-Benutzerpools, Login with Amazon, Mit Apple anmelden, Facebook, Google, SAML oder ein beliebiger OpenID-Connect-Anbieter) bzw. einem Entwickleranbieter (Ihr eigener Backend-Authentifizierungsprozess) authentifiziert werden. Nicht authentifizierte Identitäten gehören in der Regel Gastbenutzern.

Für jeden Identitätstyp existiert eine zugewiesene Rolle. Mit dieser Rolle ist eine Richtlinie verknüpft, die festlegt, auf AWS-Services welche Rolle sie zugreifen kann. Wenn Amazon Cognito eine Anfrage empfängt, ermittelt der Service den Identitätstyp sowie die ihm zugewiesene Rolle und verwendet für seine Antwort die Richtlinie, die dieser Rolle zugewiesen ist. Indem Sie eine Richtlinie ändern oder einem Identitätstyp eine andere Rolle zuweisen, können Sie steuern, auf welchen AWS-Services Identitätstyp zugegriffen werden kann. Wenn Sie die Richtlinien, die den Rollen in Ihrem Identitäten-Pool zugewiesen sind, anzeigen oder ändern möchten, verwenden Sie die AWS -IAM-Konsole.

So ändern Sie die standardmäßige authentifizierte oder nicht authentifizierte Rolle des Identitätspools

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.

3. Suchen Sie nach Gastzugang oder Authentifizierter Zugriff. In einem Identitätspool, der derzeit nicht für diesen Zugriffstyp konfiguriert ist, ist der Status Inaktiv. Wählen Sie Bearbeiten aus.
4. Wählen Sie eine Standard-IAM-Rolle für Gast- oder authentifizierte Benutzer in Ihrem Identitätspool.

    a. Wählen Sie Neue IAM-Rolle erstellen, wenn Sie möchten, dass Amazon Cognito für Sie eine neue Rolle mit grundlegenden Berechtigungen und einer Vertrauensbeziehung zu Ihrem Identitätspool erstellt. Geben Sie einen IAM-Rollen-Namen ein, um Ihre neue Rolle zu identifizieren, zum Beispiel `myidentitypool_authenticatedrole`. Wählen Sie Richtliniendokument anzeigen aus, um die Berechtigungen zu überprüfen, die Amazon Cognito Ihrer neuen IAM-Rolle zuweist.

    b. Sie können wählen, ob Sie eine bestehende IAM-Rolle verwenden möchten, wenn Sie bereits eine Rolle in Ihrer haben AWS-Konto , die Sie verwenden möchten. Sie müssen Ihre IAM-Rollen-Vertrauensrichtlinie so konfigurieren, dass sie `cognito-identity.amazonaws.com` beinhaltet. Konfigurieren Sie Ihre Rollen-Vertrauensrichtlinie so, dass Amazon Cognito die Rolle nur übernehmen kann, wenn nachgewiesen wird, dass die Anforderung von einem authentifizierten Benutzer in Ihrem spezifischen Identitätspool stammt. Weitere Informationen finden Sie unter Vertrauensstellungen und Berechtigungen für Rollen.
5. Wählen Sie Änderungen speichern aus.

Identitätsanbieter bearbeiten

Wenn Sie Ihren Benutzern erlauben, sich über öffentliche Identitätsanbieter zu authentifizieren (z. B. Amazon-Cognito-Benutzerpools, Anmelden mit Amazon, Anmelden mit Apple, Facebook oder Google), können Sie Ihre Anwendungskennungen in der Konsole der Amazon-Cognito-Identitätspools (verbundene Identitäten) angeben. Dadurch wird die (vom öffentlichen Anmeldeanbieter bereitgestellte) Anwendungs-ID mit dem Identitäten-Pool verknüpft.

Sie können über diese Seite auch Authentifizierungsregeln für die einzelnen Anbieter konfigurieren. Jeder Anbieter lässt bis zu 25 Regeln zu. Die Regeln werden in der Reihenfolge angewendet, in der sie für die einzelnen Anbieter gespeichert werden. Weitere Informationen finden Sie unter Verwenden der rollenbasierten Zugriffskontrolle.

> **Warning**
>
> Das Ändern der Anwendungs-ID, mit der ein Identitätspool verknüpft ist, verhindert, dass sich vorhandene Benutzer mit diesem Identitätspool authentifizieren. Weitere Informationen finden Sie unter Externe Identitätsanbieter von Identitäten-Pools.

So aktualisieren Sie einen Identitätspool-Identitätsanbieter (IdP)

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Suchen Sie Identitätsanbieter. Wählen Sie den Identitätsanbieter aus, den Sie bearbeiten möchten. Wenn Sie einen neuen IdP hinzufügen möchten, wählen Sie Identitätsanbieter hinzufügen aus.
   - Wenn Sie Identitätsanbieter hinzufügen ausgewählt haben, wählen Sie einen der Identitätstypen aus, die Sie hinzufügen möchten.
4. Um die Anwendungs-ID zu ändern, wählen Sie in den Identitätsanbieterinformationen die Option Bearbeiten aus.
5. Um die Rolle zu ändern, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen für Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, wählen Sie in den Rolleneinstellungen die Option Bearbeiten aus.
   - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen. Bei einem Amazon-Cognito-Benutzerpool-IdP können Sie auch eine Rolle mit preferred_role in Token auswählen. Weitere Informationen zur `cognito:preferred_role`-Anforderung finden Sie unter Zuweisen von Prioritätswerten zu Gruppen.
     i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

6. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, wählen Sie unter Attribute für die Zugriffskontrolle die Option Bearbeiten.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

7. Wählen Sie Änderungen speichern aus.

Löschen eines Identitätspools

Sie können das Löschen eines Identitätspools nicht rückgängig machen. Nachdem Sie einen Identitätspool gelöscht haben, funktionieren alle Apps und Benutzer, die davon abhängig sind, nicht mehr.

Löschen eines Identitätspools

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Markieren Sie das Kontrollkästchen neben dem Identitätspool, den Sie löschen möchten.
2. Wählen Sie Löschen aus.
3. Geben oder fügen Sie den Namen Ihres Identitätspools ein und wählen Sie Löschen aus.

> **Warning**
>
> Wenn Sie auf die Schaltfläche zum Löschen klicken, löschen Sie Ihren Identitätspool und alle enthaltenen Benutzerdaten dauerhaft. Das Löschen eines Identitätspools bewirkt, dass Anwendungen und andere Services, die den Identitätspool nutzen, nicht mehr ausgeführt werden.

Löschen einer Identität aus einem Identitätspool

Wenn Sie eine Identität aus einem Identitätspool löschen, entfernen Sie die identifizierenden Informationen, die Amazon Cognito für diesen Verbundbenutzer gespeichert hat. Wenn Ihr Benutzer erneut Anmeldeinformationen anfordert, erhält er eine neue Identitäts-ID, sofern Ihr Identitätspool seinem Identitätsanbieter weiterhin vertraut. Dieser Vorgang kann nicht rückgängig gemacht werden.

So löschen Sie eine Identität

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Identitätsbrowser.
3. Markieren Sie das Kontrollkästchen neben den Identitäten, die Sie löschen möchten, und wählen Sie Löschen. Bestätigen Sie, dass Sie die Identitäten löschen möchten, und klicken Sie dann auf Löschen.

Verwenden von Amazon Cognito Sync mit Identitätspools

Amazon Cognito Sync ist eine AWS-Service Client-Bibliothek, die es ermöglicht, anwendungsbezogene Benutzerdaten geräteübergreifend zu synchronisieren. Amazon Cognito Sync kann Benutzerprofildaten über mobile Geräte und das Web synchronisieren, ohne Ihr eigenes Backend verwenden zu müssen. Die Client-Bibliotheken speichern Daten lokal zwischen, sodass Ihre App Daten unabhängig vom Konnektivitätsstatus des Geräts lesen und schreiben kann. Wenn das Gerät online ist, können Sie Daten synchronisieren. Wenn Sie die Push-Synchronisierung einrichten, können Sie andere Geräte umgehend benachrichtigen, wenn ein Update verfügbar ist.

Verwalten von Datensätzen

Wenn Sie die Amazon-Cognito-Sync-Funktionalität in Ihrer Anwendung implementiert haben, ermöglicht die Amazon-Cognito-Identitätspool-Konsole das manuelle Erstellen und Löschen von Datensätzen und Akten für einzelne Identitäten. Alle Änderungen, die Sie am Datensatz oder an Datensätzen einer Identität in dem Amazon-Cognito-Identitätspool vornehmen, werden erst gespeichert, wenn Sie in der Konsole Synchronisieren auswählen. Die Änderung ist für den Endbenutzer erst sichtbar, wenn die Identität Synchronize aufruft. Die Daten, die von anderen Geräten für einzelne Identitäten synchronisiert werden, sind sichtbar, sobald Sie die Seite zum Auflisten von Datasets für eine bestimmte Identität aktualisieren.

Erstellen eines Datensatzes für eine Identität

Amazon Cognito Sync verknüpft einen Datensatz mit einer Identität. Sie können Ihren Datensatz mit identifizierenden Informationen über den Benutzer, für den die Identität steht, füllen und diese Informationen dann mit allen Geräten Ihres Benutzers synchronisieren.

So fügen Sie einer Identität einen Datensatz und Datensatzeinträge hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Identitätsbrowser.
3. Wählen Sie die Identität aus, die Sie bearbeiten möchten.
4. Wählen Sie unter Datensätze die Option Datensatz erstellen aus.
5. Geben Sie einen Datensatznamen ein, und wählen Sie Datensatz erstellen aus.
6. Wenn Sie Ihrem Datensatz Datensätze hinzufügen möchten, wählen Sie Ihren Datensatz aus den Identitätsdetails aus. Wählen Sie unter Datensätze die Option Datensatz erstellen aus.
7. Geben Sie einen Schlüssel und einen Wert für Ihren Datensatz ein. Wählen Sie Bestätigen aus. Wiederholen Sie den Vorgang, um weitere Datensätze hinzuzufügen.

Löschen eines mit einer Identität verknüpften Datensatzes

So löschen Sie einen Datensatz und seine Datensätze aus einer Identität

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Identitätsbrowser.
3. Klicken Sie auf den Namen des Identitätspools mit de, Datensatz, den Sie löschen möchten.
4. Markieren Sie unter Datensätze das Kontrollkästchen neben dem Datensatz, den Sie löschen möchten.
5. Wählen Sie Löschen aus. Überprüfen Sie Ihre Auswahl, und wählen Sie erneut Löschen aus.

Massen-Veröffentlichen von Daten

Mit der Funktion zum Massen-Veröffentlichen können Sie die bereits in Ihrem Amazon-Cognito-Sync-Speicher gespeicherten Daten in einen Amazon-Kinesis-Stream exportieren. Anweisungen für die Massen-Veröffentlichung aller Streams finden Sie unter Amazon-Cognito-Streams.

Aktivieren der Push-Synchronisierung

Amazon Cognito verfolgt die Zuordnung zwischen Identitäten und Geräten automatisch nach. Mit der Push-Sync-Funktion können Sie dafür sorgen, dass jede Instance einer bestimmten Identität benachrichtigt wird, wenn sich die Identitätsdaten ändern. Damit wird sichergestellt, dass alle einer Identität zugeordneten Geräte bei Änderung des Datensatzes für diese Identität eine automatische Push-Benachrichtigung erhalten, die sie über die Änderung informiert.

Sie können die Push-Synchronisierung über die Amazon-Cognito-Konsole aktivieren.

So aktivieren Sie die Push-Synchronisierung

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Identitätspool-Eigenschaften.
3. Wählen Sie unter Push-Synchronisierung die Option Bearbeiten
4. Wählen Sie Push-Synchronisierung mit Ihrem Identitätspool aktivieren aus.
5. Wählen Sie eine der Amazon Simple Notification Service (Amazon SNS)-Plattformanwendungen aus, die Sie in der aktuellen AWS-Region erstellt haben. Amazon Cognito veröffentlicht Push-Benachrichtigungen für Ihre Plattformanwendung. Wählen Sie Plattformanwendung erstellen aus, um zur Amazon-SNS-Konsole zu navigieren und eine neue zu erstellen.
6. Für die Veröffentlichung in Ihrer Plattformanwendung übernimmt Amazon Cognito eine IAM-Rolle in Ihrem AWS-Konto. Wählen Sie Neue IAM-Rolle erstellen, wenn Sie möchten, dass Amazon Cognito für Sie eine neue Rolle mit grundlegenden Berechtigungen und einer Vertrauensbeziehung zu Ihrem Identitätspool erstellt. Geben Sie einen IAM-Rollen-Namen ein, um Ihre neue Rolle zu identifizieren, zum Beispiel `myidentitypool_authenticatedrole`. Wählen Sie Richtliniendokument anzeigen aus, um die Berechtigungen zu überprüfen, die Amazon Cognito Ihrer neuen IAM-Rolle zuweist.
7. Sie können sich dafür entscheiden, eine bestehende IAM-Rolle zu verwenden, wenn Sie bereits eine Rolle in Ihrer haben AWS-Konto , die Sie verwenden möchten. Sie müssen Ihre IAM-Rollen-Vertrauensrichtlinie so konfigurieren, dass sie `cognito-identity.amazonaws.com` beinhaltet. Konfigurieren Sie Ihre Rollen-Vertrauensrichtlinie so, dass Amazon Cognito die Rolle nur übernehmen kann, wenn nachgewiesen wird, dass die Anforderung von einem authentifizierten Benutzer in Ihrem spezifischen Identitätspool stammt. Weitere Informationen finden Sie unter Vertrauensstellungen und Berechtigungen für Rollen.
8. Wählen Sie Änderungen speichern aus.

Einrichten von Amazon-Cognito-Streams

Amazon-Cognito-Streams bietet Entwicklern Kontrolle und einen detaillierten Überblick über ihre in Amazon-Cognito-Sync gespeicherten Daten. Entwickler können einen Kinesis-Stream nun für den Empfang von Ereignissen als Daten konfigurieren. Amazon Cognito kann jede Datensatzänderung per Push in Echtzeit an einen Kinesis-Stream in Ihrem Besitz übertragen. Anweisungen zum Einrichten von Amazon-Cognito-Streams in der Amazon-Cognito-Konsole finden Sie unter Amazon-Cognito-Streams.

Einrichten von Amazon-Cognito-Ereignissen

Mit Amazon Cognito Events können Sie eine AWS Lambda Funktion als Reaktion auf wichtige Ereignisse in Amazon Cognito Sync ausführen. Amazon-Cognito-Sync löst das Sync-Auslöser-Ereignis aus, wenn ein Datensatz synchronisiert wird. Sie können das Sync Trigger-Ereignis verwenden, um eine Aktion auszuführen, wenn ein Benutzer Daten aktualisiert. Anweisungen zum Einrichten von Amazon-Cognito-Ereignissen über die Konsole finden Sie unter Amazon-Cognito-Ereignisse.

Weitere Informationen AWS Lambda dazu finden Sie unter. AWS Lambda

Identitäten-Pool-Konzepte

Sie können Amazon-Cognito-Identitätspools verwenden, um eindeutige Identitäten für Benutzer zu erstellen und sie mithilfe von Identitätsanbietern zu authentifizieren. Mit einer Identität können Sie temporäre AWS Anmeldeinformationen mit eingeschränkten Rechten abrufen, um auf andere zuzugreifen. AWS-Services Amazon-Cognito-Identitätspools unterstützen öffentliche Identitätsanbieter – Amazon, Apple, Facebook und Google – sowie nicht authentifizierte Identitäten. Es werden außerdem entwicklerauthentifizierte Identitäten unterstützt, sodass Sie Benutzer mit eigenen Backend-Authentifizierungsprozessen registrieren und authentifizieren können.

Weitere Informationen zur regionalen Verfügbarkeit von Amazon-Cognito-Identitäten-Pools finden Sie unter Regionale Verfügbarkeit von AWS -Services. Weitere Informationen zu Amazon Cognito;-Identitäten-Pool-Konzepten finden Sie in den folgenden Themen.

Themen

- Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf
- IAM-Rollen
- Vertrauensstellungen und Berechtigungen für Rollen

Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf

Mit Amazon Cognito erstellen Sie eindeutige, geräte- und plattformübergreifend konsistente IDs für Ihre Endbenutzer. Amazon Cognito stellt Ihrer Anwendung auch temporäre Anmeldeinformationen mit eingeschränkten Rechten für den Zugriff auf Ressourcen zur Verfügung. AWS Auf dieser Seite werden die Grundlagen der Authentifizierung in Amazon Cognito und der Lebenszyklus einer Identität im Identitätspool erklärt.

Authentifzierungsablauf – externer Anbieter

Bei einer Benutzerauthentifizierung mit Amazon Cognito wird für den Bootstrap der Anmeldeinformationen ein Prozess mit mehreren Schritten verwendet. Amazon Cognito bietet zwei verschiedene Authentifizierungsflüsse für öffentliche Anbieter: erweitert und standardmäßig.

Sobald Sie einen dieser Abläufe abgeschlossen haben, können Sie AWS-Services gemäß den Zugriffsrichtlinien Ihrer Rolle auf andere zugreifen. Standardmäßig erstellt die Amazon-Cognito-Konsole Rollen mit Zugriff auf den Amazon-Cognito-Sync-Speicher und auf Amazon Mobile Analytics. Mehr Informationen zum Erteilen von weiteren Zugriffsberechtigungen finden Sie unter IAM-Rollen.

Identitätspools akzeptieren die folgenden Artefakte von Anbietern:

Anbieter	Authentifizierungsartefakt
Amazon-Cognito-Benutzerpool	ID-Token
OpenID Connect (OIDC)	ID-Token
SAML 2.0	SAML-Assertion
Sozialer Anbieter	Zugriffstoken

Erweiterter (vereinfachter) Authentifzierungsablauf

Wenn Sie den erweiterten Authflow verwenden, legt Ihre App in einer Anfrage zunächst einen Authentifizierungsnachweis von einem autorisierten Amazon Cognito Cognito-Benutzerpool oder einem GetIdexternen Identitätsanbieter vor.

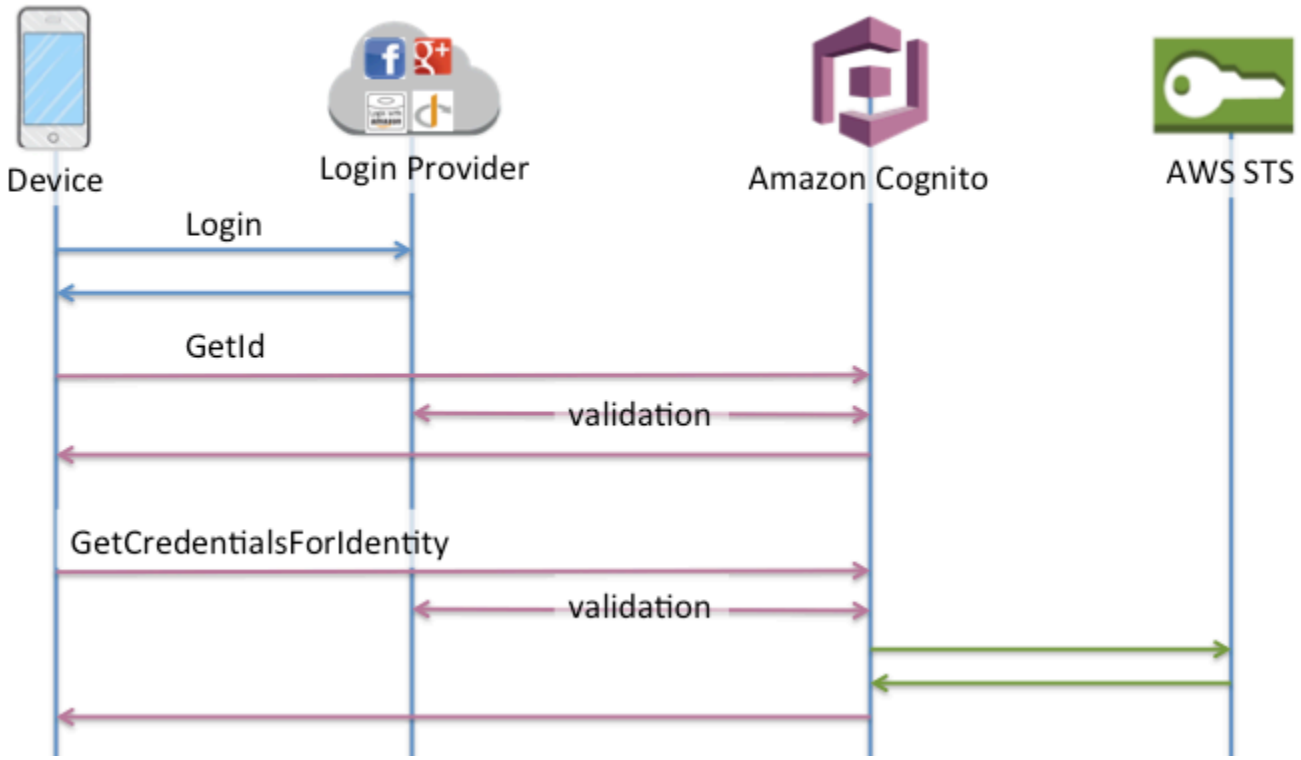
1. Ihre Anwendung präsentiert in einer GetID-Anfrage einen Authentifizierungsnachweis — ein JSON-Webtoken oder eine SAML-Assertion — von einem autorisierten Amazon Cognito Cognito-Benutzerpool oder einem externen Identitätsanbieter.

2. Ihr Identitätspool gibt eine Identitäts-ID zurück.
3. Ihre Anwendung kombiniert die Identitäts-ID mit demselben Authentifizierungsnachweis in einer GetCredentialsForIdentityAnfrage.
4. Ihr Identitätspool gibt AWS Anmeldeinformationen zurück.
5. Ihre Anwendung signiert AWS API-Anfragen mit den temporären Anmeldeinformationen.

Die erweiterte Authentifizierung verwaltet die Logik der IAM-Rollenauswahl und des Abrufs von Anmeldeinformationen in Ihrer Identitätspool-Konfiguration. Sie können Ihren Identitätspool so konfigurieren, dass er eine Standardrolle auswählt und die Prinzipien der attributebasierten Zugriffskontrolle (ABAC) oder der rollenbasierten Zugriffskontrolle (RBAC) auf die Rollenauswahl anwendet. Die AWS Anmeldeinformationen für die erweiterte Authentifizierung sind eine Stunde lang gültig.

Reihenfolge der Vorgänge bei der erweiterten Authentifizierung

1. `GetId`
2. `GetCredentialsForIdentity`



Klassischer Standardauthentifizierungsablauf

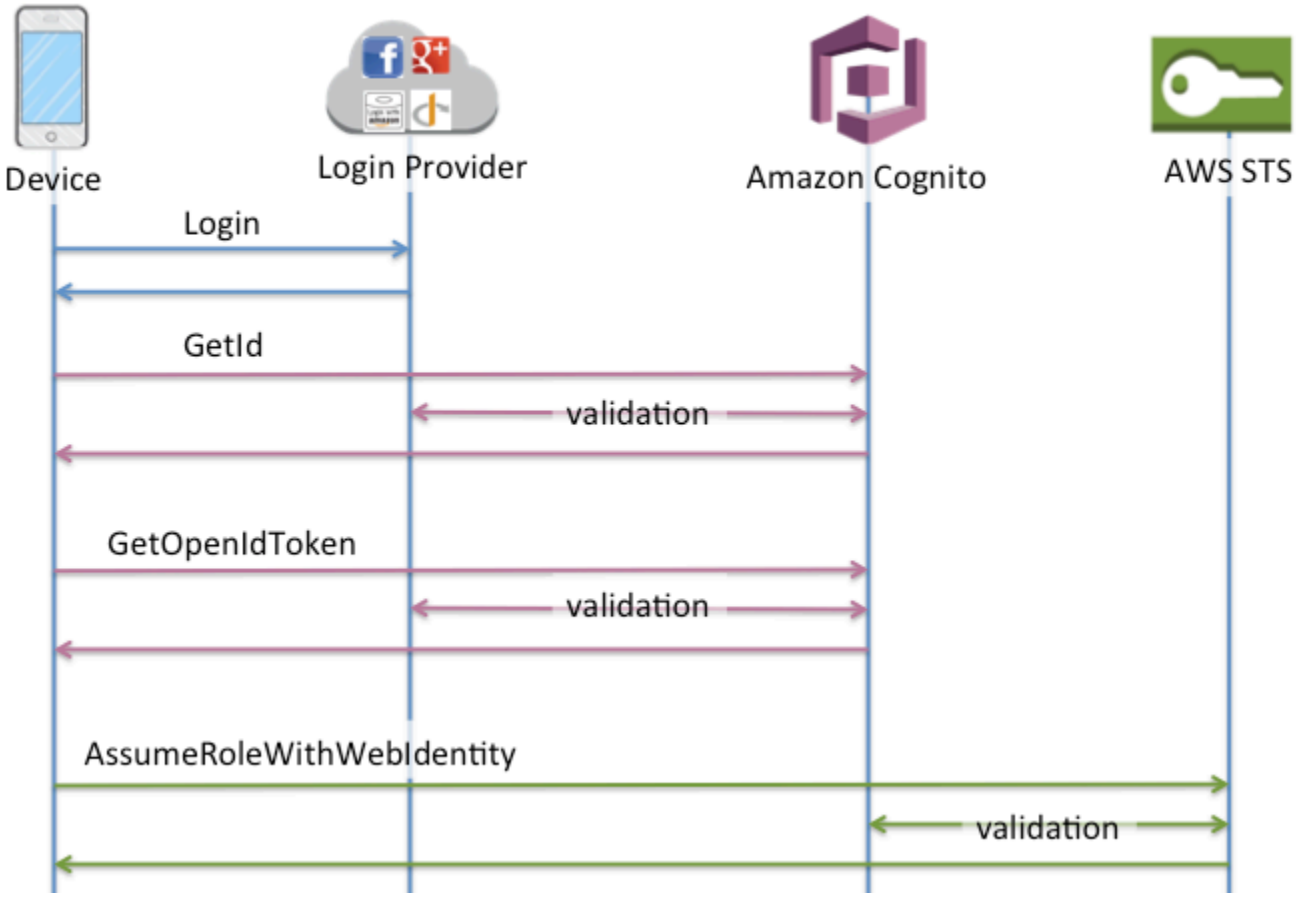
Wenn Sie den grundlegenden Authflow verwenden,

1. Ihre Anwendung präsentiert in einer GetID-Anfrage einen Authentifizierungsnachweis — ein JSON-Webtoken oder eine SAML-Assertion — von einem autorisierten Amazon Cognito Cognito-Benutzerpool oder einem externen Identitätsanbieter.
2. Ihr Identitätspool gibt eine Identitäts-ID zurück.
3. Ihre Anwendung kombiniert die Identitäts-ID mit demselben Authentifizierungsnachweis in einer GetOpenIdTokenAnfrage.
4. `GetOpenIdToken`gibt ein neues OAuth 2.0-Token zurück, das von Ihrem Identitätspool ausgestellt wurde.
5. Ihre Anwendung präsentiert das neue Token in einer AssumeRoleWithWebIdentityAnfrage.
6. AWS Security Token Service AWS STS) gibt AWS Anmeldeinformationen zurück.
7. Ihre Anwendung signiert AWS API-Anfragen mit den temporären Anmeldeinformationen.

Der grundlegende Workflow gibt Ihnen eine genauere Kontrolle über die Anmeldeinformationen, die Sie an Ihre Benutzer verteilen. Die `GetCredentialsForIdentity`-Anforderung des erweiterten Authentifizierungsablaufs fordert eine Rolle basierend auf dem Inhalt eines Zugriffstokens an. Die `AssumeRoleWithWebIdentity` Anfrage im klassischen Workflow gewährt Ihrer App eine bessere Möglichkeit, Anmeldeinformationen für jede AWS Identity and Access Management Rolle anzufordern, die Sie mit einer ausreichenden Vertrauensrichtlinie konfiguriert haben. Sie können auch eine benutzerdefinierte Rollensitzungsdauer anfordern.

Reihenfolge der Vorgänge bei der Standardauthentifizierung

1. `GetId`
2. `GetOpenIdToken`
3. `AssumeRoleWithWebIdentity`



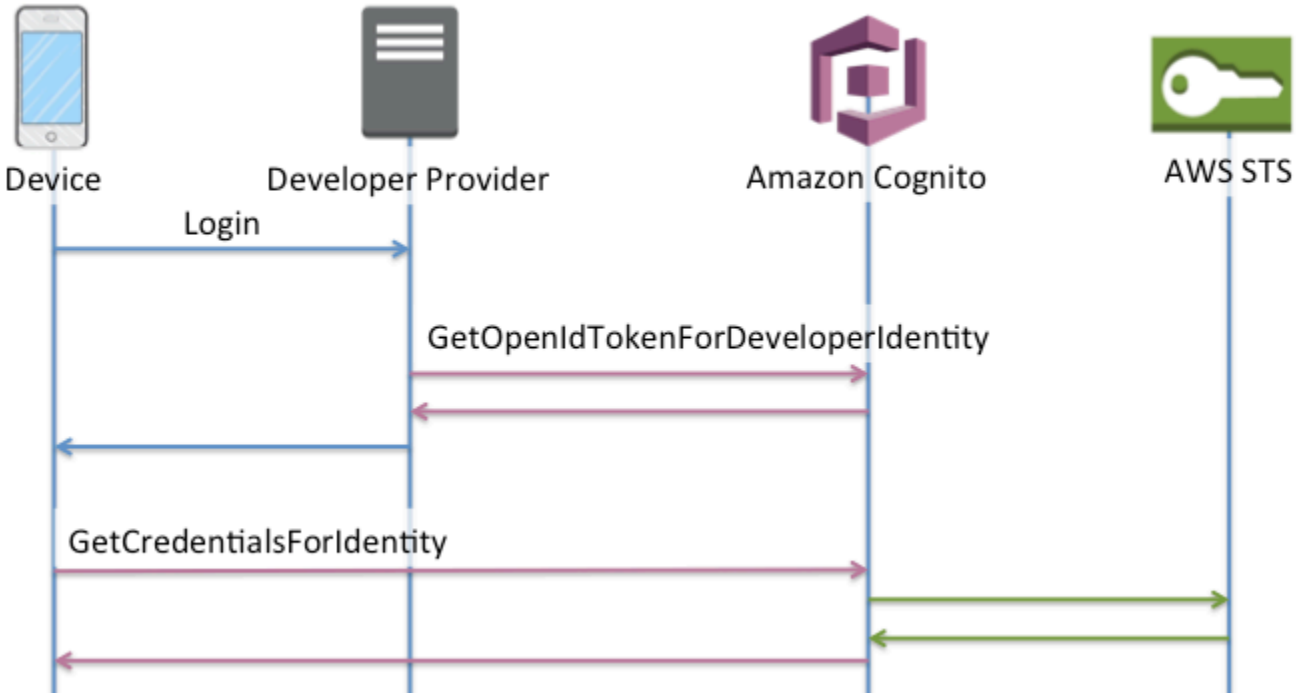
Authentifzierungsablauf für entwicklerauthentifizierte Identitäten

Wenn Sie Entwicklerauthentifizierte Identitäten (Identitätspools) verwenden, nutzt der Client einen anderen Authentifzierungsablauf, der Code außerhalb von Amazon Cognito enthält, um den Benutzer in Ihrem eigenen Authentifizierungssystem zu validieren. Code außerhalb von Amazon Cognito ist als solcher markiert.

Erweiterter Authentifzierungsablauf

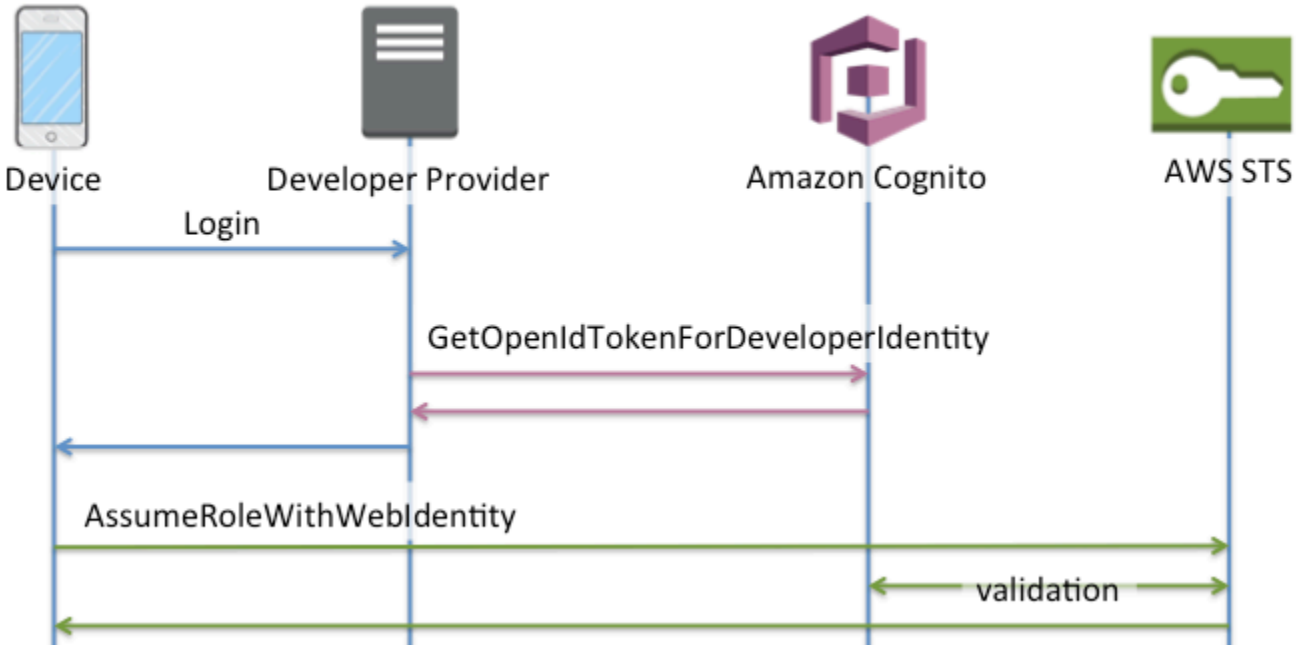
Reihenfolge der Vorgänge bei der erweiterten Authentifizierung bei einem Entwickleranbieter

1. Anmeldung über den Entwickleranbieter (Code außerhalb von Amazon Cognito)
2. Validierung der Benutzeranmeldung (Code außerhalb von Amazon Cognito)
3. GetOpenIdTokenForDeveloperIdentity
4. GetCredentialsForIdentity



Reihenfolge der Vorgänge bei der Standardauthentifizierung mit einem Entwickleranbieter

1. Implementieren Sie Logik außerhalb des Identitätspools, um sich anzumelden und eine Entwickler-Anbieter-ID zu generieren.
2. Rufen Sie gespeicherte serverseitige Anmeldeinformationen ab. AWS
3. Senden Sie die ID des Entwickler-Anbieters in einer GetOpenIdTokenForDeveloperIdentityAPI-Anfrage, die mit autorisierten AWS Anmeldeinformationen signiert ist.
4. Fordern Sie Anmeldeinformationen für die Anwendung an mit AssumeRoleWithWebIdentity.



Welchen Authentifzierungsablauf sollte ich verwenden?

Der erweiterte Ablauf ist die sicherste Wahl mit dem geringsten Aufwand für Entwickler:

- Der verbesserte Ablauf reduziert die Komplexität, Größe und Geschwindigkeit von API-Anfragen.
- Ihre Anwendung muss keine zusätzlichen API-Anfragen an stellen AWS STS.
- Ihr Identitätspool bewertet Ihre Benutzer im Hinblick auf die IAM-Rollenanmeldedaten, die sie erhalten sollten. Sie müssen keine Logik für die Rollenauswahl in Ihren Client einbetten.

> **Important**
>
> Wenn Sie einen neuen Identitätspool erstellen, sollten Sie als bewährte Methode die Standardauthentifizierung (klassische Authentifizierung) nicht standardmäßig aktivieren. Um die Standardauthentifizierung zu implementieren, bewerten Sie zunächst die Vertrauensbeziehungen Ihrer IAM-Rollen für Webidentitäten. Integrieren Sie dann die Logik für die Rollenauswahl in Ihren Client und schützen Sie den Client vor Änderungen durch Benutzer.

Der grundlegende Authentifizierungsablauf delegiert die Logik der IAM-Rollenauswahl an Ihre Anwendung. In diesem Ablauf validiert Amazon Cognito die authentifizierte oder nicht authentifizierte Sitzung Ihres Benutzers und gibt ein Token aus, mit dem Sie Anmeldeinformationen austauschen

können. AWS STS Benutzer können die Token der Standardauthentifizierung gegen alle IAM-Rollen austauschen, die Ihrem Identitätspool vertrauen, oder gegen den Status authentifiziert/nicht authentifiziert. `amr`

Machen Sie sich auch darüber im Klaren, dass die Entwicklerauthentifizierung eine Abkürzung für die Validierung der Identitätsanbieter-Authentifizierung ist. Amazon Cognito vertraut den AWS Anmeldeinformationen, die eine GetOpenIdTokenForDeveloperIdentityAnfrage autorisieren, ohne dass der Inhalt der Anfrage zusätzlich überprüft wird. Schützen Sie die Geheimnisse, die die Entwicklerauthentifizierung autorisieren, vor dem Zugriff durch Benutzer.

Übersicht über API-Befehle

GetId

Der GetIdAPI-Aufruf ist der erste Aufruf, der zur Einrichtung einer neuen Identität in Amazon Cognito erforderlich ist.

Nicht authentifizierter Zugriff

Amazon Cognito kann nicht authentifizierten Gästen Zugriff auf Ihre Anwendungen gewähren. Wenn diese Funktion im Identitätspool aktiviert ist, können die Benutzer über die `GetId`-API jederzeit eine neue Identitäts-ID anfordern. Diese Identitäts-ID wird von der Anwendung für nachfolgende Aufrufe an Amazon Cognito zwischengespeichert. Die AWS mobilen SDKs und das AWS SDK für JavaScript den Browser verfügen über Anbieter von Anmeldeinformationen, die dieses Caching für Sie übernehmen.

Authentifizierter Zugriff

Wenn Ihre Anwendung so konfiguriert ist, dass ein öffentlicher Anmeldeanbieter unterstützt wird (Facebook, Google+, Login with Amazon oder Mit Apple anmelden), können Benutzer auch Token übermitteln (OAuth- oder OpenID Connect-Token), um sich bei diesen Anbietern zu identifizieren. Bei der Verwendung in einem `GetId`-Aufruf erstellt Amazon Cognito entweder eine neue authentifizierte Identität oder gibt die Identität zurück, die bereits mit dieser bestimmten Anmeldung verknüpft ist. Amazon Cognito führt hierzu die Validierung des Tokens beim Anbieter durch und stellt Folgendes sicher:

- Das Token ist gültig und vom konfigurierten Anbieter;
- Das Token ist nicht abgelaufen;
- Das Token entspricht der durch den Anbieter erstellten Anwendungs-ID (z. B. Facebook-App-ID);
- Das Token entspricht der Benutzer-ID.

GetCredentialsForIdentity

Die GetCredentialsForIdentityAPI kann aufgerufen werden, nachdem Sie eine Identitäts-ID eingerichtet haben. Dieser Vorgang entspricht also AssumeRoleWithWebIdentityfunktionell dem Aufrufen GetOpenIdToken.

Damit Amazon Cognito `AssumeRoleWithWebIdentity` für Sie aufrufen kann, müssen Ihrem Identitäten-Pool IAM-Rollen zugewiesen sein. Sie können dies über die Amazon Cognito Cognito-Konsole oder manuell über die SetIdentityPoolRolesBedienung tun.

GetOpenIdToken

Stellen Sie eine GetOpenIdTokenAPI-Anfrage, nachdem Sie eine Identitäts-ID eingerichtet haben. Zwischenspeichern Sie Identitäts-IDs nach Ihrer ersten Anfrage und starten Sie nachfolgende grundlegende (klassische) Sitzungen für diese Identität mit `GetOpenIdToken`.

Die Antwort auf eine `GetOpenIdToken`-API-Anfrage ist ein Token, das Amazon Cognito generiert. Sie können dieses Token als `WebIdentityToken` Parameter in einer AssumeRoleWithWebIdentityAnfrage einreichen.

Bevor Sie das OpenID-Token einreichen, überprüfen Sie es in Ihrer App. Sie können OIDC-Bibliotheken in Ihrem SDK oder eine Bibliothek wie aws-jwt-verify verwenden, um zu bestätigen, dass Amazon Cognito das Token ausgegeben hat. Die Signaturschlüssel-ID, oder `kid`, des OpenID-Tokens ist eine der im jwks_uri-Dokument† von Amazon Cognito Identity aufgeführten IDs. Diese Schlüssel können sich ändern. Ihre Funktion, die Amazon-Cognito-Identitätstokens überprüft, sollte ihre Schlüsselliste regelmäßig aus dem Dokument jwks_uri aktualisieren. Amazon Cognito legt die Aktualisierungsdauer im Cache-Control-Antwort-Header jwks_uri fest, die derzeit auf ein `max-age` von 30 Tagen festgelegt ist.

Nicht authentifizierter Zugriff

Um ein Token für eine nicht authentifizierte Identität abzurufen, benötigen Sie nur die Identitäts-ID. Es ist nicht möglich, ein nicht authentifiziertes Token für authentifizierte oder von Ihnen deaktivierte Identitäten zu erhalten.

Authentifizierter Zugriff

Wenn Sie eine authentifizierte Identität haben, müssen Sie mindestens ein gültiges Token für eine Anmeldung übergeben, die dieser Identität bereits zugeordnet ist. Alle während des `GetOpenIdToken`-Aufrufs übergebenen Token müssen dieselbe zuvor erwähnte Validierung bestehen; Wenn eines der Token fehlschlägt, schlägt der gesamte Aufruf fehl. Die Antwort des

`GetOpenIdToken`-Aufrufs umfasst auch die Identitäts-ID. Der Grund hierfür ist, dass die von Ihnen übergebene Identitäts-ID u. U. nicht identisch mit der ID ist, die zurückgegeben wird.

Verknüpfen von Anmeldungen

Wenn Sie ein Token für eine Anmeldung übergeben, die noch keiner Identität zugewiesen ist, gilt die Anmeldung als mit der zugehörigen Identität „verknüpft“. Sie können nur eine Anmeldung pro öffentlichem Anbieter verknüpfen. Wenn Sie versuchen, mehr als eine Anmeldung mit einem öffentlichen Anbieter zu verknüpfen, wird eine `ResourceConflictException`-Fehlermeldung ausgelöst. Wenn eine Anmeldung nur mit einer vorhandenen Identität verknüpft ist, stimmt die vom `GetOpenIdToken`-Aufruf zurückgegebene Identitäts-ID mit der übergebenen ID überein.

Zusammenführen von Identitäten

Wenn Sie ein Token für eine Anmeldung übergeben, die derzeit nicht mit der angegebenen, sondern mit einer anderen Identität verknüpft ist, werden die beiden Identitäten zusammengeführt. Nach der Zusammenführung wird eine Identität zur übergeordneten bzw. zum Eigentümer aller zugehörigen Anmeldungen. Die andere Identität wird deaktiviert. In diesem Fall wird die übergeordnete Identitäts-ID zurückgegeben. Sie müssen Ihren lokalen Cache aktualisieren, wenn dieser Wert abweicht. Die Anbieter in den AWS mobilen SDKs oder JavaScript im AWS SDK für den Browser führen diesen Vorgang für Sie durch.

GetOpenIdTokenForDeveloperIdentity

Dieser GetOpenIdTokenForDeveloperIdentityVorgang ersetzt die Verwendung von GetIdund GetOpenIdTokenvom Gerät, wenn vom Entwickler authentifizierte Identitäten verwendet werden. Da Ihre Anwendung Anfragen für diesen API-Vorgang mit AWS Anmeldeinformationen signiert, vertraut Amazon Cognito darauf, dass die in der Anfrage angegebene Benutzer-ID gültig ist. Die Entwicklerauthentifizierung ersetzt die Token-Validierung, die Amazon Cognito mit externen Anbietern durchführt.

Die Payload für diese API umfasst eine `logins` Map. Diese Map muss den Schlüssel Ihres Entwickleranbieters und einen Wert als Kennung für den Benutzer in Ihrem System enthalten. Wenn die Benutzer-ID noch nicht mit einer vorhandenen Identität verknüpft ist, erstellt Amazon Cognito eine neue Identität und gibt die neue Identitäts-ID sowie ein OpenID-Connect-Token für diese Identität zurück. Wenn die Benutzer-ID bereits verknüpft ist, gibt Amazon Cognito die bereits vorhandene Identitäts-ID und ein OpenID-Connect-Token zurück. Zwischenspeichern Sie die Entwickleridentitätskennungen nach Ihrer ersten Anfrage und starten Sie nachfolgende grundlegende (klassische) Sitzungen für diese Identität mit `GetOpenIdTokenForDeveloperIdentity`.

Die Antwort auf eine `GetOpenIdTokenForDeveloperIdentity`-API-Anfrage ist ein Token, das Amazon Cognito generiert. Sie können dieses Token als `WebIdentityToken`-Parameter in einer `AssumeRoleWithWebIdentity`-Anfrage einreichen.

Bevor Sie das OpenID-Connect-Token einreichen, überprüfen Sie es in Ihrer App. Sie können OIDC-Bibliotheken in Ihrem SDK oder eine Bibliothek wie aws-jwt-verify verwenden, um zu bestätigen, dass Amazon Cognito das Token ausgegeben hat. Die Signaturschlüssel-ID, oder `kid`, des OpenID-Connect-Tokens ist eine der im jwks_uri-Dokument von Amazon Cognito Identity aufgeführten IDs. Diese Schlüssel können sich ändern. Ihre Funktion, die Amazon-Cognito-Identitätstokens überprüft, sollte ihre Schlüsselliste regelmäßig aus dem Dokument jwks_uri aktualisieren. Amazon Cognito legt die Aktualisierungsdauer im `cache-control`-Antwort-Header jwks_uri fest, die derzeit auf ein `max-age` von 30 Tagen festgelegt ist.

Verknüpfen von Anmeldungen

Wenn zusätzliche Anmeldungen übergeben werden, die nicht bereits mit einer Identität verknüpft sind, erfolgt wie bei den externen Anbietern eine implizite Verknüpfung dieser Anmeldungen mit dieser Identität. Wenn Sie die Anmeldung eines externen Anbieters mit einer Identität verknüpfen, kann der Benutzer den Authentifizierungsablauf des externen Anbieters für diesen Anbieter verwenden. Er kann jedoch den Namen Ihres Entwickleranbieters in der Anmeldungsübersicht nicht verwenden, wenn er `GetId` oder `GetOpenIdToken` aufruft.

Zusammenführen von Identitäten

Mit vom Entwickler authentifizierten Identitäten unterstützt Amazon Cognito sowohl implizites Zusammenführen als auch explizites Zusammenführen über die API. MergeDeveloperIdentities Bei dieser expliziten Zusammenführung können Sie zwei Identitäten mit Benutzer-IDs in Ihrem System als eine einzelne Identität kennzeichnen. Wenn Sie die Quell- und Ziel-Benutzer-IDs angeben, werden sie von Amazon Cognito zusammengeführt. Wenn Sie das nächste Mal ein OpenID-Connect-Token für eine der Benutzer-IDs anfragen, wird die gleiche Identitäts-ID zurückgegeben.

AssumeRoleWithWebIdentity

Sobald Sie ein OpenID Connect-Token haben, können Sie dieses über die AssumeRoleWithWebIdentityAPI-Anfrage an AWS Security Token Service (AWS STS) gegen temporäre AWS Anmeldeinformationen eintauschen.

Da es keine Einschränkung für die Anzahl der Identitäten gibt, die Sie erstellen können, ist es wichtig, die Ihren Benutzern erteilten Benutzerberechtigungen zu verstehen. Richten Sie verschiedene IAM-Rollen für Ihre Anwendung ein: eine für nicht authentifizierte Benutzer und eine

für authentifizierte Benutzer. Die Amazon Cognito Cognito-Konsole kann Standardrollen erstellen, wenn Sie Ihren Identitätspool zum ersten Mal einrichten. Für diese Rollen wurden praktisch keine Berechtigungen erteilt. Passen Sie sie an Ihre Bedürfnisse an.

Weitere Informationen zu Vertrauensstellungen und Berechtigungen für Rollen.

† Das Standarddokument jwks_uri von Amazon Cognito Identity enthält in den meisten AWS-Regionen Informationen über die Schlüssel, mit denen Tokens für Identitätspools signiert werden. Die folgenden Regionen haben unterschiedliche jwks_uri-Dokumente.

Amazon Cognito Identity JSON web key URIs in other AWS-Regionen

AWS-Region	Pfad zum jwks_uri-Dokument
AWS GovCloud (US-West)	`https://cognito-identity.us-gov-west-1.amazonaws.com/.well-known/jwks_uri`
China (Peking)	`https://cognito-identity.cn-north-1.amazonaws.com.cn/.well-known/jwks_uri`
Opt-in-Regionen wie Europa (Mailand) und Afrika (Kapstadt)	`https://cognito-identity.`*`Region`*`.amazonaws.com/.well-known/jwks_uri`

Sie können die jwks_uri auch vom Aussteller oder dem `iss`, den Sie im OpenID-Token von Amazon Cognito erhalten, extrapolieren. Der Discovery-Endpunkt nach OIDC-Standard `<issuer>/.well-known/openid-configuration` listet einen Pfad zur jwks_uri für Ihr Token auf.

IAM-Rollen

Während der Erstellung eines Identitätspools werden Sie aufgefordert, die von Ihren Benutzern übernommenen IAM-Rollen zu aktualisieren. IAM-Rollen funktionieren wie folgt: Wenn sich ein Benutzer bei Ihrer App anmeldet, generiert Amazon Cognito temporäre AWS Anmeldeinformationen für den Benutzer. Diese temporären Anmeldeinformationen sind mit einer bestimmten IAM-Rolle verknüpft. Mit der IAM-Rolle können Sie eine Reihe von Berechtigungen für den Zugriff auf Ihre Ressourcen definieren. AWS

Sie können unterschiedliche Standard-IAM-Rollen für authentifizierte und nicht authentifizierte Benutzer angeben. Zusätzlich können Sie Regeln definieren, um die Rollen der einzelnen Benutzer basierend auf den Ansprüchen im ID-Token eines Benutzers auszuwählen. Weitere Informationen finden Sie unter Verwenden der rollenbasierten Zugriffskontrolle.

Standardmäßig erstellt die Amazon-Cognito-Konsole IAM-Rollen, die den Zugriff auf Amazon Mobile Analytics und Amazon Cognito Sync ermöglichen. Alternativ können Sie vorhandene IAM-Rollen verwenden.

Ändern Sie IAM-Rollen, um den Zugriff auf andere Services zu ermöglichen oder einzuschränken. Melden Sie sich bei der IAM-Konsole an. Klicken Sie auf Rollen und wählen Sie eine Rolle aus. Die Richtlinien, die der Rolle zugeordnet sind, werden auf der Registerkarte Permissions aufgeführt. Sie können eine Zugriffsrichtlinie anpassen, indem Sie auf den entsprechenden Link Manage Policy (Richtlinie verwalten) klicken. Weitere Informationen zum Verwenden und Definieren von Richtlinien finden Sie unter Übersicht über IAM-Richtlinien.

> **Note**
>
> Es hat sich bewährt, Richtlinien zu definieren, die dem Prinzip für das Erteilen der geringsten Rechte folgen. Anders ausgedrückt enthalten die Richtlinien nur diejenigen Berechtigungen, die Benutzer zum Ausführen ihrer Aufgaben benötigen. Weitere Informationen finden Sie unter Gewähren von geringsten Rechten im IAM-Benutzerhandbuch.
> Nicht authentifizierte Identitäten werden von Benutzern angenommen, die sich nicht in Ihrer App anmelden. In der Regel sollten Sie nicht authentifizierten Identitäten geringere Berechtigungen zuweisen als authentifizierten Identitäten.

Themen

Einrichten einer Vertrauensrichtlinie

Amazon Cognito verwendet IAM-Rollen, um temporäre Anmeldeinformationen für die Benutzer Ihrer Anwendung zu generieren. Der Zugriff auf die Berechtigungen wird von den Rollenvertrauensstellungen bestimmt. Weitere Informationen zu Vertrauensstellungen und Berechtigungen für Rollen.

Das Token, dem präsentiert AWS STS wird, wird von einem Identitätspool generiert, der einen Benutzerpool, ein soziales Netzwerk oder ein OIDC-Provider-Token oder eine SAML-Assertion in ein eigenes Token übersetzt. Das Identitätspool-Token enthält einen `aud`-Anspruch, der die Identitätspool-ID ist.

Das folgende Beispiel für eine Rollenvertrauensrichtlinie ermöglicht es dem Verbunddienstprinzipal, die API aufzurufen. `cognito-identity.amazonaws.com` AWS STS `AssumeRoleWithWebIdentity` Die Anfrage ist nur dann erfolgreich, wenn das Identitätspool-Token in der API-Anforderung die folgenden Ansprüche hat.

1. Ein `aud`-Anspruch auf die Identitätspool-ID `us-west-2:abcdefg-1234-5678-910a-0e8443553f95`.
2. Ein `amr`-Anspruch auf `authenticated`, der hinzugefügt wird, wenn sich der Benutzer angemeldet hat und kein Gastbenutzer ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "cognito-identity.amazonaws.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "cognito-identity.amazonaws.com:aud": "us-
west-2:abcdefg-1234-5678-910a-0e8443553f95"
                },
                "ForAnyValue:StringLike": {
                    "cognito-identity.amazonaws.com:amr": "authenticated"
                }
            }
        }
    ]
}
```

Vertrauensrichtlinien für IAM-Rollen bei der Standardauthentifizierung (Classic)

Sie müssen mindestens eine Bedingung anwenden, die die Vertrauensrichtlinien für Rollen einschränkt, die Sie mit Identitätspools verwenden. Wenn Sie Richtlinien zur Rollenvertrauensstellung für Identitätspools erstellen oder aktualisieren, gibt IAM einen Fehler zurück, wenn Sie versuchen, Ihre Änderungen ohne mindestens einen Bedingungsschlüssel zu speichern, der Quellidentitäten einschränkt. AWS STS erlaubt keine kontenübergreifenden AssumeRoleWithWebIdentityOperationen von Identitätspools zu IAM-Rollen, für die eine Bedingung dieser Art fehlt.

Dieses Thema umfasst mehrere Bedingungen, die Quellidentitäten für Identitätspools einschränken. Eine vollständige Liste finden Sie unter Verfügbare Schlüssel für den AWS Web-Identitätsverbund.

Bei der einfachen oder klassischen Authentifizierung mit einem Identitätspool können Sie jede IAM-Rolle übernehmen, AWS STS sofern sie über die richtige Vertrauensrichtlinie verfügt. IAM-Rollen für Amazon Cognito Cognito-Identitätspools vertrauen darauf, dass der Service Principal `cognito-identity.amazonaws.com` die Rolle übernimmt. Diese Konfiguration reicht nicht aus, um Ihre IAM-Rollen vor unbeabsichtigtem Zugriff auf Ressourcen zu schützen. Für Rollen dieses Typs muss eine zusätzliche Bedingung für die Rollenvertrauensrichtlinie gelten. Sie können Rollen für Identitätspools nur erstellen oder ändern, wenn mindestens eine der folgenden Bedingungen erfüllt ist.

`cognito-identity.amazonaws.com:aud`

Beschränkt die Rolle auf Operationen aus einem oder mehreren Identitätspools. Amazon Cognito gibt den Quell-Identitätspool im `aud` Anspruch im Identitätspool-Token an.

`cognito-identity.amazonaws.com:amr`

Beschränkt die Rolle entweder auf Benutzer `authenticated` oder `unauthenticated` (Gast-) Benutzer. Amazon Cognito gibt den Authentifizierungsstatus im `amr` Anspruch im Identitätspool-Token an.

`cognito-identity.amazonaws.com:sub`

Schränkt die Rolle anhand der UUID auf einen oder mehrere Benutzer ein. Diese UUID ist die Identitäts-ID des Benutzers im Identitätspool. Dieser Wert entspricht nicht dem `sub` Wert des ursprünglichen Identitätsanbieters des Benutzers. Amazon Cognito gibt diese UUID im `sub` Antrag im Identitätspool-Token an.

Für die Enhanced-Flow-Authentifizierung muss sich die IAM-Rolle im selben AWS-Konto Identitätspool befinden. Dies ist jedoch bei der Standardauthentifizierung nicht der Fall.

Zusätzliche Überlegungen gelten für Amazon Cognito Cognito-Identitätspools, die kontoübergreifende IAM-Rollen übernehmen. Die Vertrauensrichtlinien dieser Rollen müssen den `cognito-identity.amazonaws.com` Service Principal akzeptieren und die spezifische Bedingung enthalten. `cognito-identity.amazonaws.com:aud` Um unbeabsichtigten Zugriff auf Ihre AWS Ressourcen zu verhindern, beschränkt der `aud` Bedingungsschlüssel die Rolle auf Benutzer aus den Identitätspools im Bedingungswert.

Das Token, das ein Identitätspool für eine Identität ausgibt, enthält Informationen über den Ursprung AWS-Konto des Identitätspools. Wenn Sie ein Identitätspool-Token in einer AssumeRoleWithWebIdentityAPI-Anfrage angeben, wird AWS STS überprüft, ob sich der ursprüngliche Identitätspool in derselben Rolle AWS-Konto wie die IAM-Rolle befindet. Wenn AWS STS festgestellt wird, dass es sich bei der Anfrage um eine kontoübergreifende Anfrage handelt, wird geprüft, ob die Vertrauensrichtlinie für Rollen eine `aud` Bedingung erfüllt. Der Aufruf zur Rollenübernahme schlägt fehl, wenn in der Richtlinie zur Rollenvertrauensstellung keine derartigen Bedingungen erfüllt sind. Wenn die Anfrage nicht kontenübergreifend ist, wird diese Einschränkung AWS STS nicht durchgesetzt. Es hat sich bewährt, immer eine Bedingung dieses Typs auf die Vertrauensrichtlinien Ihrer Identitätspool-Rollen anzuwenden.

Zusätzliche Bedingungen für die Vertrauensrichtlinie

Wiederverwenden von Rollen über Identitäten-Pools hinweg

Wenn Sie eine Rolle übergreifend für mehrere Identitätspools verwenden möchten, da sie einen gemeinsamen Berechtigungssatz haben, können Sie die Identitätspools wie folgt hinzufügen:

```
"StringEquals": {
    "cognito-identity.amazonaws.com:aud": [
        "us-east-1:12345678-abcd-abcd-abcd-123456790ab",
        "us-east-1:98765432-dcba-dcba-dcba-123456790ab"
    ]
}
```

Einschränken des Zugriffs auf bestimmte Identitäten

Erstellen Sie eine Richtlinie, die nur für eine bestimmte Gruppe von App-Benutzern gilt, indem Sie den Wert von prüfen `cognito-identity.amazonaws.com:sub`:

```
"StringEquals": {
    "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
    "cognito-identity.amazonaws.com:sub": [
```

```
        "us-east-1:12345678-1234-1234-1234-123456790ab",
        "us-east-1:98765432-1234-1234-1243-123456790ab"
    ]
}
```

Einschränken des Zugriffs auf bestimmte Anbieter

Erstellen Sie eine Richtlinie nur für die Benutzer, die sich über einen bestimmten Anbieter angemeldet haben (eventuell Ihr eigener Anmeldeanbieter), indem Sie den Wert für überprüfen `cognito-identity.amazonaws.com:amr`:

```
"ForAnyValue:StringLike": {
    "cognito-identity.amazonaws.com:amr": "login.myprovider.myapp"
}
```

Zum Beispiel hat eine App, die nur Facebook vertraut, die folgende AMR-Klausel:

```
"ForAnyValue:StringLike": {
    "cognito-identity.amazonaws.com:amr": "graph.facebook.com"
}
```

Zugriffsrichtlinien

Die Berechtigungen, die Sie einer Rolle zuweisen, gelten für alle Benutzer, die diese Rolle übernehmen. Um den Zugriff Ihrer Benutzer aufzuteilen, verwenden Sie Richtlinienbedingungen und Variablen. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags. Sie können die `sub`-Bedingung verwenden, um die Aktionen auf Amazon-Cognito-Identitäts-IDs in Ihren Zugriffsrichtlinien zu beschränken. Verwenden Sie diese Option mit Vorsicht, insbesondere für nicht authentifizierte Identitäten ohne konsistente Benutzer-ID. Weitere Informationen zu den IAM-Richtlinienvariablen für den Webverbund mit Amazon Cognito finden Sie unter IAM- und AWS STS Bedingungskontextschlüssel im AWS Identity and Access Management Benutzerhandbuch.

Um zusätzliche Sicherheit zu bieten, wendet Amazon Cognito eine Eingrenzungsrichtlinie auf Anmeldeinformationen an, die Sie Ihren nicht authentifizierten Benutzern im erweiterten Authentifizierungsablauf unter Verwendung von `GetCredentialsForIdentity` zuweisen. Die Eingrenzungsrichtlinie fügt den IAM-Richtlinien, die Sie auf Ihre nicht authentifizierte Rolle anwenden, eine Eingebundene Sitzungsrichtlinie und einen AWS Richtlinie für verwaltete Sitzungen hinzu. Da Sie sowohl in den IAM-Richtlinien für Ihre Rolle als auch in den Sitzungsrichtlinien Zugriff gewähren müssen, beschränkt die Eingrenzungsrichtlinie den Zugriff der Benutzer auf nicht in der folgenden Liste aufgeführte Services.

> **Note**
>
> Beim einfachen (klassischen) Ablauf stellen Sie Ihre eigene API-Anforderung AssumeRoleWithWebIdentity und können diese Einschränkungen auf die Anforderung anwenden. Laut einer bewährten Sicherheitsmethode sollten Sie nicht authentifizierten Benutzern keine über diese Eingrenzungsrichtlinie hinausgehenden Berechtigungen zuweisen.

Amazon Cognito verhindert außerdem, dass authentifizierte und nicht authentifizierte Benutzer API-Anforderungen an Amazon-Cognito-Identitätspools und Amazon Cognito Sync stellen. Andere AWS-Services könnten den Zugriff auf Dienste über Web-Identitäten einschränken.

Bei einer erfolgreichen Anforderung mit dem erweiterten Authentifizierungsablauf stellt Amazon Cognito im Hintergrund eine API-Anforderung `AssumeRoleWithWebIdentity`. In den Parametern in dieser Anforderung schließt Amazon Cognito Folgendes ein.

1. Die Identitäts-ID Ihres Benutzers.
2. Den ARN der IAM-Rolle, die Ihr Benutzer übernehmen möchte.
3. Einen Parameter `policy`, der eine eingebundene Sitzungsrichtlinie hinzufügt.
4. Ein `PolicyArns.member.N` Parameter, dessen Wert eine AWS verwaltete Richtlinie ist, die zusätzliche Berechtigungen in Amazon gewährt CloudWatch.

Services, auf die nicht authentifizierte Benutzer zugreifen können

Wenn Sie den erweiterten Ablauf verwenden, verhindern die Scope-down-Richtlinien, die Amazon Cognito auf die Sitzung Ihres Benutzers anwendet, dass dieser andere als die in der folgenden Tabelle aufgeführten Services nutzt. Für einen Teil der Services sind nur bestimmte Aktionen zulässig.

Kategorie	Service
Analysen	Amazon Data Firehose
	Amazon Managed Service für Apache Flink
Anwendungsintegration	Amazon Simple Queue Service

Kategorie	Service
AR und VR	Amazon Sumerian[1]
Geschäftsanwendungen	Amazon Mobile Analytics Amazon Simple Email Service
Datenverarbeitung	AWS Lambda
Kryptografie und PKI	AWS Key Management Service[1]
Datenbank	Amazon-DynamoDB Amazon SimpleDB
Front-End Web und Mobil	AWS AppSync Amazon Location Service Amazon Simple Notification Service Amazon Pinpoint
Entwicklung von Spielen	Amazon GameLift
Internet of Things (IoT)	AWS IoT

Kategorie	Service
Machine Learning	Amazon CodeWhisperer Amazon Comprehend Amazon Lex Amazon Machine Learning Amazon Personalize Amazon Polly Amazon Rekognition Amazon SageMaker [1] Amazon Textract[1] Amazon Transcribe Amazon Translate
Management und Governance	Amazon CloudWatch CloudWatch Amazon-Protokolle
Netzwerk und Bereitstellung von Inhalten	Amazon API Gateway
Sicherheit, Identität und Compliance	Amazon-Cognito-Benutzerpools
Speicher	Amazon Simple Storage Service

[1] Für die AWS-Services in der folgenden Tabelle aufgeführten Fälle gewährt die Inline-Richtlinie eine Teilmenge von Aktionen. In der Tabelle werden die jeweils verfügbaren Aktionen angezeigt.

AWS-Service	Maximale Berechtigungen für nicht authentifizierte Benutzer mit erweitertem Authentifizierungsablauf

AWS Key Management Service	`Encrypt`
	`Decrypt`
	`ReEncrypt`
	`GenerateDataKey`

Amazon SageMaker	`InvokeEndpoint`
Amazon Textract	`DetectDocumentText` `AnalyzeDocument`
Amazon Sumerian	`View*`

Um Zugriff zu gewähren, der AWS-Services über diese Liste hinausgeht, aktivieren Sie den grundlegenden (klassischen) Authentifizierungsablauf in Ihrem Identitätspool. Wenn Ihre Benutzer `NotAuthorizedException`-Fehler von AWS-Services erhalten, die durch die Richtlinien, die der IAM-Rolle für nicht authentifizierte Benutzer zugewiesen sind, erlaubt werden, sollten Sie prüfen, ob Sie den betreffenden Service aus Ihrem Anwendungsfall entfernen können. Wenn dies nicht möglich ist, wechseln Sie zum Basisablauf.

Die eingebundene Sitzungsrichtlinie

Die Richtlinie für Inlinesitzungen schränkt die effektiven Berechtigungen Ihres Benutzers so ein, dass auch der Zugriff auf die in der folgenden Liste aufgeführten AWS-Services Berechtigungen nicht möglich ist. AWS-Services In den Richtlinien, die Sie für die IAM-Rolle des Benutzers anwenden, müssen Sie diesen auch Berechtigungen gewähren. Die effektiven Berechtigungen eines Benutzers für eine Sitzung mit übernommener Rolle entsprechen der Schnittmenge der seiner Rolle zugewiesenen Richtlinien und seiner Sitzungsrichtlinie. Weitere Informationen finden Sie unter Sitzungsrichtlinien im Benutzerhandbuch zu AWS Identity and Access Management .

Amazon Cognito fügt die folgende Inline-Richtlinie zu Sitzungen für Ihre Benutzer in AWS-Regionen hinzu, die standardmäßig aktiviert sind.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "logs:*",
                "dynamodb:*",
                "kinesis:*",
                "mobileanalytics:*",
                "s3:*",
                "ses:*",
                "sns:*",
                "sqs:*",
                "lambda:*",
                "machinelearning:*",
                "execute-api:*",
                "iot:*",
                "gamelift:*",
                "scs:*",
                "cognito-identity:*",
                "cognito-idp:*",
                "lex:*",
                "polly:*",
                "comprehend:*",
                "translate:*",
                "transcribe:*",
                "rekognition:*",
                "mobiletargeting:*",
                "firehose:*",
                "appsync:*",
                "personalize:*",
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "sagemaker:InvokeEndpoint",
                "cognito-sync:*",
                "sumerian:View*",
                "codewhisperer:*",
                "textract:DetectDocumentText",
                "textract:AnalyzeDocument",
```

```
                "sdb:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Für alle anderen Regionen umfasst die Inline-Scope-down-Richtlinie alles, was in den Standardregionen aufgeführt ist, mit Ausnahme der folgenden `Action`-Anweisungen.

```
                "cognito-sync:*",
                "sumerian:View*",
                "codewhisperer:*",
                "textract:DetectDocumentText",
                "textract:AnalyzeDocument",
                "sdb:*"
```

Die Richtlinie für AWS verwaltete Sitzungen

Amazon Cognito begrenzt darüber hinaus den Umfang der Berechtigungen nicht authentifizierter Benutzer mit der von AWS verwalteten Richtlinie `AmazonCognitoUnAuthedIdentitiesSessionPolicy` auf Ihre nicht authentifizierten Benutzer im erweiterten Ablauf. Sie müssen diese Berechtigung auch in den Richtlinien gewähren, die Sie Ihrer nicht authentifizierten IAM-Rolle zuordnen.

Die verwaltete `AmazonCognitoUnAuthedIdentitiesSessionPolicy`-Richtlinie enthält die folgenden Berechtigungen.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "rum:PutRumEvents",
            "polly:*",
            "comprehend:*",
            "translate:*",
            "transcribe:*",
            "rekognition:*",
```

```
            "mobiletargeting:*",
            "firehose:*",
            "personalize:*",
            "sagemaker:InvokeEndpoint"
        ],
        "Resource": "*"
    }]
}
```

Beispiele für Zugriffsrichtlinien

In diesem Abschnitt finden Sie Beispiele für Amazon-Cognito-Zugriffsrichtlinien, die Ihren Benutzern nur die grundlegenden Berechtigungen gewähren, die zum Ausführen einer bestimmten Operation erforderlich sind. Sie können die Berechtigungen für eine bestimmte Identitäts-ID weiter einschränken, indem Sie nach Möglichkeit Richtlinienvariablen verwenden. Beispiel: Verwenden Sie ${cognito-identity.amazonaws.com:sub}. Weitere Informationen finden Sie unter Grundlegendes zur Amazon-Cognito-Authentifizierung, Teil 3: Rollen und Richtlinien im AWS -Mobile-Blog.

> **Note**
>
> Als bewährte Sicherheitsmethode sollten Richtlinien nur die Berechtigungen enthalten, die Benutzer zum Ausführen ihrer Aufgaben benötigen. Das bedeutet, dass Sie versuchen sollten, den Zugriff nach Möglichkeit immer auf eine einzelne Identität für Objekte zu beschränken.

Einer Identität Lesezugriff auf ein einzelnes Objekt in Amazon S3 gewähren

Die folgende Zugriffsrichtlinie erteilt einer Identität Leseberechtigungen für den Abruf eines einzigen Objekts aus einem bestimmten S3-Bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/assets/my_picture.jpg"]
```

```
    }
  ]
}
```

Einer Identität sowohl Lese- als auch Schreibzugriff auf identitätsspezifische Pfade in Amazon S3 gewähren

Die folgende Zugriffsrichtlinie gewährt Lese- und Schreibberechtigungen für den Zugriff auf ein bestimmtes Präfix "folder" in einem S3-Bucket, indem das Präfix der Variablen zugeordnet wird.

Mit dieser Richtlinie kann eine über `us-east-1:12345678-1234-1234-1234-123456790ab` eingefügte Identität wie `${cognito-identity.amazonaws.com:sub}` Objekte in `arn:aws:s3:::mybucket/us-east-1:12345678-1234-1234-1234-123456790ab` abrufen, ablegen und auflisten. Allerdings wird der Identität kein Zugriff auf andere Objekte in gewährt `arn:aws:s3:::mybucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${cognito-
identity.amazonaws.com:sub}/*"]}}
    },
    {
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/${cognito-identity.amazonaws.com:sub}/*"]
    }
  ]
}
```

Zuweisen des differenzierten Zugriffs von Identitäten auf Amazon DynamoDB

Die folgende Zugriffsrichtlinie bietet eine differenzierte Zugriffskontrolle auf DynamoDB-Ressourcen mithilfe von Amazon Cognito-Umgebungsvariablen. Diese Variablen erteilen Zugriff auf Elemente in

DynamoDB anhand der Identitäts-ID. Weitere Informationen finden Sie unter Verwenden von IAM-Richtlinienbedingungen für die differenzierte Zugriffskontrolle im Entwicklerhandbuch für Amazon DynamoDB .

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
      "Resource": [
        "arn:aws:dynamodb:us-west-2:123456789012:table/MyTable"
      ],
      "Condition": {
        "ForAllValues:StringEquals": {
          "dynamodb:LeadingKeys":  ["${cognito-identity.amazonaws.com:sub}"]
        }
      }
    }
  ]
}
```

Einer Identität die Berechtigung zum Aufrufen einer Lambda-Funktion zuweisen

Die folgende Zugriffsrichtlinie erteilt einer Identität Berechtigungen zum Aufrufen einer Lambda-Funktion.

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
             "Effect": "Allow",
             "Action": "lambda:InvokeFunction",
             "Resource": [
```

```
                "arn:aws:lambda:us-west-2:123456789012:function:MyFunction"
            ]
        }
    ]
}
```

Einer Identität die Berechtigung zur Veröffentlichung von Kinesis-Daten-Streams zuweisen

Die folgende Zugriffsrichtlinie erlaubt es einer Identität, die Operation `PutRecord` bei einem beliebigen Kinesis Data Stream einzusetzen. Sie kann auf Benutzer angewendet werden, die Datensätze zu allen Streams in einem Konto hinzufügen müssen. Weitere Informationen finden Sie unter Steuern des Zugriffs auf Amazon Kinesis Data Streams-Ressourcen mithilfe von IAM im Amazon Kinesis Data Streams-Entwicklerleitfaden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kinesis:PutRecord",
            "Resource": [
                "arn:aws:kinesis:us-east-1:111122223333:stream/stream1"
            ]
        }
    ]
}
```

Einer Identität Zugriff auf ihre Daten im Amazon-Cognito-Sync-Speicher zuweisen

Die folgende Zugriffsrichtlinie erteilt einer Identität nur Berechtigungen auf ihre eigenen Daten im Amazon-Cognito-Sync-Speicher.

```
{
  "Version": "2012-10-17",
  "Statement":[{
      "Effect":"Allow",
      "Action":"cognito-sync:*",
      "Resource":["arn:aws:cognito-sync:us-east-1:123456789012:identitypool/${cognito-
identity.amazonaws.com:aud}/identity/${cognito-identity.amazonaws.com:sub}/*"]
      }]
  }
```

Vertrauensstellungen und Berechtigungen für Rollen

Diese Rollen unterscheiden sich in ihren Vertrauensstellungen. Im folgenden Beispiel wird eine Vertrauensrichtlinie für eine nicht authentifizierte Rolle dargestellt:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-corner-
cafe-123456790ab"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "unauthenticated"
        }
      }
    }
  ]
}
```

Die Richtlinie erteilt Verbundbenutzer aus `cognito-identity.amazonaws.com` (Aussteller des OpenID Connect-Tokens) die Berechtigung, diese Rolle zu übernehmen. Darüber hinaus schränkt die Richtlinie den `aud`-Anspruch des Tokens (in diesem Fall die Identitäten-Pool-ID) dahingehend ein, dass er mit dem Identitäten-Pool übereinstimmen muss. Schließlich gibt die Richtlinie an, dass eines der Array-Mitglieder des mehrwertigen `amr`-Anspruchs des Tokens, das von der Amazon Cognito `GetOpenIdToken`-API-Operation ausgegeben wird, den Wert `unauthenticated` hat.

Wenn Amazon Cognito ein Token erstellt, wird die `amr` des Tokens entweder als `unauthenticated` oder `authenticated` festgelegt. Wenn `amr` `authenticated` ist, enthält das Token alle Anbieter, die während der Authentifizierung verwendet werden. Das bedeutet, dass Sie eine Rolle erstellen können, die nur Benutzern vertraut, die sich über Facebook angemeldet haben, indem Sie wie folgt die `amr`-Bedingung ändern:

```
"ForAnyValue:StringLike": {
  "cognito-identity.amazonaws.com:amr": "graph.facebook.com"
}
```

Gehen Sie bei der Änderung von Vertrauensstellungen für Ihre Rollen oder der übergreifenden Verwendung von Rollen für mehrere Identitätspools sorgfältig vor. Wenn Sie Ihre Rolle nicht korrekt konfigurieren, damit sie dem Identitätspool korrekt vertraut, wird Ihnen eine STS-Ausnahme wie die Folgende angezeigt:

```
AccessDenied -- Not authorized to perform sts:AssumeRoleWithWebIdentity
```

Wenn Sie diese Nachricht sehen, überprüfen Sie noch einmal, ob Sie die richtige Rolle für den Identitätspool und die Authentifizierungsart verwenden.

Bewährte Sicherheitsmethoden für Amazon Cognito Cognito-Identitätspools

Amazon Cognito Cognito-Identitätspools stellen temporäre AWS Anmeldeinformationen für Ihre Anwendung bereit. AWS-Konten enthalten häufig sowohl die Ressourcen, die Ihre Anwendungsbenutzer benötigen, als auch private Back-End-Ressourcen. Die IAM-Rollen und -Richtlinien, aus denen sich die AWS Anmeldeinformationen zusammensetzen, können Zugriff auf jede dieser Ressourcen gewähren.

Die wichtigste bewährte Methode bei der Konfiguration von Identitätspools besteht darin, sicherzustellen, dass Ihre Anwendung ihre Aufgaben ohne übermäßige oder unbeabsichtigte Zugriffsrechte ausführen kann. Um Sicherheitsfehler zu vermeiden, sollten Sie sich diese Empfehlungen vor dem Start jeder Anwendung, die Sie für die Produktion freigeben möchten, durchlesen.

Themen

Bewährte Methoden für die IAM-Konfiguration

Wenn ein Gast oder ein authentifizierter Benutzer eine Sitzung in Ihrer Anwendung initiiert, für die Identitätspool-Anmeldeinformationen erforderlich sind, ruft Ihre Anwendung temporäre

AWS Anmeldeinformationen für eine IAM-Rolle ab. Die Anmeldeinformationen können für eine Standardrolle, eine Rolle, die durch Regeln in Ihrer Identitätspool-Konfiguration ausgewählt wurde, oder für eine benutzerdefinierte Rolle, die von Ihrer App ausgewählt wurde, verwendet werden. Mit den jeder Rolle zugewiesenen Berechtigungen erhält Ihr Benutzer Zugriff auf Ihre AWS Ressourcen.

Weitere Informationen zu den allgemeinen Best Practices für IAM finden Sie unter Bewährte Methoden für IAM im AWS Identity and Access Management Benutzerhandbuch.

Verwenden Sie Vertrauensrichtlinienbedingungen in IAM-Rollen

IAM erfordert, dass Rollen für Identitätspools mindestens eine Bedingung für eine Vertrauensrichtlinie haben. Mit dieser Bedingung kann beispielsweise der Geltungsbereich der Rolle auf nur authentifizierte Benutzer festgelegt werden. AWS STS erfordert außerdem, dass kontoübergreifende Standardauthentifizierungsanforderungen zwei spezifische Bedingungen erfüllen: `cognito-identity.amazonaws.com:aud` und. `cognito-identity.amazonaws.com:amr` Es hat sich bewährt, diese beiden Bedingungen auf alle IAM-Rollen anzuwenden, die dem Identitätspools-Dienstprinzipal vertrauen. `cognito-identity.amazonaws.com`

- `cognito-identity.amazonaws.com:aud`: Der Aud-Anspruch im Identitätspool-Token muss mit einer vertrauenswürdigen Identitätspool-ID übereinstimmen.
- `cognito-identity.amazonaws.com:amr`: Der AMR-Anspruch im Identitätspool-Token muss entweder authentifiziert oder nicht authentifiziert sein. Mit dieser Bedingung können Sie den Zugriff auf eine Rolle nur nicht authentifizierten Gästen oder nur authentifizierten Benutzern vorbehalten. Sie können den Wert dieser Bedingung weiter verfeinern, um die Rolle beispielsweise auf Benutzer eines bestimmten Anbieters zu beschränken. `graph.facebook.com`

Das folgende Beispiel für eine Vertrauensrichtlinie für Rollen gewährt Zugriff auf eine Rolle unter den folgenden Bedingungen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
```

```
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-east-1:a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
      }
    }
  ]
}
```

Elemente, die sich auf Identitätspools beziehen

- `"Federated": "cognito-identity.amazonaws.com"`: Benutzer müssen aus einem Identitätspool stammen.
- `"cognito-identity.amazonaws.com:aud": "us-east-1:a1b2c3d4-5678-90ab-cdef-example11111"`: Benutzer müssen aus einem bestimmten Identitätspool stammen`us-east-1:a1b2c3d4-5678-90ab-cdef-example11111`.
- `"cognito-identity.amazonaws.com:amr": "authenticated"`: Benutzer müssen authentifiziert sein. Gastbenutzer können die Rolle nicht übernehmen.

Wenden Sie Berechtigungen mit den geringsten Rechten an

Wenn Sie mit IAM-Richtlinien Berechtigungen für authentifizierten Zugriff oder Gastzugriff festlegen, gewähren Sie nur die spezifischen Berechtigungen, die für die Ausführung bestimmter Aufgaben erforderlich sind, oder die Berechtigungen mit den geringsten Rechten. Die folgende Beispiel-IAM-Richtlinie gewährt, wenn sie auf eine Rolle angewendet wird, schreibgeschützten Zugriff auf eine einzelne Image-Datei in einem Amazon S3 S3-Bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/assets/my_picture.jpg"]
```

```
    }
  ]
}
```

Bewährte Methoden zur Konfiguration des Identitätspo

Identitätspools bieten flexible Optionen für die Generierung von AWS Anmeldeinformationen. Verwenden Sie keine Abkürzungen beim Design, wenn Ihre Anwendung mit den sichersten Methoden arbeiten kann.

Machen Sie sich mit den Auswirkungen des Gastzugangs vertraut

Ein nicht authentifizierter Gastzugriff ermöglicht es Benutzern, Daten von Ihnen abzurufen, AWS-Konto bevor sie sich anmelden. Jeder, der Ihre Identitätspool-ID kennt, kann nicht authentifizierte Anmeldeinformationen anfordern. Ihre Identitätspool-ID ist keine vertrauliche Information. Wenn Sie den Gastzugriff aktivieren, sind die AWS Berechtigungen, die Sie für nicht authentifizierte Sitzungen gewähren, für alle verfügbar.

Es hat sich bewährt, den Gastzugriff deaktiviert zu lassen und die erforderlichen Ressourcen erst abzurufen, nachdem sich die Benutzer authentifiziert haben. Wenn Ihre Anwendung vor der Anmeldung Zugriff auf Ressourcen benötigt, treffen Sie die folgenden Vorsichtsmaßnahmen.

- Machen Sie sich mit den automatischen Beschränkungen für Rollen ohne Authentifizierung vertraut.
- Überwachen Sie die Berechtigungen Ihrer nicht authentifizierten IAM-Rollen und passen Sie sie an die spezifischen Anforderungen Ihrer Anwendung an.
- Gewähren Sie Zugriff auf bestimmte Ressourcen.
- Sichern Sie sich die Vertrauensrichtlinie Ihrer standardmäßigen, nicht authentifizierten IAM-Rolle.
- Aktivieren Sie den Gastzugriff nur, wenn Sie sicher sind, dass Sie die Berechtigungen in Ihrer IAM-Rolle jedem im Internet gewähren würden.

Verwenden Sie standardmäßig die erweiterte Authentifizierung

Bei der (klassischen) Standardauthentifizierung delegiert Amazon Cognito die Auswahl der IAM-Rolle an Ihre App. Im Gegensatz dazu verwendet der erweiterte Ablauf die zentralisierte Logik in Ihrem Identitätspool, um die IAM-Rolle zu bestimmen. Außerdem bietet er zusätzliche Sicherheit für nicht authentifizierte Identitäten mit einer Scopedown-Richtlinie, die eine Obergrenze für

IAM-Berechtigungen festlegt. Der erweiterte Ablauf ist die sicherste Option mit dem geringsten Aufwand für Entwickler. Weitere Informationen zu diesen Optionen finden Sie unterIdentitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf.

Der grundlegende Ablauf kann die clientseitige Logik offenlegen, die bei der Rollenauswahl und Zusammenstellung der AWS STS-API-Anforderung für Anmeldeinformationen verwendet wird. Der erweiterte Ablauf verbirgt sowohl die Logik als auch die Anfrage zur Rollenübernahme hinter der Automatisierung des Identitätspools.

Wenden Sie bei der Konfiguration der Standardauthentifizierung die bewährten IAM-Methoden auf Ihre IAM-Rollen und deren Berechtigungen an.

Verwenden Sie Entwickler-Anbieter auf sichere Weise

Von Entwicklern authentifizierte Identitäten sind eine Funktion von Identitätspools für serverseitige Anwendungen. Der einzige Authentifizierungsnachweis, den Identitätspools für die Entwicklerauthentifizierung benötigen, sind die AWS Anmeldeinformationen eines Identitätspool-Entwicklers. Identitätspools erzwingen keine Einschränkungen der Gültigkeit der Entwickler-/Anbieter-Identifikatoren, die Sie in diesem Authentifizierungsablauf angeben.

Es hat sich bewährt, Entwickleranbieter nur unter den folgenden Bedingungen zu implementieren:

- Um die Verantwortung für die Verwendung von vom Entwickler authentifizierten Anmeldeinformationen zu übernehmen, sollten Sie den Namen und die Kennungen Ihres Entwickler-Anbieters so gestalten, dass sie die Authentifizierungsquelle angeben. Zum Beispiel: `"Logins" : {"MyCorp provider" : "[provider application ID]"}`.
- Vermeiden Sie langlebige Benutzeranmeldedaten. Konfigurieren Sie Ihren serverseitigen Client so, dass er Identitäten mit serviceverknüpften Rollen wie EC2-Instanzprofilen und Lambda-Ausführungsrollen anfordert.
- Vermeiden Sie es, interne und externe Vertrauensquellen im selben Identitätspool zu vermischen. Fügen Sie Ihren Entwickleranbieter und Ihre Single Sign-On-Anbieter (SSO) in separaten Identitätspools hinzu.

Verwenden von Attributen für Zugriffskontrolle

Die attributbasierte Zugriffskontrolle ist die Amazon–Cognito-Identitätspool-Implementierung attributbasierter Zugriffskontrolle (ABAC). Sie können IAM-Richtlinien verwenden, um den Zugriff

auf AWS-Ressourcen über Amazon-Cognito-Identitätspools basierend auf Benutzerattributen zu steuern. Diese Attribute können von Anbietern sozialer Identitäten und Anbietern für Identitäten für Unternehmen bezogen werden. Sie können Attribute innerhalb von Zugriffs- und ID-Token oder SAML-Assertionen Tags zuordnen, auf die in den IAM-Berechtigungsrichtlinien verwiesen werden kann.

Sie können Standardmappings auswählen oder eigene benutzerdefinierte Mappings in Amazon-Cognito-Identitätspools erstellen. Mit den Standardmappings können Sie IAM-Richtlinien basierend auf einem festen Satz von Benutzerattributen schreiben. Mit benutzerdefinierten Mappings können Sie einen benutzerdefinierten Satz von Benutzerattributen auswählen, auf die in den IAM-Berechtigungsrichtlinien verwiesen wird. Die Attributnamen in der Amazon-Cognito-Konsole werden dem Tag-Schlüssel für den Prinzipal zugeordnet, bei dem es sich um die Tags handelt, auf die in der IAM-Berechtigungsrichtlinie verwiesen wird.

Angenommen, Sie besitzen einen Medien-Streaming-Service mit einer kostenlosen und einer kostenpflichtigen Mitgliedschaft. Sie speichern die Mediendateien in Amazon S3 und markieren sie mit kostenlosen oder Premium-Tags. Sie können Attribute für die Zugriffskontrolle verwenden, um den Zugriff auf kostenlose und kostenpflichtige Inhalte basierend auf der Ebene der Benutzermitgliedschaft zu ermöglichen, die Teil des Benutzerprofils ist. Sie können das Mitgliedschaftsattribut einem Tag-Schlüssel zuordnen, der an die IAM-Berechtigungsrichtlinie weitergegeben werden soll. Auf diese Weise können Sie eine einzelne Berechtigungsrichtlinie erstellen und den Zugriff auf Premium-Inhalte basierend auf dem Wert der Mitgliedschaftsstufe und des Tags für die Inhaltsdateien bedingt zulassen.

Themen

Das Verwenden von Attributen zum Steuern des Zugriffs bietet mehrere Vorteile:

- Die Berechtigungsverwaltung ist effizienter, wenn Sie Attribute für die Zugriffskontrolle verwenden. Sie können eine grundlegende Berechtigungsrichtlinie erstellen, die Benutzerattribute verwendet, anstatt mehrere Richtlinien für verschiedene Auftragsfunktionen zu erstellen.
- Sie müssen Ihre Richtlinien nicht aktualisieren, wenn Sie Ressourcen oder Benutzer für Ihre Anwendung hinzufügen oder entfernen. Die Berechtigungsrichtlinie gewährt den Zugriff

nur Benutzern mit den übereinstimmenden Benutzerattributen. Beispielsweise müssen Sie möglicherweise den Zugriff auf bestimmte S3 Buckets basierend auf dem Auftragstitel der Benutzer steuern. In diesem Fall können Sie eine Berechtigungsrichtlinie erstellen, um den Zugriff auf diese Dateien nur Benutzern innerhalb des definierten Auftragstitels zu ermöglichen. Weitere Informationen finden Sie unter IAM-Tutorial: Verwenden von SAML-Sitzungs-Tags für ABAC.

- Attribute können als Prinzipal-Tags an eine Richtlinie übergeben werden, die Berechtigungen basierend auf den Werten dieser Attribute zulässt oder verweigert.

Verwenden von Attributen für die Zugriffskontrolle mit Amazon-Cognito-Identitätspools

Bevor Sie Attribute für die Zugriffskontrolle verwenden können, müssen Sie die folgenden Voraussetzungen erfüllen:

- Ein AWS-Konto
- Benutzerpool
- Identitäten-Pool
- Einrichten eines SDK
- Integrierte Identitätsanbieter
- Anmeldeinformation

Um Attribute für die Zugriffskontrolle zu verwenden, legt die Anforderung, die Sie als Datenquelle angeben, den Wert des von Ihnen ausgewählten Tag-Schlüssels fest. Amazon Cognito wendet den Tag-Schlüssel und den Wert auf die Sitzung Ihres Benutzers an. Ihre IAM-Richtlinien können den Zugriff Ihres Benutzers anhand der `${aws:PrincipalTag/tagkey}`-Bedingung evaluieren. IAM evaluiert den Wert des Tags Ihres Benutzers anhand der Richtlinie.

Sie müssen IAM-Rollen vorbereiten, deren Anmeldeinformationen Sie an Ihre Benutzer weitergeben. Die Vertrauensrichtlinie dieser Rollen muss es Amazon Cognito erlauben, die Rolle für Ihren Benutzer zu übernehmen. Bei Attributen für die Zugriffskontrolle müssen Sie Amazon Cognito außerdem erlauben, Prinzipal-Tags auf die temporäre Sitzung Ihres Benutzers anzuwenden. Erteilen Sie mit der Aktion AssumeRoleWithWebIdentity die Berechtigung, die Rolle zu übernehmen. Erteilen Sie mit der „nur mit Berechtigung“-Aktion `sts:TagSession` die Berechtigung, die Sitzungen der Benutzer mit Tags zu versehen. Weitere Informationen finden Sie unter Übergeben von Sitzungs-Tags in AWS Security Token Service im AWS Identity and Access Management-Benutzerhandbuch. Ein Beispiel

einer Vertrauensrichtlinie, die dem Service-Prinzipal `cognito-identity.amazonaws.com` von Amazon Cognito `sts:AssumeRoleWithWebIdentity` und `sts:TagSession`-Berechtigungen gewährt, finden Sie unter Verwenden von Attributen für die Zugriffskontrollrichtlinie (Beispiel).

So konfigurieren Sie Attribute für die Zugriffskontrolle in der Konsole

1. Melden Sie sich bei der Amazon-Cognito-Konsole an und wählen Sie Identitätspools aus. Wählen Sie einen Identitätspool aus.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Suchen Sie Identitätsanbieter. Wählen Sie den Identitätsanbieter aus, den Sie bearbeiten möchten. Wenn Sie einen neuen IdP hinzufügen möchten, wählen Sie Identitätsanbieter hinzufügen aus.
4. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, wählen Sie unter Attribute für die Zugriffskontrolle die Option Bearbeiten.
   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.
   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.
   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.
5. Wählen Sie Save Changes (Änderungen speichern) aus.

Verwenden von Attributen für die Zugriffskontrollrichtlinie (Beispiel)

Stellen Sie sich ein Szenario vor, in dem ein Mitarbeiter der Rechtsabteilung eines Unternehmens alle Dateien in Buckets auflisten muss, die zu seiner Abteilung gehören und mit ihrer Sicherheitsstufe klassifiziert sind. Angenommen, das Token, das dieser Mitarbeiter vom Identitätsanbieter erhält, enthält die folgenden Ansprüche.

Ansprüche

```
{ .
  .
```

```
            "sub" : "57e7b692-4f66-480d-98b8-45a6729b4c88",
            "department" : "legal",
            "clearance" : "confidential",
             .
             .
            }
```

Diese Attribute können Tags zugeordnet und in IAM-Berechtigungsrichtlinien als Prinzipal-Tags referenziert werden. Sie können den Zugriff jetzt verwalten, indem Sie das Benutzerprofil beim Identitätsanbieter ändern. Alternativ können Sie Attribute auf der Ressourcenseite ändern, indem Sie Namen oder Tags verwenden, ohne die Richtlinie selbst zu ändern.

Die folgende Berechtigungsrichtlinie bewirkt zweierlei:

- Ermöglicht den Listenzugriff auf alle S3-Buckets, die mit einem Präfix enden, das dem Abteilungsnamen des Benutzers entspricht.
- Ermöglicht den Lesezugriff auf Dateien in diesen Buckets, solange das Freigabe-Tag der Datei mit dem Freigabe-Attribut des Benutzers übereinstimmt.

Berechtigungsrichtlinie

```
{
 "Version": "2012-10-17",
 "Statement": [
     {
         "Effect": "Allow",
         "Action": "s3:List*",
         "Resource": "arn:aws:s3:::*-${aws:PrincipalTag/department}"
     },
     {
         "Effect": "Allow",
         "Action": "s3:GetObject*",
         "Resource": "arn:aws:s3:::*-${aws:PrincipalTag/department}/*",
         "Condition": {
             "StringEquals": {
                 "s3:ExistingObjectTag/clearance": "${aws:PrincipalTag/clearance}"
             }
         }
     }
```

```
    ]
}
```

Die Vertrauensrichtlinie bestimmt, wer diese Rolle übernehmen kann.
Die Vertrauensbeziehungsrichtlinie ermöglicht die Verwendung von `sts:AssumeRoleWithWebIdentity` und `sts:TagSession`, um den Zugriff zu ermöglichen. Fügt Bedingungen hinzu, um die Richtlinie auf den von Ihnen erstellten Identitätspool zu beschränken, und stellt sicher, dass sie für eine authentifizierte Rolle bestimmt ist.

Vertrauensrichtlinie

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "IDENTITY-POOL-ID"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
      }
    }
  ]
}
```

Attribute für die Zugriffskontrolle deaktivieren (Konsole)

Gehen Sie folgendermaßen vor, um Attribute für die Zugriffskontrolle zu deaktivieren.

So deaktivieren Sie Attribute für die Zugriffskontrolle in der Konsole

1. Melden Sie sich bei der Amazon-Cognito-Konsole an und wählen Sie Identätspools aus. Wählen Sie einen Identitätspool aus.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Suchen Sie Identitätsanbieter. Wählen Sie den Identitätsanbieter aus, den Sie bearbeiten möchten.
4. Wählen Sie unter Attribute für die Zugriffskontrolle die Option Bearbeiten aus.
5. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv.
6. Wählen Sie Save Changes (Änderungen speichern) aus.

Standard-Anbietermappings

Die folgende Tabelle enthält die Standardzuweisungsinformationen für die Authentifizierungsanbieter, die Amazon Cognito unterstützt.

Anbieter	Token-Typ	Prinzipal-Tagwerte	Beispiel
Amazon-Cognito-Benutzerpool	ID-Token	aud (Client-ID) und sub (Benutzer-ID)	"6jk8ltokc7ac9es6jrtg9q572f", "57e7b692-4f66-480d-98b8-45a6729b4c88"
Facebook	Zugriffstoken	aud (app_id), sub (user_id)	„492844718097981“, „112177216992379“
Google	ID-Token	aud (Client-ID) und sub (Benutzer-ID)	„620493171733-eebk7c0hcp5lj3e1tlqp1gntt3k0rncv.apps.googleusercontent.com“, „109220063452404746097“
SAML	Assertionen	„http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier“,	"auth0|5e28d196f8f55a0eaaa95de3", "user123@gmail.com"

Anbieter	Token-Typ	Prinzipal-Tagwerte	Beispiel
		„http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name“	
Apple	ID-Token	aud (Client-ID) und sub (Benutzer-ID)	„com.amazonaws.ec2-54-80-172-243.compute-1.client“, „001968.a6ca34e9c1e742458a26cf8005854be9.0733“
Amazon	Zugriffstoken	aud (Client-ID auf Amzn Dev Ac), user_id (Benutzer-ID)	„amzn1.application-oa2-client.9d70d9382d3446108aaee3dd763a0fa6“, „amzn1.account.aghnifjqmfsbg3g6xcpvb35orqaa“
Standard-OIDC-Anbieter	ID und Zugriffstoken	aud (als Client-ID) und sub (als Benutzer-ID)	„620493171733-eebk7c0hcp5lj3e1tlqp1gntt3k0rncv.apps.googleusercontent.com“, „109220063452404746097“
Twitter	Zugriffstoken	aud (App-ID; App-Geheimnis), sub (Benutzer-ID)	"DfwifTtKEX1FiIBRnOTIR0CFK; Xgj5xb8xIrIVCPjXgLIdkW7fXmwcJJrFvnoK9gwZkLexo1y5z1", "1269003884292222976"
DevAuth	Map	Nicht zutreffend	„tag1“, „tag2“

> **Note**
>
> Die Option für Standard-Attributmappings wird automatisch für den Tag-Schlüssel für Prinzipal und Attributnamen ausgefüllt. Standardmappings können nicht geändert werden.

Verwenden der rollenbasierten Zugriffskontrolle

Amazon Cognito Cognito-Identitätspools weisen Ihren authentifizierten Benutzern eine Reihe temporärer Anmeldeinformationen mit eingeschränkten Rechten für den Zugriff auf Ihre Ressourcen zu. AWS Die Berechtigung aller Benutzer werden durch die von Ihnen erstellen IAM-Rollen gesteuert. Sie können Regeln definieren, um die Rollen der einzelnen Benutzer basierend auf den Ansprüchen im ID-Token eines Benutzers auszuwählen. Sie sind in der Lage, eine Standardrolle für authentifizierte Benutzer zu erstellen. Darüber hinaus können Sie eine separate IAM-Rolle mit beschränkten Berechtigungen für Gastbenutzer anlegen, die nicht authentifiziert wurden.

Erstellen von Rollen für das Rollen-Mapping

Es ist wichtig, die entsprechende Vertrauensrichtlinie für jede Rolle hinzufügen, so dass sie von Amazon Cognito nur für authentifizierte Benutzer in Ihrem Identitäten-Pool übernommen wird. Hier finden Sie ein Beispiel für eine Vertrauensrichtlinie:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-corner-
cafe-123456790ab"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
```

```
      }
    }
  ]
}
```

Mit dieser Richtlinie können verbundene Benutzer aus `cognito-identity.amazonaws.com` (Aussteller des OpenID Connect-Tokens) diese Rolle übernehmen. Darüber hinaus schränkt die Richtlinie den `aud`-Anspruch des Tokens (in diesem Fall die Identitäten-Pool-ID) dahingehend ein, dass er mit dem Identitäten-Pool übereinstimmen muss. Schließlich gibt die Richtlinie an, dass eines der Array-Mitglieder des mehrwertigen `amr`-Anspruchs des Tokens, das von der Amazon Cognito `GetOpenIdToken`-API-Aktion ausgegeben wird, den Wert `authenticated` hat.

Gewähren der Berechtigung zum Übergeben einer Rolle

Um einem Benutzer die Festlegung von Rollen mit Berechtigungen zu ermöglichen, die über die vorhandenen Berechtigungen des Benutzers für einen Identitätspool hinausgehen, gewähren Sie dem betreffenden Benutzer die Berechtigung `iam:PassRole`, damit die Rolle an die API `set-identity-pool-roles` übergeben werden kann. Wenn der Benutzer beispielsweise nicht in Amazon S3 schreiben kann, aber die IAM-Rolle, die der Benutzer im Identitätspool festlegt, Amazon S3 Schreibberechtigung erteilt, kann der Benutzer diese Rolle nur festlegen, wenn die `iam:PassRole`-Berechtigung für die Rolle erteilt wurde. Die folgende Beispielrichtlinie zeigt, wie Sie die Berechtigung `iam:PassRole` erteilen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:role/myS3WriteAccessRole"
            ]
        }
    ]
}
```

In diesem Richtlinienbeispiel wird die Berechtigung `iam:PassRole` für die Rolle `myS3WriteAccessRole` gewährt. Die Rolle wird mit dem Amazon-Ressourcennamen (ARN) der Rolle angegeben. Sie müssen diese Richtlinie auch an Ihren Benutzer anfügen. Weitere Informationen finden Sie unter Arbeiten mit verwalteten Richtlinien.

> **Note**
>
> Lambda-Funktionen verwenden eine ressourcenbasierte Richtlinie, wobei die Richtlinie direkt an die Lambda-Funktion selbst angefügt wird. Wenn Sie eine Regel erstellen, die eine Lambda-Funktion aufruft, übergeben Sie keine Rolle, sodass der Benutzer die `iam:PassRole`-Berechtigung zum Erstellen der Regel nicht benötigt. Weitere Informationen zur Lambda-Funktionsautorisierung finden Sie unter Verwalten von Berechtigungen: Verwenden einer Lambda-Funktionsrichtlinie.

Zuweisen von Rollen zu Benutzern mit Token

Für Benutzer, die sich über Amazon-Cognito-Benutzerpools anmelden, können Rollen in dem vom Benutzerpool zugewiesenen ID-Token übergeben werden. Die Rollen werden in den folgenden Ansprüchen im ID-Token angezeigt:

- Der Anspruch `cognito:preferred_role` ist der Rollen-ARN.
- Der Anspruch `cognito:roles` ist eine durch Komma getrennte Zeichenfolge mit einem Satz zulässiger Rollen-ARNs.

Die Ansprüche werden wie folgt festgelegt:

- Der Anspruch `cognito:preferred_role` wird auf die Rolle aus der Gruppe mit dem besten (niedrigsten) `Precedence`-Wert gesetzt. Wenn es nur eine zulässige Rolle gibt, wird `cognito:preferred_role` auf diese Rolle gesetzt. Gibt es mehrere Rollen und hat keine von ihnen den höchsten Vorrang, wird der Anspruch nicht festgelegt.
- Der Anspruch `cognito:roles` wird festgelegt, wenn mindestens eine Rolle vorhanden ist.

Bei der Verwendung von Tokens zum Zuweisen von Rollen wählen Amazon-Cognito-Identitäten-Pools (Verbundidentitäten) die Rolle wie folgt aus, falls es mehrere Rollen gibt, die einem Benutzer zugewiesen werden können:

- Verwenden Sie den GetCredentialsForIdentity`CustomRoleArn`Parameter, wenn er festgelegt ist und er einer Rolle im Anspruch entspricht. `cognito:roles` Entspricht dieser Parameter keiner Rolle in `cognito:roles`, verweigern Sie den Zugriff.
- Wenn der Anspruch `cognito:preferred_role` festgelegt ist, verwenden Sie ihn.
- Wenn der `cognito:preferred_role` Anspruch nicht festgelegt ist, der `cognito:roles` Anspruch festgelegt und nicht im Aufruf von angegeben `CustomRoleArn` ist`GetCredentialsForIdentity`, dann wird die Einstellung für die Rollenauflösung in der Konsole oder im `AmbiguousRoleResolution` Feld (im `RoleMappings` SetIdentityPoolRolesAPI-Parameter) verwendet, um die zuzuweisende Rolle zu bestimmen.

Verwendung des regelbasierten Mappings, um Benutzern Rollen zuzuweisen

Mit Regeln können Sie IAM-Rollen Ansprüche aus einem Identitätsanbieter-Token zuordnen.

Jede Regel gibt einen Token-Anspruch (beispielsweise ein Benutzerattribut im ID-Token aus einem Amazon-Cognito-Benutzerpool), den Übereinstimmungstyp, einen Wert und eine IAM-Rolle an. Der Übereinstimmungstyp kann `Equals`, `NotEqual`, `StartsWith` oder `Contains` sein. Wenn ein Benutzer einen übereinstimmen Wert für den Anspruch hat, kann er die jeweilige Rolle annehmen, wenn er die Anmeldeinformationen erhält. Sie können beispielsweise eine Regel erstellen, die eine bestimmte IAM-Rolle für Benutzer mit dem benutzerdefinierten `custom:dept`-Attributwert `Sales` zuweist.

> **Note**
>
> In den Regeleinstellungen erfordern benutzerdefinierte Attribute das Präfix `custom:`, damit sie von Standardattributen unterschieden werden können.

Regeln werden in einer bestimmten Reihenfolge ausgewertet und die IAM-Rolle für die erste übereinstimmende Regel wird verwendet, es sei denn, `CustomRoleArn` wird angegeben und setzt die Reihenfolge außer Kraft. Weitere Informationen zu Benutzerattributen in Amazon-Cognito-Benutzerpools finden Sie unter Attribute für den Benutzerpool.

Sie können mehrere Regeln für einen Authentifizierungsanbieter in der Identitäten-Pool- (Verbundidentitäten-) Konsole festlegen. Regeln werden in einer bestimmten Reihenfolge

angewendet. Diese können Sie durch Ziehen der Regeln Bestellung ändern. Die erste übereinstimmende Regel hat Vorrang. Wenn der Übereinstimmungstyp `NotEqual` ist und der Anspruch nicht existiert, wird die Regel nicht ausgewertet. Wenn keine Regeln übereinstimmen, wird die Einstellung Rollenauflösung entweder auf Authentifizierte Standard-Rolle verwenden oder Anforderung ablehnen angewendet.

In der API und CLI können Sie die Rolle angeben, die zugewiesen werden soll, wenn keine Regeln im `AmbiguousRoleResolution` Feld des RoleMappingTyps übereinstimmen, der im `RoleMappings` Parameter der SetIdentityPoolRolesAPI angegeben ist.

Sie können ein regelbasiertes Mapping für OpenID Connect (OIDC) und SAML-Identitätsanbieter in der AWS CLI oder API mit dem Feld des Typs einrichten. `RulesConfiguration` RoleMapping Sie können dieses Feld im Parameter der API angeben. `RoleMappings` SetIdentityPoolRoles AWS Management Console Derzeit können Sie keine Regeln für OIDC- oder SAML-Anbieter hinzufügen.

Mit dem folgenden AWS CLI Befehl wird beispielsweise eine Regel hinzugefügt, die die Rolle Benutzern `arn:aws:iam::123456789012:role/Sacramento_team_S3_admin` an Ihrem Standort in Sacramento zuweist, die von OIDC IdP authentifiziert wurden: `arn:aws:iam::123456789012:oidc-provider/myOIDCIdP`

```
aws cognito-identity set-identity-pool-roles --region us-east-1 --cli-input-json
 file://role-mapping.json
```

Inhalt von **`role-mapping.json`**:

```
{
    "IdentityPoolId": "us-east-1:12345678-corner-cafe-123456790ab",
    "Roles": {
        "authenticated": "arn:aws:iam::123456789012:role/myS3WriteAccessRole",
        "unauthenticated": "arn:aws:iam::123456789012:role/myS3ReadAccessRole"
    },
    "RoleMappings": {
        "arn:aws:iam::123456789012:oidc-provider/myOIDCIdP": {
            "Type": "Rules",
            "AmbiguousRoleResolution": "AuthenticatedRole",
            "RulesConfiguration": {
                "Rules": [
                    {
                        "Claim": "locale",
                        "MatchType": "Equals",
```

```
                        "Value": "Sacramento",
                        "RoleARN": "arn:aws:iam::123456789012:role/
Sacramento_team_S3_admin"
                    }
                ]
            }
        }
    }
}
```

Für jeden Benutzerpool oder anderen Authentifizierungsanbieter, den Sie für einen Identitätspool konfigurieren, können Sie bis zu 25 Regeln erstellen. Diese Grenze ist nicht einstellbar. Weitere Informationen finden Sie unter Kontingente in Amazon Cognito.

Token-Ansprüche zur Verwendung in regelbasiertem Mapping

Amazon Cognito

Ein Amazon-Cognito-ID-Token wird als JSON-Web-Token (JWT) dargestellt. Das Token enthält Ansprüche bezüglich der Identität des authentifizierten Benutzers, beispielsweise `name`, `family_name` und `phone_number`. Weitere Informationen zu Standardansprüchen finden Sie in der OpenID Connect-Spezifikation. Neben den Standardansprüchen sind die folgenden zusätzlichen Ansprüche spezifisch für Amazon Cognito.

- `cognito:groups`
- `cognito:roles`
- `cognito:preferred_role`

Amazon

Die folgenden Ansprüche können zusammen mit ihren möglichen Werte mit Login with Amazon verwendet werden:

- `iss`: www.amazon.com
- `aud`: App-ID
- `sub`: `sub` aus dem Login with Amazon-Token

Facebook

Die folgenden Ansprüche können zusammen mit ihren möglichen Werte mit Facebook verwendet werden:

- `iss`: graph.facebook.com
- `aud`: App-ID
- `sub`: `sub` aus dem Facebook-Token

Google

Alle Ansprüche im OpenID-Token stehen für das regelbasierte Mapping zur Verfügung: Alle Ansprüche im OpenID-Token stehen für die regelbasierte Zuweisung zur Verfügung. Weitere Informationen zu den Ansprüchen, die über das Google-Token verfügbar sind, finden Sie auf der Google-Website zu OpenID-Connect.

Apple

Ein Apple-Token enthält Standardansprüche aus der OpenID Connect-Spezifikation. Weitere Informationen zu den Ansprüchen, die über das Apple-Token verfügbar sind, erfahren Sie in der Apple-Dokumentation unterAuthentifizieren von Benutzern mit „Mit Apple anmelden“ . Token von Apple enthalten nicht immer `email`.

OpenID

Alle Ansprüche im OpenID-Token stehen für das regelbasierte Mapping zur Verfügung. Weitere Informationen zu Standardansprüchen finden Sie in der OpenID Connect-Spezifikation. Weitere Informationen zu zusätzlichen verfügbaren Ansprüchen finden Sie in der Dokumentation des OpenID-Anbieters.

SAML

Ansprüche werden von der empfangenen SAML-Assertion analysiert. Alle in der SAML-Assertion verfügbaren Ansprüche können im regelbasierten Mapping genutzt werden.

Bewährte Methoden für rollenbasierte Zugriffskontrolle

> **Important**
>
> Wenn ein Anspruch, den Sie einer Rolle zuordnen, durch den Endbenutzer geändert werden kann, kann jeder beliebige Endbenutzer Ihr Rolle übernehmen und die Richtlinie

entsprechend festlegen. Ordnen Sie nur Ansprüche zu, die vom Endbenutzer nicht direkt für Rollen mit erhöhten Berechtigungen festgelegt werden können. In einem Amazon-Cognito-Benutzerpool können Sie Lese- und Schreibberechtigungen pro App für jedes Benutzerattribut festlegen.

Important

Wenn Sie Rollen für Gruppen in einem Amazon-Cognito-Benutzerpool festlegen, werden diese Rollen durch das ID-Token des Benutzers übergeben. Um diese Rollen zu verwenden, müssen Sie auch Choose role from token (Rolle aus Token auswählen) für die Auswahl authentifizierter Rollen für den Identitäten-Pool festlegen.
Sie können die Einstellung für die Rollenauflösung in der Konsole und den `RoleMappings` SetIdentityPoolRolesAPI-Parameter verwenden, um das Standardverhalten anzugeben, wenn die richtige Rolle nicht anhand des Tokens bestimmt werden kann.

Abrufen von Anmeldeinformationen

Sie können Amazon Cognito verwenden, um temporäre Anmeldeinformationen mit eingeschränkten Rechten für Ihre Anwendung bereitzustellen, sodass Ihre Benutzer auf Ressourcen zugreifen können. AWS In diesem Abschnitt wird beschrieben, wie Sie Anmeldeinformationen erhalten und eine Amazon-Cognito-Identität aus Ihrem Identitäts-Pool abrufen.

Amazon Cognito unterstützt sowohl authentifizierte als auch nicht authentifizierte Identitäten. Für nicht authentifizierte Benutzer wird die Identität nicht verifiziert, sodass diese Rolle für Gastbenutzer Ihrer Anwendung geeignet ist, oder in Fällen, in denen es keine Rolle spielt, ob Benutzer ihre Identität verifizieren lassen. Authentifizierte Benutzer melden sich bei Ihrer Anwendung über einen Drittanbieter oder einen Benutzerpool an, der ihre Identität überprüft. Vergewissern Sie sich, dass Sie die Berechtigungen der Ressourcen entsprechend anpassen, damit Sie keinen Zugriff von nicht authentifizierten Benutzern darauf gewähren.

Amazon-Cognito-Identitäten sind keine Anmeldeinformationen. Sie werden mithilfe der Unterstützung von Web Identity Federation in der AWS Security Token Service () gegen Anmeldeinformationen ausgetauscht.AWS STS Die empfohlene Methode zum Erhalt von AWS -Anmeldeinformationen für Ihre App-Benutzer ist die Verwendung von `AWS.CognitoIdentityCredentials`. Die Identität im Anmeldeinformationsobjekt wird dann gegen Anmeldeinformationen mit ausgetauscht AWS STS.

> **Note**
>
> Wenn Sie Ihren Identitätspool vor Februar 2015 erstellt haben, müssen Sie Ihre Rollen erneut mit Ihrem Identitätspool verknüpfen, um den `AWS.CognitoIdentityCredentials`-Konstruktor ohne die Rollen als Parameter verwenden zu können. Öffnen Sie dazu die Amazon-Cognito-Konsole und wählen Sie Identitätspools verwalten, Ihren Identitätspool und danach Identitätspool bearbeiten aus, legen Sie Ihre authentifizierten und nicht authentifizierten Rollen fest und speichern Sie die Änderungen.

Anbieter von Anmeldeinformationen für Web-Identitäten sind Teil der standardmäßigen Anbieterkette für Anmeldeinformationen in AWS -SDKs. Um Ihr Identitätspool-Token in einer lokalen `config` Datei für ein AWS SDK oder das festzulegen AWS CLI, fügen Sie einen `web_identity_token_file` Profileintrag hinzu. Weitere Informationen finden Sie im Referenzhandbuch zu AWS SDKs und Tools unter Übernehmen Sie die Rolle des Anbieters von Anmeldeinformationen.

Weitere Informationen zum Eingeben von Web-Identitäts-Anmeldeinformationen in Ihrem SDK finden Sie im SDK-Entwicklerhandbuch. Die besten Ergebnisse erzielen Sie, wenn Sie Ihr Projekt mit der integrierten Identitätspool-Integration beginnen. AWS Amplify

AWS SDK-Ressourcen zum Abrufen und Einrichten von Anmeldeinformationen mit Identitätspools

- Identity Pool Federation (Android) im Amplify Dev Center
- Identity Pool Federation (iOS) im Amplify Dev Center
- Verwenden von Amazon Cognito Identity zur Benutzerauthentifizierung im Entwicklerhandbuch AWS SDK for JavaScript
- Anbieter von Amazon Cognito Cognito-Anmeldeinformationen im AWS SDK for .NET Entwicklerhandbuch
- Geben Sie die Anmeldeinformationen im Entwicklerhandbuch programmgesteuert an AWS SDK for Go
- Geben Sie temporäre Anmeldeinformationen im Code im Entwicklerhandbuch ein AWS SDK for Java 2.x
- assumeRoleWithWebIdentityCredentialProviderAnbieter im AWS SDK for PHP Entwicklerhandbuch
- Übernehmen einer Rolle mit einem Web-Identitätsanbieter in der AWS SDK for Python (Boto3) - Dokumentation

- Geben Sie Ihre Anmeldeinformationen und die Standardregion im AWS SDK for Rust Entwicklerhandbuch an

Die folgenden Abschnitte enthalten Beispielcode in einigen älteren AWS SDKs.

Android

Sie können Amazon Cognito verwenden, um temporäre Anmeldeinformationen mit eingeschränkten Rechten für Ihre Anwendung bereitzustellen, sodass Ihre Benutzer auf Ressourcen zugreifen können. AWS Amazon Cognito unterstützt sowohl authentifizierte als auch nicht authentifizierte Identitäten. Gehen Sie wie folgt vor, um AWS Anmeldeinformationen für Ihre App bereitzustellen.

Um einen Amazon Cognito Cognito-Identitätspool in einer Android-App zu verwenden, richten AWS Amplify Sie ihn ein. Weitere Informationen finden Sie unter Authentifizierung im Amplify Dev Center.

Amazon-Cognito-Identität abrufen

Wenn Sie nicht authentifizierte Benutzer zulassen, können Sie sofort eine eindeutige Amazon-Cognito-ID (Identitäts-ID) für den Endbenutzer abrufen. Wenn Sie Benutzern authentifizieren, können Sie die Identitäts-ID abrufen, nachdem Sie die Anmelde-Token im Anmeldeinformationsanbieter festgelegt haben:

```
String identityId = credentialsProvider.getIdentityId();
Log.d("LogTag", "my ID is " + identityId);
```

> **Note**
>
> Rufen Sie `getIdentityId()`, `refresh()` oder `getCredentials()` nicht im Haupt-Thread Ihrer Anwendung auf. Ab Android 3.0 (API Level 11) schlägt Ihre App automatisch fehl und gibt einen Fehler aus, NetworkOnMainThreadExceptionwenn Sie Netzwerk-I/O im Hauptanwendungs-Thread ausführen. Sie müssen den Code mit `AsyncTask` in einen Hintergrund-Thread verschrieben. Weitere Informationen finden Sie in der Android-Dokumentation. Sie können zum Abrufen einer ID auch `getCachedIdentityId()` aufrufen, aber nur, wenn bereits eine ID lokal zwischengespeichert ist. Andernfalls gibt die Methode null zurück.

iOS – Objective-C

Sie können Amazon Cognito verwenden, um temporäre Anmeldeinformationen mit eingeschränkten Rechten für Ihre Anwendung bereitzustellen, sodass Ihre Benutzer auf Ressourcen zugreifen können. AWS Amazon-Cognito-Identitäten-Pools unterstützen authentifizierte und nicht authentifizierte Identitäten. Gehen Sie wie folgt vor, um AWS Anmeldeinformationen für Ihre App bereitzustellen.

Um einen Amazon Cognito Cognito-Identitätspool in einer iOS-App zu verwenden, richten AWS Amplify Sie ihn ein. Weitere Informationen finden Sie unter Swift-Authentifizierung und Flutter-Authentifizierung im Amplify Dev Center.

Amazon-Cognito-Identität abrufen

Wenn Sie nicht authentifizierte Benutzer zulassen, können Sie sofort eine eindeutige Amazon-Cognito-ID (Identitäts-ID) für Ihren Endbenutzer abrufen. Sofern Sie Benutzer authentifizieren, können Sie dies nach dem Festlegen des Anmelde-Tokens in den Anmeldeinformationsanbieter tun:

```
// Retrieve your Amazon Cognito ID
[[credentialsProvider getIdentityId] continueWithBlock:^id(AWSTask *task) {
    if (task.error) {
        NSLog(@"Error: %@", task.error);
    }
    else {
        // the task result will contain the identity id
        NSString *cognitoId = task.result;
    }
    return nil;
}];
```

> **Note**
>
> `getIdentityId` ist ein asynchroner Aufruf. Wenn für Ihren Anbieter bereits eine Identitäts-ID festgelegt ist, können Sie `credentialsProvider.identityId` aufrufen, um diese lokal zwischengespeicherte Identität abzurufen. Ist für den Anbieter jedoch keine Identitäts-ID festgelegt, gibt der Aufruf von `credentialsProvider.identityId` `nil` zurück. Weitere Informationen finden Sie in der Referenz zu Amplify iOS SDK.

iOS – Swift

Sie können Amazon Cognito verwenden, um temporäre Anmeldeinformationen mit eingeschränkten Rechten für Ihre Anwendung bereitzustellen, sodass Ihre Benutzer auf Ressourcen zugreifen können. AWS Amazon Cognito unterstützt sowohl authentifizierte als auch nicht authentifizierte Identitäten. Gehen Sie wie folgt vor, um AWS Anmeldeinformationen für Ihre App bereitzustellen.

Um einen Amazon Cognito Cognito-Identitätspool in einer iOS-App zu verwenden, richten AWS Amplify Sie ihn ein. Weitere Informationen finden Sie unter Swift-Authentifizierung im Amplify Dev Center.

Amazon-Cognito-Identität abrufen

Wenn Sie nicht authentifizierte Benutzer zulassen, können Sie sofort eine eindeutige Amazon-Cognito-ID (Identitäts-ID) für Ihren Endbenutzer abrufen. Sofern Sie Benutzer authentifizieren, können Sie dies nach dem Festlegen des Anmelde-Tokens in den Anmeldeinformationsanbieter tun:

```
// Retrieve your Amazon Cognito ID
credentialsProvider.getIdentityId().continueWith(block: { (task) -> AnyObject? in
    if (task.error != nil) {
        print("Error: " + task.error!.localizedDescription)
    }
    else {
        // the task result will contain the identity id
        let cognitoId = task.result!
        print("Cognito id: \(cognitoId)")
    }
    return task;
})
```

> **Note**
>
> `getIdentityId` ist ein asynchroner Aufruf. Wenn für Ihren Anbieter bereits eine Identitäts-ID festgelegt ist, können Sie `credentialsProvider.identityId` aufrufen, um diese lokal zwischengespeicherte Identität abzurufen. Ist für den Anbieter jedoch keine Identitäts-ID festgelegt, gibt der Aufruf von `credentialsProvider.identityId` `nil` zurück. Weitere Informationen finden Sie in der Referenz zu Amplify iOS SDK.

JavaScript

Wenn Sie noch keinen Identitätspool erstellt haben, erstellen Sie einen in der Amazon-Cognito-Konsole, bevor Sie `AWS.CognitoIdentityCredentials` verwenden.

Nachdem Sie einen Identitäten-Pool bei Ihren Identitätsanbietern konfiguriert haben, können Sie mit `AWS.CognitoIdentityCredentials` Benutzer authentifizieren. Um die Anmeldeinformationen für Ihre Anwendung so zu konfigurieren, dass Sie `AWS.CognitoIdentityCredentials` verwenden können, setzen Sie die `credentials`-Eigenschaft für `AWS.Config` oder Sie verwenden eine servicespezifische Konfiguration. Im folgenden Beispiel wird verwende `AWS.Config`:

```
// Set the region where your identity pool exists (us-east-1, eu-west-1)
AWS.config.region = 'us-east-1';

// Configure the credentials provider to use your identity pool
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'IDENTITY_POOL_ID',
    Logins: { // optional tokens, used for authenticated login
        'graph.facebook.com': 'FBTOKEN',
        'www.amazon.com': 'AMAZONTOKEN',
        'accounts.google.com': 'GOOGLETOKEN',
        'appleid.apple.com': 'APPLETOKEN'
    }
});

// Make the call to obtain credentials
AWS.config.credentials.get(function(){

    // Credentials will be available when this function is called.
    var accessKeyId = AWS.config.credentials.accessKeyId;
    var secretAccessKey = AWS.config.credentials.secretAccessKey;
    var sessionToken = AWS.config.credentials.sessionToken;

});
```

Die optionale `Logins`-Eigenschaft ist eine Abbildung der Namen des Identitätsanbieters auf die Identitäts-Token für diese Anbieter. Wie Sie den Token von Ihrem Identitätsanbieter erhalten, hängt davon ab, welchen Anbieter Sie verwenden. Ist beispielsweise Facebook einer Ihrer Identitätsanbieter, könnten sie die `FB.login`-Funktion aus dem Facebook SDK verwenden, um ein Identitätsanbieter-Token zu erhalten:

```
FB.login(function (response) {
    if (response.authResponse) { // logged in
        AWS.config.credentials = new AWS.CognitoIdentityCredentials({
          IdentityPoolId: 'us-east-1:1699ebc0-7900-4099-b910-2df94f52a030',
          Logins: {
            'graph.facebook.com': response.authResponse.accessToken
          }
        });

        console.log('You are now logged in.');
    } else {
        console.log('There was a problem logging you in.');
    }
});
```

Amazon-Cognito-Identität abrufen

Wenn Sie nicht authentifizierte Benutzer zulassen, können Sie sofort eine eindeutige Amazon-Cognito-ID (Identitäts-ID) für Ihren Endbenutzer abrufen. Sofern Sie Benutzer authentifizieren, können Sie dies nach dem Festlegen des Anmelde-Tokens in den Anmeldeinformationsanbieter tun:

```
var identityId = AWS.config.credentials.identityId;
```

Unity

Sie können Amazon Cognito verwenden, um temporäre Anmeldeinformationen mit eingeschränkten Rechten für Ihre Anwendung bereitzustellen, sodass Ihre Benutzer auf Ressourcen zugreifen können. AWS Amazon Cognito unterstützt sowohl authentifizierte als auch nicht authentifizierte Identitäten. Gehen Sie wie folgt vor, um AWS Anmeldeinformationen für Ihre App bereitzustellen.

Das AWS -SDK for Unity ist jetzt Teil von AWS SDK for .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im AWS SDK for .NET AWS SDK for .NET Developer Guide unter Amazon Cognito Credentials Provider. Oder im Amplify Dev Center finden Sie Optionen zum Erstellen einer App mit AWS Amplify.

Amazon-Cognito-Identität abrufen

Wenn Sie nicht authentifizierte Benutzer zulassen, können Sie sofort eine eindeutige Amazon-Cognito-ID (Identitäts-ID) für Ihren Endbenutzer abrufen. Sofern Sie Benutzer authentifizieren, können Sie dies nach dem Festlegen des Anmelde-Tokens in den Anmeldeinformationsanbieter tun:

```
credentials.GetIdentityIdAsync(delegate(AmazonCognitoIdentityResult<string> result) {
    if (result.Exception != null) {
        //Exception!
    }
    string identityId = result.Response;
});
```

Xamarin

Sie können Amazon Cognito verwenden, um temporäre Anmeldeinformationen mit eingeschränkten Rechten für Ihre Anwendung bereitzustellen, sodass Ihre Benutzer auf Ressourcen zugreifen können. AWS Amazon Cognito unterstützt sowohl authentifizierte als auch nicht authentifizierte Identitäten. Gehen Sie wie folgt vor, um AWS Anmeldeinformationen für Ihre App bereitzustellen.

Das AWS -SDK für Xamarin ist jetzt Teil von AWS SDK for .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im AWS SDK for .NET AWS SDK for .NET Developer Guide unter Amazon Cognito Credentials Provider. Oder im Amplify Dev Center finden Sie Optionen zum Erstellen einer App mit AWS Amplify.

> **Note**
>
> Hinweis: Wenn Sie den Identitätspool vor Februar 2015 erstellt haben, müssen Sie dem Identitätspool die Rollen neu zuweisen, um diesen Konstruktor ohne die Rollen als Parameter zu verwenden. Öffnen Sie dazu die Amazon-Cognito-Konsole und wählen Sie Identitätspools verwalten, Ihren Identitätspool und danach Identitätspool bearbeiten aus, legen Sie Ihre authentifizierten und nicht authentifizierten Rollen fest und speichern Sie die Änderungen.

Amazon-Cognito-Identität abrufen

Wenn Sie nicht authentifizierte Benutzer zulassen, können Sie sofort eine eindeutige Amazon-Cognito-ID (Identitäts-ID) für Ihren Endbenutzer abrufen. Sofern Sie Benutzer authentifizieren, können Sie dies nach dem Festlegen des Anmelde-Tokens in den Anmeldeinformationsanbieter tun:

```
var identityId = await credentials.GetIdentityIdAsync();
```

Zugreifen auf Dienste AWS

Nachdem Sie Ihren Amazon Cognito Cognito-Anmeldeinformationsanbieter konfiguriert und AWS Anmeldeinformationen abgerufen haben, können Sie einen AWS-Service Client erstellen.

AWS SDK-Ressourcen für die Erstellung eines Clients

- AWS Die Client-Konfiguration im AWS SDK for C++ Developer Guide
- Verwenden von AWS SDK for Go V2 mit AWS-Services im AWS SDK for Go Entwicklerhandbuch
- Konfiguration von HTTP-Clients im AWS SDK for Java 2.x Entwicklerhandbuch
- Serviceobjekte im AWS SDK for JavaScript Developer Guide erstellen und aufrufen
- Clients in der AWS SDK for Python (Boto3) Dokumentation erstellen
- Einen Service-Client im AWS SDK for Rust Developer Guide erstellen
- Verwendung von Clients im AWS SDK for Swift Developer Guide

Der folgende Ausschnitt initialisiert beispielsweise einen Amazon-DynamoDB-Client:

Android

Um einen Amazon Cognito Cognito-Identitätspool in einer Android-App zu verwenden, richten AWS Amplify Sie ihn ein. Weitere Informationen finden Sie unter Authentifizierung im Amplify Dev Center.

```
// Create a service client with the provider
AmazonDynamoDB client = new AmazonDynamoDBClient(credentialsProvider);
```

Der Anmeldeinformationsanbieter kommuniziert mit Amazon Cognito und ruft sowohl die eindeutige Kennung für authentifizierte und nicht authentifizierte Benutzer als auch temporäre Anmeldeinformationen mit eingeschränkten Rechten für das Mobile SDK ab. AWS AWS Die abgerufenen Anmeldeinformationen gelten für eine Stunde. Der Anbieter wird aktualisiert, wenn sie ablaufen.

iOS – Objective-C

Um einen Amazon Cognito Cognito-Identitätspool in einer iOS-App zu verwenden, richten AWS Amplify Sie ihn ein. Weitere Informationen finden Sie unter Swift-Authentifizierung und Flutter-Authentifizierung im Amplify Dev Center.

```
// create a configuration that uses the provider
```

```
AWSServiceConfiguration *configuration = [AWSServiceConfiguration
 configurationWithRegion:AWSRegionUSEast1 provider:credentialsProvider];
// get a client with the default service configuration
AWSDynamoDB *dynamoDB = [AWSDynamoDB defaultDynamoDB];
```

Der Anmeldeinformationsanbieter kommuniziert mit Amazon Cognito und ruft sowohl die eindeutige Kennung für authentifizierte und nicht authentifizierte Benutzer als auch temporäre Anmeldeinformationen mit eingeschränkten Rechten für das Mobile SDK ab. AWS AWS Die abgerufenen Anmeldeinformationen gelten für eine Stunde. Der Anbieter wird aktualisiert, wenn sie ablaufen.

iOS – Swift

Um einen Amazon Cognito Cognito-Identitätspool in einer iOS-App zu verwenden, richten AWS Amplify Sie ihn ein. Weitere Informationen finden Sie unter Swift-Authentifizierung im Amplify Dev Center.

```
// get a client with the default service configuration
let dynamoDB = AWSDynamoDB.default()

// get a client with a custom configuration
AWSDynamoDB.register(with: configuration!, forKey: "USWest2DynamoDB");
let dynamoDBCustom = AWSDynamoDB(forKey: "USWest2DynamoDB")
```

Der Anmeldeinformationsanbieter kommuniziert mit Amazon Cognito und ruft sowohl die eindeutige Kennung für authentifizierte und nicht authentifizierte Benutzer als auch temporäre Anmeldeinformationen mit eingeschränkten Rechten für das Mobile SDK ab. AWS AWS Die abgerufenen Anmeldeinformationen gelten für eine Stunde. Der Anbieter wird aktualisiert, wenn sie ablaufen.

JavaScript

```
// Create a service client with the provider
var dynamodb = new AWS.DynamoDB({region: 'us-west-2'});
```

Der Anmeldeinformationsanbieter kommuniziert mit Amazon Cognito und ruft sowohl die eindeutige Kennung für authentifizierte und nicht authentifizierte Benutzer als auch temporäre Anmeldeinformationen mit eingeschränkten AWS Rechten für das Mobile SDK ab. AWS Die

abgerufenen Anmeldeinformationen gelten für eine Stunde. Der Anbieter wird aktualisiert, wenn sie ablaufen.

Unity

Das AWS -SDK for Unity ist jetzt Teil von AWS SDK for .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im AWS SDK for .NET AWS SDK for .NET Developer Guide unter Amazon Cognito Credentials Provider. Oder im Amplify Dev Center finden Sie Optionen zum Erstellen einer App mit AWS Amplify.

```
// create a service client that uses credentials provided by Cognito
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials, REGION);
```

Der Anmeldeinformationsanbieter kommuniziert mit Amazon Cognito und ruft sowohl die eindeutige Kennung für authentifizierte und nicht authentifizierte Benutzer als auch temporäre Anmeldeinformationen mit eingeschränkten AWS Rechten für das Mobile SDK ab. AWS Die abgerufenen Anmeldeinformationen gelten für eine Stunde. Der Anbieter wird aktualisiert, wenn sie ablaufen.

Xamarin

Das AWS -SDK für Xamarin ist jetzt Teil von AWS SDK for .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im AWS SDK for .NET AWS SDK for .NET Developer Guide unter Amazon Cognito Credentials Provider. Oder im Amplify Dev Center finden Sie Optionen zum Erstellen einer App mit AWS Amplify.

```
// create a service client that uses credentials provided by Cognito
var client = new AmazonDynamoDBClient(credentials, REGION)
```

Der Anmeldeinformationsanbieter kommuniziert mit Amazon Cognito und ruft sowohl die eindeutige Kennung für authentifizierte und nicht authentifizierte Benutzer als auch temporäre Anmeldeinformationen mit eingeschränkten AWS Rechten für das Mobile SDK ab. AWS Die abgerufenen Anmeldeinformationen gelten für eine Stunde. Der Anbieter wird aktualisiert, wenn sie ablaufen.

Externe Identitätsanbieter von Identitäten-Pools

Mit der `logins`-Eigenschaft können Sie Anmeldeinformationen einrichten, die Sie von einem Identitätsanbieter (IdP) erhalten haben. Darüber hinaus können Sie einen Identitätspool mehreren

IdPs zuordnen. Sie können beispielsweise sowohl das Facebook- als auch das Google-Token in der `logins`-Eigenschaft festlegen, sodass die eindeutige Amazon-Cognito-Identität beiden IdP-Anmeldungen zugeordnet wird. Der Benutzer kann sich bei beiden Konten authentifizieren, Amazon Cognito gibt jedoch dieselbe Benutzer-ID zurück.

Die folgenden Anweisungen führen Sie durch IdPs die Authentifizierung mit den von Amazon Cognito unterstützten Identitätspools.

Themen

Facebook als Identitätspools einrichten (IdP)

Dank der Integration von Amazon-Cognito-Identitätspools mit Facebook ist die verbundene Authentifizierung für Ihre Mobil-Anwendungsbenutzer möglich. In diesem Abschnitt wird erklärt, wie Sie Ihre Anwendung bei Facebook als IdP registrieren und einrichten.

Einrichten von Facebook

Registrieren Sie Ihre Anwendung bei Facebook, bevor Sie Facebook-Benutzer authentifizieren und mit Facebook-APIs interagieren.

Das Facebook Developer-Portal führt Sie durch die Einrichtung Ihrer Anwendung. Führen Sie dieses Verfahren durch, bevor Sie Facebook in Ihren Amazon-Cognito-Identitäten-Pool integrieren:

Einrichten von Facebook

1. Melden Sie beim Facebook Developer-Portal mit Ihren Facebook-Anmeldeinformationen an.
2. Wählen Sie im Menü Apps die Option Neue App hinzufügen aus.
3. Wählen Sie eine Plattform aus und schließen Sie den Schnellstart-Prozess ab.

Android

Weitere Informationen zur Integration der Facebook-Anmeldung in Android-Apps finden Sie unter Facebook-Handbuch Erste Schritte.

iOS – Objective-C

Weitere Informationen zur Integration der Facebook-Anmeldung in iOS Objective-C-Apps finden Sie unter Facebook-Handbuch Erste Schritte.

iOS – Swift

Weitere Informationen zur Integration der Facebook-Anmeldung in iOS Swift-Apps finden Sie unter Facebook-Handbuch Erste Schritte.

JavaScript

Weitere Informationen zur Integration von JavaScript Web-Apps in die Facebook-Anmeldung finden Sie im Facebook-Leitfaden „Erste Schritte“.

Unity

Weitere Informationen zur Integration der Facebook-Anmeldung in Unity-Apps finden Sie unter Facebook-Handbuch Erste Schritte.

Xamarin

Wenn Sie die Facebook-Authentifizierung hinzufügen möchten, integrieren Sie zunächst das Facebook-SDK in Ihre Anwendung. Verwenden Sie dazu die entsprechenden Anleitungen unten. Amazon-Cognito-Identitäten-Pools verwenden das Facebook-Zugriffstoken zur Erstellung einer eindeutigen Benutzerkennung, die mit einer Amazon-Cognito-Identität verbunden ist.

- Facebook-iOS-SDK von Xamarin
- Facebook-Android-SDK von Xamarin

Konfigurieren eines Identitätsanbieters in der Amazon-Cognito-Identitätspool-Konsole

Gehen Sie wie folgt vor, um Ihren Identitätsanbieter zu konfigurieren.

So fügen Sie einen neuen Facebook-Identitätsanbieter (IdP) hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.

2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Facebook.
5. Geben Sie die App-ID des OAuth-Projekts ein, das Sie bei Meta for Developers erstellt haben. Weitere Informationen finden Sie unter Facebook-Anmeldung in den Dokumenten zu Meta for Developers.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

    - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

        i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

        ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

    a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

    b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

    c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Verwenden von Facebook

Android

Wenn Sie die Facebook-Authentifizierung hinzufügen möchten, integrieren Sie zunächst das Facebook-SDK anhand der Anleitungen im Facebook-Leitfaden in Ihre Anwendung. Fügen Sie dann die Schaltfläche Mit Facebook anmelden Ihrer Android-Benutzeroberfläche hinzu. Das Facebook-SDK verwendet ein Sitzungsobjekt, um seinen Status nachzuverfolgen. Amazon Cognito verwendet das Zugriffstoken von diesem Sitzungsobjekt, um den Benutzer zu authentifizieren, die eindeutige Kennung zu generieren und dem Benutzer bei Bedarf Zugriff auf andere AWS Ressourcen zu gewähren.

Nachdem Sie den Benutzer mit dem Facebook-SDK authentifiziert haben, fügen Sie das Sitzungs-Token zu den Amazon-Cognito-Anmeldeinformationen des Anbieters hinzu.

Facebook-SDK 4.0 oder höher:

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("graph.facebook.com", AccessToken.getCurrentAccessToken().getToken());
credentialsProvider.setLogins(logins);
```

Facebook-SDK vor 4.0:

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("graph.facebook.com", Session.getActiveSession().getAccessToken());
credentialsProvider.setLogins(logins);
```

Die Facebook-Anmeldung initialisiert eine Singleton-Sitzung im SDK. Das Facebook-Sitzungsobjekt enthält ein OAuth-Token, das Amazon Cognito verwendet, um AWS Anmeldeinformationen für Ihren authentifizierten Endbenutzer zu generieren. Darüber hinaus verwendet Amazon Cognito das Token, um in Ihrer Benutzer-Datenbank zu überprüfen, ob ein Benutzer mit dieser speziellen Facebook-Identität existiert. Wenn der Benutzer bereits vorhanden ist, gibt API gibt den vorhandenen Bezeichner zurück. Andernfalls gibt die API einen neuen Bezeichner zurück. Das Client-SDK legt Bezeichner automatisch im Cache des lokalen Geräts ab.

> **Note**
>
> Nachdem Sie die Login-Zuordnung eingerichtet haben, rufen Sie die Anmeldeinformationen an `refresh` oder `get` rufen Sie sie ab. AWS

iOS – Objective-C

Wenn Sie die Facebook-Authentifizierung hinzufügen möchten, integrieren Sie zunächst das Facebook-SDK anhand der Anleitungen im Facebook-Leitfaden in Ihre Anwendung. Fügen Sie dann eine Schaltfläche "Mit Facebook anmelden" zu Ihrer Benutzeroberfläche hinzu. Das Facebook-SDK verwendet ein Sitzungsobjekt, um seinen Status nachzuverfolgen. Amazon Cognito verwendet den Zugriffstoken von diesem Sitzungsobjekt zur Authentifizierung des Benutzers, und um ihn mit einem eindeutigen Amazon-Cognito-Identitäten-Pool (Verbundidentität) zu verbinden.

Wenn Sie das Facebook-Zugriffs-Token für Amazon Cognito bereitstellen möchten, implementieren Sie das AWSIdentityProviderManager-Protokoll.

Wenn Sie die `logins`-Methode implementieren, geben Sie ein Wörterbuch zurück, das `AWSIdentityProviderFacebook` enthält. Dieses Wörterbuch fungiert als Schlüssel und das aktuelle Zugriffstoken des authentifizierten Facebook-Benutzers als Wert, wie im folgenden Codebeispiel gezeigt.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    FBSDKAccessToken* fbToken = [FBSDKAccessToken currentAccessToken];
    if(fbToken){
        NSString *token = fbToken.tokenString;
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook : token }];
    }else{
            return [AWSTask taskWithError:[NSError errorWithDomain:@"Facebook Login"
                                                              code:-1
                                                          userInfo:@{@"error":@"No current
 Facebook access token"}]];
    }
}
```

Beim Instantiieren des `AWSCognitoCredentialsProvider` leiten Sie die Klasse, die `AWSIdentityProviderManager` implementiert, als den Wert von `identityProviderManager` im Konstruktor weiter. Weitere Informationen finden Sie auf der AWSCognitoCredentialsProviderReferenzseite und wählen Sie initWithRegionTyp:identityPoolId: identityProviderManager.

iOS – Swift

Wenn Sie die Facebook-Authentifizierung hinzufügen möchten, integrieren Sie zunächst das Facebook-SDK anhand der Anleitungen im Facebook-Leitfaden in Ihre Anwendung. Fügen Sie dann

eine Schaltfläche "Mit Facebook anmelden" zu Ihrer Benutzeroberfläche hinzu. Das Facebook-SDK verwendet ein Sitzungsobjekt, um seinen Status nachzuverfolgen. Amazon Cognito verwendet den Zugriffstoken von diesem Sitzungsobjekt zur Authentifizierung des Benutzers, und um ihn mit einem eindeutigen Amazon-Cognito-Identitäten-Pool (Verbundidentität) zu verbinden.

Wenn Sie das Facebook-Zugriffs-Token für Amazon Cognito bereitstellen möchten, implementieren Sie das AWSIdentityProviderManager-Protokoll.

Wenn Sie die `logins`-Methode implementieren, geben Sie ein Wörterbuch zurück, das `AWSIdentityProviderFacebook` enthält. Dieses Wörterbuch fungiert als Schlüssel und das aktuelle Zugriffstoken des authentifizierten Facebook-Benutzers als Wert, wie im folgenden Codebeispiel gezeigt.

```
class FacebookProvider: NSObject, AWSIdentityProviderManager {
    func logins() -> AWSTask<NSDictionary> {
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error:NSError(domain: "Facebook Login", code: -1 , userInfo:
 ["Facebook" : "No current Facebook access token"]))
    }
}
```

Beim Instantiieren des `AWSCognitoCredentialsProvider` leiten Sie die Klasse, die `AWSIdentityProviderManager` implementiert, als den Wert von `identityProviderManager` im Konstruktor weiter. Weitere Informationen finden Sie auf der AWSCognitoCredentialsProviderReferenzseite und wählen Sie initWithRegionTyp:identityPoolId: identityProviderManager.

JavaScript

Wenn Sie die Facebook-Authentifizierung hinzufügen möchten, befolgen Sie die Anleitungen unter Facebook-Anmeldung für das Web und fügen Sie die Schaltfläche Mit Facebook anmelden auf Ihrer Website hinzu. Das Facebook-SDK verwendet ein Sitzungsobjekt, um seinen Status nachzuverfolgen. Amazon Cognito verwendet das Zugriffstoken von diesem Sitzungsobjekt, um den Benutzer zu authentifizieren, die eindeutige Kennung zu generieren und dem Benutzer bei Bedarf Zugriff auf andere AWS Ressourcen zu gewähren.

Nachdem Sie den Benutzer mit dem Facebook-SDK authentifiziert haben, fügen Sie das Sitzungs-Token zu den Amazon-Cognito-Anmeldeinformationen des Anbieters hinzu.

```
FB.login(function (response) {

  // Check if the user logged in successfully.
  if (response.authResponse) {

    console.log('You are now logged in.');

    // Add the Facebook access token to the Amazon Cognito credentials login map.
    AWS.config.credentials = new AWS.CognitoIdentityCredentials({
      IdentityPoolId: 'IDENTITY_POOL_ID',
      Logins: {
        'graph.facebook.com': response.authResponse.accessToken
      }
    });

    // Obtain AWS credentials
    AWS.config.credentials.get(function(){
        // Access AWS resources here.
    });

  } else {
    console.log('There was a problem logging you in.');
  }

});
```

Das Facebook-SDK erhält ein OAuth-Token, das Amazon Cognito verwendet, um AWS Anmeldeinformationen für Ihren authentifizierten Endbenutzer zu generieren. Darüber hinaus verwendet Amazon Cognito das Token, um in Ihrer Benutzer-Datenbank zu überprüfen, ob ein Benutzer mit dieser speziellen Facebook-Identität existiert. Wenn der Benutzer bereits vorhanden ist, gibt API gibt den vorhandenen Bezeichner zurück. Andernfalls wird ein neuer Bezeichner zurückgegeben. Bezeichner werden automatisch vom Client-SDK auf dem lokalen Gerät zwischengespeichert.

> **Note**
>
> Nachdem Sie die Anmeldezuordnung festgelegt haben, geben Sie einen Aufruf an `refresh` oder `get` aus, um die Anmeldeinformationen zu erhalten. Ein Codebeispiel finden Sie in der JavaScript README-Datei unter „Anwendungsfall 17, Integrieren von Benutzerpools mit Cognito Identity“.

Unity

Wenn Sie die Facebook-Authentifizierung hinzufügen möchten, integrieren Sie zunächst das Facebook-SDK anhand der Anleitungen im Facebook-Leitfaden in Ihre Anwendung. Amazon Cognito verwendet das Facebook-Zugriffstoken vom FB-Objekt zum Generieren einer eindeutigen Benutzer-ID, die einer Amazon-Cognito-Identität zugeordnet ist.

Nachdem Sie den Benutzer mit dem Facebook-SDK authentifiziert haben, fügen Sie das Sitzungs-Token zu den Amazon-Cognito-Anmeldeinformationen des Anbieters hinzu.

```
void Start()
{
    FB.Init(delegate() {
        if (FB.IsLoggedIn) { //User already logged in from a previous session
            AddFacebookTokenToCognito();
        } else {
            FB.Login ("email", FacebookLoginCallback);
        }
    });
}

void FacebookLoginCallback(FBResult result)
{
    if (FB.IsLoggedIn)
    {
        AddFacebookTokenToCognito();
    }
    else
    {
        Debug.Log("FB Login error");
    }
}

void AddFacebookTokenToCognito()
{
    credentials.AddLogin ("graph.facebook.com",
 AccessToken.CurrentAccessToken.TokenString);
}
```

Bevor Sie `FB.AccessToken` verwenden, rufen Sie `FB.Login()` auf und stellen Sie sicher, dass `FB.IsLoggedIn` wahr ist.

Xamarin

Xamarin für Android:

```
public void InitializeFacebook() {
    FacebookSdk.SdkInitialize(this.ApplicationContext);
    callbackManager = CallbackManagerFactory.Create();
    LoginManager.Instance.RegisterCallback(callbackManager, new FacebookCallback &lt;
 LoginResult &gt; () {
      HandleSuccess = loginResult = &gt; {
        var accessToken = loginResult.AccessToken;
        credentials.AddLogin("graph.facebook.com", accessToken.Token);
        //open new activity
      },
      HandleCancel = () = &gt; {
        //throw error message
      },
      HandleError = loginError = &gt; {
        //throw error message
      }
    });
    LoginManager.Instance.LogInWithReadPermissions(this, new List &lt; string &gt; {
      "public_profile"
    });
  }
```

Xamarin für iOS:

```
public void InitializeFacebook() {
  LoginManager login = new LoginManager();
  login.LogInWithReadPermissions(readPermissions.ToArray(),
 delegate(LoginManagerLoginResult result, NSError error) {
    if (error != null) {
      //throw error message
    } else if (result.IsCancelled) {
      //throw error message
    } else {
      var accessToken = loginResult.AccessToken;
      credentials.AddLogin("graph.facebook.com", accessToken.Token);
      //open new view controller
    }
  });
}
```

Login with Amazon als Identitätspools (IdP) einrichten

In Amazon Cognito ist Login with Amazon integriert, um Verbundauthentifizierung für Ihre mobilen und Webanwendungsbenutzer bereitzustellen. In diesem Abschnitt wird erklärt, wie Sie Ihre Anwendung mit Login with Amazon als Identitätsanbieter (IdP) registrieren und einrichten.

Richten Sie Login with Amazon im Entwicklerportal ein, damit es mit Amazon Cognito funktioniert. Weitere Informationen finden Sie unter Einrichten von „Login with Amazon” in den häufig gestellten Fragen zu Login with Amazon.

> **Note**
>
> Für Xamarin befolgen Sie die Anweisungen im Xamarin Handbuch „Erste Schritte“, um Login with Amazon in Ihre Xamarin-Anwendung zu integrieren.

> **Note**
>
> Sie können Login with Amazon auf der Unity-Plattform nicht nativ integrieren. Verwenden Sie stattdessen eine Webansicht und folgen Sie dem Browser-Anmeldungsablauf.

Einrichten von „Login with Amazon”

Implementierung von Login with Amazon

Im Amazon-Entwicklerportal können Sie eine OAuth-Anwendung einrichten, um sie in Ihren Identitäten-Pool zu integrieren. Hier finden Sie außerdem Dokumentation zu Login with Amazon und können SDKs herunterladen. Wählen Sie Developer console (Entwickler-Konsole) und dann Login with Amazon im Entwicklerportal aus. Sie können ein Sicherheitsprofil für Ihre Anwendung erstellen und dann Authentifizierungsmechanismen für Login with Amazon in Ihrer App erstellen. Weitere Informationen darüber, wie Sie Login-with-Amazon-Authentifizierung in Ihre App integrieren können, finden Sie unter Abrufen von Anmeldeinformationen.

Amazon gibt eine OAuth 2.0-Client-ID für Ihr neues Sicherheitsprofil aus. Sie finden die client ID (Client-ID) auf der Registerkarte Web Settings (Web-Einstellungen) im Sicherheitsprofil. Geben Sie in Ihrem Identitätspool die Sicherheitsprofil-ID im Feld App ID des Login-with-Amazon-IDP ein.

> **Note**
>
> Sie geben in Ihrem Identitätspool die Sicherheitsprofil-ID im Feld App ID des Login-with-Amazon-IDP ein. Bei Benutzerpools wird stattdessen die Client-ID verwenden.

Konfigurieren des externen Anbieters in der Amazon-Cognito-Konsole

So fügen Sie eine Anmeldung mit dem Identitätsanbieter (IdP) Amazon hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Mit Amazon anmelden.
5. Geben Sie die App-ID des OAuth-Projekts ein, das Sie unter Mit Amazon anmelden erstellt haben. Weitere Informationen finden Sie in der Dokumentation zur Anmeldung mit Amazon.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

   - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

     i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

     ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.
7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Verwenden von „Login with Amazon”: Android

Nachdem Sie die Amazon-Anmeldung authentifiziert haben, können Sie das Token in der onSuccess-Methode der Schnittstelle an den Amazon Cognito-Anmeldeinformationsanbieter übergeben. TokenListener Der Code sieht folgendermaßen aus:

```
@Override
public void onSuccess(Bundle response) {
    String token = response.getString(AuthzConstants.BUNDLE_KEY.TOKEN.val);
    Map<String, String> logins = new HashMap<String, String>();
    logins.put("www.amazon.com", token);
    credentialsProvider.setLogins(logins);
}
```

Verwenden von Login with Amazon: iOS – Objective-C

Nachdem Sie die Amazon-Anmeldung authentifiziert haben, können Sie das Token in der requestDidSucceed AMZN-Methode an den Amazon Cognito-Anmeldeinformationsanbieter übergeben: AccessTokenDelegate

```
- (void)requestDidSucceed:(APIResult \*)apiResult {
    if (apiResult.api == kAPIAuthorizeUser) {
        [AIMobileLib getAccessTokenForScopes:[NSArray arrayWithObject:@"profile"]
 withOverrideParams:nil delegate:self];
    }
    else if (apiResult.api == kAPIGetAccessToken) {
        credentialsProvider.logins = @{ @(AWSCognitoLoginProviderKeyLoginWithAmazon):
 apiResult.result };
    }
```

```
}}
```

Verwenden von „Login with Amazon”: iOS – Swift

Nachdem Sie die Amazon-Anmeldung authentifiziert haben, können Sie in der `requestDidSucceed`-Methode des `AMZNAccessTokenDelegate` das Token an den Amazon-Cognito-Anmeldeinformationsanbieter weiterleiten.

```
func requestDidSucceed(apiResult: APIResult!) {
    if apiResult.api == API.AuthorizeUser {
        AIMobileLib.getAccessTokenForScopes(["profile"], withOverrideParams: nil,
 delegate: self)
    } else if apiResult.api == API.GetAccessToken {
        credentialsProvider.logins =
 [AWSCognitoLoginProviderKey.LoginWithAmazon.rawValue: apiResult.result]
    }
}
```

Verwenden Sie „Login with Amazon“: JavaScript

Nachdem der Benutzer mit Login with Amazon authentifiziert und wieder zurück zu Ihrer Website weitergeleitet wurde, wird das Anmelden mit Amazon Zugriffs-Token in der Abfragezeichenfolge bereitgestellt. Leiten Sie das Token an die Anmeldeinformationen weiter.

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
   IdentityPoolId: 'IDENTITY_POOL_ID',
   Logins: {
       'www.amazon.com': 'Amazon Access Token'
   }
});
```

Verwenden von „Login with Amazon”: Xamarin

Xamarin für Android

```
AmazonAuthorizationManager manager = new AmazonAuthorizationManager(this,
 Bundle.Empty);

var tokenListener = new APIListener {
  Success = response => {
```

```
    // Get the auth token
    var token = response.GetString(AuthzConstants.BUNDLE_KEY.Token.Val);
    credentials.AddLogin("www.amazon.com", token);
  }
};

// Try and get existing login
manager.GetToken(new[] {
  "profile"
}, tokenListener);
```

Xamarin für iOS

Fügen Sie `AppDelegate.cs` Folgendes ein:

```
public override bool OpenUrl (UIApplication application, NSUrl url, string
 sourceApplication, NSObject annotation)
{
    // Pass on the url to the SDK to parse authorization code from the url
    bool isValidRedirectSignInURL = AIMobileLib.HandleOpenUrl (url, sourceApplication);
    if(!isValidRedirectSignInURL)
        return false;

    // App may also want to handle url
    return true;
}
```

Führen Sie dann im Bereich `ViewController.cs` die folgenden Schritte aus:

```
public override void ViewDidLoad ()
{
    base.LoadView ();

    // Here we create the Amazon Login Button
    btnLogin = UIButton.FromType (UIButtonType.RoundedRect);
    btnLogin.Frame = new RectangleF (55, 206, 209, 48);
    btnLogin.SetTitle ("Login using Amazon", UIControlState.Normal);
    btnLogin.TouchUpInside += (sender, e) => {
        AIMobileLib.AuthorizeUser (new [] { "profile"}, new AMZNAuthorizationDelegate
 ());
    };
    View.AddSubview (btnLogin);
}
```

```
// Class that handles Authentication Success/Failure
public class AMZNAuthorizationDelegate : AIAuthenticationDelegate
{
  public override void RequestDidSucceed(ApiResult apiResult)
    {
      // Your code after the user authorizes application for requested scopes
      var token = apiResult["access_token"];
      credentials.AddLogin("www.amazon.com",token);
    }

    public override void RequestDidFail(ApiError errorResponse)
    {
      // Your code when the authorization fails
      InvokeOnMainThread(() => new UIAlertView("User Authorization Failed",
 errorResponse.Error.Message, null, "Ok", null).Show());
    }
}
```

Google als Identitätspool-IdP einrichten

Amazon Cognito wird in Google integriert, um verbundene Authentifizierung für Ihre mobile Anwendungsbenutzer zu bieten. In diesem Abschnitt wird erklärt, wie Sie Ihre Anwendung bei Google als IdP registrieren und einrichten.

Android

> **Note**
>
> Wenn Ihre App Google verwendet und auf mehreren mobilen Plattformen verfügbar ist, sollten Sie sie als OpenID-Connect-Anbieter konfigurieren. Fügen Sie alle erstellten Client-IDs als zusätzliche Zielgruppenwerte hinzu, um eine bessere Integration zu ermöglichen. Weitere Informationen zu Google Cross-Client-Identität finden Sie unter Cross-Client-Identity-Modell.

Einrichten von Google

Zum Aktivieren der Google-Anmeldung für Android müssen Sie ein Google-Developer-Konsolenprojekt für Ihre Anwendung erstellen.

1. Gehen Sie zur Google Developers-Konsole und erstellen Sie ein neues Projekt.
2. Wählen Sie APIs & Services (APIs und Services) und dann OAuth consent screen (OAuth-Zustimmungsbildschirm) aus. Passen Sie die Informationen an, die Google Ihren Benutzern anzeigt, wenn Google um ihre Zustimmung zur Weitergabe ihrer Profildaten mit Ihrer App bittet.
3. Wählen Sie Credentials (Anmeldeinformationen) und dann Create credentials (Anmeldeinformationen erstellen) aus. Wählen Sie OAuth client ID (OAuth-Client-ID) aus. Wählen Sie Android als Application type (Anwendungstyp) aus. Erstellen Sie eine separate Client-ID für jede Plattform, auf der Sie Ihre App entwickeln.
4. Wählen Sie unter Credentials (Anmeldeinformationen) die Option Manage service accounts(Service-Konten verwalten) aus. Wählen Sie Create service account (Service-Konto erstellen) aus. Geben Sie Ihre Service-Kontodaten ein und wählen Sie Create and continue (Erstellen und fortfahren) aus.
5. Gewähren Sie dem Service-Konto Zugriff auf Ihr Projekt. Gewähren Sie Benutzern Zugriff auf das Service-Konto je nach Erfordernissen der App.
6. Wählen Sie Ihr neues Service-Konto und dann die Registerkarte Keys (Schlüssel) und die Option Add key (Schlüssel hinzufügen) aus. Erstellen Sie einen neuen JSON-Schlüssel und laden Sie ihn herunter.

Weitere Informationen zur Verwendung der Google-Developer-Konsole finden Sie unter Erstellen und Verwalten von Projekten in der Google-Cloud-Dokumentation.

Weitere Informationen zur Integration von Google in Ihre Android-App finden Sie in der Google Identity-Dokumentation unter Nutzer mit Anmeldung bei Google authentifizieren.

So fügen Sie einen Google-Identitätsanbieter (IdP) hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Google.
5. Geben Sie die Client-ID des OAuth-Projekts ein, das Sie auf der Google-Cloud-Plattform erstellt haben. Weitere Informationen finden Sie unter Einrichten von OAuth 2.0 in der Google-Cloud-Plattform-Konsole-Hilfe.

6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

   - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

     i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

     ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Verwenden von Google

Zum Aktivieren der Anmeldung mit Google in Ihrer Anwendung folgen Sie den Anleitungen in der Google-Dokumentation für Android. Wenn sich ein Benutzer anmeldet, fordert er ein OpenID-Connect-Authentifizierungstoken von Google an. Amazon Cognito verwendet das Token dann, um den Benutzer zu authentifizieren und eine eindeutige ID zu generieren.

Der folgende Beispiel-Code zeigt, wie Sie das Authentifizierungs-Token vom Google Play Service abrufen:

```
GooglePlayServicesUtil.isGooglePlayServicesAvailable(getApplicationContext());
AccountManager am = AccountManager.get(this);
Account[] accounts = am.getAccountsByType(GoogleAuthUtil.GOOGLE_ACCOUNT_TYPE);
String token = GoogleAuthUtil.getToken(getApplicationContext(), accounts[0].name,
        "audience:server:client_id:YOUR_GOOGLE_CLIENT_ID");
Map<String, String> logins = new HashMap<String, String>();
logins.put("accounts.google.com", token);
credentialsProvider.setLogins(logins);
```

iOS – Objective-C

> **Note**
>
> Wenn Ihre App Google verwendet und auf mehreren mobilen Plattformen verfügbar ist, konfigurieren Sie Google als OpenID-Connect-Anbieter. Fügen Sie alle erstellten Client-IDs als zusätzliche Zielgruppenwerte hinzu, um eine bessere Integration zu ermöglichen. Weitere Informationen zu Google Cross-Client-Identität finden Sie unter Cross-Client-Identity-Modell.

Einrichten von Google

Zum Aktivieren der Google-Anmeldung für iOS erstellen Sie ein Google-Developer-Konsolenprojekt für Ihre Anwendung.

1. Gehen Sie zur Google Developers-Konsole und erstellen Sie ein neues Projekt.
2. Wählen Sie APIs & Services (APIs und Services) und dann OAuth consent screen (OAuth-Zustimmungsbildschirm) aus. Passen Sie die Informationen an, die Google Ihren Benutzern anzeigt, wenn Google um ihre Zustimmung zur Weitergabe ihrer Profildaten mit Ihrer App bittet.
3. Wählen Sie Credentials (Anmeldeinformationen) und dann Create credentials (Anmeldeinformationen erstellen) aus. Wählen Sie OAuth client ID (OAuth-Client-ID) aus. Wählen Sie iOS als Application type (Anwendungstyp) aus. Erstellen Sie eine separate Client-ID für jede Plattform, auf der Sie Ihre App entwickeln.
4. Wählen Sie unter Credentials (Anmeldeinformationen) die Option Manage service accounts(Service-Konten verwalten) aus. Wählen Sie Create service account (Service-Konto erstellen) aus. Geben Sie Ihre Service-Kontodaten ein und wählen Sie Create and continue (Erstellen und fortfahren) aus.

5. Gewähren Sie dem Service-Konto Zugriff auf Ihr Projekt. Gewähren Sie Benutzern Zugriff auf das Service-Konto je nach Erfordernissen der App.
6. Wählen Sie Ihr neues Service-Konto aus. Wählen Sie die Registerkarte Keys (Schlüssel) und Add key (Schlüssel hinzufügen) aus. Erstellen Sie einen neuen JSON-Schlüssel und laden Sie ihn herunter.

Weitere Informationen zur Verwendung der Google-Developer-Konsole finden Sie unter Erstellen und Verwalten von Projekten in der Google-Cloud-Dokumentation.

Weitere Informationen zur Integration von Google in Ihre iOS-App finden Sie in der Google-Identitätsdokumentation unter Google-Anmeldung für iOS.

So fügen Sie einen Google-Identitätsanbieter (IdP) hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Google.
5. Geben Sie die Client-ID des OAuth-Projekts ein, das Sie auf der Google-Cloud-Plattform erstellt haben. Weitere Informationen finden Sie unter Einrichten von OAuth 2.0 in der Google-Cloud-Plattform-Konsole-Hilfe.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

    - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

        i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Verwenden von Google

Um die Anmeldung mit Google in Ihrer Anwendung zu aktivieren, befolgen Sie die Anleitung in der Google-Dokumentation für iOS. Bei einer erfolgreichen Authentifizierung wird ein OpenID Connect-Authentifizierungs-Token erstellt, mit dem Amazon Cognito den Benutzer authentifiziert und eindeutige Kennung generiert.

Bei einer erfolgreichen Authentifizierung wird ein `GTMOAuth2Authentication`-Objekt mit einem `id_token` erstellt, mit dem Amazon Cognito den Benutzer authentifiziert und eine eindeutige Kennung generiert:

```
- (void)finishedWithAuth: (GTMOAuth2Authentication *)auth error: (NSError *) error {
        NSString *idToken = [auth.parameters objectForKey:@"id_token"];
        credentialsProvider.logins = @{ @(AWSCognitoLoginProviderKeyGoogle): idToken };
    }
```

iOS – Swift

> **Note**
>
> Wenn Ihre App Google verwendet und auf mehreren mobilen Plattformen verfügbar ist, konfigurieren Sie Google als OpenID-Connect-Anbieter. Fügen Sie alle erstellten Client-IDs

als zusätzliche Zielgruppenwerte hinzu, um eine bessere Integration zu ermöglichen. Weitere Informationen zu Google Cross-Client-Identität finden Sie unter Cross-Client-Identity-Modell.

Einrichten von Google

Zum Aktivieren der Google-Anmeldung für iOS erstellen Sie ein Google-Developer-Konsolenprojekt für Ihre Anwendung.

1. Gehen Sie zur Google Developers-Konsole und erstellen Sie ein neues Projekt.
2. Wählen Sie APIs & Services (APIs und Services) und dann OAuth consent screen (OAuth-Zustimmungsbildschirm) aus. Passen Sie die Informationen an, die Google Ihren Benutzern anzeigt, wenn Google um ihre Zustimmung zur Weitergabe ihrer Profildaten mit Ihrer App bittet.
3. Wählen Sie Credentials (Anmeldeinformationen) und dann Create credentials (Anmeldeinformationen erstellen) aus. Wählen Sie OAuth client ID (OAuth-Client-ID) aus. Wählen Sie iOS als Application type (Anwendungstyp) aus. Erstellen Sie eine separate Client-ID für jede Plattform, auf der Sie Ihre App entwickeln.
4. Wählen Sie unter Credentials (Anmeldeinformationen) die Option Manage service accounts(Service-Konten verwalten) aus. Wählen Sie Create service account (Service-Konto erstellen) aus. Geben Sie Ihre Service-Kontodaten ein und wählen Sie Create and continue (Erstellen und fortfahren) aus.
5. Gewähren Sie dem Service-Konto Zugriff auf Ihr Projekt. Gewähren Sie Benutzern Zugriff auf das Service-Konto je nach Erfordernissen der App.
6. Wählen Sie Ihr neues Service-Konto und dann die Registerkarte Keys (Schlüssel) und die Option Add key (Schlüssel hinzufügen) aus. Erstellen Sie einen neuen JSON-Schlüssel und laden Sie ihn herunter.

Weitere Informationen zur Verwendung der Google-Developer-Konsole finden Sie unter Erstellen und Verwalten von Projekten in der Google-Cloud-Dokumentation.

Weitere Informationen zur Integration von Google in Ihre iOS-App finden Sie in der Google-Identitätsdokumentation unter Google-Anmeldung für iOS.

Wählen Sie Identitäten-Pools verwalten in der Homepage der Amazon-Cognito-Konsole aus:

Konfigurieren des externen Anbieters in der Amazon-Cognito-Konsole

1. Wählen Sie den Namen des Identitäten-Pools aus, für den Sie Google als externen Anbieter aktivieren möchten. Die Seite Dashboard für Ihren Identitäten-Pool wird angezeigt.
2. Klicken Sie in der oberen rechten Ecke der Seite auf Dashboard und dann auf Edit identity pool (Identitäten-Pool bearbeiten). Die Seite "Edit identity pool" wird angezeigt.
3. Scrollen Sie nach unten und klicken Sie zum Erweitern des Abschnitts auf Authentication providers (Authentifizierungsanbieter).
4. Wählen Sie die Registerkarte Google.
5. Wählen Sie Unlock (Entsperren) aus.
6. Geben Sie die Google-Client-ID ein, die Sie von Google erhalten haben, und wählen Sie dann Save Changes (Änderungen speichern) aus.

Verwenden von Google

Um die Anmeldung mit Google in Ihrer Anwendung zu aktivieren, befolgen Sie die Anleitung in der Google-Dokumentation für iOS. Bei einer erfolgreichen Authentifizierung wird ein OpenID-Connect-Authentifizierungs-Token erstellt, mit dem Amazon Cognito den Benutzer authentifiziert und eine eindeutige ID generiert.

Eine erfolgreiche Authentifizierung führt zu einem `GTMOAuth2Authentication`-Objekt, das ein `id_token` enthält. Amazon Cognito verwendet dieses Token, um den Benutzer zu authentifizieren und eine eindeutige ID zu generieren.

```
func finishedWithAuth(auth: GTMOAuth2Authentication!, error: NSError!) {
    if error != nil {
      print(error.localizedDescription)
    }
    else {
      let idToken = auth.parameters.objectForKey("id_token")
      credentialsProvider.logins = [AWSCognitoLoginProviderKey.Google.rawValue:
 idToken!]
    }
}
```

JavaScript

> **Note**
>
> Wenn Ihre App Google verwendet und auf mehreren mobilen Plattformen verfügbar ist, sollten Sie sie als OpenID-Connect-Anbieter konfigurieren. Fügen Sie alle erstellten Client-IDs als zusätzliche Zielgruppenwerte hinzu, um eine bessere Integration zu ermöglichen. Weitere Informationen zu Google Cross-Client-Identität finden Sie unter Cross-Client-Identity-Modell.

Einrichten von Google

Um die Google-Anmeldung für eine JavaScript Web-App zu aktivieren, erstellen Sie ein Google Developers Console-Projekt für Ihre Anwendung.

1. Gehen Sie zur Google Developers-Konsole und erstellen Sie ein neues Projekt.
2. Wählen Sie APIs & Services (APIs und Services) und dann OAuth consent screen (OAuth-Zustimmungsbildschirm) aus. Passen Sie die Informationen an, die Google Ihren Benutzern anzeigt, wenn Google um ihre Zustimmung zur Weitergabe ihrer Profildaten mit Ihrer App bittet.
3. Wählen Sie Credentials (Anmeldeinformationen) und dann Create credentials (Anmeldeinformationen erstellen) aus. Wählen Sie OAuth client ID (OAuth-Client-ID) aus. Wählen Sie Web application (Webanwendung) als Application type (Anwendungstyp) aus. Erstellen Sie eine separate Client-ID für jede Plattform, auf der Sie Ihre App entwickeln.
4. Wählen Sie unter Credentials (Anmeldeinformationen) die Option Manage service accounts(Service-Konten verwalten) aus. Wählen Sie Create service account (Service-Konto erstellen) aus. Geben Sie Ihre Service-Kontodaten ein und wählen Sie Create and continue (Erstellen und fortfahren) aus.
5. Gewähren Sie dem Service-Konto Zugriff auf Ihr Projekt. Gewähren Sie Benutzern Zugriff auf das Service-Konto je nach Erfordernissen der App.
6. Wählen Sie Ihr neues Service-Konto und dann die Registerkarte Keys (Schlüssel) und die Option Add key (Schlüssel hinzufügen) aus. Erstellen Sie einen neuen JSON-Schlüssel und laden Sie ihn herunter.

Weitere Informationen zur Verwendung der Google-Developer-Konsole finden Sie unter Erstellen und Verwalten von Projekten in der Google-Cloud-Dokumentation.

Weitere Informationen zur Integration von Google in Ihre Web-App finden Sie in der Google-Identitätsdokumentation unter Mit Google anmelden.

Konfigurieren des externen Anbieters in der Amazon-Cognito-Konsole

So fügen Sie einen Google-Identitätsanbieter (IdP) hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Google.
5. Geben Sie die Client-ID des OAuth-Projekts ein, das Sie auf der Google-Cloud-Plattform erstellt haben. Weitere Informationen finden Sie unter Einrichten von OAuth 2.0 in der Google-Cloud-Plattform-Konsole-Hilfe.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

   - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

     i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

     ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.
7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Verwenden von Google

Um die Anmeldung mit Google in Ihrer Anwendung zu aktivieren, befolgen Sie die Google Dokumentation für Web.

Bei einer erfolgreichen Authentifizierung wird ein Antwortobjekt mit einem `id_token` erstellt, mit dem Amazon Cognito den Benutzer authentifiziert und eine eindeutige ID generiert:

```
function signinCallback(authResult) {
  if (authResult['status']['signed_in']) {

     // Add the Google access token to the Amazon Cognito credentials login map.
     AWS.config.credentials = new AWS.CognitoIdentityCredentials({
        IdentityPoolId: 'IDENTITY_POOL_ID',
        Logins: {
           'accounts.google.com': authResult['id_token']
        }
     });

     // Obtain AWS credentials
     AWS.config.credentials.get(function(){
        // Access AWS resources here.
     });
  }
}
```

Unity

Einrichten von Google

Zum Aktivieren der Google-Anmeldung für eine Unity-App erstellen Sie ein Google-Developer-Konsolenprojekt für Ihre Anwendung.

1. Gehen Sie zur Google Developers-Konsole und erstellen Sie ein neues Projekt.
2. Wählen Sie APIs & Services (APIs und Services) und dann OAuth consent screen (OAuth-Zustimmungsbildschirm) aus. Passen Sie die Informationen an, die Google Ihren Benutzern anzeigt, wenn Google um ihre Zustimmung zur Weitergabe ihrer Profildaten mit Ihrer App bittet.
3. Wählen Sie Credentials (Anmeldeinformationen) und dann Create credentials (Anmeldeinformationen erstellen) aus. Wählen Sie OAuth client ID (OAuth-Client-ID) aus. Wählen Sie Web application (Webanwendung) als Application type (Anwendungstyp) aus. Erstellen Sie eine separate Client-ID für jede Plattform, auf der Sie Ihre App entwickeln.
4. Erstellen Sie für Unity eine zusätzliche OAuth client ID (OAuth-Client-ID) für Android und eine weitere für iOS.
5. Wählen Sie unter Credentials (Anmeldeinformationen) die Option Manage service accounts(Service-Konten verwalten) aus. Wählen Sie Create service account (Service-Konto erstellen) aus. Geben Sie Ihre Service-Kontodaten ein und wählen Sie Create and continue (Erstellen und fortfahren) aus.
6. Gewähren Sie dem Service-Konto Zugriff auf Ihr Projekt. Gewähren Sie Benutzern Zugriff auf das Service-Konto je nach Erfordernissen der App.
7. Wählen Sie Ihr neues Service-Konto und dann die Registerkarte Keys (Schlüssel) und die Option Add key (Schlüssel hinzufügen) aus. Erstellen Sie einen neuen JSON-Schlüssel und laden Sie ihn herunter.

Weitere Informationen zur Verwendung der Google-Developer-Konsole finden Sie unter Erstellen und Verwalten von Projekten in der Google-Cloud-Dokumentation.

Einen OpenID-Anbieter in der IAM Console erstellen

1. Erstellen Sie einen OpenID-Anbieter in der IAM-Konsole. Weitere Informationen zum Einrichten eines OpenID-Anbieters finden Sie unter Verwenden von OpenID-Connect-Identitätsanbietern.
2. Wenn Sie zur Eingabe Ihrer Provider-URL aufgefordert werden, geben Sie ein `"https://accounts.google.com"`.
3. Wenn Sie zur Eingabe eines Wert in das Feld Zielgruppe aufgefordert werden, geben Sie eine der drei Client-IDs ein, die Sie in den vorherigen Schritten erstellt haben.
4. Wählen Sie den Anbieternamen aus und fügen Sie zwei weitere Zielgruppen mit den beiden anderen Client-IDs hinzu.

Konfigurieren des externen Anbieters in der Amazon-Cognito-Konsole

Wählen Sie Identitäten-Pools verwalten in der Homepage der Amazon-Cognito-Konsole aus:

So fügen Sie einen Google-Identitätsanbieter (IdP) hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Google.
5. Geben Sie die Client-ID des OAuth-Projekts ein, das Sie auf der Google-Cloud-Plattform erstellt haben. Weitere Informationen finden Sie unter Einrichten von OAuth 2.0 in der Google-Cloud-Plattform-Konsole-Hilfe.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

   - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

     i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

     ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.
7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Installieren des Unity Google Plug-In

1. Fügen Sie das Google Play Games-Plug-In für Unity zu Ihrem Unity-Projekt hinzu.
2. Konfigurieren Sie in Unity das Plug-In über das Menü Windows mithilfe der drei IDs für die Android- und iOS-Plattformen.

Verwenden von Google

Der folgende Beispiel-Code zeigt, wie Sie das Authentifizierungs-Token vom Google Play Service abrufen:

```
void Start()
{
  PlayGamesClientConfiguration config = new
 PlayGamesClientConfiguration.Builder().Build();
  PlayGamesPlatform.InitializeInstance(config);
  PlayGamesPlatform.DebugLogEnabled = true;
  PlayGamesPlatform.Activate();
  Social.localUser.Authenticate(GoogleLoginCallback);
}

void GoogleLoginCallback(bool success)
{
  if (success)
  {
    string token = PlayGamesPlatform.Instance.GetIdToken();
    credentials.AddLogin("accounts.google.com", token);
  }
  else
  {
    Debug.LogError("Google login failed. If you are not running in an actual Android/
iOS device, this is expected.");
  }
```

```
}
```

Xamarin

> **Note**
>
> Amazon Cognito unterstützt Google auf der Xamarin-Plattform nicht nativ. Die Integration erfordert derzeit die Verwendung einer Webansicht für den Browser-Anmeldungsablauf. Um zu erfahren, wie Google Integration mit anderen SDKs funktioniert, wählen Sie eine andere Plattform.

Um die Anmeldung mit Google in Ihrer Anwendung zu aktivieren, müssen Sie Ihre Benutzer authentifizieren und ein OpenID-Connect-Token von ihnen erhalten. Amazon Cognito verwendet dieses Token zum Generieren einer eindeutigen Benutzer-ID, die einer Amazon-Cognito-Identität zugeordnet ist. Leider gestattet Ihnen das Google SDK für Xamarin nicht, das OpenID-Connect-Token abzurufen. Verwenden Sie daher einen alternativen Client in einer Webansicht.

Sobald Sie das Token erhalten haben, können Sie es in Ihren `CognitoAWSCredentials` einrichten:

```
credentials.AddLogin("accounts.google.com", token);
```

> **Note**
>
> Wenn Ihre App Google verwendet und auf mehreren mobilen Plattformen verfügbar ist, sollten Sie sie als OpenID-Connect-Anbieter konfigurieren. Fügen Sie alle erstellten Client-IDs als zusätzliche Zielgruppenwerte hinzu, um eine bessere Integration zu ermöglichen. Weitere Informationen zu Google Cross-Client-Identität finden Sie unter Cross-Client-Identity-Modell.

Mit Apple anmelden als Identitätspool-IdP einrichten

Amazon Cognito fügt sich in „Mit Apple anmelden“ ein, um verbundene Authentifizierung für Benutzer Ihrer mobilen Anwendungen und Webanwendungen bereitzustellen. In diesem Abschnitt wird erklärt, wie Sie Ihre Anwendung mittels „Mit Apple anmelden“ als Identitätsanbieter (IdP) registrieren und einrichten.

Das Hinzufügen von „Mit Apple anmelden“ als Authentifizierungsanbieter zu einem Identitäten-Pool ist ein zweistufiger Prozess. Zuerst integrieren Sie „Mit Apple anmelden“ in eine Anwendung und dann konfigurieren Sie „Mit Apple anmelden“ in Identitäten-Pools. Die meisten up-to-date Informationen zur Einrichtung von Sign in with Apple finden Sie unter Konfiguration Ihrer Umgebung für die Anmeldung mit Apple in der Apple-Dokumentation für Entwickler.

Einrichten von „Mit Apple anmelden“

Zum Konfigurieren von „Mit Apple anmelden“ als IdP müssen Sie Ihre Anwendung bei Apple registrieren, um die Client-ID zu erhalten.

1. Erstellen Sie ein Entwickler-Konto bei Apple.
2. Melden Sie sich mit Ihren Apple-Anmeldeinformationen an.
3. Wählen Sie in der linken Navigationsleiste Zertifikate, IDs und Profile aus.
4. Wählen Sie im linken Navigationsbereich Kennungen aus.
5. Wählen Sie auf der Seite Kennungen das Symbol + aus.
6. Wählen Sie auf der Seite Neue Kennung registrieren die Option App-IDs und dann Weiter aus.
7. Machen Sie auf der Seite Registrieren einer App-ID das Folgende:

    a. Geben Sie unter Beschreibung eine Beschreibung ein.

    b. Geben Sie unter Bundle-ID eine Kennung ein. Notieren Sie sich diese Bundle ID (Bundle-ID), da Sie diesen Wert benötigen, um Apple als Anbieter im Identitäten-Pool zu konfigurieren.

    c. Wählen Sie unter Funktionen die Option Mit Apple anmelden und dann Bearbeiten aus.

    d. Wählen Sie auf der Seite Anmeldung mit Apple: App-ID-Konfiguration die entsprechende Einstellung für Ihre App aus. Wählen Sie dann Speichern.

    e. Klicken Sie auf Continue.
8. Wählen Sie auf der Seite App-ID bestätigen die Option Registrieren aus.
9. Fahren Sie mit Schritt 10 fort, wenn Sie „Mit Apple anmelden“ in eine native iOS-Anwendung integrieren möchten. Schritt 11 ist für Anwendungen vorgesehen, die Sie in die Anmeldung mit Apple JS integrieren möchten.
10. Wählen Sie auf der Seite Identifiers (IDs) das Menü App IDs (App-IDs) und dann Service IDs (Service-IDs) aus. Klicken Sie auf das Pluszeichen (+).
11. Wählen Sie auf der Seite Neue Kennung registrieren die Option Services-IDs und dann Weiter aus.

12. Machen Sie auf der Seite Registrieren einer Service-ID das Folgende:

    a. Geben Sie unter Beschreibung eine Beschreibung ein.

    b. Geben Sie unter Kennungen eine Kennung ein. Notieren Sie sich die Service-ID, da Sie diesen Wert benötigen, um Apple als Anbieter in Ihrem Identitäten-Pool zu konfigurieren.

    c. Wählen Sie Mit Apple anmelden und dann Konfigurieren aus.

    d. Wählen Sie auf der Webauthentifizierungs-Konfiguration eine Primäre App-ID. Wählen Sie unter Website URLs (Website-URLs) das Symbol + aus. Geben Sie für Domänen und Subdomänen den Domänennamen Ihrer App ein. Geben Sie unter Return URLs (Rückgabe-URLs) die Rückruf-URL ein, an die die Autorisierung nach der Authentifizierung durch „Mit Apple anmelden“ weitergeleitet wird.

    e. Wählen Sie Weiter aus.

    f. Wählen Sie Weiter und dann Registrieren aus.

13. Wählen Sie im linken Navigationsbereich die Option Schlüssel aus.

14. Klicken Sie auf der Seite Schlüssel auf das Symbol +.

15. Machen Sie auf der Seite Registrieren eines neuen Schlüssels das Folgende:

    a. Geben Sie unter Schlüsselname einen Schlüsselnamen ein.

    b. Wählen Sie Mit Apple anmelden und dann Konfigurieren aus.

    c. Wählen Sie auf der Seite Schlüssel konfigurieren eine Primäre App-ID und dann Speichern aus.

    d. Wählen Sie Weiter und dann Registrieren aus.

> **Note**
>
> Informationen zur Integration von „Mit Apple anmelden“ in eine native iOS-Anwendung finden Sie unter Implementing User Authentication with Sign in with Apple.
> Informationen zur Integration von „Mit Apple anmelden“ in eine andere Plattform als natives iOS finden Sie unter Anmelden mit Apple JS.

Konfigurieren des externen Anbieters in der Amazon-Cognito-Verbundidentitäten-Konsole

Gehen Sie wie folgt vor, um Ihren externen Anbieter zu konfigurieren.

So fügen Sie eine Anmeldung mit dem Identitätsanbieter (IdP) Apple hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Mit Apple anmelden.
5. Geben Sie die Services-ID des OAuth-Projekts ein, das Sie mit Apple Developer erstellt haben. Weitere Informationen finden Sie unter Authentifizieren von Benutzern mit „Mit Apple anmelden“ in der Dokumentation zu „Mit Apple anmelden“ .
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

   - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

     i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

     ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.
7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-

Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Anmelden mit Apple als Anbieter in den Amazon-Cognito-Verbundidentitäten-CLI-Beispielen

In diesem Beispiel wird ein Identitäten-Pool mit dem Namen `MyIdentityPool` mit „Mit Apple anmelden“ als IdP erstellt.

```
aws cognito-identity create-identity-pool --identity-
pool-name MyIdentityPool --supported-login-providers
appleid.apple.com="sameple.apple.clientid"
```

Weitere Informationen finden Sie unter Erstellen eines Identitätspools.

Erstellen Sie eine Amazon-Cognito-Identität-ID

In diesem Beispiel wird eine Amazon-Cognito-ID generiert (oder abgerufen). Da dies eine öffentliche API ist, benötigen Sie keine Anmeldeinformationen, um diese API aufzurufen.

```
aws cognito-identity get-id --identity-pool-id SampleIdentityPoolId --
logins appleid.apple.com="SignInWithAppleIdToken"
```

Weitere Informationen finden Sie unter get-id.

Anfordern von Anmeldeinformationen für eine Amazon-Cognito-Identitäts-ID

In diesem Beispiel werden Anmeldeinformationen für die angegebene Identitäts-ID und für „Mit Apple anmelden“ zurückgegeben. Da dies eine öffentliche API ist, benötigen Sie keine Anmeldeinformationen, um diese API aufzurufen.

```
aws cognito-identity get-credentials-for-identity --identity-id
SampleIdentityId --logins appleid.apple.com="SignInWithAppleIdToken"
```

Weitere Informationen finden Sie unter get-credentials-for-identity

„Mit Apple anmelden“ verwenden: Android

Apple stellt kein SDK bereit, das „Mit Apple anmelden“ für Android unterstützt. Sie können stattdessen den Webflow in einer Webansicht verwenden.

- Um „Mit Apple anmelden“ in Ihrer Anwendung zu konfigurieren, folgen Sie in der Apple-Dokumentation der SeiteKonfigurieren Ihrer Webseite für „Mit Apple anmelden“.
- Um der Android-Benutzeroberfläche eine Schaltfläche Sign in with Apple (Mit Apple anmelden) hinzuzufügen, folgen Sie Anzeigen und Konfigurieren von „Mit Apple anmelden“-Schaltflächen in der Apple-Dokumentation.
- Um Benutzer sicher mit „Mit Apple anmelden“ zu authentifizieren, folgen Sie der Anleitung unter Authenticating Users with Sign in with Apple (Konfigurieren Ihrer Webseite für „Mit Apple anmelden“) in der Apple-Dokumentation.

Das Anmelden mit Apple verwendet ein Sitzungsobjekt, um seinen Status nachzuverfolgen. Amazon Cognito verwendet das ID-Token aus diesem Sitzungsobjekt, um den Benutzer zu authentifizieren, die eindeutige Kennung zu generieren und dem Benutzer bei Bedarf Zugriff auf andere AWS Ressourcen zu gewähren.

```
@Override
public void onSuccess(Bundle response) {
    String token = response.getString("id_token");
    Map<String, String> logins = new HashMap<String, String>();
    logins.put("appleid.apple.com", token);
    credentialsProvider.setLogins(logins);
}
```

„Mit Apple anmelden“ verwenden: iOS – Objective-C

Apple bietet SDK-Support für „Mit Apple anmelden“ in nativen iOS-Anwendungen. Um die Benutzerauthentifizierung mit „Mit Apple anmelden“ in nativen iOS-Geräten zu implementieren, folgen Sie Implementieren der Benutzerauthentifizierung mit „Mit Apple anmelden“ in der Apple-Dokumentation.

Amazon Cognito verwendet das ID-Token, um den Benutzer zu authentifizieren, die eindeutige Kennung zu generieren und dem Benutzer bei Bedarf Zugriff auf andere AWS Ressourcen zu gewähren.

```
(void)finishedWithAuth: (ASAuthorizationAppleIDCredential *)auth error: (NSError *)
 error {
        NSString *idToken = [ASAuthorizationAppleIDCredential
 objectForKey:@"identityToken"];
        credentialsProvider.logins = @{ "appleid.apple.com": idToken };
```

```
    }
```

„Mit Apple anmelden“ verwenden: iOS – Swift

Apple bietet SDK-Support für „Mit Apple anmelden“ in nativen iOS-Anwendungen. Um die Benutzerauthentifizierung mit „Mit Apple anmelden“ in nativen iOS-Geräten zu implementieren, folgen Sie Implementieren der Benutzerauthentifizierung mit „Mit Apple anmelden“ in der Apple-Dokumentation.

Amazon Cognito verwendet das ID-Token, um den Benutzer zu authentifizieren, die eindeutige Kennung zu generieren und dem Benutzer bei Bedarf Zugriff auf andere AWS Ressourcen zu gewähren.

Weitere Informationen zum Einrichten von „Mit Apple anmelden“ in iOS finden Sie unter Einrichten von „Mit Apple anmelden“.

```
func finishedWithAuth(auth: ASAuthorizationAppleIDCredential!, error: NSError!) {
    if error != nil {
      print(error.localizedDescription)
    }
    else {
      let idToken = auth.identityToken,
      credentialsProvider.logins = ["appleid.apple.com": idToken!]
    }
}
```

Verwenden Sie „Mit Apple anmelden“: JavaScript

Apple bietet kein SDK an, das die Anmeldung mit Apple für unterstützt JavaScript. Sie können stattdessen den Webflow in einer Webansicht verwenden.

- Um „Mit Apple anmelden“ in Ihrer Anwendung zu konfigurieren, folgen Sie in der Apple-Dokumentation der SeiteKonfigurieren Ihrer Webseite für „Mit Apple anmelden“.
- Wie du deiner JavaScript Benutzeroberfläche die Schaltfläche „Mit Apple anmelden“ hinzufügst, erfährst du unter „Anmeldung mit Apple-Tasten anzeigen und konfigurieren“ in der Apple-Dokumentation.
- Um Benutzer sicher mit „Mit Apple anmelden“ zu authentifizieren, folgen Sie der Anleitung unter Konfigurieren Ihrer Webseite für „Mit Apple anmelden“ in der Apple-Dokumentation.

Das Anmelden mit Apple verwendet ein Sitzungsobjekt, um seinen Status nachzuverfolgen. Amazon Cognito verwendet das ID-Token aus diesem Sitzungsobjekt, um den Benutzer zu authentifizieren, die eindeutige Kennung zu generieren und dem Benutzer bei Bedarf Zugriff auf andere AWS Ressourcen zu gewähren.

```
function signinCallback(authResult) {
     // Add the apple's id token to the Amazon Cognito credentials login map.
     AWS.config.credentials = new AWS.CognitoIdentityCredentials({
        IdentityPoolId: 'IDENTITY_POOL_ID',
        Logins: {
           'appleid.apple.com': authResult['id_token']
        }
     });

     // Obtain AWS credentials
     AWS.config.credentials.get(function(){
        // Access AWS resources here.
     });
}
```

„Mit Apple anmelden“ verwenden: Xamarin

Wir haben kein SDK, das „Mit Apple anmelden“ für Xamarin unterstützt. Sie können stattdessen den Webflow in einer Webansicht verwenden.

- Um „Mit Apple anmelden“ in Ihrer Anwendung zu konfigurieren, folgen Sie in der Apple-Dokumentation der SeiteKonfigurieren Ihrer Webseite für „Mit Apple anmelden“.
- Um der Xamarin-Benutzeroberfläche eine Schaltfläche Sign in with Apple (Mit Apple anmelden) hinzuzufügen, folgen Sie Anzeigen und Konfigurieren von „Mit Apple anmelden“-Schaltflächen in der Apple-Dokumentation.
- Um Benutzer sicher mit „Mit Apple anmelden“ zu authentifizieren, folgen Sie der Anleitung unter Konfigurieren Ihrer Webseite für „Mit Apple anmelden“ in der Apple-Dokumentation.

Das Anmelden mit Apple verwendet ein Sitzungsobjekt, um seinen Status nachzuverfolgen. Amazon Cognito verwendet das ID-Token aus diesem Sitzungsobjekt, um den Benutzer zu authentifizieren, die eindeutige Kennung zu generieren und dem Benutzer bei Bedarf Zugriff auf andere AWS Ressourcen zu gewähren.

Sobald Sie das Token erhalten haben, können Sie es in Ihren `CognitoAWSCredentials` einrichten:

```
credentials.AddLogin("appleid.apple.com", token);
```

Einrichtung eines OIDC-Anbieters als Identitätspool-IdP

OpenID Connect ist ein offener Standard für die Authentifizierung, der von einer Reihe von Anmeldungsanbietern unterstützt wird. Amazon Cognito unterstützt die Verknüpfung von Identitäten mit OpenID-Connect-Anbietern, die Sie über AWS Identity and Access Management konfigurieren.

Hinzufügen eines OpenID-Connect-Anbieters

Weitere Informationen darüber, wie Sie einen OpenID-Connect-Anbieter erstellen, finden Sie unter Erstellen von OpenID-Connect (IDC)-Identitätsanbietern in der AWS Identity and Access Management -Benutzeranleitung.

Zuordnen eines Anbieters mit Amazon Cognito

So fügen Sie einen OIDC-Identitätsanbieter (IdP) hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie OpenID Connect (OIDC) aus.
5. Wählen Sie einen OIDC-Identitätsanbieter aus dem IAM in Ihrem. IdPs AWS-Konto Wenn Sie einen neuen SAML-Anbieter hinzufügen möchten, wählen Sie Neuen Anbieter erstellen, um zur IAM-Konsole zu navigieren.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

    - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

        i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die

Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.

7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.

8. Wählen Sie Änderungen speichern aus.

Sie können mehrere OpenID Connect-Anbieter einem einzigen Identitätspool zuordnen.

Verwenden von OpenID Connect

Weitere Informationen darüber, wie Sie sich anmelden und ein ID-Token erhalten, finden Sie in der Dokumentation des Anbieters.

Nachdem Sie ein Token erhalten haben, fügen Sie es der Anmeldezuordnung hinzu. Verwenden Sie den URI Ihres Anbieters als Schlüssel.

Überprüfen eines OpenID-Connect-Tokens

Bei der ersten Integration in Amazon Cognito erhalten Sie möglicherweise eine `InvalidToken`-Ausnahme. Es ist wichtig zu verstehen, wie Amazon Cognito OpenID Connect (OIDC)-Token überprüft.

> **Note**
>
> Wie unter https://tools.ietf.org/html/rfc7523 angegeben, lässt Amazon Cognito eine Übergangsfrist von fünf Minuten zu, damit der Taktversatz zwischen Systemen korrigiert werden kann.

1. Der Parameter `iss` muss dem Schlüssel in der Anmeldezuordnung (z. B. login.provider.com) entsprechen.
2. Die Signatur muss gültig sein. Die Signatur müssen über einen öffentlichen RSA-Schlüssel verifizierbar sein.
3. Der Fingerabdruck des öffentlichen Schlüssels des Zertifikats stimmt mit dem Fingerabdruck überein, den Sie bei der Erstellung Ihres OIDC-Anbieters in IAM festgelegt haben.
4. Wenn der Parameter `azp` vorhanden ist, prüfen Sie diesen Wert anhand der in Ihrem OIDC-Anbieter aufgeführten Client-IDs.
5. Wenn der Parameter `azp` nicht vorhanden ist, gleichen Sie den Parameter `aud` mit den in Ihrem OIDC-Anbieter aufgeführten Client-IDs ab.

Die Website jwt.io ist eine wertvolle Ressource zum Decodieren von Token und Überprüfen dieser Werte.

Android

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("login.provider.com", token);
credentialsProvider.setLogins(logins);
```

iOS – Objective-C

```
credentialsProvider.logins = @{ "login.provider.com": token }
```

iOS – Swift

Um Amazon Cognito das OIDC-ID-Token bereitzustellen, implementieren Sie das `AWSIdentityProviderManager`-Protokoll.

Geben Sie bei der Implementierung der `logins`-Methode ein Wörterbuch zurück, das den von Ihnen konfigurierten OIDC-Anbieternamen enthält. Dieses Wörterbuch fungiert als Schlüssel und das aktuelle ID-Token des authentifizierten Benutzers als Wert, wie im folgenden Codebeispiel gezeigt.

```
class OIDCProvider: NSObject, AWSIdentityProviderManager {
    func logins() -> AWSTask<NSDictionary> {
        let completion = AWSTaskCompletionSource<NSString>()
        getToken(tokenCompletion: completion)
        return completion.task.continueOnSuccessWith { (task) -> AWSTask<NSDictionary>?
 in
            //login.provider.name is the name of the OIDC provider as setup in the
 Amazon Cognito console
            return AWSTask(result:["login.provider.name":task.result!])
        } as! AWSTask<NSDictionary>

    }

    func getToken(tokenCompletion: AWSTaskCompletionSource<NSString>) -> Void {
        //get a valid oidc token from your server, or if you have one that hasn't
 expired cached, return it

        //TODO code to get token from your server
        //...

        //if error getting token, set error appropriately
        tokenCompletion.set(error:NSError(domain: "OIDC Login", code: -1 , userInfo:
 ["Unable to get OIDC token" : "Details about your error"]))
        //else
        tokenCompletion.set(result:"result from server id token")
    }
}
```

Wenn Sie den instanziierenAWSCognitoCredentialsProvider, übergeben Sie die Klasse, die implementiert wird, AWSIdentityProviderManager als Wert von im Konstruktor. identityProviderManager Weitere Informationen finden Sie auf der AWSCognitoCredentialsProviderReferenzseite und wählen Sie initWithRegionType::. identityPoolId identityProviderManager

JavaScript

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
 IdentityPoolId: 'IDENTITY_POOL_ID',
```

```
 Logins: {
    'login.provider.com': token
 }
});
```

Unity

```
credentials.AddLogin("login.provider.com", token);
```

Xamarin

```
credentials.AddLogin("login.provider.com", token);
```

Einrichtung eines SAML-Anbieters als Identitätspool-IdP

Amazon Cognito unterstützt die Authentifizierung mit Identitätsanbietern (IdPs) über Security Assertion Markup Language 2.0 (SAML 2.0). Sie können einen IdP verwenden, der SAML mit Amazon Cognito unterstützt, um einen einfachen Onboarding-Ablauf für Ihre Benutzer bereitzustellen. Ihr SAML-unterstützender IDP gibt die IAM-Rollen an, die Ihre Benutzer annehmen können. Auf diese Weise können verschiedene Benutzer verschiedene Berechtigungssätze erhalten.

Konfigurieren des Identitäten-Pools für einen SAML-IdP

In den folgenden Schritten wird beschrieben, wie Sie Ihren Identitäten-Pool für die Verwendung eines SAML-basierten IdP konfigurieren.

> **Note**
>
> Bevor Sie Ihren Identitäten-Pool zur Unterstützung eines SAML-Anbieters konfigurieren, nehmen Sie zunächst eine Konfiguration des SAML-IdP in der IAM-Konsole vor. Weitere Informationen finden Sie unter Integrieren von Drittanbieter-SAML-Lösungsanbietern mit AWS im IAM-Benutzerhandbuch.

So fügen Sie einen SAML-Identitätsanbieter (IdP) hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.

2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie SAML.
5. Wählen Sie einen SAML-Identitätsanbieter aus dem IAM in Ihrem. IdPs AWS-Konto Wenn Sie einen neuen SAML-Anbieter hinzufügen möchten, wählen Sie Neuen Anbieter erstellen, um zur IAM-Konsole zu navigieren.
6. Um die Rolle festzulegen, die Amazon Cognito bei der Ausgabe von Anmeldeinformationen an Benutzer anfordert, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Rolleneinstellungen.

   - Sie können Benutzern dieses IdPs die Standardrolle zuweisen, die Sie bei der Konfiguration Ihrer authentifizierten Rolle eingerichtet haben, oder die Rolle mit Regeln wählen.

     i. Wenn Sie Rolle mit Regeln wählen ausgewählt haben, geben Sie die Quell-Anforderung aus der Benutzerauthentifizierung, den Operator, mit dem Sie die Anforderung vergleichen möchten, den Wert, der zu einer Übereinstimmung mit dieser Rollenauswahl führt, und die Rolle ein, die Sie zuweisen möchten, wenn die Rollenzuweisung übereinstimmt. Wählen Sie Weitere hinzufügen aus, um eine zusätzliche Regel zu erstellen, die auf einer anderen Bedingung basiert.

     ii. Wählen Sie eine Rollenauflösung. Wenn die Anforderungen Ihres Benutzers nicht Ihren Regeln entsprechen, können Sie Anmeldeinformationen verweigern oder Anmeldeinformationen für Ihre Authentifizierte Rolle ausgeben.
7. Um die Prinzipal-Tags zu ändern, die Amazon Cognito Benutzern zuweist, wenn es Anmeldeinformationen an Benutzer ausgibt, die sich bei diesem Anbieter authentifiziert haben, konfigurieren Sie die Attribute für die Zugriffskontrolle.

   a. Um keine Prinzipal-Tags anzuwenden, wählen Sie Inaktiv aus.

   b. Wählen Sie Standardzuordnungen verwenden, um Prinzipal-Tags auf der Grundlage von `sub`- und `aud`-Anforderungen anzuwenden.

   c. Um Ihr eigenes benutzerdefiniertes Schema von Attributen für Prinzipal-Tags zu erstellen, wählen Sie Benutzerdefinierte Zuordnungen verwenden. Geben Sie dann einen Tag-Schlüssel ein, den Sie aus jeder Anforderung beziehen möchten, die Sie in einem Tag repräsentieren möchten.
8. Wählen Sie Änderungen speichern aus.

Konfigurieren Ihres SAML-IDP

Nachdem Sie den SAML-Anbieter erstellt haben, konfigurieren Sie Ihren SAML-IdP, um die Vertrauensstellung für die vertrauende Seite zwischen Ihrem IdP und AWS hinzuzufügen. Bei vielen können Sie eine URL angeben IdPs, die der IdP verwenden kann, um Informationen und Zertifikate der vertrauenden Partei aus einem XML-Dokument zu lesen. Für AWS können Sie https://signin.aws.amazon.com/static/saml-metadata.xml verwenden. Der nächste Schritt besteht darin, die SAML-Assertion-Antwort Ihres IdP so zu konfigurieren, dass die benötigten Ansprüche aufgefüllt werden. AWS Weitere Informationen zur Konfiguration der Ansprüche finden Sie unter Konfigurieren von SAML-Assertions für die Authentifizierungsantwort.

Wenn Ihr SAML-IdP mehr als ein Signaturzertifikat in SAML-Metadaten enthält, stellt Ihr Benutzerpool bei der Anmeldung fest, dass die SAML-Assertion gültig ist, sofern sie mit einem Zertifikat in den SAML-Metadaten übereinstimmt.

Anpassen Ihrer Benutzerrolle mit SAML

Wenn Sie SAML mit Amazon Cognito Identity verwenden, können Sie die Rolle für den Endbenutzer anpassen. Amazon Cognito unterstützt nur den erweiterten Ablauf mit dem SAML-basierten IDP. Sie brauchen keine authentifizierte oder nicht authentifizierte Rolle für den Identitäten-Pool anzugeben, um einen SAML-basierten IdP zu verwenden. Das Anspruch-Attribut `https://aws.amazon.com/SAML/Attributes/Role` legt ein oder mehrere Paare von durch Komma getrennten Rollen und Anbieter-ARNs fest. Dies sind die Rollen, die der Benutzer annehmen kann. Sie können den SAML-IdP so konfigurieren, dass er die Rollenattribute basierend auf den Benutzerattributinformationen einrichtet, die vom IdP verfügbar sind. Wenn Sie mehrere Rollen in der SAML-Assertion erhalten, füllen Sie den optionalen `customRoleArn`-Parameter aus, wenn Sie `getCredentialsForIdentity` aufrufen. Der Benutzer nimmt diesen `customRoleArn` an, wenn die Rolle mit einer Rolle im Anspruch der SAML-Assertion übereinstimmt.

Authentifizieren von Benutzern mit einem SAML-IdP

Um eine Verbindung mit dem SAML-basierten IdP herzustellen, ermitteln Sie die URL, unter der der Benutzer die Anmeldung initiiert. AWS Der Verband verwendet eine vom IDP initiierte Anmeldung. In AD FS 2.0 hat die URL die Form `https://<fqdn>/adfs/ls/IdpInitiatedSignOn.aspx?loginToRp=urn:amazon:webservices`.

Zur Bereitstellung von Unterstützung für Ihren SAML-IdP in Amazon Cognito müssen Sie zuerst Benutzer mit Ihrem SAML-Identitätsanbieter von Ihrer iOS- oder Android-App aus authentifizieren. Der Code, mit dem Sie den SAML-IDP integrieren und authentifizieren, ist für den SAML-Anbieter

spezifisch. Nachdem Sie Ihren Benutzer authentifiziert haben, können Sie die resultierende SAML-Assertion über Amazon-Cognito-APIs an Amazon Cognito Identity weiterleiten.

Sie können keine SAML-Assertion der `Logins`-Zuordnung Ihrer Identitätspool-API-Anfrage wiederholen oder erneut wiedergeben. Eine erneut wiedergegebene SAML-Assertion hat eine Assertion-ID, die die ID einer früheren API-Antwort dupliziert. Zu den API-Vorgängen, die eine SAML-Assertion in der `Logins` Map akzeptieren können GetId, gehören, GetCredentialsForIdentity, GetOpenIdTokenund ID. GetOpen TokenForDeveloperIdentity Sie können eine SAML-Assertion-ID einmal pro API-Anfrage in einem Identitätspool-Authentifizierungsablauf erneut wiedergeben. Sie können beispielsweise dieselbe SAML-Assertion in einer `GetId`-Anfrage und in einer nachfolgenden `GetCredentialsForIdentity`-Anforderung angeben, jedoch nicht in einer zweiten `GetId`-Anforderung.

Android

Wenn Sie das Android-SDK verwenden, können Sie die Anmeldezuordnung wie folgt mit der SAML-Assertion ausfüllen.

```
Map logins = new HashMap();
logins.put("arn:aws:iam::aws account id:saml-provider/name", "base64 encoded assertion
 response");
// Now this should be set to CognitoCachingCredentialsProvider object.
CognitoCachingCredentialsProvider credentialsProvider = new
 CognitoCachingCredentialsProvider(context, identity pool id, region);
credentialsProvider.setLogins(logins);
// If SAML assertion contains multiple roles, resolve the role by setting the custom
 role
credentialsProvider.setCustomRoleArn("arn:aws:iam::aws account id:role/
customRoleName");
// This should trigger a call to the Amazon Cognito service to get the credentials.
credentialsProvider.getCredentials();
```

iOS

Wenn Sie das iOS-SDK verwenden, können Sie die SAML-Assertion wie folgt in `AWSIdentityProviderManager` bereitstellen.

```
- (AWSTask<NSDictionary<NSString*,NSString*> *> *) logins {
    //this is hardcoded for simplicity, normally you would asynchronously go to your
 SAML provider
    //get the assertion and return the logins map using a AWSTaskCompletionSource
```

```
    return [AWSTask taskWithResult:@{@"arn:aws:iam::aws account id:saml-provider/
name":@"base64 encoded assertion response"}];
}

// If SAML assertion contains multiple roles, resolve the role by setting the custom
 role.
// Implementing this is optional if there is only one role.
- (NSString *)customRoleArn {
    return @"arn:aws:iam::accountId:role/customRoleName";
}
```

Entwicklerauthentifizierte Identitäten (Identitätspools)

Amazon Cognito unterstützt entwicklerauthentifizierte Identitäten zusätzlich zum Web-Identitätsverbund über Facebook als Identitätspools einrichten (IdP), Google als Identitätspool-IdP einrichten, Login with Amazon als Identitätspools (IdP) einrichten und Mit Apple anmelden als Identitätspool-IdP einrichten. Mit entwicklerauthentifizierten Identitäten können Sie Benutzer über Ihren eigenen vorhandenen Authentifizierungsprozess registrieren und authentifizieren und weiterhin Amazon Cognito verwenden, um Benutzerdaten zu synchronisieren und auf AWS Ressourcen zuzugreifen. Die Verwendung von entwicklerauthentifizierten Identitäten beinhaltet die Interaktion zwischen dem Endbenutzergerät, Ihrem Backend für die Authentifizierung und Amazon Cognito. Weitere Informationen finden Sie unter Grundlegendes zur Amazon Cognito-Authentifizierung, Teil 2: Entwicklerauthentifizierte Identitäten im - AWS Blog.

Erläuterungen zum Authentifizierungsfluss

Der GetOpenIdTokenForDeveloperIdentity API-Vorgang kann die Entwicklerauthentifizierung sowohl für die erweiterte als auch für die grundlegende Authentifizierung initiieren. Diese API authentifiziert eine Anforderung mit Administratoranmeldeinformationen. Die `Logins` Zuordnung ist ein Name eines Identitätspool-Entwickleranbieters wie `login.mydevprovider` in Kombination mit einer benutzerdefinierten Kennung.

Beispiel:

```
"Logins": {
        "login.mydevprovider": "my developer identifier"
    }
```

Erweiterte Authentifizierung

Rufen Sie die GetCredentialsForIdentity API-Operation mit einer `Logins` Zuordnung mit dem Namen `cognito-identity.amazonaws.com` und einem Wert des Tokens aus auf`GetOpenIdTokenForDeveloperIdentity`.

Beispiel:

```
"Logins": {
        "cognito-identity.amazonaws.com": "eyJra12345EXAMPLE"
    }
```

`GetCredentialsForIdentity` mit entwicklerauthentifizierten Identitäten gibt temporäre Anmeldeinformationen für die standardmäßig authentifizierte Rolle des Identitätspools zurück.

Basisauthentifizierung

Rufen Sie die AssumeRoleWithWebIdentity -API-Operation auf und fordern Sie die `RoleArn` aller IAM-Rollen an, für die eine entsprechende Vertrauensstellung definiert ist. Setzen Sie den Wert von `WebIdentityToken` auf das Token, das Sie von erhalten haben`GetOpenIdTokenForDeveloperIdentity`.

Informationen zum Authentifizierungsablauf für entwicklerauthentifizierte Identitäten und dazu, wie sie sich von Identitäten externer Anbieter unterscheiden, finden Sie unter Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf.

Definieren eines Entwickleranbieternamens und Zuordnen zu einem Identitäten-Pool

Zur Verwendung von entwicklerauthentifizierten Identitäten muss dem Entwickleranbieter ein Identitätspool zugeordnet werden. Führen Sie dazu die folgenden Schritte aus:

So fügen Sie einen benutzerdefinierten Entwickleranbieter hinzu

1. Wählen Sie Identitätspools in der Amazon-Cognito-Konsole aus. Wählen Sie einen Identitätspool.
2. Wählen Sie die Registerkarte Datenzugriff aus.
3. Wählen Sie Identitätsanbieter hinzufügen aus.
4. Wählen Sie Benutzerdefinierter Entwickleranbieter aus.
5. Geben Sie einen Namen für den Entwickleranbieter ein. Wenn Sie Ihren Entwickleranbieter eingegeben haben, können Sie ihn nicht mehr ändern oder löschen.

6. Wählen Sie Änderungen speichern aus.

Hinweis: Sobald der Anbieternamen festgelegt wurde, kann er nicht mehr geändert werden.

Weitere Anweisungen zur Verwendung der Amazon-Cognito-Konsole finden Sie unter Verwenden der Amazon-Cognito-Konsole.

Implementieren eines Identitätsanbieters

Android

Zur Verwendung von entwicklerauthentifizierten Identitäten implementieren Sie eine eigene Identitätsanbieterklasse, die `AWSAbstractCognitoIdentityProvider` erweitert. Die Identitätsanbieterklasse sollte ein Antwortobjekt zurückgeben, das das Token als Attribut enthält.

Es folgt ein einfaches Beispiel für einen Identitätsanbieter.

```
public class DeveloperAuthenticationProvider extends
 AWSAbstractCognitoDeveloperIdentityProvider {

  private static final String developerProvider = "<Developer_provider_name>";

  public DeveloperAuthenticationProvider(String accountId, String identityPoolId,
 Regions region) {
    super(accountId, identityPoolId, region);
    // Initialize any other objects needed here.
  }

  // Return the developer provider name which you choose while setting up the
  // identity pool in the &COG; Console

  @Override
  public String getProviderName() {
    return developerProvider;
  }

  // Use the refresh method to communicate with your backend to get an
  // identityId and token.

  @Override
  public String refresh() {
```

```
    // Override the existing token
    setToken(null);

    // Get the identityId and token by making a call to your backend
    // (Call to your backend)

    // Call the update method with updated identityId and token to make sure
    // these are ready to be used from Credentials Provider.

    update(identityId, token);
    return token;

  }

  // If the app has a valid identityId return it, otherwise get a valid
  // identityId from your backend.

  @Override
  public String getIdentityId() {

    // Load the identityId from the cache
    identityId = cachedIdentityId;

    if (identityId == null) {
       // Call to your backend
    } else {
       return identityId;
    }

  }
}
```

Zur Verwendung dieses Anbieters müssen Sie ihn an übergeben `CognitoCachingCredentialsProvider`. Ein Beispiel:

```
DeveloperAuthenticationProvider developerProvider = new
 DeveloperAuthenticationProvider( null, "IDENTITYPOOLID", context, Regions.USEAST1);
CognitoCachingCredentialsProvider credentialsProvider = new
 CognitoCachingCredentialsProvider( context, developerProvider, Regions.USEAST1);
```

iOS – Objective-C

Zur Verwendung von entwicklerauthentifizierten Identitäten implementieren Sie eine eigene Identitätsanbieterklasse, die AWSCognitoCredentialsProviderHelper erweitert. Die Identitätsanbieterklasse sollte ein Antwortobjekt zurückgeben, das das Token als Attribut enthält.

```
@implementation DeveloperAuthenticatedIdentityProvider
/*
 * Use the token method to communicate with your backend to get an
 * identityId and token.
 */

- (AWSTask <NSString*> *) token {
    //Write code to call your backend:
    //Pass username/password to backend or some sort of token to authenticate user
    //If successful, from backend call getOpenIdTokenForDeveloperIdentity with logins
 map
    //containing "your.provider.name":"enduser.username"
    //Return the identity id and token to client
    //You can use AWSTaskCompletionSource to do this asynchronously

    // Set the identity id and return the token
    self.identityId = response.identityId;
    return [AWSTask taskWithResult:response.token];
}

@end
```

Zur Verwendung dieses Identitätsanbieters müssen Sie ihn wie in folgendem Beispiel gezeigt an `AWSCognitoCredentialsProvider` übergeben:

```
DeveloperAuthenticatedIdentityProvider * devAuth =
 [[DeveloperAuthenticatedIdentityProvider alloc]
 initWithRegionType:AWSRegionYOUR_IDENTITY_POOL_REGION
                                         identityPoolId:@"YOUR_IDENTITY_POOL_ID"
                                        useEnhancedFlow:YES
                                identityProviderManager:nil];
AWSCognitoCredentialsProvider *credentialsProvider = [[AWSCognitoCredentialsProvider
 alloc]

 initWithRegionType:AWSRegionYOUR_IDENTITY_POOL_REGION
                                                        identityProvider:devAuth];
```

Wenn Sie sowohl nicht authentifizierte Identitäten als auch entwicklerauthentifizierte Identitäten unterstützen möchten, überschreiben Sie die Methode `logins` in der `AWSCognitoCredentialsProviderHelper`-Implementierung.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else{
        return [super logins];
    }
}
```

Wenn Sie sowohl entwicklerauthentifizierte Identitäten als auch Social-Identity-Anbieter unterstützen möchten, müssen Sie den verwendeten aktuellen Anbieter in der `logins`-Implementierung von `AWSCognitoCredentialsProviderHelper` verwalten.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else if (/*logic to determine if user is Facebook*/){
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook :
 [FBSDKAccessToken currentAccessToken] }];
    }else {
        return [super logins];
    }
}
```

iOS – Swift

Zur Verwendung von entwicklerauthentifizierten Identitäten implementieren Sie eine eigene Identitätsanbieterklasse, die AWSCognitoCredentialsProviderHelper erweitert. Die Identitätsanbieterklasse sollte ein Antwortobjekt zurückgeben, das das Token als Attribut enthält.

```
import AWSCore
/*
 * Use the token method to communicate with your backend to get an
 * identityId and token.
 */
class DeveloperAuthenticatedIdentityProvider : AWSCognitoCredentialsProviderHelper {
    override func token() -> AWSTask<NSString> {
    //Write code to call your backend:
```

```
    //pass username/password to backend or some sort of token to authenticate user, if
 successful,
    //from backend call getOpenIdTokenForDeveloperIdentity with logins map containing
 "your.provider.name":"enduser.username"
    //return the identity id and token to client
    //You can use AWSTaskCompletionSource to do this asynchronously

    // Set the identity id and return the token
    self.identityId = resultFromAbove.identityId
    return AWSTask(result: resultFromAbove.token)
}
```

Zur Verwendung dieses Identitätsanbieters müssen Sie ihn wie in folgendem Beispiel gezeigt an `AWSCognitoCredentialsProvider` übergeben:

```
let devAuth =
 DeveloperAuthenticatedIdentityProvider(regionType: .YOUR_IDENTITY_POOL_REGION,
 identityPoolId: "YOUR_IDENTITY_POOL_ID", useEnhancedFlow: true,
 identityProviderManager:nil)
let credentialsProvider =
 AWSCognitoCredentialsProvider(regionType: .YOUR_IDENTITY_POOL_REGION,
 identityProvider:devAuth)
let configuration = AWSServiceConfiguration(region: .YOUR_IDENTITY_POOL_REGION,
 credentialsProvider:credentialsProvider)
AWSServiceManager.default().defaultServiceConfiguration = configuration
```

Wenn Sie sowohl nicht authentifizierte Identitäten als auch entwicklerauthentifizierte Identitäten unterstützen möchten, überschreiben Sie die Methode `logins` in der `AWSCognitoCredentialsProviderHelper`-Implementierung.

```
override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else {
        return super.logins()
    }
}
```

Wenn Sie sowohl entwicklerauthentifizierte Identitäten als auch Social-Identity-Anbieter unterstützen möchten, müssen Sie den verwendeten aktuellen Anbieter in der `logins`-Implementierung von `AWSCognitoCredentialsProviderHelper` verwalten.

```
override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else if (/*logic to determine if user is Facebook*/){
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error:NSError(domain: "Facebook Login", code: -1 , userInfo:
 ["Facebook" : "No current Facebook access token"]))
    }else {
        return super.logins()
    }
}
```

JavaScript

Nachdem Sie eine Identitäts-ID und ein Sitzungs-Token von Ihrem Backend erhalten haben, übergeben Sie diese an den `AWS.CognitoIdentityCredentials`-Anbieter. Ein Beispiel:

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
   IdentityPoolId: 'IDENTITY_POOL_ID',
   IdentityId: 'IDENTITY_ID_RETURNED_FROM_YOUR_PROVIDER',
   Logins: {
      'cognito-identity.amazonaws.com': 'TOKEN_RETURNED_FROM_YOUR_PROVIDER'
   }
});
```

Unity

Zur Verwendung von entwicklerauthentifizierten Identitäten müssen Sie `CognitoAWSCredentials` erweitern und die Methode `RefreshIdentity` überschreiben, um die Benutzeridentitäts-ID und das Token vom Backend abzurufen und zurückzugeben. Nachstehend finden Sie ein einfaches Beispiel für einen Identitätsanbieter, der ein hypothetisches Backend unter „example.com“ kontaktiert:

```
using UnityEngine;
using System.Collections;
using Amazon.CognitoIdentity;
using System.Collections.Generic;
using ThirdParty.Json.LitJson;
using System;
using System.Threading;
```

```
public class DeveloperAuthenticatedCredentials : CognitoAWSCredentials
{
    const string PROVIDER_NAME = "example.com";
    const string IDENTITY_POOL = "IDENTITY_POOL_ID";
    static readonly RegionEndpoint REGION = RegionEndpoint.USEast1;

    private string login = null;

    public DeveloperAuthenticatedCredentials(string loginAlias)
        : base(IDENTITY_POOL, REGION)
    {
        login = loginAlias;
    }

    protected override IdentityState RefreshIdentity()
    {
        IdentityState state = null;
        ManualResetEvent waitLock = new ManualResetEvent(false);
        MainThreadDispatcher.ExecuteCoroutineOnMainThread(ContactProvider((s) =>
        {
            state = s;
            waitLock.Set();
        }));
        waitLock.WaitOne();
        return state;
    }

    IEnumerator ContactProvider(Action<IdentityState> callback)
    {
        WWW www = new WWW("http://example.com/?username="+login);
        yield return www;
        string response = www.text;

        JsonData json = JsonMapper.ToObject(response);

        //The backend has to send us back an Identity and a OpenID token
        string identityId = json["IdentityId"].ToString();
        string token = json["Token"].ToString();

        IdentityState state = new IdentityState(identityId, PROVIDER_NAME, token,
 false);
        callback(state);
    }
```

```
}
```

Der obige Code verwendet ein Thread-Dispatcher-Objekt zum Aufrufen einer Coroutine. Wenn Sie in Ihrem Projekt nicht die Möglichkeit dazu haben, können Sie das folgende Skript in Ihren Szenen verwenden:

```
using System;
using UnityEngine;
using System.Collections;
using System.Collections.Generic;

public class MainThreadDispatcher : MonoBehaviour
{
    static Queue<IEnumerator> _coroutineQueue = new Queue<IEnumerator>();
    static object _lock = new object();

    public void Update()
    {
        while (_coroutineQueue.Count > 0)
        {
            StartCoroutine(_coroutineQueue.Dequeue());
        }
    }

    public static void ExecuteCoroutineOnMainThread(IEnumerator coroutine)
    {
        lock (_lock) {
            _coroutineQueue.Enqueue(coroutine);
        }
    }
}
```

Xamarin

Zur Verwendung von entwicklerauthentifizierten Identitäten müssen Sie `CognitoAWSCredentials` erweitern und die Methode `RefreshIdentity` überschreiben, um die Benutzeridentitäts-ID und das Token vom Backend abzurufen und zurückzugeben. Nachstehend finden Sie ein einfaches Beispiel für einen Identitätsanbieter, der ein hypothetisches Backend unter „example.com“ kontaktiert:

```
public class DeveloperAuthenticatedCredentials : CognitoAWSCredentials
{
```

```
    const string PROVIDER_NAME = "example.com";
    const string IDENTITY_POOL = "IDENTITY_POOL_ID";
    static readonly RegionEndpoint REGION = RegionEndpoint.USEast1;
    private string login = null;

    public DeveloperAuthenticatedCredentials(string loginAlias)
        : base(IDENTITY_POOL, REGION)
    {
        login = loginAlias;
    }

    protected override async Task<IdentityState> RefreshIdentityAsync()
    {
        IdentityState state = null;
        //get your identity and set the state
        return state;
    }
}
```

Aktualisieren der Anmeldezuweisung (nur Android und iOS)

Android

Nach erfolgreicher Authentifizierung des Benutzers mit Ihrem Authentifizierungssystem aktualisieren Sie die Anmeldezuweisung mit dem Entwickleranbieter-Namen und einer Entwicklerbenutzer-ID. Dies ist eine alphanumerische Zeichenfolge, die einen Benutzer in Ihrem Authentifizierungssystem eindeutig identifiziert. Stellen Sie sicher, dass nach der Aktualisierung der Anmeldezuweisung die Methode `refresh` da die `identityId` sich möglicherweise geändert hat:

```
HashMap<String, String> loginsMap = new HashMap<String, String>();
loginsMap.put(developerAuthenticationProvider.getProviderName(),
 developerUserIdentifier);

credentialsProvider.setLogins(loginsMap);
credentialsProvider.refresh();
```

iOS – Objective-C

Das iOS-SDK ruft die Methode `logins` nur auf, um die aktuelle Anmeldezuweisung abzurufen, wenn keine Anmeldeinformationen vorliegen oder diese abgelaufen sind. Wenn Sie das Abrufen neuer Anmeldeinformationen im SDK erzwingen möchten (weil beispielsweise ein nicht authentifizierter

Endbenutzer jetzt authentifiziert ist und Sie Anmeldeinformationen für den authentifizierten Benutzer benötigen), rufen Sie `clearCredentials` in `credentialsProvider` auf.

```
[credentialsProvider clearCredentials];
```

iOS – Swift

Das iOS-SDK ruft die Methode `logins` nur auf, um die aktuelle Anmeldezuweisung abzurufen, wenn keine Anmeldeinformationen vorliegen oder diese abgelaufen sind. Wenn Sie das Abrufen neuer Anmeldeinformationen im SDK erzwingen möchten (weil beispielsweise ein nicht authentifizierter Endbenutzer jetzt authentifiziert ist und Sie Anmeldeinformationen für den authentifizierten Benutzer benötigen), rufen Sie `clearCredentials` in `credentialsProvider` auf.

```
credentialsProvider.clearCredentials()
```

Aufrufen eines Tokens (Serverseite)

Sie erhalten ein Token, indem Sie aufrufenGetOpenIdTokenForDeveloperIdentity. Diese API muss von Ihrem Backend aus mit AWS Entwickleranmeldeinformationen aufgerufen werden. Sie darf nicht über das Client-SDK aufgerufen werden. Die API empfängt die Cognito-Identitätspool-ID, eine Anmeldezuweisung mit Ihrem Identitätsanbieter-Namen als Schlüssel und der ID als Wert sowie optional eine Cognito-Identitäts-ID (d. h. Sie ändern einen nicht authentifizierten in einen authentifizierten Benutzer). Die ID kann der Benutzername Ihres Benutzers, eine E-Mail-Adresse oder ein numerischer Wert sein. Die API beantwortet den Aufruf mit einer eindeutigen Cognito-ID für Ihren Benutzer und einem OpenID Connect-Token für den Endbenutzer.

Im Folgenden noch einige Hinweise zu dem durch zurückgegebenen Token `GetOpenIdTokenForDeveloperIdentity`:

- Sie können eine benutzerdefinierte Ablaufzeit für das Token angeben, sodass Sie es zwischenspeichern können. Wenn Sie keine benutzerdefinierte Ablaufzeit bereitstellen, ist das Token für 15 Minuten gültig.
- Die maximale Token-Dauer, die Sie festlegen können, ist 24 Stunden.
- Beachten Sie, dass eine Verlängerung der Token-Dauer zu Sicherheitsproblemen führen kann. Wenn ein Angreifer dieses Token erhält, kann er es für die Token-Dauer gegen AWS Anmeldeinformationen für den Endbenutzer eintauschen.

Der folgende Java-Codeausschnitt zeigt, wie Sie einen Amazon-Cognito-Client initiieren und ein Token für eine entwicklerauthentifizierte Identität abrufen.

```
// authenticate your end user as appropriate
// ....

// if authenticated, initialize a cognito client with your AWS developer credentials
AmazonCognitoIdentity identityClient = new AmazonCognitoIdentityClient(
  new BasicAWSCredentials("access_key_id", "secret_access_key")
);

// create a new request to retrieve the token for your end user
GetOpenIdTokenForDeveloperIdentityRequest request =
  new GetOpenIdTokenForDeveloperIdentityRequest();
request.setIdentityPoolId("YOUR_COGNITO_IDENTITY_POOL_ID");

request.setIdentityId("YOUR_COGNITO_IDENTITY_ID"); //optional, set this if your client
 has an
                                                    //identity ID that you want to link
 to this
                                                    //developer account

// set up your logins map with the username of your end user
HashMap<String,String> logins = new HashMap<>();
logins.put("YOUR_IDENTITY_PROVIDER_NAME","YOUR_END_USER_IDENTIFIER");
request.setLogins(logins);

// optionally set token duration (in seconds)
request.setTokenDuration(60 * 15l);
GetOpenIdTokenForDeveloperIdentityResult response =
  identityClient.getOpenIdTokenForDeveloperIdentity(request);

// obtain identity id and token to return to your client
String identityId = response.getIdentityId();
String token = response.getToken();

//code to return identity id and token to client
//...
```

Nach den oben aufgeführten Schritten können Sie entwicklerauthentifizierte Identitäten in Ihre App integrieren. Bei Problemen oder Fragen können Sie gerne einen Beitrag in unseren Foren posten.

Verbinden mit einer vorhandenen Social Identity

Alle Verknüpfungen von Anbietern müssen über Ihr Backend erfolgen, wenn Sie entwicklerauthentifizierte Identitäten verwenden. Um eine benutzerdefinierte Identität mit der Social Identity eines Benutzers (Login with Amazon, Mit Apple anmelden, Facebook oder Google) zu verbinden, fügen Sie das Identitätsanbieter-Token der Anmeldezuweisung hinzu, wenn Sie aufrufenGetOpenIdTokenForDeveloperIdentity. Um dies zu ermöglichen, müssen Sie zusätzlich das Token des Social-Identity-Anbieter an den Endbenutzer übergeben, wenn Sie das Backend zur Authentifizierung des Endbenutzers über das Client-SDK aufrufen.

Wenn Sie beispielsweise eine benutzerdefinierte Identität mit Facebook verknüpfen möchten, fügen Sie der Anmeldezuweisung beim Aufrufen von zusätzlich zu Ihrer Identitätsanbieter-ID das Facebook-Token hinzu `GetOpenIdTokenForDeveloperIdentity`.

```
logins.put("YOUR_IDENTITY_PROVIDER_NAME","YOUR_END_USER_IDENTIFIER");
logins.put("graph.facebook.com","END_USERS_FACEBOOK_ACCESSTOKEN");
```

Unterstützen des Anbieterwechsels

Android

Ihre Anwendung erfordert möglicherweise die Unterstützung sowohl von nicht authentifizierten Identitäten oder authentifizierten Identitäten unter Verwendung öffentlicher Anbieter (Anmeldung mit Amazon, Apple, Facebook oder Google) als auch von entwicklerauthentifizierten Identitäten. Der wesentliche Unterschied zwischen entwicklerauthentifizierten Identitäten und anderen Identitäten (nicht authentifizierten Identitäten und authentifizierten Identitäten unter Verwendung von öffentlichen Anbietern) ist die Art, in der die Identitäts-ID und das Token abgerufen werden. Für andere Identitäten interagiert die mobile Anwendung direkt mit Amazon Cognito, anstatt Ihr Authentifizierungssystem zu kontaktieren. Die mobile Anwendung muss daher in Abhängigkeit von der Auswahl durch den App-Benutzer zwei verschiedene Abläufe unterstützen können. Hierzu müssen Sie einige Änderungen am benutzerdefinierten Identitätsanbieter vornehmen.

Die `refresh`-Methode überprüft die Anmeldezuordnung. Ist diese nicht leer, sondern enthält einen Schlüssel mit dem Entwickleranbieter-Namen, müssen Sie Ihr Backend aufrufen. Rufen Sie andernfalls die getIdentityId Methode auf und geben Sie null zurück.

```
public String refresh() {
```

```
    setToken(null);

    // If the logins map is not empty make a call to your backend
    // to get the token and identityId
    if (getProviderName() != null &&
        !this.loginsMap.isEmpty() &&
        this.loginsMap.containsKey(getProviderName())) {

        /**
         * This is where you would call your backend
         **/

        // now set the returned identity id and token in the provider
        update(identityId, token);
        return token;

    } else {
        // Call getIdentityId method and return null
        this.getIdentityId();
        return null;
    }
}
```

Entsprechend weist die Methode `getIdentityId` zwei Abläufe auf, die vom Inhalt der Anmeldezuweisung abhängen:

```
public String getIdentityId() {

    // Load the identityId from the cache
    identityId = cachedIdentityId;

    if (identityId == null) {

        // If the logins map is not empty make a call to your backend
        // to get the token and identityId

        if (getProviderName() != null && !this.loginsMap.isEmpty()
            && this.loginsMap.containsKey(getProviderName())) {

            /**
              * This is where you would call your backend
             **/
```

```
            // now set the returned identity id and token in the provider
            update(identityId, token);
            return token;

         } else {
            // Otherwise call &COG; using getIdentityId of super class
            return super.getIdentityId();
         }

      } else {
         return identityId;
      }

}
```

iOS – Objective-C

Ihre Anwendung erfordert möglicherweise die Unterstützung sowohl von nicht authentifizierten Identitäten oder authentifizierten Identitäten unter Verwendung öffentlicher Anbieter (Anmeldung mit Amazon, Apple, Facebook oder Google) als auch von entwicklerauthentifizierten Identitäten. Überschreiben Sie dazu die -AWSCognitoCredentialsProviderHelper`logins`Methode, um die richtige Anmeldezuweisung basierend auf dem aktuellen Identitätsanbieter zurückgeben zu können. Das folgende Beispiel zeigt, wie Sie zwischen einer authentifizierten Identität, Facebook und einer entwicklerauthentifizierten Identität wechseln können.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else if (/*logic to determine if user is Facebook*/){
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook :
 [FBSDKAccessToken currentAccessToken] }];
    }else {
        return [super logins];
    }
}
```

Beim Wechsel von einem nicht authentifizierten zu einem authentifizierten Anbieter müssen Sie `[credentialsProvider clearCredentials];` aufrufen und so erzwingen, dass das SDK die neuen authentifizierten Anmeldeinformationen abruft. Wenn Sie zwischen zwei authentifizierten Anbietern wechseln und nicht versuchen, die beiden Anbieter zu verknüpfen (d. h. Sie stellen im Anmeldeverzeichnis keine Token für mehrere Anbieter bereit), müssen

Sie `[credentialsProvider clearKeychain];` aufrufen. Dadurch werden sowohl die Anmeldeinformationen als auch die Identität gelöscht und das SDK ruft neue Informationen ab.

iOS – Swift

Ihre Anwendung erfordert möglicherweise die Unterstützung sowohl von nicht authentifizierten Identitäten oder authentifizierten Identitäten unter Verwendung öffentlicher Anbieter (Anmeldung mit Amazon, Apple, Facebook oder Google) als auch von entwicklerauthentifizierten Identitäten. Überschreiben Sie dazu die -AWSCognitoCredentialsProviderHelper`logins`Methode, um die richtige Anmeldezuweisung basierend auf dem aktuellen Identitätsanbieter zurückgeben zu können. Das folgende Beispiel zeigt, wie Sie zwischen einer authentifizierten Identität, Facebook und einer entwicklerauthentifizierten Identität wechseln können.

```
override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else if (/*logic to determine if user is Facebook*/){
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error:NSError(domain: "Facebook Login", code: -1 , userInfo:
 ["Facebook" : "No current Facebook access token"]))
    }else {
        return super.logins()
    }
}
```

Beim Wechsel von einem nicht authentifizierten zu einem authentifizierten Anbieter müssen Sie `credentialsProvider.clearCredentials()` aufrufen und so erzwingen, dass das SDK die neuen authentifizierten Anmeldeinformationen abruft. Wenn Sie zwischen zwei authentifizierten Anbietern wechseln und nicht versuchen, die beiden Anbieter zu verknüpfen (d. h. Sie stellen im Anmeldeverzeichnis keine Token für mehrere Anbieter bereit), müssen Sie aufrufen `credentialsProvider.clearKeychain()`. Dadurch werden sowohl die Anmeldeinformationen als auch die Identität gelöscht und das SDK ruft neue Informationen ab.

Unity

Ihre Anwendung erfordert möglicherweise die Unterstützung sowohl von nicht authentifizierten Identitäten oder authentifizierten Identitäten unter Verwendung öffentlicher Anbieter (Anmeldung mit Amazon, Apple, Facebook oder Google) als auch von entwicklerauthentifizierten Identitäten. Der

wesentliche Unterschied zwischen entwicklerauthentifizierten Identitäten und anderen Identitäten (nicht authentifizierten Identitäten und authentifizierten Identitäten unter Verwendung von öffentlichen Anbietern) ist die Art, in der die Identitäts-ID und das Token abgerufen werden. Für andere Identitäten interagiert die mobile Anwendung direkt mit Amazon Cognito, anstatt Ihr Authentifizierungssystem zu kontaktieren. Die mobile Anwendung muss daher je nach Auswahl durch den App-Benutzer zwei verschiedene Abläufe unterstützen können. Hierzu müssen Sie einige Änderungen am benutzerdefinierten Identitätsanbieter vornehmen.

Die empfohlene Methode, dies in Unity zu tun, besteht darin, Ihren Identitätsanbieter von AmazonCognitoEnhancedIdentityProvide anstelle von zu erweitern und die übergeordnete RefreshAsync Methode anstelle Ihrer eigenen Methode aufzurufen AbstractCognitoIdentityProvider, falls der Benutzer nicht über Ihr eigenes Backend authentifiziert ist. Wenn der Benutzer authentifiziert wurde, können Sie den gleichen Ablauf verwenden, der oben bereits erläutert wurde.

Xamarin

Ihre Anwendung erfordert möglicherweise die Unterstützung sowohl von nicht authentifizierten Identitäten oder authentifizierten Identitäten unter Verwendung öffentlicher Anbieter (Anmeldung mit Amazon, Apple, Facebook oder Google) als auch von entwicklerauthentifizierten Identitäten. Der wesentliche Unterschied zwischen entwicklerauthentifizierten Identitäten und anderen Identitäten (nicht authentifizierten Identitäten und authentifizierten Identitäten unter Verwendung von öffentlichen Anbietern) ist die Art, in der die Identitäts-ID und das Token abgerufen werden. Für andere Identitäten interagiert die mobile Anwendung direkt mit Amazon Cognito, anstatt Ihr Authentifizierungssystem zu kontaktieren. Die mobile Anwendung muss daher je nach Auswahl durch den App-Benutzer zwei verschiedene Abläufe unterstützen können. Hierzu müssen Sie einige Änderungen am benutzerdefinierten Identitätsanbieter vornehmen.

Wechseln von nicht authentifizierten Benutzern zu authentifizierten Benutzern (Identitäten-Pools)

Amazon-Cognito-Identitäten-Pools unterstützen authentifizierte und nicht authentifizierte Benutzer. Nicht authentifizierte Benutzer erhalten Zugriff auf Ihre AWS-Ressourcen, auch wenn diese nicht mit Ihren Identitätsanbietern (IdPs) angemeldet sind. Dieser Grad des Zugriffs ist nützlich, um Inhalte für Benutzer anzuzeigen, bevor sie sich anmelden. Jeder nicht authentifizierte Benutzer hat eine eindeutige Identität im Identitäten-Pool, auch wenn er nicht einzeln angemeldet und authentifiziert ist.

In diesem Abschnitt wird beschrieben, wie Benutzer vorgehen, wenn sie von einer Anmeldung als nicht authentifizierte Identität zur Verwendung einer authentifizierten Identität wechseln möchten.

Android

Benutzer können sich als nicht authentifizierte Gäste bei Ihrer Anwendung anmelden. Schließlich können sie sich dazu entscheiden, sich mit einem der unterstützten IdPs anzumelden. Amazon Cognito stellt sicher, dass eine ältere Identität dieselbe eindeutige ID beibehält wie die neue und dass die Profildaten automatisch zusammengeführt werden.

Ihre Anwendung wird über die Schnittstelle `IdentityChangedListener` über die Profilzusammenführung informiert. Implementieren Sie die Methode `identityChanged` in der Schnittstelle, um diese Meldungen zu erhalten:

```
@override
public void identityChanged(String oldIdentityId, String newIdentityId) {
    // handle the change
}
```

iOS – Objective-C

Benutzer können sich als nicht authentifizierte Gäste bei Ihrer Anwendung anmelden. Schließlich können sie sich dazu entscheiden, sich mit einem der unterstützten IdPs anzumelden. Amazon Cognito stellt sicher, dass eine ältere Identität dieselbe eindeutige ID beibehält wie die neue und dass die Profildaten automatisch zusammengeführt werden.

`NSNotificationCenter` informiert die Anwendung über eine Profilzusammenführung:

```
[[NSNotificationCenter defaultCenter] addObserver:self
                                      selector:@selector(identityIdDidChange:)
                                      name:AWSCognitoIdentityIdChangedNotification
                                      object:nil];

-(void)identityDidChange:(NSNotification*)notification {
    NSDictionary *userInfo = notification.userInfo;
    NSLog(@"identity changed from %@ to %@",
        [userInfo objectForKey:AWSCognitoNotificationPreviousId],
        [userInfo objectForKey:AWSCognitoNotificationNewId]);
}
```

iOS – Swift

Benutzer können sich als nicht authentifizierte Gäste bei Ihrer Anwendung anmelden. Schließlich können sie sich dazu entscheiden, sich mit einem der unterstützten IdPs anzumelden. Amazon Cognito stellt sicher, dass eine ältere Identität dieselbe eindeutige ID beibehält wie die neue und dass die Profildaten automatisch zusammengeführt werden.

`NSNotificationCenter` informiert die Anwendung über eine Profilzusammenführung:

```
[NSNotificationCenter.defaultCenter().addObserver(observer: self
   selector:"identityDidChange"
   name:AWSCognitoIdentityIdChangedNotification
   object:nil)

func identityDidChange(notification: NSNotification!) {
  if let userInfo = notification.userInfo as? [String: AnyObject] {
    print("identity changed from: \(userInfo[AWSCognitoNotificationPreviousId])
    to: \(userInfo[AWSCognitoNotificationNewId])")
  }
}
```

JavaScript

Anfänglich nicht authentifizierter Benutzer

Benutzer beginnen in der Regel mit der nicht authentifizierten Rolle. Für diese Rolle stellen Sie die Eigenschaft für die Anmeldeinformationen Ihres Konfigurationsobjekt so ein, dass sie keine Anmeldeeigenschaft enthält. In diesem Fall könnte Ihre Standardkonfiguration folgendermaßen aussehen:

```

// set the default config object
var creds = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'us-east-1:1699ebc0-7900-4099-b910-2df94f52a030'
});
AWS.config.credentials = creds;
```

Wechseln Sie zum authentifizierten Benutzer

Wenn ein nicht authentifizierter Benutzer sich bei einem IdP anmeldet und Sie ein Token haben, können Sie den Benutzer vom nicht authentifizierten in einen authentifizierten Benutzer ändern,

indem Sie eine benutzerdefinierte Funktion aufrufen, welche das Anmeldeobjekt aktualisiert und das Anmelde-Token hinzufügt:

```
// Called when an identity provider has a token for a logged in user
function userLoggedIn(providerName, token) {
    creds.params.Logins = creds.params.Logins || {};
    creds.params.Logins[providerName] = token;

    // Expire credentials to refresh them on the next request
    creds.expired = true;
}
```

Darüber hinaus können Sie ein `CognitoIdentityCredentials`-Objekt erstellen. Wenn Sie dies tun, müssen Sie die Anmeldeeigenschaften von vorhandenen Serviceobjekten zurücksetzen, um die aktualisierten Konfigurationsdaten für die Anmeldeinformationen darzustellen. Weitere Informationen finden Sie unter Verwenden des Global Configuration Object.

Weitere Informationen zum `CognitoIdentityCredentials`-Objekt finden Sie unter AWS-CognitoIdentityCredentials in der AWS SDK for JavaScript-API-Referenz.

Unity

Benutzer können sich als nicht authentifizierte Gäste bei Ihrer Anwendung anmelden. Schließlich können sie sich dazu entscheiden, sich mit einem der unterstützten IdPs anzumelden. Amazon Cognito stellt sicher, dass eine ältere Identität dieselbe eindeutige ID beibehält wie die neue und dass die Profildaten automatisch zusammengeführt werden.

Sie können das `IdentityChangedEvent` abonnieren, um Benachrichtigungen über Profilzusammenführungen zu erhalten:

```
credentialsProvider.IdentityChangedEvent += delegate(object sender,
 CognitoAWSCredentials.IdentityChangedArgs e)
{
    // handle the change
    Debug.log("Identity changed from " + e.OldIdentityId + " to " + e.NewIdentityId);
};
```

Xamarin

Benutzer können sich als nicht authentifizierte Gäste bei Ihrer Anwendung anmelden. Schließlich können sie sich dazu entscheiden, sich mit einem der unterstützten IdPs anzumelden. Amazon Cognito stellt sicher, dass eine ältere Identität dieselbe eindeutige ID beibehält wie die neue und dass die Profildaten automatisch zusammengeführt werden.

```
credentialsProvider.IdentityChangedEvent += delegate(object sender,
 CognitoAWSCredentials.IdentityChangedArgs e){
    // handle the change
    Console.WriteLine("Identity changed from " + e.OldIdentityId + " to " +
 e.NewIdentityId);
};
```

Amazon Cognito Sync

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync ist AWS AppSync ein Service zum Synchronisieren von Anwendungsdaten zwischen verschiedenen Geräten.
> Es ermöglicht Benutzerdaten wie App-Einstellungen oder Spielstatus synchronisiert werden. Darüber hinaus erweitert es diese Möglichkeiten, indem mehrere Benutzer gemeinsam genutzte Daten synchronisieren und diese in Echtzeit zusammen nutzen können.

Amazon Cognito Sync ist ein AWS-Service und eine Client-Bibliothek, mit der die geräteübergreifende Synchronisierung von anwendungsbezogenen Benutzerdaten möglich ist. Amazon Cognito Sync kann Benutzerprofildaten über mobile Geräte und das Web synchronisieren, ohne Ihr eigenes Backend verwenden zu müssen. Die Client-Bibliotheken speichern Daten lokal zwischen, sodass Ihre App Daten unabhängig vom Konnektivitätsstatus des Geräts lesen und schreiben kann. Wenn das Gerät online ist, können Sie Daten synchronisieren. Wenn Sie die Push-Synchronisierung einrichten, können Sie andere Geräte umgehend benachrichtigen, wenn ein Update verfügbar ist.

Weitere Informationen zu regionalen Amazon-Cognito-Identitäten-Pools finden Sie unter Regionale Verfügbarkeit von AWS-Services.

Weitere Informationen zu Amazon Cognito Sync finden Sie in den folgenden Themen.

Themen

Erste Schritte mit Amazon Cognito Sync

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync ist AWS AppSync ein Service zum Synchronisieren von Anwendungsdaten zwischen verschiedenen Geräten.
> Es ermöglicht Benutzerdaten wie App-Einstellungen oder Spielstatus synchronisiert werden. Darüber hinaus erweitert es diese Möglichkeiten, indem mehrere Benutzer gemeinsam genutzte Daten synchronisieren und diese in Echtzeit zusammen nutzen können.

Amazon Cognito Sync ist ein AWS-Service und eine Client-Bibliothek, mit der die geräteübergreifende Synchronisierung von anwendungsbezogenen Benutzerdaten möglich ist. Sie können diese Funktion zum Synchronisieren von Profildaten für Benutzer auf mobilen Geräten und in Webanwendungen verwenden. Der Client-Bibliotheken speichern Daten lokal zwischen, sodass Ihre App Daten unabhängig vom Konnektivitätsstatus des Geräts lesen und schreiben kann. Wenn das Gerät online ist, können Sie Daten synchronisieren und, sofern Push-Synchronisierung eingerichtet ist, andere Geräte umgehend benachrichtigen, wenn ein Update verfügbar ist.

Einrichten eines Identitätspools in Amazon Cognito

Amazon Cognito Sync erfordert einen Amazon-Cognito-Identitätspool, um Benutzeridentitäten bereitzustellen. Bevor Sie Amazon Cognito Sync verwenden, müssen Sie zuerst einen Identitätspool einrichten. Befolgen Sie die Anleitung unter Erste Schritte mit Amazon Cognito Cognito-Identitätspools zum Erstellen eines Identitäten-Pools und zum Installieren des SDK.

Speichern und Synchronisieren von Daten

Nach dem Einrichten eines Identitäten-Pools und der Installation des SDK können Sie mit dem Speichern und Synchronisieren von Daten zwischen Geräten beginnen. Weitere Informationen finden Sie unter Synchronisieren von Daten.

Synchronisieren von Daten

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync ist AWS AppSync ein Service zum Synchronisieren von Anwendungsdaten zwischen verschiedenen Geräten.

Es ermöglicht Benutzerdaten wie App-Einstellungen oder Spielstatus synchronisiert werden. Darüber hinaus erweitert es diese Möglichkeiten, indem mehrere Benutzer gemeinsam genutzte Daten synchronisieren und diese in Echtzeit zusammen nutzen können.

Mit Amazon Cognito können Sie Benutzerdaten in Datensätzen speichern, die Schlüssel-Wert-Paare enthalten. Amazon-Cognito verknüpft diese Daten mit einer Identität in Ihrem Identitäten-Pool sodass Ihre App mit verschiedenen Anmeldedaten und Geräten darauf zugreifen kann. Um diese Daten zwischen dem Amazon-Cognito-Service und Geräten eines Endbenutzers zu synchronisieren, rufen Sie die Synchronisierungsmethode auf. Jeder Datensatz kann eine maximale Größe von 1 MB haben. Sie können bis zu 20 Datensätze mit einer Identität assoziieren.

Der Amazon-Cognito-Sync-Client erstellt einen lokalen Cache für die Identitätsdaten. Ihre App kommuniziert mit diesem lokalen Cache, wenn sie Schlüssel liest und schreibt. Diese Kommunikation stellt sicher, dass alle am Gerät vorgenommen Änderungen sofort auf dem Gerät zur Verfügung stehen, auch wenn Sie offline sind. Wenn die Synchronisierungsmethode aufgerufen wird, werden Änderungen vom Service auf das Gerät gezogen, und alle lokalen Änderungen werden per Push-Verfahren an den Service übertragen. Zu diesem Punkt stehen die Änderungen auch anderen Geräten zur Synchronisierung zur Verfügung.

Initialisieren des Amazon-Cognito-Sync-Clients

Zum Initialisieren des Amazon-Cognito-Sync-Clients müssen Sie zunächst einen Anmeldeinformationsanbieter erstellen. Der Anmeldeinformationsanbieter ruft temporäre AWS-Anmeldeinformationen ab, damit Ihre App auf Ihre AWS-Ressourcen zugreifen kann. Sie müssen auch die erforderlichen Header-Dateien importieren. Gehen Sie zur Initialisierung des Amazon-Cognito-Sync-Client folgendermaßen vor.

Android

1. Erstellen Sie einen Anmeldeinformationen-Anbieter gemäß den Anweisungen unter Abrufen von Anmeldeinformationen.
2. Importieren Sie folgendes Amazon-Cognito-Paket: `import com.amazonaws.mobileconnectors.cognito.*;`
3. Initialisieren Sie Amazon Cognito Sync. Übermitteln Sie wie folgt den Android App-Kontext, die Identitäten-Pool-ID, eine AWS-Region und einen initialisierten Amazon Cognito-Anmeldeinformationsanbieter:

```
CognitoSyncManager client = new CognitoSyncManager(
    getApplicationContext(),
    Regions.YOUR_REGION,
    credentialsProvider);
```

iOS – Objective-C

1. Erstellen Sie einen Anmeldeinformationen-Anbieter gemäß den Anweisungen unter Abrufen von Anmeldeinformationen.
2. Importieren Sie `AWSCore` und `Cognito` initialisieren Sie `AWSCognito` wie folgt:

```
#import <AWSiOSSDKv2/AWSCore.h>
#import <AWSCognitoSync/Cognito.h>

AWSCognito *syncClient = [AWSCognito defaultCognito];
```

3. Wenn Sie CocoaPods verwenden, ersetzen Sie `<AWSiOSSDKv2/AWSCore.h>` durch `AWSCore.h`. Folgen Sie der gleichen Syntax für den Import von Amazon Cognito.

iOS – Swift

1. Erstellen Sie einen Anmeldeinformationen-Anbieter gemäß den Anweisungen unter Abrufen von Anmeldeinformationen.
2. Importieren und initialisieren Sie `AWSCognito` wie folgt:

```
import AWSCognito
let syncClient = AWSCognito.default()!
```

JavaScript

1. Laden Sie den Amazon-Cognito-Sync-Manager für JavaScript herunter.
2. Fügen Sie die Sync-Manager-Bibliothek in Ihr Projekt ein.
3. Erstellen Sie einen Anmeldeinformationen-Anbieter gemäß den Anweisungen unter Abrufen von Anmeldeinformationen.
4. Initialisieren Sie den Sync-Manager wie folgt:

```
var syncManager = new AWS.CognitoSyncManager();
```

Unity

1. Erstellen Sie eine Instance von `CognitoAWSCredentials` gemäß den Anweisungen unter Abrufen von Anmeldeinformationen.
2. Erstellen Sie eine Instance von `CognitoSyncManager`. Übergeben Sie das `CognitoAwsCredentials`-Objekt und ein `AmazonCognitoSyncConfig` und schließen Sie wie folgt mindestens die Region ein:

```
AmazonCognitoSyncConfig clientConfig = new AmazonCognitoSyncConfig { RegionEndpoint =
 REGION };
CognitoSyncManager syncManager = new CognitoSyncManager(credentials, clientConfig);
```

Xamarin

1. Erstellen Sie eine Instance von `CognitoAWSCredentials` gemäß den Anweisungen unter Abrufen von Anmeldeinformationen.
2. Erstellen Sie eine Instance von `CognitoSyncManager`. Übergeben Sie das `CognitoAwsCredentials`-Objekt und ein `AmazonCognitoSyncConfig` und schließen Sie wie folgt mindestens die Region ein:

```
AmazonCognitoSyncConfig clientConfig = new AmazonCognitoSyncConfig { RegionEndpoint =
 REGION };
CognitoSyncManager syncManager = new CognitoSyncManager(credentials, clientConfig);
```

Grundlegendes zu Datensätzen

Bei Amazon Cognito werden Benutzerprofildaten in Datensätzen organisiert. Jeder Datensatz kann bis zu 1 MB Daten in Form von Schlüssel-Wert-Paaren enthalten. Ein Datensatz ist die genaueste Entität, die Sie synchronisieren können. Lese- und Schreibvorgänge auf einem Datensatz haben nur Auswirkungen auf den lokalen Speicher, bis die Synchronisierungsmethode aufgerufen wird. Amazon Cognito identifiziert einen Datensatz durch eine eindeutige Zeichenfolge. Sie können wie folgt einen neuen Datensatz erstellen oder einen vorhandenen öffnen.

Android

```
Dataset dataset = client.openOrCreateDataset("datasetname");
```

Um einen Datensatz zu löschen, rufen Sie zuerst die Methode auf, mit der dieser aus dem lokalen Speicher gelöscht wird. Rufen Sie anschließend wie folgt die `synchronize`-Methode zum Löschen des Datensatzes aus Amazon Cognito auf:

```
dataset.delete();
dataset.synchronize(syncCallback);
```

iOS – Objective-C

```
AWSCognitoDataset *dataset = [syncClient openOrCreateDataset:@"myDataSet"];
```

Um einen Datensatz zu löschen, rufen Sie zuerst die Methode auf, mit der dieser aus dem lokalen Speicher gelöscht wird. Rufen Sie anschließend wie folgt die `synchronize`-Methode zum Löschen des Datensatzes aus Amazon Cognito auf:

```
[dataset clear];
[dataset synchronize];
```

iOS – Swift

```
let dataset = syncClient.openOrCreateDataset("myDataSet")!
```

Um einen Datensatz zu löschen, rufen Sie zuerst die Methode auf, mit der dieser aus dem lokalen Speicher gelöscht wird. Rufen Sie anschließend wie folgt die `synchronize`-Methode zum Löschen des Datensatzes aus Amazon Cognito auf:

```
dataset.clear()
dataset.synchronize()
```

JavaScript

```
syncManager.openOrCreateDataset('myDatasetName', function(err, dataset) {
   // ...
```

```
});
```

Unity

```
string myValue = dataset.Get("myKey");
dataset.Put("myKey", "newValue");
```

Um einen Schlüssel aus einem Datensatz zu löschen, verwenden Sie `Remove` wie folgt:

```
dataset.Remove("myKey");
```

Xamarin

```
Dataset dataset = syncManager.OpenOrCreateDataset("myDatasetName");
```

Um einen Datensatz zu löschen, rufen Sie zuerst die Methode auf, mit der dieser aus dem lokalen Speicher gelöscht wird. Rufen Sie anschließend wie folgt die `synchronize`-Methode zum Löschen des Datensatzes aus Amazon Cognito auf:

```
dataset.Delete();
dataset.SynchronizeAsync();
```

Lesen und Schreiben von Daten in Datensätze

Amazon-Cognito-Datensätze fungieren als Wörterbücher, auf deren Werte mit einem Schlüssel zugegriffen werden kann: Sie können die Schlüssel und Werte eines Datensatzes wie in den folgenden Beispielen gezeigt lesen, hinzufügen oder ändern, als wäre der Datensatz ein Wörterbuch.

Beachten Sie, dass sich die von Ihnen in einen Datensatz geschriebenen Werte nur auf die lokale, zwischengespeicherte Kopie der Daten auswirken, bis Sie die Synchronisierungsmethode aufrufen.

Android

```
String value = dataset.get("myKey");
dataset.put("myKey", "my value");
```

iOS – Objective-C

```
[dataset setString:@"my value" forKey:@"myKey"];
```

```
NSString *value = [dataset stringForKey:@"myKey"];
```

iOS – Swift

```
dataset.setString("my value", forKey:"myKey")
let value = dataset.stringForKey("myKey")
```

JavaScript

```
dataset.get('myKey', function(err, value) {
  console.log('myRecord: ' + value);
});

dataset.put('newKey', 'newValue', function(err, record) {
  console.log(record);
});

dataset.remove('oldKey', function(err, record) {
  console.log(success);
});
```

Unity

```
string myValue = dataset.Get("myKey");
dataset.Put("myKey", "newValue");
```

Xamarin

```
//obtain a value
string myValue = dataset.Get("myKey");

// Create a record in a dataset and synchronize with the server
dataset.OnSyncSuccess += SyncSuccessCallback;
dataset.Put("myKey", "myValue");
dataset.SynchronizeAsync();

void SyncSuccessCallback(object sender, SyncSuccessEventArgs e) {
  // Your handler code here
}
```

Android

Verwenden Sie die `remove`-Methode zum Entfernen von Schlüsseln aus einem Datensatz:

```
dataset.remove("myKey");
```

iOS – Objective-C

Um einen Schlüssel aus einem Datensatz zu löschen, verwenden Sie `removeObjectForKey` wie folgt:

```
[dataset removeObjectForKey:@"myKey"];
```

iOS – Swift

Um einen Schlüssel aus einem Datensatz zu löschen, verwenden Sie `removeObjectForKey` wie folgt:

```
dataset.removeObjectForKey("myKey")
```

Unity

Um einen Schlüssel aus einem Datensatz zu löschen, verwenden Sie `Remove` wie folgt:

```
dataset.Remove("myKey");
```

Xamarin

Sie können `Remove` zum Löschen eines Schlüssels aus einer Datenbank verwenden.

```
dataset.Remove("myKey");
```

Synchronisieren lokaler Daten mit dem Sync Store

Android

Die `synchronize`-Methode vergleicht lokal zwischengespeicherte Daten mit den im Amazon-Cognito-Sync-Store gespeicherten Daten. Remote-Änderungen werden aus dem Amazon-Cognito-Sync-Store entnommen; Konfliktauflösung wird aufgerufen, wenn irgendwelche Konflikte auftreten,

und aktualisierte Werte auf dem Gerät werden per Push-Verfahren an den Service weitergeleitet. Zum Synchronisieren des Datensatzes rufen Sie die `synchronize`-Methode auf:

```
dataset.synchronize(syncCallback);
```

Die `synchronize`-Methode erhält eine Implementierung der `SyncCallback`-Schnittstelle, die im Folgenden näher erörtert wird.

Die `synchronizeOnConnectivity()`-Methode versucht die Synchronisierung, wenn eine Verbindung verfügbar ist. Wenn umgehend Konnektivität verfügbar ist, verhält sich `synchronizeOnConnectivity()` wie `synchronize()`. Andernfalls überwacht es auf Änderungen in der Konnektivität und führt eine Synchronisierung durch, sobald eine Verbindung verfügbar ist. Wenn `synchronizeOnConnectivity()` mehrere Male aufgerufen wird, wird nur die letzte Synchronisationsanforderung behalten, und nur der letzte Rückruf wird ausgelöst. Wenn für den Datensatz oder die Rückruffunktion eine Speicherbereinigung durchgeführt wurde, erfolgt mit dieser Methode keine Synchronisierung, und der Rückruf wird nicht ausgelöst.

Weitere Informationen zur Datensatzsynchronisierung und den verschiedenen Rückrufen finden Sie unter Umgang mit Callbacks.

iOS – Objective-C

Die `synchronize`-Methode vergleicht lokal zwischengespeicherte Daten mit den im Amazon-Cognito-Sync-Store gespeicherten Daten. Remote-Änderungen werden aus dem Amazon-Cognito-Sync-Store entnommen; Konfliktauflösung wird aufgerufen, wenn irgendwelche Konflikte auftreten, und aktualisierte Werte auf dem Gerät werden per Push-Verfahren an den Service weitergeleitet. Zum Synchronisieren des Datensatzes rufen Sie die `synchronize`-Methode auf:

Die `synchronize`-Methode ist asynchron und gibt ein `AWSTask`-Objekt zur Verarbeitung der Antwort zurück:

```
[[dataset synchronize] continueWithBlock:^id(AWSTask *task) {
    if (task.isCancelled) {
        // Task cancelled.
    } else if (task.error) {
        // Error while executing task.
    } else {
        // Task succeeded. The data was saved in the sync store.
    }
    return nil;
```

```
}];
```

Die `synchronizeOnConnectivity`-Methode versucht die Synchronisierung, wenn auf dem Gerät eine Verbindung verfügbar ist. Zunächst prüft `synchronizeOnConnectivity` auf Konnektivität und ruft sofort die Synchronisierung auf, wenn das Gerät online ist, und gibt das mit dem Versuch verknüpfte `AWSTask`-Objekt zurück.

Wenn das Gerät offline ist, plant `synchronizeOnConnectivity` 1) eine Synchronisierung, wenn das Gerät das nächste Mal online ist und 2) gibt ein `AWSTask`-Objekt mit einem nil-Ergebnis zurück. Die geplante Synchronisierung ist nur für den Lebenszyklus des Datensatzes gültig. Die Daten werden nicht synchronisiert, wenn die App beendet wird, bevor die Konnektivität wiederhergestellt wurde. Wenn Sie benachrichtigt werden sollen, wenn Ereignisse während der geplanten Synchronisierung auftreten, müssen Sie Beobachter der in gefundenen Benachrichtigungen hinzufügen `AWSCognito`.

Weitere Informationen zur Datensatzsynchronisierung und den verschiedenen Rückrufen finden Sie unter Umgang mit Callbacks.

iOS – Swift

Die `synchronize`-Methode vergleicht lokal zwischengespeicherte Daten mit den im Amazon-Cognito-Sync-Store gespeicherten Daten. Remote-Änderungen werden aus dem Amazon-Cognito-Sync-Store entnommen; Konfliktauflösung wird aufgerufen, wenn irgendwelche Konflikte auftreten, und aktualisierte Werte auf dem Gerät werden per Push-Verfahren an den Service weitergeleitet. Zum Synchronisieren des Datensatzes rufen Sie die `synchronize`-Methode auf:

Die `synchronize`-Methode ist asynchron und gibt ein `AWSTask`-Objekt zur Verarbeitung der Antwort zurück:

```
dataset.synchronize().continueWith(block: { (task) -> AnyObject? in

            if task.isCancelled {
                // Task cancelled.
            } else if task.error != nil {
                // Error while executing task
            } else {
                // Task succeeded. The data was saved in the sync store.
            }
            return task
})
```

Die `synchronizeOnConnectivity`-Methode versucht die Synchronisierung, wenn auf dem Gerät eine Verbindung verfügbar ist. Zunächst prüft `synchronizeOnConnectivity` auf Konnektivität und ruft sofort `synchronize` auf, wenn das Gerät online ist, und gibt das mit dem Versuch verknüpfte `AWSTask`-Objekt zurück.

Wenn das Gerät offline ist, plant `synchronizeOnConnectivity` 1) eine Synchronisierung, wenn das Gerät das nächste Mal online ist und 2) gibt ein `AWSTask`-Objekt mit einem nil-Ergebnis zurück. Die geplante Synchronisierung ist nur für den Lebenszyklus des Datensatzes gültig. Die Daten werden nicht synchronisiert, wenn die App beendet wird, bevor die Konnektivität wiederhergestellt wurde. Wenn Sie benachrichtigt werden sollen, wenn Ereignisse während der geplanten Synchronisierung auftreten, müssen Sie Beobachter der in gefundenen Benachrichtigungen hinzufügen `AWSCognito`.

Weitere Informationen zur Datensatzsynchronisierung und den verschiedenen Rückrufen finden Sie unter Umgang mit Callbacks.

JavaScript

Die `synchronize`-Methode vergleicht lokal zwischengespeicherte Daten mit den im Amazon-Cognito-Sync-Store gespeicherten Daten. Remote-Änderungen werden aus dem Amazon-Cognito-Sync-Store entnommen; Konfliktauflösung wird aufgerufen, wenn irgendwelche Konflikte auftreten, und aktualisierte Werte auf dem Gerät werden per Push-Verfahren an den Service weitergeleitet. Zum Synchronisieren des Datensatzes rufen Sie die `synchronize`-Methode auf:

```
dataset.synchronize();
```

Weitere Informationen zur Datensatzsynchronisierung und den verschiedenen Rückrufen finden Sie unter Umgang mit Callbacks.

Unity

Die Synchronisations-Methode vergleicht lokal zwischengespeicherte Daten mit den im Amazon-Cognito-Sync-Store gespeicherten Daten. Remote-Änderungen werden aus dem Amazon-Cognito-Sync-Store entnommen; Konfliktauflösung wird aufgerufen, wenn irgendwelche Konflikte auftreten, und aktualisierte Werte auf dem Gerät werden per Push-Verfahren an den Service weitergeleitet. Zum Synchronisieren des Datensatzes rufen Sie die `synchronize`-Methode auf:

```
dataset.Synchronize();
```

Die Synchronisierung erfolgt asynchron, und zuletzt wird einer der verschiedenen Callbacks aufgerufen, die Sie im Datensatz angeben können.

Weitere Informationen zur Datensatzsynchronisierung und den verschiedenen Rückrufen finden Sie unter Umgang mit Callbacks.

Xamarin

Die `synchronize`-Methode vergleicht lokal zwischengespeicherte Daten mit den im Amazon-Cognito-Sync-Store gespeicherten Daten. Remote-Änderungen werden aus dem Amazon-Cognito-Sync-Store entnommen; Konfliktauflösung wird aufgerufen, wenn irgendwelche Konflikte auftreten, und aktualisierte Werte auf dem Gerät werden per Push-Verfahren an den Service weitergeleitet. Zum Synchronisieren des Datensatzes rufen Sie die `synchronize`-Methode auf:

```
dataset.SynchronizeAsync();
```

Weitere Informationen zur Datensatzsynchronisierung und den verschiedenen Rückrufen finden Sie unter Umgang mit Callbacks.

Umgang mit Callbacks

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync ist AWS AppSync ein Service zum Synchronisieren von Anwendungsdaten zwischen verschiedenen Geräten.
> Es ermöglicht Benutzerdaten wie App-Einstellungen oder Spielstatus synchronisiert werden. Darüber hinaus erweitert es diese Möglichkeiten, indem mehrere Benutzer gemeinsam genutzte Daten synchronisieren und diese in Echtzeit zusammen nutzen können.

In diesem Abschnitt wird beschrieben, wie Sie Callbacks verarbeiten.

Android

SyncCallback-Schnittstelle

Durch die Implementierung der `SyncCallback`-Schnittstelle können Sie auf Ihrer App Benachrichtigungen über die Datensatz- Synchronisierung empfangen. Ihre App kann dann aktive Entscheidungen zum Löschen von lokalen Daten, zum Zusammenführen authentifizierter und nicht

authentifizierter Profile und zum Beheben von Konflikten treffen. Sie sollten die folgenden Methoden implementieren, die von der Schnittstelle benötigt werden:

- `onSuccess()`
- `onFailure()`
- `onConflict()`
- `onDatasetDeleted()`
- `onDatasetsMerged()`

Beachten Sie, dass, Sie auch die Klasse `DefaultSyncCallback` verwenden, können, wenn Sie nicht alle Callbacks angeben möchten. Diese bietet standardmäßige, leere Implementierungen für alle.

onSuccess

Der `onSuccess()`-Callback wird ausgelöst, wenn ein Datensatz erfolgreich vom Sync Store heruntergeladen wird.

```
@Override
public void onSuccess(Dataset dataset, List<Record> newRecords) {
}
```

onFailure

onFailure() wird aufgerufen, wenn während der Synchronisierung eine Ausnahmebedingung eintritt.

```
@Override
public void onFailure(DataStorageException dse) {
}
```

onConflict

Konflikte können auftreten, wenn der gleiche Schlüssel im lokalen Speicher und im Sync Store geändert wurde. Die `onConflict()`-Methode erledigt die Konfliktlösung. Wenn Sie diese Methode nicht implementieren, wird der Amazon-Cognito-Sync-Client standardmäßig auf die letzte Änderung zurückgesetzt.

```
@Override
```

```
public boolean onConflict(Dataset dataset, final List<SyncConflict> conflicts) {
    List<Record> resolvedRecords = new ArrayList<Record>();
    for (SyncConflict conflict : conflicts) {
        /* resolved by taking remote records */
        resolvedRecords.add(conflict.resolveWithRemoteRecord());

        /* alternately take the local records */
        // resolvedRecords.add(conflict.resolveWithLocalRecord());

        /* or customer logic, say concatenate strings */
        // String newValue = conflict.getRemoteRecord().getValue()
        //     + conflict.getLocalRecord().getValue();
        // resolvedRecords.add(conflict.resolveWithValue(newValue);
    }
    dataset.resolve(resolvedRecords);

    // return true so that synchronize() is retried after conflicts are resolved
    return true;
}
```

onDatasetDeleted

Wenn ein Datensatz gelöscht wird, ermittelt der Amazon-Cognito-Client anhand der `SyncCallback`-Schnittstelle, ob auch die lokal zwischengespeicherte Kopie des Datensatzes gelöscht werden sollte. Implementieren Sie die `onDatasetDeleted()`-Methode, um dem Client-SDK mitzuteilen, was mit den lokalen Daten geschehen soll.

```
@Override
public boolean onDatasetDeleted(Dataset dataset, String datasetName) {
    // return true to delete the local copy of the dataset
    return true;
}
```

onDatasetMerged

Wenn zwei zuvor unverbundene Identitäten verknüpft werden, werden alle ihre Datensätze zusammengeführt. Anwendungen werden von der Zusammenführung über die `onDatasetsMerged()`-Methode benachrichtigt:

```
@Override
public boolean onDatasetsMerged(Dataset dataset, List<String> datasetNames) {
    // return false to handle Dataset merge outside the synchronization callback
```

```
    return false;
}
```

iOS – Objective-C

Sync-Benachrichtigungen

Der Amazon-Cognito-Client gibt eine Reihe von `NSNotification`-Ereignissen während eines Synchronisierungsaufrufs aus. Sie können sich registrieren, um diese Benachrichtigungen über das standardmäßige zu überwachen `NSNotificationCenter`:

```
[NSNotificationCenter defaultCenter]
  addObserver:self
  selector:@selector(myNotificationHandler:)
  name:NOTIFICATION_TYPE
  object:nil];
```

Amazon Cognito unterstützt fünf Benachrichtigungstypen, die nachstehend aufgeführt sind.

AWSCognitoDidStartSynchronizeNotification

Wird aufgerufen, wenn eine Synchronisierung gestartet wird. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes.

AWSCognitoDidEndSynchronizeNotification

Wird aufgerufen, wenn ein Synchronisierungsvorgang abgeschlossen wird (erfolgreich oder nicht). Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes.

AWSCognitoDidFailToSynchronizeNotification

Wird aufgerufen, wenn eine Synchronisierung fehlschlägt. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes, und den Schlüsselfehler mit dem Fehler, der den Ausfall verursacht hat.

AWSCognitoDidChangeRemoteValueNotification

Wird aufgerufen, wenn lokale Änderungen erfolgreich im Push-Verfahren auf Amazon Cognito übertragen wurden. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des

synchronisierten Datensatzes, und die Schlüssel, die einen NSArray der Datensatzschlüssel enthalten, die im Push-Verfahren übertragen wurden.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Wird aufgerufen, wenn ein lokaler Wert sich aufgrund eines Synchronisationsvorgangs ändert. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes, und die Schlüssel, die einen NSArray der Datensatzschlüssel enthalten, die geändert wurden.

Konfliktlösung-Handler

Während einer Synchronisierungsoperation können Konflikte auftreten, wenn der gleichen Schlüssel im lokalen Speicher und im Sync Store geändert wurde. Wenn Sie keinen Konfliktauflösung-Handler festgelegt haben, wählt Amazon Cognito standardmäßig die letzte Aktualisierung.

Durch die Implementierung und Zuweisung eines AWSCognitoRecordConflictHandler können Sie die Standard-Konfliktauflösung ändern. Der AWSCognitoConflict-Eingabeparameterkonflikt enthält ein AWSCognitoRecord-Objekt für die lokal zwischengespeicherten Daten und für den im Konflikt stehenden Datensatz im Sync Store. AWSCognitoConflict dient zur Konfliktauflösung mit dem lokalen Datensatz: [conflict resolveWithLocalRecord], dem Remote-Datensatz: [conflict resolveWithRemoteRecord] oder einem neuen Wert: [conflict resolveWithValue:value]. Wenn von dieser Methode nil zurückgegeben wird, kann die Synchronisierung nicht fortfahren, und die Konflikte treten erneut auf, wenn Sie den Sync-Prozess das nächste Mal starten.

Sie können den Konfliktlösung-Handler auf der Client-Ebene festlegen:

```
client.conflictHandler = ^AWSCognitoResolvedConflict* (NSString *datasetName,
 AWSCognitoConflict *conflict) {
    // always choose local changes
    return [conflict resolveWithLocalRecord];
};
```

Oder auf Datensatzebene:

```
dataset.conflictHandler = ^AWSCognitoResolvedConflict* (NSString *datasetName,
 AWSCognitoConflict *conflict) {
    // override and always choose remote changes
    return [conflict resolveWithRemoteRecord];
};
```

Dataset gelöscht-Handler

Wenn ein Datensatz gelöscht wird, ermittelt der Amazon-Cognito-Client anhand des `AWSCognitoDatasetDeletedHandler`, ob auch die lokal zwischengespeicherte Kopie des Datensatzes gelöscht werden sollte. Wenn kein `AWSCognitoDatasetDeletedHandler` implementiert ist, werden die lokalen Daten automatisch gelöscht. Implementieren Sie einen `AWSCognitoDatasetDeletedHandler`, wenn Sie eine Kopie der lokalen Daten aufbewahren möchten, bevor Sie diese löschen, oder um die lokalen Daten beizubehalten.

Sie können den Datensatz gelöscht-Handler auf der Client-Ebene festlegen:

```
client.datasetDeletedHandler = ^BOOL (NSString *datasetName) {
    // make a backup of the data if you choose
    ...
    // delete the local data (default behavior)
    return YES;
};
```

Oder auf Datensatzebene:

```
dataset.datasetDeletedHandler = ^BOOL (NSString *datasetName) {
    // override default and keep the local data
    return NO;
};
```

Dataset zusammenfügen-Handler

Wenn zwei zuvor unverbundene Identitäten verknüpft werden, werden alle ihre Datensätze zusammengeführt. Anwendungen werden von der Zusammenführung über die -Methode benachrichtigt `DatasetMergeHandler`. Der Handler erhält den Namen des Root-Datensatzes sowie eine Reihe von Datensatzamen, die als Zusammenführungen des Root-Datensatzes markiert sind.

Wenn kein `DatasetMergeHandler` implementiert ist, werden diese Datensätze ignoriert, nehmen jedoch weiterhin Platz in den maximal 20 Datensätzen der Identität ein.

Sie können den Datensatz zusammenführen-Handler auf der Client-Ebene festlegen:

```
client.datasetMergedHandler = ^(NSString *datasetName, NSArray *datasets) {
    // Blindly delete the datasets
    for (NSString *name in datasets) {
        AWSCognitoDataset *merged = [[AWSCognito defaultCognito]
 openOrCreateDataset:name];
```

```
        [merged clear];
        [merged synchronize];
    }
};
```

Oder auf Datensatzebene:

```
dataset.datasetMergedHandler = ^(NSString *datasetName, NSArray *datasets) {
    // Blindly delete the datasets
    for (NSString *name in datasets) {
        AWSCognitoDataset *merged = [[AWSCognito defaultCognito]
 openOrCreateDataset:name];
        // do something with the data if it differs from existing dataset
        ...
        // now delete it
        [merged clear];
        [merged synchronize];
    }
};
```

iOS – Swift

Sync-Benachrichtigungen

Der Amazon-Cognito-Client gibt eine Reihe von `NSNotification`-Ereignissen während eines Synchronisierungsaufrufs aus. Sie können sich registrieren, um diese Benachrichtigungen über das standardmäßige zu überwachen `NSNotificationCenter`:

```
NSNotificationCenter.defaultCenter().addObserver(observer: self,
   selector: "myNotificationHandler",
   name:NOTIFICATION_TYPE,
   object:nil)
```

Amazon Cognito unterstützt fünf Benachrichtigungstypen, die nachstehend aufgeführt sind.

AWSCognitoDidStartSynchronizeNotification

Wird aufgerufen, wenn eine Synchronisierung gestartet wird. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes.

AWSCognitoDidEndSynchronizeNotification

Wird aufgerufen, wenn ein Synchronisierungsvorgang abgeschlossen wird (erfolgreich oder nicht). Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes.

AWSCognitoDidFailToSynchronizeNotification

Wird aufgerufen, wenn eine Synchronisierung fehlschlägt. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes, und den Schlüsselfehler mit dem Fehler, der den Ausfall verursacht hat.

AWSCognitoDidChangeRemoteValueNotification

Wird aufgerufen, wenn lokale Änderungen erfolgreich im Push-Verfahren auf Amazon Cognito übertragen wurden. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes, und die Schlüssel, die einen NSArray der Datensatzschlüssel enthalten, die im Push-Verfahren übertragen wurden.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Wird aufgerufen, wenn ein lokaler Wert sich aufgrund eines Synchronisationsvorgangs ändert. Die `userInfo` enthalten den Schlüssel-Datensatz, d. h. den Namen des synchronisierten Datensatzes, und die Schlüssel, die einen NSArray der Datensatzschlüssel enthalten, die geändert wurden.

Konfliktlösung-Handler

Während einer Synchronisierungsoperation können Konflikte auftreten, wenn der gleichen Schlüssel im lokalen Speicher und im Sync Store geändert wurde. Wenn Sie keinen Konfliktauflösung-Handler festgelegt haben, wählt Amazon Cognito standardmäßig die letzte Aktualisierung.

Durch die Implementierung und Zuweisung eines `AWSCognitoRecordConflictHandler` können Sie die Standard-Konfliktauflösung ändern. Der `AWSCognitoConflict`-Eingabeparameterkonflikt enthält ein `AWSCognitoRecord`-Objekt für die lokal zwischengespeicherten Daten und für den im Konflikt stehenden Datensatz im Sync Store. `AWSCognitoConflict` dient zur Konfliktauflösung mit dem lokalen Datensatz: [conflict resolveWithLocalRecord], dem Remote-Datensatz: [conflict resolveWithRemoteRecord] oder einem neuen Wert: [conflict resolveWithValue:value]. Wenn von dieser Methode nil zurückgegeben wird, kann die Synchronisierung nicht fortfahren, und die Konflikte treten erneut auf, wenn Sie den Sync-Prozess das nächste Mal starten.

Sie können den Konfliktlösung-Handler auf der Client-Ebene festlegen:

```
client.conflictHandler = {
```

```
    (datasetName: String?, conflict: AWSCognitoConflict?) ->
 AWSCognitoResolvedConflict? in
    return conflict.resolveWithLocalRecord()
}
```

Oder auf Datensatzebene:

```
dataset.conflictHandler = {
    (datasetName: String?, conflict: AWSCognitoConflict?) ->
 AWSCognitoResolvedConflict? in
    return conflict.resolveWithLocalRecord()
}
```

Dataset gelöscht-Handler

Wenn ein Datensatz gelöscht wird, ermittelt der Amazon-Cognito-Client anhand des `AWSCognitoDatasetDeletedHandler`, ob auch die lokal zwischengespeicherte Kopie des Datensatzes gelöscht werden sollte. Wenn kein `AWSCognitoDatasetDeletedHandler` implementiert ist, werden die lokalen Daten automatisch gelöscht. Implementieren Sie einen `AWSCognitoDatasetDeletedHandler`, wenn Sie eine Kopie der lokalen Daten aufbewahren möchten, bevor Sie diese löschen, oder um die lokalen Daten beizubehalten.

Sie können den Datensatz gelöscht-Handler auf der Client-Ebene festlegen:

```
client.datasetDeletedHandler = {
      (datasetName: String!) -> Bool in
      // make a backup of the data if you choose
      ...
      // delete the local data (default behaviour)
      return true
}
```

Oder auf Datensatzebene:

```
dataset.datasetDeletedHandler = {
      (datasetName: String!) -> Bool in
      // make a backup of the data if you choose
      ...
      // delete the local data (default behaviour)
      return true
```

```
}
```

Datensatz-zusammenfügen-Handler

Wenn zwei zuvor unverbundene Identitäten verknüpft werden, werden alle ihre Datensätze zusammengeführt. Anwendungen werden von der Zusammenführung über die -Methode benachrichtigt `DatasetMergeHandler`. Der Handler erhält den Namen des Root-Datensatzes sowie eine Reihe von Datensatzamen, die als Zusammenführungen des Root-Datensatzes markiert sind.

Wenn kein `DatasetMergeHandler` implementiert ist, werden diese Datensätze ignoriert, nehmen jedoch weiterhin Platz in den maximal 20 Datensätzen der Identität ein.

Sie können den Datensatz zusammenführen-Handler auf der Client-Ebene festlegen:

```
client.datasetMergedHandler = {
    (datasetName: String!, datasets: [AnyObject]!) -> Void in
    for nameObject in datasets {
        if let name = nameObject as? String {
            let merged = AWSCognito.defaultCognito().openOrCreateDataset(name)
            merged.clear()
            merged.synchronize()
        }
    }
}
```

Oder auf Datensatzebene:

```
dataset.datasetMergedHandler = {
    (datasetName: String!, datasets: [AnyObject]!) -> Void in
    for nameObject in datasets {
        if let name = nameObject as? String {
            let merged = AWSCognito.defaultCognito().openOrCreateDataset(name)
            // do something with the data if it differs from existing dataset
              ...
              // now delete it
            merged.clear()
            merged.synchronize()
        }
    }
}
```

JavaScript

Synchronisierungs-Callbacks

Beim Synchronisieren () eines Datensatzes können Sie optional Callbacks für die folgenden Status festlegen:

```
dataset.synchronize({

   onSuccess: function(dataset, newRecords) {
      //...
   },

   onFailure: function(err) {
      //...
   },

   onConflict: function(dataset, conflicts, callback) {
      //...
   },

   onDatasetDeleted: function(dataset, datasetName, callback) {
      //...
   },

  onDatasetMerged: function(dataset, datasetNames, callback) {
      //...
  }

});
```

onSuccess()

Der `onSuccess()`-Callback wird ausgelöst, wenn ein Datensatz erfolgreich vom Sync Store aktualisiert wird. Wenn Sie keinen Callback definieren, erfolgt keine Benachrichtigung bei einer erfolgreichen Synchronisierung.

```
onSuccess: function(dataset, newRecords) {
   console.log('Successfully synchronized ' + newRecords.length + ' new records.');
}
```

onFailure()

`onFailure()` wird aufgerufen, wenn während der Synchronisierung eine Ausnahmebedingung eintritt. Wenn Sie keinen Callback definieren, erfolgt keine Benachrichtigung bei einer fehlgeschlagenen Synchronisierung.

```
onFailure: function(err) {
   console.log('Synchronization failed.');
   console.log(err);
}
```

onConflict()

Konflikte können auftreten, wenn der gleiche Schlüssel im lokalen Speicher und im Sync Store geändert wurde. Die `onConflict()`-Methode erledigt die Konfliktlösung. Wenn Sie diese Methode nicht implementieren, wird die Synchronisierung abgebrochen, wenn ein Konflikt auftritt.

```
onConflict: function(dataset, conflicts, callback) {

   var resolved = [];

   for (var i=0; i<conflicts.length; i++) {

      // Take remote version.
      resolved.push(conflicts[i].resolveWithRemoteRecord());

      // Or... take local version.
      // resolved.push(conflicts[i].resolveWithLocalRecord());

      // Or... use custom logic.
      // var newValue = conflicts[i].getRemoteRecord().getValue() +
 conflicts[i].getLocalRecord().getValue();
      // resolved.push(conflicts[i].resovleWithValue(newValue);

   }

   dataset.resolve(resolved, function() {
      return callback(true);
   });

   // Or... callback false to stop the synchronization process.
   // return callback(false);

}
```

onDatasetDeleted()

Wenn ein Datensatz gelöscht wird, ermittelt der Amazon-Cognito-Client anhand des `onDatasetDeleted()`-Callback, ob auch die lokal zwischengespeicherte Kopie des Datensatzes gelöscht werden sollte. Standardmäßig wird der Datensatz nicht gelöscht.

```
onDatasetDeleted: function(dataset, datasetName, callback) {

   // Return true to delete the local copy of the dataset.
   // Return false to handle deleted datasets outside the synchronization callback.

   return callback(true);

}
```

onDatasetMerged()

Wenn zwei zuvor unverbundene Identitäten verknüpft werden, werden alle ihre Datensätze zusammengeführt. Anwendungen werden von der Zusammenführung über den `onDatasetsMerged()`-Callback benachrichtigt:

```
onDatasetMerged: function(dataset, datasetNames, callback) {

   // Return true to continue the synchronization process.
   // Return false to handle dataset merges outside the synchronization callback.

   return callback(false);

}
```

Unity

Nachdem Sie einen Datensatz geöffnet oder erstellt haben, können Sie andere Callbacks festlegen, die ausgelöst werden, wenn Sie die Synchronisieren-Methode verwenden. Dies ist die Art und Weise, Ihre Callbacks zu registrieren:

```
dataset.OnSyncSuccess += this.HandleSyncSuccess;
dataset.OnSyncFailure += this.HandleSyncFailure;
dataset.OnSyncConflict = this.HandleSyncConflict;
dataset.OnDatasetMerged = this.HandleDatasetMerged;
```

```
dataset.OnDatasetDeleted = this.HandleDatasetDeleted;
```

Beachten Sie, dass `SyncSuccess` und `SyncFailure` += anstelle von = verwenden, sodass Sie mehrere Callbacks für sie abonnieren können.

OnSyncSuccess

Der `OnSyncSuccess`-Callback wird ausgelöst, wenn ein Datensatz erfolgreich von der Cloud aktualisiert wird. Wenn Sie keinen Callback definieren, erfolgt keine Benachrichtigung bei einer erfolgreichen Synchronisierung.

```
private void HandleSyncSuccess(object sender, SyncSuccessEvent e)
{
    // Continue with your game flow, display the loaded data, etc.
}
```

OnSyncFailure

`OnSyncFailure` wird aufgerufen, wenn während der Synchronisierung eine Ausnahmebedingung eintritt. Wenn Sie keinen Callback definieren, erfolgt keine Benachrichtigung bei einer fehlgeschlagenen Synchronisierung.

```
private void HandleSyncFailure(object sender, SyncFailureEvent e)
{
    Dataset dataset = sender as Dataset;
    if (dataset.Metadata != null) {
        Debug.Log("Sync failed for dataset : " + dataset.Metadata.DatasetName);
    } else {
        Debug.Log("Sync failed");
    }
    // Handle the error
    Debug.LogException(e.Exception);
}
```

OnSyncConflict

Konflikte können auftreten, wenn der gleiche Schlüssel im lokalen Speicher und im Sync Store geändert wurde. Der `OnSyncConflict`-Callback erledigt die Konfliktlösung. Wenn Sie diese Methode nicht implementieren, wird die Synchronisierung abgebrochen, wenn ein Konflikt auftritt.

```
private bool HandleSyncConflict(Dataset dataset, List < SyncConflict > conflicts)
```

```
{
  if (dataset.Metadata != null) {
    Debug.LogWarning("Sync conflict " + dataset.Metadata.DatasetName);
  } else {
    Debug.LogWarning("Sync conflict");
  }
  List < Amazon.CognitoSync.SyncManager.Record > resolvedRecords = new List <
 Amazon.CognitoSync.SyncManager.Record > ();
  foreach(SyncConflict conflictRecord in conflicts) {
    // SyncManager provides the following default conflict resolution methods:
    //      ResolveWithRemoteRecord - overwrites the local with remote records
    //      ResolveWithLocalRecord - overwrites the remote with local records
    //      ResolveWithValue - to implement your own logic
    resolvedRecords.Add(conflictRecord.ResolveWithRemoteRecord());
  }
  // resolves the conflicts in local storage
  dataset.Resolve(resolvedRecords);
  // on return true the synchronize operation continues where it left,
  //      returning false cancels the synchronize operation
  return true;
}
```

OnDatasetDeleted

Wenn ein Datensatz gelöscht wird, ermittelt der Amazon-Cognito-Client anhand des `OnDatasetDeleted`-Callback, ob auch die lokal zwischengespeicherte Kopie des Datensatzes gelöscht werden sollte. Standardmäßig wird der Datensatz nicht gelöscht.

```
private bool HandleDatasetDeleted(Dataset dataset)
  {
      Debug.Log(dataset.Metadata.DatasetName + " Dataset has been deleted");
      // Do clean up if necessary
      // returning true informs the corresponding dataset can be purged in the local
 storage and return false retains the local dataset
      return true;
  }
```

OnDatasetMerged

Wenn zwei zuvor unverbundene Identitäten verknüpft werden, werden alle ihre Datensätze zusammengeführt. Anwendungen werden von der Zusammenführung über den `OnDatasetsMerged`-Callback benachrichtigt:

```
public bool HandleDatasetMerged(Dataset localDataset, List<string> mergedDatasetNames)
{
    foreach (string name in mergedDatasetNames)
    {
        Dataset mergedDataset = syncManager.OpenOrCreateDataset(name);
        //Lambda function to delete the dataset after fetching it
        EventHandler<SyncSuccessEvent> lambda;
        lambda = (object sender, SyncSuccessEvent e) => {
            ICollection<string> existingValues = localDataset.GetAll().Values;
            ICollection<string> newValues = mergedDataset.GetAll().Values;

            //Implement your merge logic here

            mergedDataset.Delete(); //Delete the dataset locally
            mergedDataset.OnSyncSuccess -= lambda; //We don't want this callback to be
 fired again
            mergedDataset.OnSyncSuccess += (object s2, SyncSuccessEvent e2) => {
                localDataset.Synchronize(); //Continue the sync operation that was
 interrupted by the merge
            };
            mergedDataset.Synchronize(); //Synchronize it as deleted, failing to do so
 will leave us in an inconsistent state
        };
        mergedDataset.OnSyncSuccess += lambda;
        mergedDataset.Synchronize(); //Asnchronously fetch the dataset
    }

    // returning true allows the Synchronize to continue and false stops it
    return false;
}
```

Xamarin

Nachdem Sie einen Datensatz geöffnet oder erstellt haben, können Sie andere Callbacks festlegen, die ausgelöst werden, wenn Sie die Synchronisieren-Methode verwenden. Dies ist die Art und Weise, Ihre Callbacks zu registrieren:

```
dataset.OnSyncSuccess += this.HandleSyncSuccess;
dataset.OnSyncFailure += this.HandleSyncFailure;
dataset.OnSyncConflict = this.HandleSyncConflict;
dataset.OnDatasetMerged = this.HandleDatasetMerged;
dataset.OnDatasetDeleted = this.HandleDatasetDeleted;
```

Beachten Sie, dass `SyncSuccess` und `SyncFailure` += anstelle von = verwenden, sodass Sie mehrere Callbacks für sie abonnieren können.

OnSyncSuccess

Der `OnSyncSuccess`-Callback wird ausgelöst, wenn ein Datensatz erfolgreich von der Cloud aktualisiert wird. Wenn Sie keinen Callback definieren, erfolgt keine Benachrichtigung bei einer erfolgreichen Synchronisierung.

```
private void HandleSyncSuccess(object sender, SyncSuccessEventArgs e)
{
    // Continue with your game flow, display the loaded data, etc.
}
```

OnSyncFailure

`OnSyncFailure` wird aufgerufen, wenn während der Synchronisierung eine Ausnahmebedingung eintritt. Wenn Sie keinen Callback definieren, erfolgt keine Benachrichtigung bei einer fehlgeschlagenen Synchronisierung.

```
private void HandleSyncFailure(object sender, SyncFailureEventArgs e)
{
    Dataset dataset = sender as Dataset;
    if (dataset.Metadata != null) {
        Console.WriteLine("Sync failed for dataset : " + dataset.Metadata.DatasetName);
    } else {
        Console.WriteLine("Sync failed");
    }
}
```

OnSyncConflict

Konflikte können auftreten, wenn der gleiche Schlüssel im lokalen Speicher und im Sync Store geändert wurde. Der `OnSyncConflict`-Callback erledigt die Konfliktlösung. Wenn Sie diese Methode nicht implementieren, wird die Synchronisierung abgebrochen, wenn ein Konflikt auftritt.

```
private bool HandleSyncConflict(Dataset dataset, List < SyncConflict > conflicts)
{
  if (dataset.Metadata != null) {
    Console.WriteLine("Sync conflict " + dataset.Metadata.DatasetName);
  } else {
```

```
    Console.WriteLine("Sync conflict");
  }
  List < Amazon.CognitoSync.SyncManager.Record > resolvedRecords = new List <
 Amazon.CognitoSync.SyncManager.Record > ();
  foreach(SyncConflict conflictRecord in conflicts) {
    // SyncManager provides the following default conflict resolution methods:
    //      ResolveWithRemoteRecord - overwrites the local with remote records
    //      ResolveWithLocalRecord - overwrites the remote with local records
    //      ResolveWithValue - to implement your own logic
    resolvedRecords.Add(conflictRecord.ResolveWithRemoteRecord());
  }
  // resolves the conflicts in local storage
  dataset.Resolve(resolvedRecords);
  // on return true the synchronize operation continues where it left,
  //      returning false cancels the synchronize operation
  return true;
}
```

OnDatasetDeleted

Wenn ein Datensatz gelöscht wird, ermittelt der Amazon-Cognito-Client anhand des `OnDatasetDeleted`-Callback, ob auch die lokal zwischengespeicherte Kopie des Datensatzes gelöscht werden sollte. Standardmäßig wird der Datensatz nicht gelöscht.

```
private bool HandleDatasetDeleted(Dataset dataset)
{
    Console.WriteLine(dataset.Metadata.DatasetName + " Dataset has been deleted");
    // Do clean up if necessary
    // returning true informs the corresponding dataset can be purged in the local
 storage and return false retains the local dataset
    return true;
}
```

OnDatasetMerged

Wenn zwei zuvor unverbundene Identitäten verknüpft werden, werden alle ihre Datensätze zusammengeführt. Anwendungen werden von der Zusammenführung über den `OnDatasetsMerged`-Callback benachrichtigt:

```
public bool HandleDatasetMerged(Dataset localDataset, List<string> mergedDatasetNames)
{
    foreach (string name in mergedDatasetNames)
```

```
    {
        Dataset mergedDataset = syncManager.OpenOrCreateDataset(name);

            //Implement your merge logic here

        mergedDataset.OnSyncSuccess += lambda;
        mergedDataset.SynchronizeAsync(); //Asnchronously fetch the dataset
    }

    // returning true allows the Synchronize to continue and false stops it
    return false;
}
```

Push-Synchronisierung

⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
Wie Amazon Cognito Sync ist AWS AppSync ein Service zum Synchronisieren von Anwendungsdaten zwischen verschiedenen Geräten.
Es ermöglicht Benutzerdaten wie App-Einstellungen oder Spielstatus synchronisiert werden. Darüber hinaus erweitert es diese Möglichkeiten, indem mehrere Benutzer gemeinsam genutzte Daten synchronisieren und diese in Echtzeit zusammen nutzen können.

Amazon Cognito verfolgt die Zuordnung zwischen Identitäten und Geräten automatisch nach. Über die Push-Synchronisierung oder Push-Sync-Funktion können Sie sicherstellen, dass jede Instance einer bestimmten Identität benachrichtigt wird, wenn sich die Identitätsdaten ändern. Mit der Push-Synchronisierung wird sichergestellt, dass alle einer Identität zugeordneten Geräte bei Änderung der Synchronisationsspeicherdaten für diese Identität eine automatische Push-Benachrichtigung erhalten, die sie über die Änderung informiert.

Note

Push-Synchronisierung wird nicht für JavaScript, Unity oder Xamarin unterstützt.

Bevor Sie die Push-Synchronisierung verwenden können, müssen Sie zuerst Ihr Konto für die Push-Synchronisierung einrichten und die Push-Synchronisierung in der Amazon-Cognito-Konsole aktivieren.

Erstellen einer Amazon-Simple-Notification-Service-(Amazon-SNS)-App

Erstellen und konfigurieren Sie eine Amazon-SNS-Anwendung für die unterstützten Plattformen, wie im SNS-Entwicklerhandbuch beschrieben.

Aktivieren der Push-Synchronisierung in der Amazon-Cognito-Konsole

Sie können die Push-Synchronisierung über die Amazon-Cognito-Konsole aktivieren. Auf der Startseite der Konsole:

1. Klicken Sie auf den Namen des Identitätspools, für den Sie die Push-Synchronisierung aktivieren möchten. Die Seite Dashboard für Ihren Identitäten-Pool wird angezeigt.
2. Klicken Sie in der rechten oberen Ecke der Seite Dashboard auf Edit identity pool (Identitäten-Pool bearbeiten). Die Seite Federated Identities (Verbundidentitäten) wird angezeigt.
3. Führen Sie einen Bildlauf nach unten durch und klicken Sie zum Erweitern auf Push synchronization (Push-Synchronisierung).
4. Wählen Sie im Dropdown-Menü Service role (Servicerolle) die IAM-Rolle aus, die die Cognito-Erlaubnis zum Senden einer SNS-Benachrichtigung erteilt. Klicken Sie auf Rolle erstellen, um Rollen zu erstellen oder zu ändern, die Ihrem Identitäten-Pool in der AWS-IAM-Konsole zugeordnet sind.
5. Wählen Sie einen Plattformanwendung aus und klicken Sie dann auf Save Changes (Änderungen speichern).
6. Gewähren Sie SNS-Zugriff auf Ihre Anwendung

Konfigurieren Sie in der AWS Identity and Access Management-Konsole Ihre IAM-Rollen so, dass sie vollständigen Amazon-SNS-Zugriff haben, oder erstellen Sie eine neue Rolle, die vollen SNS-Zugriff hat. Die folgende Rollenvertrauensrichtlinie gewährt Amazon Cognito Sync eine eingeschränkte Möglichkeit, eine IAM-Rolle zu übernehmen. Amazon Cognito Sync kann diese Rolle nur übernehmen, wenn es dies im Namen des Identitätspools mit der Bedingung `aws:SourceArn` und im Namen des Kontos mit der Bedingung `aws:SourceAccount` durchführt.

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

```
                "Service": "cognito-sync.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "123456789012"
                },
                "ArnLike": {
                    "AWS:SourceArn": "arn:aws:cognito-identity:us-
east-1:123456789012:identitypool/us-east-1:177a950c-2c08-43f0-9983-28727EXAMPLE"
                }
            }
        }
    ]
}
```

Weitere Informationen zu IAM-Rollen finden Sie unter Rollen (Übertragung und Vereinigung).

Push-Synchronisierung in Ihrer App verwenden: Android

Ihre Anwendung muss die Google Play Services importieren. Sie können die neueste Version der Google Play SDK über die Android SDK Manager herunterladen. Befolgen Sie die Android-Dokumentation zur Android-Implementierung, um Ihre App zu registrieren und eine Registrierungs-ID von GCM zu erhalten. Sobald Sie die Registrierungs-ID haben, müssen Sie das Gerät bei Amazon Cognito registrieren, wie in dem nachfolgenden Codebeispiel gezeigt:

```
String registrationId = "MY_GCM_REGISTRATION_ID";
try {
    client.registerDevice("GCM", registrationId);
} catch (RegistrationFailedException rfe) {
    Log.e(TAG, "Failed to register device for silent sync", rfe);
} catch (AmazonClientException ace) {
    Log.e(TAG, "An unknown error caused registration for silent sync to fail", ace);
}
```

Sie können jetzt ein Gerät für den Empfang von Updates von einem bestimmten Datensatz abonnieren:

```
Dataset trackedDataset = client.openOrCreateDataset("myDataset");
if (client.isDeviceRegistered()) {
    try {
```

```
        trackedDataset.subscribe();
    } catch (SubscribeFailedException sfe) {
        Log.e(TAG, "Failed to subscribe to datasets", sfe);
    } catch (AmazonClientException ace) {
        Log.e(TAG, "An unknown error caused the subscription to fail", ace);
    }
}
```

Wenn Sie keine Push-Benachrichtigungen mehr von einem Datensatz erhalten möchten, rufen Sie einfach die Abmelden-Methode auf. Um alle Datensätze (oder eine bestimmte Teilmenge) im `CognitoSyncManager` Objekt zu abonnieren, verwenden Sie `subscribeAll()`:

```
if (client.isDeviceRegistered()) {
    try {
        client.subscribeAll();
    } catch (SubscribeFailedException sfe) {
        Log.e(TAG, "Failed to subscribe to datasets", sfe);
    } catch (AmazonClientException ace) {
        Log.e(TAG, "An unknown error caused the subscription to fail", ace);
    }
}
```

In Ihrer Implementierung des Android BroadcastReceiver-Objekts können Sie die neueste Version des geänderten Datensatzes prüfen und entscheiden, ob Ihre Anwendung erneut synchronisiert werden muss:

```
@Override
public void onReceive(Context context, Intent intent) {

    PushSyncUpdate update = client.getPushSyncUpdate(intent);

    // The update has the source (cognito-sync here), identityId of the
    // user, identityPoolId in question, the non-local sync count of the
    // data set and the name of the dataset. All are accessible through
    // relevant getters.

    String source = update.getSource();
    String identityPoolId = update.getIdentityPoolId();
    String identityId = update.getIdentityId();
    String datasetName = update.getDatasetName;
    long syncCount = update.getSyncCount;
```

```
    Dataset dataset = client.openOrCreateDataset(datasetName);

    // need to access last sync count. If sync count is less or equal to
    // last sync count of the dataset, no sync is required.

    long lastSyncCount = dataset.getLastSyncCount();
    if (lastSyncCount < syncCount) {
        dataset.synchronize(new SyncCallback() {
            // ...
        });
    }

}
```

Die folgenden Schlüssel sind in der Push-Benachrichtigung-Nutzlast verfügbar:

- `source`: Cognito-Synchronisierung Diese kann als ein Unterscheidungsmerkmal zwischen Benachrichtigungen dienen.
- `identityPoolId`Die Identitätspool-ID. Diese kann für die Validierung verwendet werden oder dient als zusätzliche Informationen, ist jedoch aus Sicht des Empfängers nicht unbedingt erforderlich.
- `identityId`Die Identitäts-ID innerhalb des Pools.
- `datasetName`Der Namen des aktualisierten Datensatzes. Dieser ist für den openOrCreateDataset-Aufruf verfügbar.
- `syncCount`: Die Synchronisierungszahl für den Remote-Datensatz. Damit können Sie sicherstellen, dass lokale Datensatz nicht mehr auf dem neuesten Stand ist und die eingehende Synchronisierung neu ist.

Push-Sync in Ihrer App verwenden: iOS – Objective-C

Um ein Geräte-Token für Ihre Anwendung zu erhalten, befolgen Sie die Apple Dokumentation zum Registrieren von für Remote-Benachrichtigungen. Nachdem Sie das Geräte-Token als NSData-Objekt von APNs erhalten haben, müssen Sie das Gerät bei Amazon Cognito mithilfe der `registerDevice:`-Methode des Sync-Clienten registrieren, wie unten gezeigt:

```
AWSCognito *syncClient = [AWSCognito defaultCognito];
    [[syncClient registerDevice: devToken] continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to registerDevice: %@", task.error);
```

```
        } else {
            NSLog(@"Successfully registered device with id: %@", task.result);
        }
        return nil;
      }
    ];
```

In Debug-Modus wird Ihr Gerät mit der APNs-Sandbox registriert, im Release-Modus wird es mit APNs registriert. Um Updates von einem bestimmten Datensatz zu erhalten, verwenden Sie die `subscribe`-Methode:

```
[[[syncClient openOrCreateDataset:@"MyDataset"] subscribe]
 continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to subscribe to dataset: %@", task.error);
        } else {
            NSLog(@"Successfully subscribed to dataset: %@", task.result);
        }
        return nil;
      }
    ];
```

Wenn Sie keine Push-Benachrichtigungen mehr von einem Datensatz erhalten möchten, rufen Sie einfach die `unsubscribe`-Methode auf.

```
[[[syncClient openOrCreateDataset:@”MyDataset”] unsubscribe]
 continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to unsubscribe from dataset: %@", task.error);
        } else {
            NSLog(@"Successfully unsubscribed from dataset: %@", task.result);
        }
        return nil;
      }
    ];
```

Um alle Datensätze im `AWSCognito`-Objekt zu abonnieren, rufen Sie `subscribeAll` auf:

```
[[syncClient subscribeAll] continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to subscribe to all datasets: %@", task.error);
```

```
        } else {
            NSLog(@"Successfully subscribed to all datasets: %@", task.result);
        }
        return nil;
      }
    ];
```

Bevor Sie `subscribeAll` aufrufen, sollten Sie jeden Datensatz mindestens einmal synchronisieren, damit diese Datensätze auf dem Server vorliegen.

Um auf Push-Benachrichtigungen zu reagieren, müssen Sie die `didReceiveRemoteNotification`-Methode auf Ihre App delegieren:

```
- (void)application:(UIApplication *)application didReceiveRemoteNotification:
(NSDictionary *)userInfo
    {
        [[NSNotificationCenter defaultCenter]
 postNotificationName:@"CognitoPushNotification" object:userInfo];
    }
```

Wenn Sie eine Benachrichtigung mithilfe des Benachrichtigungs-Handlers veröffentlichen, können Sie die Benachrichtigung an anderer Stelle in der Anwendung beantworten, an der Sie einen Handle für den Datensatz haben Wenn Sie die Benachrichtigung auf folgende Weise abonnieren...

```
[[NSNotificationCenter defaultCenter] addObserver:self
 selector:@selector(didReceivePushSync:)
    name: :@"CognitoPushNotification" object:nil];
```

... können Sie auf die Benachrichtigung auf folgende Weise reagieren:

```
- (void)didReceivePushSync:(NSNotification*)notification
    {
        NSDictionary * data = [(NSDictionary *)[notification object]
 objectForKey:@"data"];
        NSString * identityId = [data objectForKey:@"identityId"];
        NSString * datasetName = [data objectForKey:@"datasetName"];
        if([self.dataset.name isEqualToString:datasetName] && [self.identityId
 isEqualToString:identityId]){
            [[self.dataset synchronize] continueWithBlock:^id(AWSTask *task) {
                if(!task.error){
                    NSLog(@"Successfully synced dataset");
```

```
                }
                return nil;
            }];
        }
    }
```

Die folgenden Schlüssel sind in der Push-Benachrichtigung-Nutzlast verfügbar:

- `source`: Cognito-Synchronisierung Diese kann als ein Unterscheidungsmerkmal zwischen Benachrichtigungen dienen.
- `identityPoolId`Die Identitätspool-ID. Diese kann für die Validierung verwendet werden oder dient als zusätzliche Informationen, ist jedoch aus Sicht des Empfängers nicht unbedingt erforderlich.
- `identityId`Die Identitäts-ID innerhalb des Pools.
- `datasetName`Der Namen des aktualisierten Datensatzes. Dieser ist für den `openOrCreateDataset`-Aufruf verfügbar.
- `syncCount`: Die Synchronisierungszahl für den Remote-Datensatz. Damit können Sie sicherstellen, dass lokale Datensatz nicht mehr auf dem neuesten Stand ist und die eingehende Synchronisierung neu ist.

Push-Sync in Ihrer App verwenden: iOS – Swift

Um ein Geräte-Token für Ihre Anwendung zu erhalten, befolgen Sie die Apple Dokumentation zum Registrieren von für Remote-Benachrichtigungen. Nachdem Sie das Geräte-Token als NSData-Objekt von APNs erhalten haben, müssen Sie das Gerät bei Amazon Cognito mithilfe der registerDevice:-Methode des Sync-Clienten registrieren, wie unten gezeigt:

```
let syncClient = AWSCognito.default()
syncClient.registerDevice(devToken).continueWith(block: { (task: AWSTask!) ->
 AnyObject! in
    if (task.error != nil) {
        print("Unable to register device: " + task.error.localizedDescription)

    } else {
        print("Successfully registered device with id: \(task.result)")
    }
    return task
})
```

In Debug-Modus wird Ihr Gerät mit der APNs-Sandbox registriert, im Release-Modus wird es mit APNs registriert. Um Updates von einem bestimmten Datensatz zu erhalten, verwenden Sie die `subscribe`-Methode:

```
syncClient.openOrCreateDataset("MyDataset").subscribe().continueWith(block: { (task:
 AWSTask!) -> AnyObject! in
  if (task.error != nil) {
      print("Unable to subscribe to dataset: " + task.error.localizedDescription)

  } else {
      print("Successfully subscribed to dataset: \(task.result)")
  }
  return task
})
```

Wenn Sie keine Push-Benachrichtigungen mehr von einem Datensatz erhalten möchten, rufen Sie die `unsubscribe`-Methode auf:

```
syncClient.openOrCreateDataset("MyDataset").unsubscribe().continueWith(block: { (task:
 AWSTask!) -> AnyObject! in
  if (task.error != nil) {
      print("Unable to unsubscribe to dataset: " + task.error.localizedDescription)

  } else {
      print("Successfully unsubscribed to dataset: \(task.result)")
  }
  return task
})
```

Um alle Datensätze im `AWSCognito`-Objekt zu abonnieren, rufen Sie `subscribeAll` auf:

```
syncClient.openOrCreateDataset("MyDataset").subscribeAll().continueWith(block: { (task:
 AWSTask!) -> AnyObject! in
  if (task.error != nil) {
      print("Unable to subscribe to all datasets: " + task.error.localizedDescription)

  } else {
      print("Successfully subscribed to all datasets: \(task.result)")
  }
  return task
})
```

Bevor Sie `subscribeAll` aufrufen, sollten Sie jeden Datensatz mindestens einmal synchronisieren, damit diese Datensätze auf dem Server vorliegen.

Um auf Push-Benachrichtigungen zu reagieren, müssen Sie die `didReceiveRemoteNotification`-Methode auf Ihre App delegieren:

```
func application(application: UIApplication, didReceiveRemoteNotification userInfo:
 [NSObject : AnyObject],
  fetchCompletionHandler completionHandler: (UIBackgroundFetchResult) -> Void) {

 NSNotificationCenter.defaultCenter().postNotificationName("CognitoPushNotification",
 object: userInfo)
})
```

Wenn Sie eine Benachrichtigung mithilfe des Benachrichtigungs-Handlers veröffentlichen, können Sie die Benachrichtigung an anderer Stelle in der Anwendung beantworten, an der Sie einen Handle für den Datensatz haben Wenn Sie die Benachrichtigung auf folgende Weise abonnieren...

```
NSNotificationCenter.defaultCenter().addObserver(observer:self,
   selector:"didReceivePushSync:",
   name:"CognitoPushNotification",
   object:nil)
```

... können Sie auf die Benachrichtigung auf folgende Weise reagieren:

```
func didReceivePushSync(notification: NSNotification) {
    if let data = (notification.object as! [String: AnyObject])["data"] as? [String:
 AnyObject] {
        let identityId = data["identityId"] as! String
        let datasetName = data["datasetName"] as! String

        if self.dataset.name == datasetName && self.identityId == identityId {
          dataset.synchronize().continueWithBlock {(task) -> AnyObject! in
              if task.error == nil {
                print("Successfully synced dataset")
              }
              return nil
          }
        }
    }
}
```

Die folgenden Schlüssel sind in der Push-Benachrichtigung-Nutzlast verfügbar:

- `source`: Cognito-Synchronisierung Diese kann als ein Unterscheidungsmerkmal zwischen Benachrichtigungen dienen.
- `identityPoolId`Die Identitätspool-ID. Diese kann für die Validierung verwendet werden oder dient als zusätzliche Informationen, ist jedoch aus Sicht des Empfängers nicht unbedingt erforderlich.
- `identityId`Die Identitäts-ID innerhalb des Pools.
- `datasetName`Der Namen des aktualisierten Datensatzes. Dieser ist für den `openOrCreateDataset`-Aufruf verfügbar.
- `syncCount`: Die Synchronisierungszahl für den Remote-Datensatz. Damit können Sie sicherstellen, dass lokale Datensatz nicht mehr auf dem neuesten Stand ist und die eingehende Synchronisierung neu ist.

Amazon-Cognito-Streams

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync ist AWS AppSync ein Service zum Synchronisieren von Anwendungsdaten zwischen verschiedenen Geräten.
> Es ermöglicht Benutzerdaten wie App-Einstellungen oder Spielstatus synchronisiert werden. Darüber hinaus erweitert es diese Möglichkeiten, indem mehrere Benutzer gemeinsam genutzte Daten synchronisieren und diese in Echtzeit zusammen nutzen können.

Amazon-Cognito-Streams bietet Entwicklern Kontrolle und einen detaillierten Überblick über ihre in Amazon Cognito gespeicherten Daten. Entwickler können einen Kinesis-Stream für den Empfang von Ereignissen konfigurieren, sobald Daten aktualisiert und synchronisiert werden. Amazon Cognito kann jede Datensatzänderung per Push in Echtzeit an einen Kinesis-Stream in Ihrem Besitz übertragen.

Mit Amazon-Cognito-Streams können Sie alle Ihre Synchronisierungsdaten auf Kinesis verschieben, wo sie anschließend auf ein Data-Warehouse-Tool wie Amazon Redshift zur weiteren Analyse gestreamt werden können. Weitere Informationen zu Kinesis finden Sie unter Erste Schritte mit Amazon Kinesis.

Konfigurieren von Streams

Sie können Amazon-Cognito-Streams in der Amazon-Cognito-Konsole einrichten. Zum Aktivieren von Amazon-Cognito-Streams müssen Sie in der Amazon-Cognito-Konsole den Kinesis-Stream für die Veröffentlichung sowie eine IAM-Rolle auswählen, der Amazon Cognito die Berechtigung erteilt, Ereignisse in den ausgewählten Stream zu schreiben.

Auf der Startseite der Konsole:

1. Klicken Sie auf den Namen des Identitätspools, für den Sie Amazon-Cognito-Streams einrichten möchten. Die Seite Dashboard für Ihren Identitäten-Pool wird angezeigt.
2. Klicken Sie in der rechten oberen Ecke der Seite Dashboard auf Edit identity pool (Identitäten-Pool bearbeiten). Die Seite Manage Federated Identities wird angezeigt.
3. Scrollen Sie nach unten und wählen Sie Cognito Streams, um diesen zu erweitern.
4. Wählen Sie im Dropdown-Menü Stream name (Stream-Name) den Namen eines vorhandenen Kinesis-Streams aus. Alternativ klicken Sie auf Create stream (Stream erstellen), um ihn zu erstellen. Geben Sie dazu einen Stream-Namen und die Anzahl der Shards ein. Weitere Informationen zu Shards und Hilfe zum Schätzen der Anzahl der Shards, die für Ihren Stream erforderlich sind, finden Sie im Kinesis-Entwicklerhandbuch.
5. Wählen Sie im Dropdown-Menü Rolle veröffentlichen die IAM-Rolle aus, die Ihnen die Amazon-Cognito-Berechtigung gewährt, Streams zu veröffentlichen. Klicken Sie auf Rolle erstellen, um Rollen zu erstellen oder zu ändern, die Ihrem Identitäten-Pool in der AWS-IAM-Konsole zugeordnet sind.
6. Wählen Sie im Dropdown-Menü Stream status (Stream-Status) die Option Enabled (Aktiviert), um Stream-Aktualisierungen zu ermöglichen. Klicken Sie auf Save Changes (Änderungen speichern).

Nachdem Sie erfolgreich Amazon-Cognito-Streams konfiguriert haben, werden alle nachfolgenden Aktualisierungen an den Datensätzen in diesem Identitätspool an den Stream gesendet.

Stream-Inhalte

Jeder Datensatz, der an den Stream gesendet wird, stellt eine einzelne Synchronisierung dar. Hier finden Sie ein Beispiel für einen Datensatz, der an den Stream gesendet wurde:

```
{
    "identityPoolId": "Pool Id",
    "identityId": "Identity Id",
    "dataSetName": "Dataset Name",
    "operation": "(replace|remove)",
```

```
    "kinesisSyncRecords": [
        {
            "key": "Key",
            "value": "Value",
            "syncCount": 1,
            "lastModifiedDate": 1424801824343,
            "deviceLastModifiedDate": 1424801824343,
            "op": "(replace|remove)"
        },
        ...
    ],
    "lastModifiedDate": 1424801824343,
    "kinesisSyncRecordsURL": "S3Url",
    "payloadType": "(S3Url|Inline)",
    "syncCount": 1
}
```

Für Updates, die größer als die maximale Kinesis-Nutzlastgröße von 1 MB sind, schließt Amazon Cognito eine vorsignierte Amazon-S3-URL ein, die den vollständigen Inhalt der Aktualisierung enthält.

Nachdem Sie Amazon-Cognito-Streams konfiguriert haben, deaktivieren Sie Amazon-Cognito-Streams, wenn Sie den Kinesis-Stream löschen oder die Rollenvertrauensberechtigung ändern, sodass die Rolle nicht mehr von Amazon Cognito Sync übernommen werden kann. Sie müssen den Kinesis-Stream entweder neu erstellen oder die Rolle reparieren und anschließend den Stream erneut aktivieren.

Massen-Veröffentlichung

Sobald Sie Amazon-Cognito-Streams konfiguriert haben, können Sie eine Massen-Veröffentlichung der vorhandenen Daten in Ihrem Identitätspool ausführen. Nachdem Sie eine Massen-Veröffentlichungsoperation entweder über die Konsole oder direkt über die API initiiert haben, beginnt Amazon Cognito mit der Veröffentlichung dieser Daten auf demselben Stream, auf dem Sie Ihre Updates erhalten.

Amazon Cognito garantiert nicht die Eindeutigkeit der Daten, die bei Verwendung der Massen-Veröffentlichungsoperation an den Stream gesendet werden. Sie können die gleiche Aktualisierung sowohl als Update als auch als Teil einer Massen-Veröffentlichung erhalten. Beachten Sie dies beim Verarbeiten der Datensätze aus Ihrem Stream.

Um alle Ihre Streams als Teil einer Massen-Veröffentlichung zu veröffentlichen, führen Sie die Schritte 1 bis 6 unter "Configuring Streams" aus, und klicken Sie dann auf "Massen-Veröffentlichung

starten". Sie sind auf eine laufende Massen-Veröffentlichungsoperation zu einem bestimmten Zeitpunkt und eine erfolgreiche Massen-Veröffentlichungsanfrage alle 24 Stunden beschränkt.

Amazon-Cognito-Ereignisse

⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
Wie Amazon Cognito Sync ist AWS AppSync ein Service zum Synchronisieren von Anwendungsdaten zwischen verschiedenen Geräten.
Es ermöglicht Benutzerdaten wie App-Einstellungen oder Spielstatus synchronisiert werden. Darüber hinaus erweitert es diese Möglichkeiten, indem mehrere Benutzer gemeinsam genutzte Daten synchronisieren und diese in Echtzeit zusammen nutzen können.

Mit Amazon-Cognito-Ereignissen können Sie eine AWS Lambda-Funktion als Reaktion auf wichtige Ereignisse in Amazon Cognito ausführen. Amazon Cognito löst das Sync-Auslöser-Ereignis aus, wenn ein Datensatz synchronisiert wird. Sie können das Sync Trigger-Ereignis verwenden, um eine Aktion auszuführen, wenn ein Benutzer Daten aktualisiert. Die Funktion kann die Daten bewerten und optional manipulieren, bevor sie in der Cloud gespeichert und auf den anderen Geräten des Benutzers synchronisiert werden. Dies ist nützlich zum Validieren der Daten, die vom Gerät kommen, bevor sie mit den anderen Geräten des Benutzers synchronisiert werden, oder zum Aktualisieren anderer Werte im Datensatz basierend auf eingehenden Daten, wie z. B. die Ausstellung einer Auszeichnung, wenn ein Spieler eine neue Ebene erreicht.

Die nachfolgenden Schritte führen Sie durch die Einrichtung einer Lambda-Funktion, die bei jeder Synchronisierung eines Amazon-Cognito-Datensatzes ausgeführt wird.

> **Note**
>
> Bei der Verwendung von Amazon-Cognito-Ereignissen können Sie nur die Anmeldeinformationen verwenden, die Sie von der Amazon-Cognito-Identität erhalten haben. Wenn Sie über eine zugewiesene Lambda-Funktion verfügen, jedoch `UpdateRecords` mit AWS-Konto-Anmeldeinformationen (Entwickler-Anmeldeinformationen) aufrufen, wird Ihre Lambda-Funktion nicht aufgerufen.

Erstellen einer Funktion in AWS Lambda

Um Lambda in Amazon Cognito zu integrieren, müssen Sie zunächst eine Funktion in Lambda erstellen. Hierzu gehen Sie wie folgt vor:

Auswählen der Lambda-Funktion in Amazon Cognito

1. Öffnen Sie die Lambda-Konsole.
2. Klicken Sie auf Create a Lambda function.
3. Auf dem Bildschirm "Blueprint auswählen" suchen und wählen Sie "Cognito-Sync-Trigger".
4. Auf dem Bildschirm "Ereignisquellen konfigurieren" lassen Sie den Ereignisquellentyp auf "Cognito-Sync-Trigger" eingestellt, und wählen Sie den Identitätspool aus. Klicken Sie auf Weiter.

> **Note**
>
> Wenn Sie einen Amazon-Cognito-Sync-Trigger außerhalb der Konsole konfigurieren, müssen Sie ressourcenbasierte Berechtigungen für Lambda hinzufügen, damit Amazon Cognito die Funktion aufrufen kann. Sie können diese Berechtigung von der Lambda-Konsole aus hinzufügen (siehe Verwenden ressourcenbasierter Richtlinien für AWS Lambda oder über die Lambda-Funktion AddPermission.
> Beispiel für ressourcenbasierte Lambda-Richtlinie
> Die folgende ressourcenbasierte AWS Lambda-Richtlinie gewährt Amazon Cognito eine eingeschränkte Möglichkeit, eine Lambda-Funktion aufzurufen. Amazon Cognito kann diese Rolle nur aufrufen, wenn es dies im Namen des Identitätspools mit der Bedingung `aws:SourceArn` und im Namen des Kontos mit der Bedingung `aws:SourceAccount` durchführt.
>
> ```
> {
>     "Version": "2012-10-17",
>     "Id": "default",
>     "Statement": [
>         {
>             "Sid": "lambda-allow-cognito-my-function",
>             "Effect": "Allow",
>             "Principal": {
>                 "Service": "cognito-sync.amazonaws.com"
>             },
>             "Action": "lambda:InvokeFunction",
>             "Resource": "<your Lambda function ARN>",
>             "Condition": {
>                 "StringEquals": {
>                     "AWS:SourceAccount": "<your account number>"
> ```

```
                },
                "ArnLike": {
                    "AWS:SourceArn": "<your identity pool ARN>"
                }
            }
        }
    ]
}
```

5. Auf dem Bildschirm "Funktion konfigurieren" geben Sie einen Namen und eine Beschreibung für Ihre Funktion ein. Lassen Sie "Runtime" auf "Node.js" eingestellt. Lassen Sie den Code für unser Beispiel unverändert. Das Standardbeispiel nimmt keine Änderungen an den synchronisierten Daten vor. Es protokolliert lediglich die Tatsache, dass das Amazon-Cognito-Sync-Auslöser-Ereignis aufgetreten ist. Lassen Sie den Handler-Namen "index.handler" unverändert. Für „Rolle“ wählen Sie eine IAM-Rolle, die Ihnen Code-Berechtigung für den Zugriff auf gewährt AWS Lambda. Angaben zum Ändern von Rollen finden Sie auf der IAM-Konsole. Lassen Sie die erweiterten Einstellungen unverändert. Klicken Sie auf Weiter.
6. Auf dem Bildschirm "Prüfen" überprüfen Sie die Details und klicken Sie auf "Funktion erstellen". Auf der nächsten Seite wird Ihre neue Lambda-Funktion angezeigt.

Sie haben nun eine geeignete Funktion in Lambda geschrieben. Nun müssen Sie diese Funktion als Handler für das Amazon-Cognito-Sync-Auslöser-Ereignis auswählen. In den nachfolgenden Schritten wird dieser Prozess beschrieben.

Auf der Startseite der Konsole:

1. Klicken Sie auf den Namen des Identitätspools, für den Sie Amazon-Cognito-Ereignisse einrichten möchten. Die Seite "Dashboard" für Ihren Identitätspool wird angezeigt.
2. Klicken Sie in der oberen rechten Ecke der Seite Dashboard auf Manage Federated Identities. Die Seite Manage Federated Identities wird angezeigt.
3. Führen Sie einen Bildlauf nach unten durch und klicken Sie auf "Cognito Events", um es zu erweitern.
4. Wählen Sie im Dropdown-Menü „Sync-Auslöser“ die Lambda-Funktion aus, die Sie beim Eintreten eines Sync-Ereignisses auslösen möchten.
5. Klicken Sie auf Save Changes.

Ihre Lambda-Funktion wird jetzt jedes Mal ausgeführt, wenn ein Datensatz synchronisiert wird. Im nächsten Abschnitt wird erläutert, wie Sie die Daten in Ihrer Funktion lesen und ändern, während sie synchronisiert werden.

Schreiben einer Lambda-Funktion für Sync-Auslöser

Sync-Auslöser befolgen das Programmierungsmuster der Dienstanbieter-Schnittstellen. Amazon Cognito stellt Ihrer Lambda-Funktion die Eingabe im folgenden JSON-Format zur Verfügung.

```
{
  "version": 2,
  "eventType": "SyncTrigger",
  "region": "us-east-1",
  "identityPoolId": "identityPoolId",
  "identityId": "identityId",
  "datasetName": "datasetName",
  "datasetRecords": {
    "SampleKey1": {
      "oldValue": "oldValue1",
      "newValue": "newValue1",
      "op": "replace"
    },
    "SampleKey2": {
      "oldValue": "oldValue2",
      "newValue": "newValue2",
      "op": "replace"
    },..
  }
}
```

Amazon Cognito erwartet den Rückgabewert der Funktion im gleichen Format wie die Eingabe.

Beachten Sie beim Schreiben von Funktionen für das Sync-Auslöser-Ereignis Folgendes:

- Wenn Amazon Cognito die Lambda-Funktion während UpdateRecords aufruft, muss die Funktion innerhalb von 5 Sekunden reagieren. Andernfalls generiert der Amazon-Cognito-Sync-Service eine `LambdaSocketTimeoutException`-Ausnahme. Sie können diesen Timeout-Wert nicht erhöhen.
- Wenn Ihnen eine `LambdaThrottledException`-Ausnahme angezeigt wird, versuchen Sie den Synchronisierungsvorgang erneut, um die Datensätze zu aktualisieren.
- Amazon Cognito stellt alle Datensätze im Datensatz als Eingabe für die Funktion zur Verfügung.

- Zeichnet auf, dass in den Benutzeraktualisierungen in der App das op-Feld auf `replace` festgelegt ist. In den gelöschten Datensätzen ist das op-Feld auf `remove` festgelegt.
- Sie können jeden Datensatz ändern, selbst wenn der Benutzer der App den Datensatz nicht aktualisiert.
- Alle Felder mit Ausnahme von datasetRecords sind schreibgeschützt. Ändern Sie diese nicht. Wenn Sie diese Felder ändern, können Sie die Datensätze nicht aktualisieren.
- Um den Wert eines Datensatzes zu ändern, aktualisieren Sie den Wert und setzen Sie op auf `replace`.
- Zum Entfernen eines Datensatzes setzen Sie op auf `remove` oder setzen den Wert auf Null.
- Um einen Datensatz hinzuzufügen, fügen einen neuen Datensatz zum datasetRecords-Array hinzu.
- Amazon Cognito ignoriert jeden ausgelassenen Datensatz in der Antwort, wenn Amazon Cognito den Datensatz aktualisiert

Beispiel-Lambda-Funktion

Das folgende Beispiel einer Lambda-Funktion zeigt das Abrufen, Modifizieren und Entfernen von Daten.

```
console.log('Loading function');

exports.handler = function(event, context) {
    console.log(JSON.stringify(event, null, 2));

    //Check for the event type
    if (event.eventType === 'SyncTrigger') {

        //Modify value for a key
        if('SampleKey1' in event.datasetRecords){
            event.datasetRecords.SampleKey1.newValue = 'ModifyValue1';
            event.datasetRecords.SampleKey1.op = 'replace';
        }

        //Remove a key
        if('SampleKey2' in event.datasetRecords){
            event.datasetRecords.SampleKey2.op = 'remove';
        }

        //Add a key
        if(!('SampleKey3' in event.datasetRecords)){
```

```
            event.datasetRecords.SampleKey3={'newValue':'ModifyValue3', 'op' :
 'replace'};
        }

    }
    context.done(null, event);
};
```

Verwenden der Amazon-Cognito-Konsole

Sie können dieAmazon-Cognito-Konsole verwenden, um Benutzerpools und Identitäten-Pools zu erstellen und zu verwalten.

Dieses Handbuch bietet step-by-step exemplarische Vorgehensweisen für allgemeine Amazon Cognito Cognito-Benutzerpool-Aufgaben in der Amazon Cognito Cognito-Konsole.

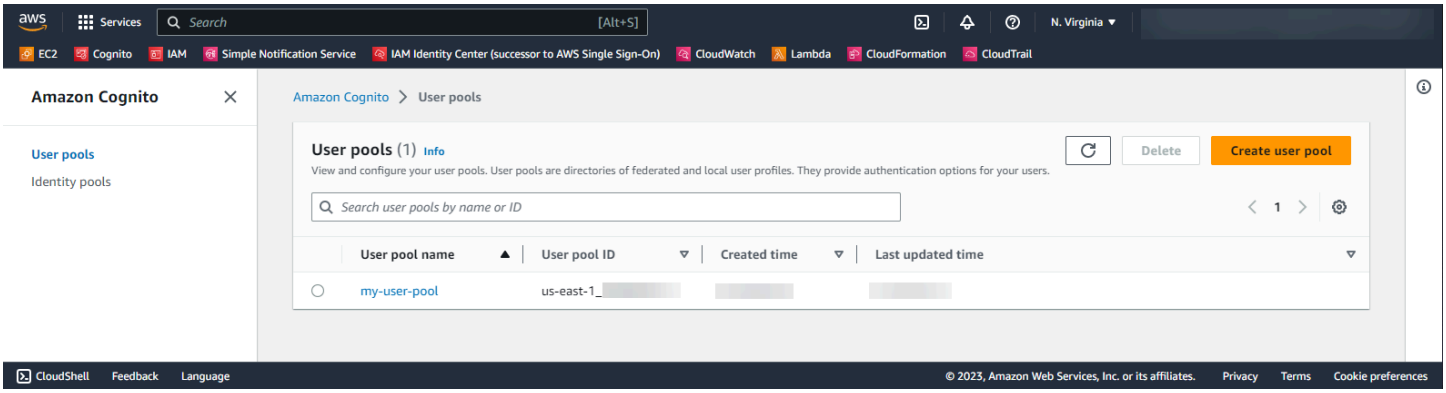
Melden Sie sich bei der Amazon-Cognito-Konsole an

1. Um Amazon Cognito nutzen zu können, müssen Sie sich für ein AWS Konto registrieren.
2. Melden Sie sich bei der Amazon-Cognito-Konsole an. Möglicherweise werden Sie zur Eingabe Ihrer AWS Anmeldeinformationen aufgefordert.
3. Um einen Benutzerpool zu erstellen oder zu bearbeiten, wählen Sie aus dem linken Navigationsbereich User Pools (Benutzerpools) aus.

   Weitere Informationen finden Sie unter Erste Schritte mit Benutzerpools.
4. Wählen Sie zum Erstellen oder Bearbeiten eines Identitätspools Identitätspools aus. Sie werden zur ursprünglichen Konsole für Amazon-Cognito-Identitätspools weitergeleitet.

   Weitere Informationen finden Sie unter Erste Schritte mit Amazon Cognito Cognito-Identitätspools.

Die Amazon Cognito Cognito-Konsole ist ein Teil der AWS Management Console, die Informationen zu Ihrem Konto und Ihrer Abrechnung bereitstellt. Weitere Informationen finden Sie unter Arbeiten mit der AWS Management Console.



Themen

Die Benutzerpool-Konsole

Wählen Sie in der Ansicht Benutzerpools der Amazon-Cognito-Konsole einen Benutzerpool aus der Liste aus, um Details anzuzeigen. In der Detailansicht finden Sie in der Benutzerpool-Übersicht oben in der Konsole grundlegende Informationen zu Ihrem Benutzerpool. Mit den folgenden Registerkarten wird Ihre Benutzerpool-Konfiguration in verwandte Funktionen organisiert.

Benutzer

Die Registerkarte Benutzer enthält Informationen über Benutzer und Benutzerimporte aus CSV-Dateien. Auf dieser Registerkarte können Sie Benutzer hinzufügen, entfernen und bearbeiten.

Referenzen

- Verwalten von Benutzern in Ihrem Benutzerpool
- Importieren von Benutzern aus einer CSV-Datei in Benutzerpools

Gruppen

Die Registerkarte Gruppen enthält Informationen über Benutzergruppen. Sie können die Mitgliedschaft in Gruppen hinzufügen, modifizieren und ändern sowie die IAM-Rollen ändern, die Gruppen für die Identitätspool-Integration zugeordnet sind.

Referenzen

- Hinzufügen von Gruppen zu einem Benutzerpool

Anmeldeerfahrung

Die Registerkarte Anmeldeerfahrung enthält Informationen darüber, wie sich Benutzer bei Ihrem Benutzerpool anmelden. Auf dieser Registerkarte finden Sie externe Identitätsanbieter, Benutzernamenoptionen, die Passwortrichtlinie, die Konfiguration der Multi-Faktor-Authentifizierung (MFA), das Verhalten bei vergessenen Passwörtern und die Geräteerkennung. Sie können Identitätsanbieter hinzufügen und modifizieren sowie das allgemeine Anmeldeverhalten Ihres Benutzerpools ändern.

Referenzen

- Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter

- Anpassen von Anmeldeattributen
- Hinzufügen von Benutzerpool-Passwortanforderungen
- Hinzufügen der MFA zu einem Benutzerpool
- Wiederherstellen von Benutzerkonten
- Verwendung von Benutzergeräten in Ihrem Benutzerpool

Registrierungserfahrung

Die Registerkarte Registrierungserfahrung enthält Informationen zur Self-Service-Registrierung, zu den erforderlichen Attributen, zur Überprüfung von Telefonnummern und E-Mail-Adressen sowie zu benutzerdefinierten Attributen.

Referenzen

- Registrieren und Bestätigen von Benutzerkonten
- Attribute für den Benutzerpool
- Überprüfen von Kontaktinformationen bei der Anmeldung

Messaging

Die Registerkarte Messaging enthält Informationen über die AWS-Services , die Sie verwenden möchten, um Ihren Benutzern E-Mail- und SMS-Nachrichten zu senden, und das Format der Nachrichten, die Sie ihnen senden möchten.

Referenzen

- E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools
- Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools
- Konfigurieren der Nachrichten zur SMS- und E-Mail-Verifizierung und zur Einladung von Benutzern

App-Integration

Die Registerkarte App-Integration enthält Informationen zu den App-Clients des Benutzerpools, zur Domain, die Sie Ihren Benutzerpool-Serviceendpunkten zuweisen, zu API-Ressourcenservern, zur gehosteten Benutzeroberfläche und zu erweiterten Sicherheitsfunktionen. Sie können die einzelnen App-Clients detailliert aufschlüsseln, um Folgendes zu konfigurieren.

1. Token-Einstellungen
2. Rückruf-URLs
3. Authentifizierungsabläufe

4. Attributberechtigungen
5. App-spezifische erweiterte Sicherheits- und gehostete Benutzeroberflächeneinstellungen
6. Amazon-Pinpoint-Analysen

Referenzen

- App-Clients für Benutzerpools
- Einrichtung und Verwendung der gehosteten Benutzeroberfläche und der Verbundendpunkte in Amazon Cognito
- Konfigurieren einer Benutzerpool-Domäne
- Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern
- Hinzufügen erweiterter Sicherheit zu einem Benutzerpool.
- Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools

Benutzerpool-Eigenschaften

Die Registerkarte Benutzerpool-Eigenschaften enthält Informationen zur Konfiguration des Benutzerpools, die nicht direkt mit Benutzern zusammenhängen: Lambda-Trigger, AWS WAF Web-ACL-Schutz, Löschschutz und Ressourcen-Tags.

Referenzen

- Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern
- Eine AWS WAF Web-ACL einem Benutzerpool zuordnen
- Löschschutz für Benutzerpools
- Markieren Sie Ihre Ressourcen AWS

Die Identitätspool-Konsole

Wählen Sie in der Ansicht Identitätspools der Amazon-Cognito-Konsole einen Identitätspool aus der Liste aus, um Details anzuzeigen. In der Detailansicht finden Sie in der Identitätspool – Übersicht oben in der Konsole grundlegende Informationen zu Ihrem Benutzerpool. Mit den folgenden Registerkarten wird Ihre Benutzerpool-Konfiguration in verwandte Funktionen organisiert.

Benutzerstatistiken

Auf der Registerkarte Benutzerstatistiken werden statistische Informationen über die Benutzer angezeigt, die Identitäten in Ihrem Identitätspool generiert haben. Auf dieser Registerkarte können Sie keine Einstellungen für den Identitätspool konfigurieren.

Identitäts-Browser

Die Registerkarte Identitäts-Browser enthält Informationen zu den einzelnen Identitäten, die Benutzer in Ihrem Identitätspool generiert haben. Sie können Identitäten anzeigen und löschen.

Referenzen

- Erste Schritte mit Amazon Cognito Cognito-Identitätspools

Benutzerzugriff

Die Registerkarte Benutzerzugriff enthält Informationen zu den Identitätsanbietern, die Sie mit Ihrem Identitätspool verknüpft haben, zu den Entwickleranbietern, den Standard-IAM-Rollen, die Identitäten zugewiesen sind, und zur Konfiguration des nicht authentifizierten Gastzugriffs. Sie können die einzelnen Identitätsanbieter detailliert aufschlüsseln, um Folgendes zu konfigurieren.

1. Rollenbasierte Zugriffskontrolle mit der Option Auswahl der IAM-Rolle
2. Attributbasierte Zugriffskontrolle mit der Option Attribute für die Zugriffskontrolle

Referenzen

- Externe Identitätsanbieter von Identitäten-Pools
- IAM-Rollen
- Authentifizierte und nicht authentifizierte Identitäten
- Entwicklerauthentifizierte Identitäten (Identitätspools)
- Verwenden der rollenbasierten Zugriffskontrolle
- Verwenden von Attributen für Zugriffskontrolle

Eigenschaften des Identitätspools

Die Registerkarte Eigenschaften des Identitätspools enthält Informationen zur Konfiguration verschiedener Identitätspools: grundlegende (klassische) Authentifizierung und Ressourcen-Tags.

- Identitäten-Pools (Verbundidentitäten) – Authentifizierungsablauf
- Verschlagworten Sie Ihre Ressourcen AWS

Sicherheit in Amazon Cognito

Cloud-Sicherheit hat AWS höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der übergreifenden Verantwortlichkeit beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der AWS . Weitere Informationen zu den Compliance-Programmen, die für Amazon Cognito 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 zeigt Ihnen, wie Sie das Modell der geteilten Verantwortung bei der Verwendung von Amazon Cognito einsetzen können. Es zeigt Ihnen, wie Sie Amazon Cognito konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, mit denen Sie Ihre Amazon Cognito Cognito-Ressourcen überwachen und sichern können.

Inhalt

Datenschutz in Amazon Cognito

Das Modell der AWS gemeinsamen Verantwortung gilt für den Datenschutz in Amazon Cognito (Amazon Cognito). Wie in diesem Modell beschrieben, AWS ist es für den Schutz der globalen Infrastruktur verantwortlich, auf der die gesamte Cloud betrieben wird. AWS Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Dieser Inhalt umfasst die Sicherheitskonfiguration und die Verwaltungsaufgaben für die AWS Dienste, die Sie verwenden. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz.

Aus Datenschutzgründen empfehlen wir, die AWS Kontoanmeldeinformationen zu schützen und individuelle Benutzerkonten mit AWS Identity and Access Management (IAM) einzurichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem sollten Sie die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Verwenden Sie SSL/TLS, um mit Ressourcen zu kommunizieren. AWS
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen Standardsicherheitskontrollen innerhalb der AWS Dienste.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu sichern.

Wir empfehlen dringend, in Freitextfeldern wie z. B. im Feld Name keine sensiblen, identifizierenden Informationen wie Kontonummern von Kunden einzugeben. Dies gilt auch, wenn Sie mit Amazon Cognito oder anderen AWS Diensten über die Konsole AWS CLI, API oder AWS SDKs arbeiten. Alle Daten, die Sie in der Amazon Cognito oder andere Services eingeben, können in Diagnoseprotokolle aufgenommen werden. Wenn Sie eine URL für einen externen Server bereitstellen, schließen Sie keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL ein.

Datenverschlüsselung

Die Datenverschlüsselung fällt normalerweise in zwei Kategorien: Verschlüsselung im Ruhezustand und Verschlüsselung während der Übertragung.

Verschlüsselung im Ruhezustand

Daten innerhalb von Amazon Cognito werden im Ruhezustand gemäß Industriestandards verschlüsselt.

Verschlüsselung während der Übertragung

Als verwalteter Service ist Amazon Cognito durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon Cognito zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Amazon-Cognito-Benutzerpools und -Identitätspools verfügen über IAM-authentifizierte, nicht authentifizierte und Token-autorisierte API-Operationen. Nicht authentifizierte und Token-autorisierte API-Operationen sind für die Verwendung durch Ihre Kunden, die Endbenutzer Ihrer App, vorgesehen. Nicht authentifizierte und Token-autorisierte API-Operationen werden im Ruhezustand und während der Übertragung verschlüsselt. Weitere Informationen finden Sie unter Authentifizierte und nicht authentifizierte API-Operationen der Amazon-Cognito-Benutzerpools.

> **Note**
>
> Amazon Cognito verschlüsselt Ihre Inhalte intern und unterstützt keine von Kunden bereitgestellten Schlüssel.

Identity and Access Management für Amazon Cognito

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu AWS kontrollieren. IAM-Administratoren steuern, wer authentifiziert (angemeldet) und autorisiert (im Besitz von Berechtigungen) ist, Amazon-Cognito-Ressourcen zu nutzen. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in Amazon Cognito ausführen.

Service-Benutzer – Wenn Sie Amazon Cognito zur Ausführung von Aufgaben verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen bereit. Wenn Sie für Ihre Arbeit weitere Amazon-Cognito-Funktionen ausführen, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Fuktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Unter Fehlerbehebung für Amazon-Cognito-Identität und -Zugriff finden Sie nützliche Informationen für den Fall, dass Sie keinen Zugriff auf eine Funktion in Amazon Cognito haben.

Service-Administrator – Wenn Sie in Ihrem Unternehmen für Amazon-Cognito-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollständigen Zugriff auf Amazon Cognito. Ihre Aufgabe besteht darin, zu bestimmen, auf welche Funktionen und Ressourcen von Amazon Cognito Ihre Servicebenutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen dazu, wie Ihr Unternehmen IAM mit Amazon Cognito verwenden kann, finden Sie unter Funktionsweise von der Amazon Cognito mit IAM.

IAM-Administrator – Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf Amazon Cognito verfassen können. Beispiele für identitätsbasierte Amazon-Cognito-Richtlinien, die Sie in IAM verwenden können, finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Cognito.

Authentifizierung mit Identitäten

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

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS , übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter So melden Sie sich bei Ihrem an AWS-Konto im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, um Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch zu signieren. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS Empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center - Benutzerhandbuch und Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Der Zugriff erfolgt, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Verbundidentität

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

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

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

IAM-Benutzer und -Gruppen

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

Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen wechseln. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center -Benutzerhandbuch.
- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.

- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon-EC2 aus oder speichert Objekte in Amazon-S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Forward Access Sessions (FAS) — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.
  - Dienstbezogene Rolle — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- Anwendungen, die auf Amazon EC2 ausgeführt werden — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS

Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

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

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

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

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

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern,

welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

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

Ressourcenbasierte Richtlinien

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

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

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter Zugriffskontrollliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

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

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Service Control Policies (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Weitere Informationen zu Organizations und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations -Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird,

ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter Bewertungslogik für Richtlinien.

Funktionsweise von der Amazon Cognito mit IAM

Bevor Sie IAM zum Verwalten des Zugriffs auf Amazon Cognito verwenden, erfahren Sie, welche IAM-Funktionen Sie mit Amazon Cognito verwenden können.

IAM-Funktionen, die Sie mit Amazon Cognito verwenden können

IAM-Feature	Unterstützung von Amazon Cognito
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Bedingungsschlüssel für die Richtlinie	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Teilweise
Temporäre Anmeldeinformationen	Ja
Hauptberechtigungen	Nein
Servicerollen	Ja
Service-verknüpfte Rollen	Ja

Einen allgemeinen Überblick darüber, wie Amazon Cognito und andere AWS Services mit den meisten IAM-Funktionen funktionieren, finden Sie im AWS IAM-Benutzerhandbuch unter Dienste, die mit IAM funktionieren.

Identitätsbasierte Richtlinien für Amazon Cognito

Unterstützt Richtlinien auf Identitätsbasis.	Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für Amazon Cognito

Beispiele für identitätsbasierte Amazon-Cognito-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Cognito.

Ressourcenbasierte Richtlinien in Amazon Cognito

Unterstützt ressourcenbasierte Richtlinien	Nein

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

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

Richtlinienaktionen für Amazon Cognito

Unterstützt Richtlinienaktionen	Ja

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

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

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

Eine Liste der Amazon-Cognito-Aktionen finden Sie unter Von Amazon Cognito definierte Aktionen in der Service-Autorisierungs-Referenz.

Richtlinienaktionen in Amazon Cognito verwenden das folgende Präfix vor der Aktion:

```
cognito-identity
```

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

```
"Action": [
      "cognito-identity:action1",
      "cognito-identity:action2"
         ]
```

Signierte und nicht signierte APIs im Vergleich

Wenn Sie Amazon Cognito Cognito-API-Anfragen mit AWS Anmeldeinformationen signieren, können Sie diese in einer AWS Identity and Access Management (IAM-) Richtlinie einschränken. API-Anforderungen, die Sie mit AWS -Anmeldeinformationen signieren müssen, umfassen die serverseitige Anmeldung mit `AdminInitiateAuth` sowie Aktionen, die Ihre Amazon-Cognito-Ressourcen erstellen, anzeigen oder ändern, wie beispielsweise `UpdateUserPool`. Weitere Informationen zu signierten API-Anfragen finden Sie unter Signieren von AWS API-Anfragen.

Da Amazon Cognito ein Verbraucheridentitätsprodukt für Apps ist, die Sie für die Öffentlichkeit zugänglich machen möchten, haben Sie Zugriff auf die folgenden nicht signierten APIs. Ihre App nimmt diese API-Anforderungen für Ihre Benutzer und Ihre potenziellen Benutzer vor. Einige APIs verlangen keine vorherige Autorisierung wie `InitiateAuth` für den Start einer neuen Authentifizierungssitzung. Einige APIs verwenden Zugriffstoken oder Sitzungsschlüssel für die Autorisierung, z. B. `VerifySoftwareToken`, um die MFA-Einrichtung für einen Benutzer mit bestehender authentifizierter Sitzung abzuschließen. Eine nicht signierte, autorisierte API der Amazon-Cognito-Benutzerpools unterstützt einen `Session`- oder`AccessToken`-Parameter in der Anforderungssyntax wie in der API-Referenz zu Amazon Cognito dargestellt. Eine nicht signierte API der Amazon-Cognito-Identität unterstützt `IdentityId`-Parameter wie in der API-Referenz zu Amazon-Cognito-Verbundidentitäten dargestellt.

Weitere Informationen zu den Autorisierungsmodellen der Amazon-Cognito-Benutzerpool-API finden Sie unter Authentifizierte und nicht authentifizierte API-Operationen der Amazon-Cognito-Benutzerpools.

API-Operationen der Amazon-Cognito-Identitätspools

- `GetId`
- `GetOpenIdToken`
- `GetCredentialsForIdentity`
- `UnlinkIdentity`

API-Operationen der Amazon-Cognito-Benutzerpools

- AssociateSoftwareToken
- ChangePassword
- ConfirmDevice
- ConfirmForgotPassword
- ConfirmSignUp
- DeleteUser
- DeleteUserAttributes
- ForgetDevice
- ForgotPassword
- GetDevice
- GetUser
- GetUserAttributeVerificationCode
- GlobalSignOut
- InitiateAuth
- ListDevices
- ResendConfirmationCode
- RespondToAuthChallenge
- RevokeToken
- SetUserMFAPreference
- SetUserSettings
- SignUp
- UpdateAuthEventFeedback
- UpdateDeviceStatus
- UpdateUserAttributes
- VerifySoftwareToken
- VerifyUserAttribute

Beispiele für identitätsbasierte Amazon-Cognito-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Cognito.

Richtlinienressourcen für Amazon Cognito

Unterstützt Richtlinienressourcen	Ja

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

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

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

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

Amazon-Ressourcennamen (ARNs)

ARNs für Amazon-Cognito-Verbundidentitäten

Mit Amazon-Cognito-Identitäten-Pools (Verbundidentitäten) können Sie den Zugriff eines IAM-Benutzers auf einen bestimmten Identitäten-Pool einschränken, indem Sie den Amazon-Ressourcennamen (ARN) mit dem Format wie im folgenden Beispiel verwenden. Weitere Informationen zu ARNs finden Sie unter IAM-IDs.

```
arn:aws:cognito-identity:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

ARNs für Amazon Cognito Sync

In Amazon Cognito Sync können Kunden den Zugriff auch anhand der Identitäten-Pool-ID, der Identitäts-ID und des Datensatznamens einschränken.

Für APIs, die für einen Identitäten-Pool verwendet werden, ist das ARN-Format des Identitäten-Pools das gleiche wie für die Amazon-Cognito-Verbundidentitäten, mit der Ausnahme, dass der Name des Service `cognito-sync` anstelle von `cognito-identity` lautet:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

Für APIs für eine einzelne Identität, beispielsweise `RegisterDevice`, können Sie mit dem folgenden ARN-Format auf diese Identität verweisen:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/
identity/IDENTITY_ID
```

Für APIs, die für Datensätze ausgeführt werden – wie `UpdateRecords` und `ListRecords` – finden Sie den einzelnen Datensatz unter Verwendung des folgenden ARN-Formats:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/
identity/IDENTITY_ID/dataset/DATASET_NAME
```

ARNs für Amazon Cognito: Benutzerpools

Für „Amazon Cognito – Eigene Benutzerpools“ ist es mit dem folgenden ARN-Format möglich, den Zugriff eines Benutzers auf einen bestimmten Benutzerpool einzuschränken:

```
arn:aws:cognito-idp:REGION:ACCOUNT_ID:userpool/USER_POOL_ID
```

Eine Liste der Amazon-Cognito-Ressourcentypen und ihrer ARNs finden Sie unter Von Amazon Cognito definierte Ressourcen in der Service-Autorisierungs-Referenz. Informationen zu den Aktionen, mit denen Sie den ARN einzelner Ressourcen angeben können, finden Sie unter Von Amazon Cognito definierte Aktionen.

Beispiele für identitätsbasierte Amazon-Cognito-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Cognito.

Richtlinienbedingungsschlüssel für Amazon Cognito

Unterstützt servicespezifische Richtlini enbedingungsschlüssel	Ja

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

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

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen `AND`-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, AWS wertet die Bedingung mithilfe einer logischen `OR` Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

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

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

Eine Liste von Amazon-Cognito-Bedingungsschlüsseln finden Sie unter Bedingungsschlüssel für Amazon Cognito in der Service-Autorisierungs-Referenz. Informationen dazu, mit welchen Aktionen und Ressourcen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von Amazon Cognito definierte Aktionen.

Beispiele für identitätsbasierte Amazon-Cognito-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Cognito.

Zugriffssteuerungslisten (ACLs) in Amazon Cognito

Unterstützt ACLs	Nein

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Attributbasierte Zugriffskontrolle (ABAC) mit Amazon Cognito

Unterstützt ABAC (Tags in Richtlinien)	Teilweise

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

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

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

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

Weitere Informationen zu ABAC finden Sie unter Was ist ABAC? im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

Verwenden temporärer Anmeldeinformationen mit Amazon Cognito

Unterstützt temporäre Anmeldeinformationen	Ja

Einige funktionieren AWS-Services nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, einschließlich Informationen, die mit temporären Anmeldeinformationen AWS-Services funktionieren AWS-Services , finden Sie im IAM-Benutzerhandbuch unter Diese Option funktioniert mit IAM.

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen AWS Management Console Methode als einem Benutzernamen und einem Passwort anmelden. Wenn

Sie beispielsweise AWS über den Single Sign-On-Link (SSO) Ihres Unternehmens darauf zugreifen, werden bei diesem Vorgang automatisch temporäre Anmeldeinformationen erstellt. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Wechseln von Rollen finden Sie unter Wechseln zu einer Rolle (Konsole) im IAM-Benutzerhandbuch.

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

Serviceübergreifende Prinzipal-Berechtigungen für Amazon Cognito

Unterstützt Forward Access Sessions (FAS)	Nein

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für Amazon Cognito

Unterstützt Servicerollen	Ja

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

Weitere Informationen zu Amazon-Cognito-Servicerollen finden Sie unter Aktivieren der Push-Synchronisierung und Push-Synchronisierung.

> **Warning**
>
> Das Ändern der Berechtigungen für eine Servicerolle könnte die Funktionalität von Amazon Cognito beeinträchtigen. Bearbeiten Sie Servicerollen nur, wenn Amazon Cognito dazu Anleitungen bereitstellt.

Serviceverknüpfte Rollen für Amazon Cognito

Unterstützt serviceverknüpfte Rollen	Ja

Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Einzelheiten zum Erstellen oder Verwalten von serviceverknüpften Amazon-Cognito-Rollen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon Cognito.

Beispiele für identitätsbasierte Richtlinien für Amazon Cognito

Standardmäßig besitzen Benutzer und Rollen keine Berechtigungen zum Erstellen oder Ändern von Amazon-Cognito-Ressourcen. Sie können auch keine Aufgaben mithilfe der AWS API AWS Management Console, AWS Command Line Interface (AWS CLI) oder ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Einzelheiten zu Aktionen und Ressourcentypen, die von Amazon Cognito definiert werden, einschließlich des Formats der ARNs für die einzelnen Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Cognito in der Service-Authorization-Referenz.

Themen

- Bewährte Methoden für Richtlinien

- Verwenden der Amazon-Cognito-Konsole
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- Einschränken des Konsolenzugriffs auf einen bestimmten Identitäten-Pool
- Erteilen von Zugriff auf einen bestimmten Datensatz für alle Identitäten in einem Pool

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien können festlegen, ob jemand Amazon-Cognito-Ressourcen in Ihrem Konto erstellen, aufrufen oder löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien oder AWS -verwaltete Richtlinien für Auftrags-Funktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.

- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung zum IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

> **Note**
>
> Die ursprüngliche und die neue Version der Amazon-Cognito-Konsole weisen unterschiedliche Verhaltensweisen auf, wenn Sie Ihre Amazon-Cognito-Ressourcen anzeigen und ändern. Wenn Sie Berechtigungen für Aktionen unter dem Service-Präfix `cognito-idp` nur dann erteilen, wenn die Bedingung `aws:ViaAWSService` „true“ ist, kann der betroffene IAM-Prinzipal in der ursprünglichen, jedoch nicht in der neuen Konsole mit Amazon-Cognito-Ressourcen arbeiten. Um in der Amazon-Cognito-Konsole zu arbeiten, legen Sie keine `aws:ViaAWSService`-Bedingung für Amazon-Cognito-Berechtigungen in Ihrer IAM-Richtlinie fest.

Verwenden der Amazon-Cognito-Konsole

Um auf die Amazon-Cognito-Konsole zugreifen zu können, müssen Sie über einen Mindestsatz von Berechtigungen verfügen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den Amazon Cognito Cognito-Ressourcen in Ihrem AWS-Konto aufzulisten und anzuzeigen. 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 Amazon Cognito-Konsole weiterhin verwenden können, fügen Sie den Entitäten auch die Amazon Cognito `ConsoleAccess` - oder `ReadOnly` AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

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 beinhaltet Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API oder. AWS CLI 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": "*"
        }
    ]
}
```

Einschränken des Konsolenzugriffs auf einen bestimmten Identitäten-Pool

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cognito-identity:ListIdentityPools"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cognito-identity:*"
      ],
      "Resource": "arn:aws:cognito-identity:us-east-1:0123456789:identitypool/us-
east-1:1a1a1a1a-ffff-1111-9999-12345678"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cognito-sync:*"
      ],
      "Resource": "arn:aws:cognito-sync:us-east-1:0123456789:identitypool/us-
east-1:1a1a1a1a-ffff-1111-9999-12345678"
    }
  ]
}
```

Erteilen von Zugriff auf einen bestimmten Datensatz für alle Identitäten in einem Pool

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

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cognito-sync:ListRecords",
        "cognito-sync:UpdateRecords"
      ],
      "Resource": "arn:aws:cognito-sync:us-east-1:0123456789:identitypool/us-
east-1:1a1a1a1a-ffff-1111-9999-12345678/identity/*/dataset/UserProfile"
    }
  ]
}
```

Fehlerbehebung für Amazon-Cognito-Identität und -Zugriff

Verwenden Sie die folgenden Informationen, um häufige Probleme zu diagnostizieren und zu beheben, die beim Arbeiten mit Amazon Cognito und IAM auftreten könnten.

Themen

- Ich bin nicht autorisiert, eine Aktion in Amazon Cognito auszuführen.
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich bin Administrator und möchte anderen Zugriff auf Amazon Cognito gewähren.
- Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine Amazon Cognito Cognito-Ressourcen ermöglichen

Ich bin nicht autorisiert, eine Aktion in Amazon Cognito 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 `cognito-identity:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cognito-identity:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `cognito-identity: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 zur Ausführung der Aktion `iam:PassRole` autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Amazon Cognito 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 eine IAM-Benutzerin mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Amazon Cognito 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 bin Administrator und möchte anderen Zugriff auf Amazon Cognito gewähren.

Um anderen Personen oder einer Anwendung Zugriff auf Amazon Cognito zu gewähren, müssen Sie eine IAM-Entität (Benutzer oder Rolle) für die Person oder Anwendung erstellen, die Zugriff benötigt. Sie werden die Anmeldeinformationen für diese Einrichtung verwenden, um auf AWS zuzugreifen. Anschließend müssen Sie der Entität eine Richtlinie anfügen, die dieser die korrekten Berechtigungen in Amazon Cognito gewährt.

Informationen zum Einstieg finden Sie unter Erstellen Ihrer ersten delegierten IAM-Benutzer und -Gruppen im IAM-Benutzerhandbuch.

Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine Amazon Cognito Cognito-Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen dazu, ob Amazon Cognito diese Funktionen unterstützt, finden Sie unter Funktionsweise von der Amazon Cognito mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontenübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.

Verwendung von serviceverknüpften Rollen für Amazon Cognito

Amazon Cognito verwendet AWS Identity and Access Management (IAM) serviceverknüpfte Rollen. Eine serviceverknüpfte Rolle ist ein einzigartiger Typ von IAM-Rolle mit einer Vertrauensrichtlinie, die es einem AWS-Service ermöglicht, die Rolle zu übernehmen. Servicebezogene Rollen sind von Amazon Cognito vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon Cognito einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon Cognito definiert die Berechtigungen seiner serviceverknüpften Rollen, und sofern nicht anders definiert, kann nur Amazon

Cognito seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre Amazon-Cognito-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Informationen zu anderen Services, die servicegebundene Rollen unterstützen, finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Servicegebundene Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon Cognito

Amazon Cognito verwendet die folgenden serviceverknüpften Rollen.

- AWSServiceRoleForAmazonCognitoIdpEmailService— Ermöglicht dem Amazon Cognito Cognito-Benutzerpool-Service, Ihre Amazon SES SES-Identitäten zum Senden von E-Mails zu verwenden.

- AWSServiceRoleForAmazonCognitoIdp— Ermöglicht Amazon Cognito Cognito-Benutzerpools, Ereignisse zu veröffentlichen und Endpunkte für Ihre Amazon Pinpoint Pinpoint-Projekte zu konfigurieren.

AWSServiceRoleForAmazonCognitoIdpEmailService

Die serviceverknüpfte Rolle `AWSServiceRoleForAmazonCognitoIdpEmailService` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `email.cognito-idp.amazonaws.com`

Die Richtlinie für Rollenberechtigungen erlaubt Amazon Cognito, die folgenden Aktionen auf den angegebenen Ressourcen durchzuführen:

Zulässige Aktionen für: AWSServiceRoleForAmazonCognitoIdpEmailService

- Aktion: `ses:SendEmail` und `ses:SendRawEmail`
- Ressource: *

Die Richtlinie verweigert Amazon Cognito die Durchführung der folgenden Aktionen für die angegebenen Ressourcen:

Verweigerte Aktionen

- Aktion: `ses:List*`
- Ressource: *

Mit diesen Berechtigungen kann Amazon Cognito Ihre verifizierten E-Mail-Adressen in Amazon SES nur verwenden, um Ihren Benutzern eine E-Mail zu senden. Amazon Cognito sendet Ihren Benutzern E-Mails, wenn sie bestimmte Aktionen in der Client-App für einen Benutzerpool ausführen, z. B. sich anmelden oder ein Passwort zurücksetzen.

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine servicegebundene Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

AWSServiceRoleForAmazonCognitoIdp

Die AWSServiceRoleForAmazonCognitoIdp dienstbezogene Rolle vertraut darauf, dass die folgenden Dienste die Rolle übernehmen:

- `email.cognito-idp.amazonaws.com`

Die Richtlinie für Rollenberechtigungen erlaubt Amazon Cognito, die folgenden Aktionen auf den angegebenen Ressourcen durchzuführen:

Zulässige Aktionen für AWSServiceRoleForAmazonCognitoIdp

- Aktion: `cognito-idp:Describe`
- Ressource: *

Mit dieser Berechtigung kann Amazon Cognito `Describe`-Amazon-Cognito-API-Operationen für Sie aufrufen.

> **Note**
>
> Wenn Sie Amazon Cognito mit Amazon Pinpoint mithilfe von `createUserPoolClient` und `updateUserPoolClient` integrieren, werden Ressourcenberechtigungen

als Inline-Richtlinie zum SLR hinzugefügt. Die Inline-Richtlinie stellt `mobiletargeting:UpdateEndpoint`- und `mobiletargeting:PutEvents`-Berechtigungen bereit. Diese Berechtigungen ermöglichen Amazon Cognito, Ereignisse zu veröffentlichen und Endpunkte für Pinpoint-Projekte zu konfigurieren, die Sie in Cognito integrieren.

Erstellen einer serviceverknüpften Rolle für Amazon Cognito

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie einen Benutzerpool so konfigurieren, dass er Ihre Amazon SES SES-Konfiguration für die E-Mail-Zustellung in der AWS Management Console, der oder der AWS CLI Amazon Cognito-API verwendet, erstellt Amazon Cognito die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie einen Benutzerpool so konfigurieren, dass er Ihre Amazon-SES-Konfiguration für die E-Mail-Zustellung verwendet, erstellt Amazon Cognito die serviceverknüpfte Rolle erneut für Sie.

Bevor Amazon Cognito diese Rolle erstellen kann, muss in die IAM-Berechtigung, die Sie zur Einrichtung des Benutzerpools verwenden, die `iam:CreateServiceLinkedRole`-Aktion eingefügt werden. Weitere Informationen zum Aktualisieren von Berechtigungen in IAM finden Sie unter Ändern von Berechtigungen für einen IAM-Benutzer im IAM-Benutzerhandbuch.

Bearbeiten einer serviceverknüpften Rolle für Amazon Cognito

Sie können die AmazonCognitoIdp oder die AmazonCognitoIdpEmailService dienstbezogenen Rollen in nicht bearbeiten. AWS Identity and Access Management Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach der Erstellung einer serviceverknüpften Rolle nicht bearbeitet werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon Cognito

Wenn Sie eine Funktion oder einen Service, die bzw. der eine servicegebundene Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Wenn Sie die Rolle löschen, behalten Sie nur Entitäten bei, die Amazon Cognito aktiv überwacht oder verwaltet. Bevor Sie Rollen löschen

AmazonCognitoIdp oder mit Diensten AmazonCognitoIdpEmailService verknüpfte Rollen löschen können, müssen Sie für jeden Benutzerpool, der die Rolle verwendet, einen der folgenden Schritte ausführen:

- Benutzerpool löschen.
- Die E-Mail-Einstellungen im Benutzerpool so aktualisieren, dass sie die Standard-E-Mail-Funktionalität verwenden. In der Standardeinstellung wird die serviceverknüpfte Rolle nicht verwendet.

Denken Sie daran, die Aktion jeweils AWS-Region mit einem Benutzerpool durchzuführen, der die Rolle verwendet.

> **Note**
>
> Wenn der Amazon-Cognito-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn das passiert, warten Sie einige Minuten und versuchen Sie es erneut.

So löschen Sie einen Amazon-Cognito-Benutzerpool

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon Cognito Cognito-Konsole unterhttps://console.aws.amazon.com/cognito.
2. Wählen Sie Manage User Pools (Benutzerpools verwalten).
3. Wählen Sie auf der Seite Your User Pools (Eigene Benutzerpools) den Benutzerpool aus, den Sie löschen möchten.
4. Wählen Sie Delete pool (Pool löschen).
5. Geben Sie im Fenster Delete user pool (Benutzerpool löschen) **`delete`** ein, und klicken Sie auf Delete pool (Pool löschen).

So aktualisieren Sie einen Amazon-Cognito-Benutzerpool so, dass er die Standard-E-Mail-Funktionalität verwendet

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon Cognito Cognito-Konsole unterhttps://console.aws.amazon.com/cognito.
2. Wählen Sie Manage User Pools (Benutzerpools verwalten).

3. Wählen Sie auf der Seite Your User Pools (Eigene Benutzerpools) den Benutzerpool aus, den Sie aktualisieren möchten.
4. Wählen Sie im Navigationsmenü auf der linken Seite die Option Message customizations (Nachrichtenanpassungen) aus.
5. Wählen Sie unter Do you want to send emails through your Amazon SES Configuration? (Möchten Sie Ihre E-Mails über die Amazon SES-Konfiguration versenden?), wählen Sie die OptionNo - Use Cognito (Default) (Nein, Cognito verwenden [Standard]) aus.
6. Wenn Sie alle Optionen für Ihr E-Mail-Konto eingerichtet haben, wählen Sie die Option Save changes (Änderungen speichern) aus.

So löschen Sie die servicegebundene Rolle mit IAM

Verwenden Sie die IAM-Konsole, die oder die AWS API AWS CLI, um Rollen zu löschen AmazonCognitoIdp oder AmazonCognitoIdpEmailService serviceverknüpfte Rollen zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für Amazon Cognito serviceverknüpfte Rollen

Amazon Cognito unterstützt dienstbezogene Rollen überall AWS-Regionen dort, wo der Service verfügbar ist. Weitere Informationen finden Sie unter AWS-Regionen und Endpunkte.

Protokollierung und Überwachung in Amazon Cognito

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon Cognito und Ihren anderen AWS Lösungen. Amazon Cognito unterstützt derzeit die folgenden AWS-Services , damit Sie Ihre Organisation und die dort stattfindenden Aktivitäten überwachen können.

- AWS CloudTrail — Mit können CloudTrail Sie API-Aufrufe von der Amazon Cognito Cognito-Konsole und von Codeaufrufen an die Amazon Cognito Cognito-API-Operationen erfassen. Wenn sich ein Benutzer beispielsweise authentifiziert, CloudTrail kann er Details wie die IP-Adresse in der Anfrage, wer die Anfrage gestellt hat und wann sie gestellt wurde, aufzeichnen.
- Amazon CloudWatch Logs — Mit CloudWatch Logs können Sie detaillierte Protokolle der Benutzeraktivitäten an eine Protokollgruppe senden. Sie können beispielsweise detaillierte Benutzeraktivitätsprotokolle überprüfen, um Fehler bei der Zustellung von E-Mail- und SMS-Nachrichten an Ihre Benutzer zu beheben.

- Amazon CloudWatch Metrics — Mit CloudWatch Metriken können Sie Ereignisse nahezu in Echtzeit überwachen, melden und automatische Maßnahmen ergreifen. Sie können beispielsweise CloudWatch Dashboards zu den bereitgestellten Metriken erstellen, um Ihre Amazon Cognito Cognito-Benutzerpools zu überwachen, oder Sie können CloudWatch Alarme für die bereitgestellten Metriken erstellen, um Sie bei Überschreitung eines festgelegten Schwellenwerts zu benachrichtigen.
- Amazon CloudWatch Logs Insights — Mit CloudWatch Logs Insights können Sie konfigurieren, dass Ereignisse CloudTrail CloudWatch zur Überwachung von Amazon Cognito CloudTrail Cognito-Protokolldateien gesendet werden.

Themen

Überwachung der Kosten

Amazon Cognito berechnet Gebühren für die folgenden Dimensionen Ihrer Nutzung.

- Benutzerpool monatlich aktive Benutzer (MAUs)
- Benutzerpool MAUs, die mit OIDC oder SAML Federation angemeldet sind
- MAUs in einem Benutzerpool mit erweiterten Sicherheitsfunktionen
- Aktive Benutzer bündeln App-Clients und fordern Volumen für die Machine-to-Machine-Autorisierung (M2M) an, wobei die Kundendaten gewährt werden
- Gekaufte Nutzung über den Standardkontingenten für einige Kategorien von Benutzerpool-APIs

Darüber hinaus können Funktionen Ihres Benutzerpools wie E-Mail-Nachrichten, SMS-Nachrichten und Lambda-Trigger Kosten für abhängige Dienste verursachen. Eine vollständige Übersicht finden Sie unter Amazon Cognito Pricing.

Kosten anzeigen und antizipieren

In der AWS Billing and Cost Management Konsole können Sie Ihre AWS Kosten einsehen und entsprechende Berichte erstellen. Ihre aktuellen Gebühren für Amazon Cognito finden Sie im

Abschnitt Abrechnung und Zahlungen. Filtern Sie unter Rechnungen, Gebühren nach Service nach Service nach, `Cognito` um Ihre Nutzung einzusehen. Weitere Informationen finden Sie unter Anzeigen Ihrer Rechnung im AWS Billing -Benutzerhandbuch.

Um die API-Anforderungsraten zu überwachen, überprüfen Sie die Nutzungsmetrik in der Service-Kontingents-Konsole. Beispielsweise werden Anfragen zu Kundenanmeldedaten als Rate der ClientAuthentication Anfragen angezeigt. In Ihrer Rechnung sind diese Anfragen dem App-Client zugeordnet, der sie erstellt hat. Mit diesen Informationen können Sie die Kosten in einer Mehrmandantenarchitektur gerecht auf die Mieter verteilen.

Um die Anzahl der M2M-Anfragen für einen bestimmten Zeitraum zu ermitteln, können Sie AWS CloudTrail Ereignisse auch zur CloudWatch Analyse an Logs senden. Fragen Sie Ihre CloudTrail Ereignisse nach `Token_POST` Ereignissen ab, für die Sie Kundendaten erhalten haben. Die folgende CloudWatch Insights-Abfrage gibt diese Anzahl zurück.

```
filter eventName = "Token_POST" and @message like '"grant_type":["client_credentials"]'
 | stats count(*)
```

Verwalten von Kosten

Amazon Cognito berechnet auf der Grundlage der Benutzerzahl, der Nutzung der Funktionen und des Anforderungsvolumens. Im Folgenden finden Sie einige Tipps zur Kostenverwaltung in Amazon Cognito:

Aktivieren Sie keine inaktiven Benutzer

Typische Vorgänge, bei denen ein Benutzer aktiv wird, sind Anmeldung, Registrierung und Zurücksetzen des Passworts. Eine ausführlichere Liste finden Sie unter. Monthly active users (Aktive Benutzer pro Monat) Amazon Cognito zählt inaktive Benutzer nicht auf Ihre Rechnung. Vermeiden Sie alle Vorgänge, bei denen ein Benutzer aktiv wird. Fragen Sie Benutzer statt der AdminGetUserAPI-Operation mit der ListUsersOperation ab. Führen Sie keine umfangreichen administrativen Tests von Benutzerpool-Vorgängen mit inaktiven Benutzern durch.

Verbundene Benutzer verknüpfen

Benutzer, die sich mit einem SAML 2.0- oder OpenID Connect (OIDC) -Identitätsanbieter anmelden, haben höhere Kosten als lokale Benutzer. Sie können diese Benutzer mit einem lokalen Benutzerprofil verknüpfen. Ein verknüpfter Benutzer kann sich als lokaler Benutzer mit den Attributen und Zugriffsrechten anmelden, die für seinen Verbundbenutzer gelten. SAML- oder OIDC-Benutzern,

IdPs die sich im Laufe eines Monats nur mit einem verknüpften lokalen Konto anmelden, werden als lokale Benutzer abgerechnet.

Tarife für Anfragen verwalten

Wenn sich Ihr Benutzerpool der Obergrenze Ihres Kontingents nähert, könnten Sie erwägen, zusätzliche Kapazität zu erwerben, um das Volumen zu bewältigen. Möglicherweise können Sie das Volumen der Anfragen in Ihrer Anwendung reduzieren. Weitere Informationen finden Sie unter Optimieren Sie die Anforderungsraten für Kontingentgrenzen.

Fordern Sie nur dann ein neues Token an, wenn Sie eines benötigen

Bei der Machine-to-Machine-Autorisierung (M2M) mit der Gewährung von Kundenanmeldedaten kann es zu einer hohen Anzahl von Token-Anfragen kommen. Jede neue Token-Anfrage wirkt sich auf Ihre Quote für Anfragen und die Höhe Ihrer Rechnung aus. Um die Kosten zu optimieren, sollten Sie die Einstellungen für das Ablaufen von Token und die Token-Handhabung in das Design Ihrer Anwendungen einbeziehen.

- Zwischenspeichern Sie Zugriffstoken, sodass Ihre Anwendung, wenn sie ein neues Token anfordert, eine zwischengespeicherte Version eines zuvor ausgegebenen Tokens erhält. Wenn Sie diese Methode implementieren, schützt Ihr Caching-Proxy vor Anwendungen, die Zugriffstoken anfordern, ohne sich des Ablaufs zuvor erworbener Token bewusst zu sein. Das Zwischenspeichern von Tokens ist ideal für kurzlebige Microservices wie Lambda-Funktionen und Docker-Container.
- Implementieren Sie in Ihren Anwendungen Mechanismen zur Token-Behandlung, die den Ablauf von Tokens berücksichtigen. Fordern Sie kein neues Token an, bis frühere Token abgelaufen sind. Bewerten Sie die Vertraulichkeits- und Verfügbarkeitsanforderungen jeder Anwendung und konfigurieren Sie den Benutzerpool-App-Client so, dass Zugriffstoken mit einer angemessenen Gültigkeitsdauer ausgestellt werden. Die Dauer eines benutzerdefinierten Tokens eignet sich am besten für APIs und Server mit längerer Lebensdauer, die die Häufigkeit von Anfragen nach Anmeldeinformationen dauerhaft verwalten können.

Löschen Sie ungenutzte Client-Anmeldeinformationen (App-Clients).

Die Abrechnung der M2M-Autorisierung basiert auf zwei Faktoren: der Rate der Token-Anfragen und der Anzahl der App-Clients, die Kundenanmeldedaten gewähren. Wenn App-Clients für die M2M-Autorisierung nicht verwendet werden, löschen Sie sie oder entziehen Sie ihnen die Autorisierung zur Ausgabe von Kundenanmeldedaten. Weitere Informationen zur Verwaltung der App-Client-Konfiguration finden Sie unterApp-Clients für Benutzerpools.

Erweiterte Sicherheit verwalten

Wenn Sie erweiterte Sicherheitsfunktionen in einem Benutzerpool konfigurieren, gilt der Abrechnungstarif für erweiterte Sicherheit für alle MAUs im Benutzerpool. Wenn Sie Benutzer haben, die keine erweiterten Sicherheitsfunktionen benötigen, teilen Sie sie in einen anderen Benutzerpool auf.

Tracking von Kontingenten und Nutzung in CloudWatch Service Quotas

Sie können Amazon Cognito Cognito-Benutzerpools mithilfe von Amazon CloudWatch oder mithilfe von Service Quotas überwachen. Sie können die Nutzung von Identitätspools auch in Service Quotas überwachen. CloudWatch sammelt Rohdaten und verarbeitet sie zu lesbaren Metriken, die nahezu in Echtzeit verfügbar sind. In CloudWatch können Sie Alarme einrichten, die auf bestimmte Schwellenwerte achten und Benachrichtigungen senden oder Maßnahmen ergreifen, wenn diese Schwellenwerte erreicht werden. Informationen zum Erstellen eines CloudWatch Alarms für ein Servicekontingent finden Sie unter Alarm erstellen CloudWatch. Amazon-Cognito-Metriken werden in Intervallen von fünf Minuten bereitgestellt. Weitere Informationen zu Aufbewahrungsfristen finden Sie auf CloudWatch der Amazon-Seite mit CloudWatch häufig gestellten Fragen.

Mit Service Quotas können Sie Ihre Kontingentverwendung für Amazon-Cognito-Benutzerpools und -Identitätspools anzeigen und verwalten. Die Service-Quotas-Konsole verfügt über drei Features: Anzeigen von Servicekontingenten, Anfordern einer Erhöhung des Servicekontingents und Anzeigen der aktuellen Auslastung. Mit der ersten Funktion können Sie Kontingente anzeigen und feststellen, ob das Kontingent einstellbar ist. Sie können die zweite Funktion verwenden, um eine Erhöhung der Service Quotas anzufordern. Sie können die letzte Funktion verwenden, um die Kontingentauslastung anzuzeigen. Diese Funktion ist nur verfügbar, nachdem Ihr Konto eine Weile aktiv war. Weitere Informationen zum Anzeigen von Kontingenten in der Service-Quotas-Konsole finden Sie unter Anzeigen von Service Quotas.

> **Note**
>
> Amazon-Cognito-Metriken werden in Intervallen von fünf Minuten bereitgestellt. Weitere Informationen zu Aufbewahrungsfristen finden Sie auf CloudWatch der Amazon-Seite mit CloudWatch häufig gestellten Fragen.

Wenn Sie bei einem Konto angemeldet sind AWS-Konto , das als Monitoring-Konto in CloudWatch Cross-Account-Observability eingerichtet ist, können Sie dieses Monitoring-Konto verwenden, um

Servicequoten zu visualisieren und Alarme für Kennzahlen in den Quellkonten einzurichten, die mit diesem Monitoring-Konto verknüpft sind. Weitere Informationen finden Sie unter CloudWatch kontoübergreifende Beobachtbarkeit.

Themen

Protokollierung zusätzlicher Aktivitäten aus Amazon-Cognito-Benutzerpools

Sie können Ihren Benutzerpool so konfigurieren, dass detaillierte Protokolle einiger zusätzlicher Aktivitäten an eine CloudWatch Protokollgruppe gesendet werden. Diese Protokolle haben eine feinere Granularität als die darin enthaltenen und können bei der Fehlerbehebung in AWS CloudTrail Ihrem Benutzerpool nützlich sein. Wenn Sie diese Funktion aktivieren, können Sie die Protokollgruppe auswählen, an die Amazon Cognito Protokolle senden soll. Die Protokollierung von Benutzeraktivitäten ist nützlich, wenn Sie den Status von E-Mail- und SMS-Nachrichten ermitteln möchten, die Ihr Benutzerpool mit Amazon SNS und Amazon SES übermittelt hat.

Derzeit können Sie nur Benutzerbenachrichtigungsprotokolle auf Fehler-Ebene aus Ihrem Benutzerpool bereitstellen.

Die detaillierte Protokollierung ersetzt oder ändert die folgenden Protokollfunktionen von Benutzerpools nicht.

1. CloudTrail Protokolle routinemäßiger Benutzeraktivitäten wie Registrierung und Anmeldung.
2. Analyse der Benutzeraktivitäten im großen Maßstab mit CloudWatch Metriken.

Separat finden Sie auch Protokolle von Benutzerimportaufträgen und Lambda-Triggern unter CloudWatch Logs. Amazon Cognito und Lambda speichern diese Protokolle in anderen Protokollgruppen als denen, die Sie für detaillierte Aktivitätsprotokolle angeben.

Sie können detaillierte Aktivitätsprotokolle mit der Amazon Cognito Cognito-Benutzerpools-API in einer SetLogDeliveryConfigurationAPI-Anfrage konfigurieren. Sie können die

Protokollierungskonfiguration eines Benutzerpools in einer GetLogDeliveryConfigurationAPI-Anfrage einsehen.

Sie müssen diese Anfragen mit AWS Anmeldeinformationen autorisieren, die über die folgenden Berechtigungen verfügen.

```
{
    "Version": "2012-10-17",
    "Statement": [
            {
            "Sid": "ManageUserPoolLogs",
            "Action": [
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration",
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "CognitoLog",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "CognitoLoggingCWL",
            "Action": [
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
```

```
            ],
            "Effect": "Allow"
        }
    ]
}
```

Es folgt ein Beispielereignis in einem Benutzerpool. Dieses Protokollschema kann sich ändern. Einige Felder werden möglicherweise mit Nullwerten protokolliert.

```
{
    "eventTimestamp": "1687297330677",
    "eventSource": "USER_NOTIFICATION",
    "logLevel": "ERROR",
    "message": {
        "details": "String"
    },
    "logSourceId": {
        "userPoolId": "String"
    }
}
```

Die Bereitstellung von Protokollen von Amazon Cognito ist das optimale Szenario. Die Menge der Protokolle, die Ihr Benutzerpool bereitstellt, und Ihre Dienstkontingente für CloudWatch Protokolle können sich auf die Übermittlung von Protokollen auswirken.

CloudWatch Protokollgebühren fallen an, wenn die Protokollzustellung aktiviert ist. Weitere Informationen finden Sie unter CloudWatch Preise für verkaufte Logs in Amazon.

Um Protokolle an Protokollgruppen mit einer Ressourcenrichtlinie mit einer Größe von mehr als 5 120 Zeichen zu senden, konfigurieren Sie eine Protokollgruppe mit einem Pfad, der mit `/aws/vendedlogs` beginnt. Weitere Informationen finden Sie unter Aktivieren der Protokollierung für bestimmte AWS Dienste.

Metriken für Amazon-Cognito-Benutzerpools

In der folgenden Tabelle sind die Metriken aufgeführt, die für Amazon-Cognito-Benutzerpools verfügbar sind. Der Amazon CloudWatch -Metrik-Namespace für Amazon Cognito lautet `AWS/Cognito`. Weitere Informationen finden Sie unter Namespaces im CloudWatch Amazon-Benutzerhandbuch.

> **Note**
>
> Metriken, für die in den letzten zwei Wochen keine neuen Datenpunkte vorlagen, werden nicht in der Konsole angezeigt. Sie werden auch nicht angezeigt, wenn Sie den Metriknamen oder die Dimensionsnamen in der Konsole in das Suchfeld auf der Registerkarte Alle Metriken eingeben. Darüber hinaus werden sie nicht in den Ergebnissen eines Listenmetriks-Befehls zurückgegeben. Diese Metriken lassen sich am besten mit den `get-metric-statistics` Befehlen `get-metric-data` oder in der AWS CLI abrufen.

Metrik	Beschreibung
`SignUpSuccesses`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten erfolgreichen Benutzerregistrierungsanforderungen an. Eine erfolgreiche Benutzerregistrierungsanforderung liefert den Wert 1, eine fehlgeschlagene Anforderung den Wert 0. Eine gedrosselte Anforderung wird als fehlgeschlagene Anforderung gezählt, liefert also auch den Wert 0. Verwenden Sie den statistischen Wert `Average` für diese Metrik, um den Prozentsatz erfolgreicher Benutzerregistrierungsanforderungen zu ermitteln. Verwenden Sie den statistischen Wert `Sample Count` für diese Metrik, um die Gesamtzahl der Benutzerregistrierungsanforderungen zu ermitteln. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl erfolgreicher Benutzerregistrierungsanforderungen zu ermitteln. Um die Gesamtzahl der fehlgeschlagenen Benutzerregistrierungsanfragen zu zählen, verwenden Sie den CloudWatch `Math` Ausdruck und subtrahieren Sie die `Sum` Statistik von der `Sample Count` Statistik.

Metrik	Beschreibung
	Diese Metrik wird pro Benutzerpool pro Benutzerpool-Client veröffentlicht. Wenn die Benutzerregistrierung durch einen Administrator erfolgt, wird die Metrik mit dem Benutzerpool-Client als veröffentlicht `Admin`. Beachten Sie, dass diese Metrik für die Anwendungsfälle User import (Benutzerimport) und User migration (Benutzermigration) nicht ausgegeben wird. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl
`SignUpThrottles`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten gedrosselten Benutzerregistrierungsanforderungen an. Sobald eine Benutzerregistrierungsanforderung gedrosselt wird, wird der Wert 1 veröffentlicht. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl der gedrosselten Benutzerregistrierungsanforderungen zu ermitteln. Diese Metrik wird für jeden Benutzerpool für jeden Client veröffentlicht. Falls die Anforderung, die gedrosselt wurde, von einem Administrator stammt, wird die Metrik mit dem Benutzerpool-Client als veröffentlicht `Admin`. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl

Metrik	Beschreibung
`SignInSuccesses`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten erfolgreichen Benutzerauthentifizierungsanforderungen an. Eine Benutzerauthentifizierung gilt als erfolgreich, wenn ein Authentifizierungs-Token an den Benutzer ausgegeben wird. Eine erfolgreiche Authentifizierung liefert den Wert 1, eine fehlgeschlagene Anforderung den Wert 0. Eine gedrosselte Anforderung wird als fehlgeschlagene Anforderung gezählt, liefert also auch den Wert 0. Verwenden Sie den statistischen Wert `Average` für diese Metrik, um den Prozentsatz erfolgreicher Benutzerauthentifizierungsanforderungen zu ermitteln. Verwenden Sie den statistischen Wert `Sample Count` für diese Metrik, um die Gesamtzahl der Benutzerauthentifizierungsanforderungen zu ermitteln. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl erfolgreicher Benutzerauthentifizierungsanforderungen zu ermitteln. Um die Gesamtzahl der fehlgeschlagenen Benutzerauthentifizierungsanfragen zu zählen, verwenden Sie den CloudWatch `Math` Ausdruck und subtrahieren Sie die Statistik von der `Sum` Statistik. `Sample Count` Diese Metrik wird für jeden Benutzerpool für jeden Client veröffentlicht. Falls ein ungültiger Benutzerpool-Client mit einer Anfrage versorgt wird, enthält der entsprechende Benutzerpool-Client-Wert in der Metrik einen festen Wert `Invalid` anstelle des tatsächlich in der Anfrage gesendeten ungültigen Werts.

Metrik	Beschreibung
	Beachten Sie, dass Anforderungen zur Aktualisierung des Amazon-Cognito-Tokens in dieser Metrik nicht enthalten sind. Es gibt eine separate Metrik mit dem statistischen Wert für die Token-`Refresh`. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl

Metrik	Beschreibung
`SignInThrottles`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten gedrosselten Benutzerauthentifizierungsanforderungen an. Der Wert 1 wird veröffentlicht, wann immer eine Authentifizierungsanforderung gedrosselt wird. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl der gedrosselten Benutzerauthentifizierungsanforderungen zu ermitteln. Diese Metrik wird für jeden Benutzerpool für jeden Client veröffentlicht. Falls ein ungültiger Benutzerpool-Client mit einer Anfrage versorgt wird, enthält der entsprechende Benutzerpool-Client-Wert in der Metrik einen festen Wert `Invalid` anstelle des tatsächlich in der Anfrage gesendeten ungültigen Werts. Anforderungen zum Aktualisieren des Amazon-Cognito-Tokens sind in dieser Metrik nicht enthalten. Es gibt eine separate Metrik mit dem statistischen Wert für die Token-`Refresh`. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl

Metrik	Beschreibung
TokenRefreshSuccesses	Stellt die Gesamtzahl an den Amazon-Cognito-Benutzerpool gerichteter erfolgreicher Anforderungen zum Aktualisieren eines Amazon-Cognito-Tokens bereit. Eine erfolgreiche Anforderung zum Aktualisieren eines Amazon-Cognito-Tokens liefert den Wert 1, eine fehlgeschlagene Anforderung dagegen den Wert 0. Eine gedrosselte Anforderung wird als fehlgeschlagene Anforderung gezählt, liefert also auch den Wert 0. Verwenden Sie den statistischen Wert `Average` für diese Metrik, um den Prozentsatz erfolgreicher Anforderungen zum Aktualisieren eines Amazon-Cognito-Tokens zu ermitteln. Verwenden Sie den statistischen Wert `SampleCount` für diese Metrik, um die Gesamtzahl der Anforderungen zum Aktualisieren eines Amazon-Cognito-Tokens zu ermitteln. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl der erfolgreichen Anforderungen zum Aktualisieren eines Amazon-Cognito-Tokens zu ermitteln. Um die Gesamtzahl der fehlgeschlagenen Anfragen zur Aktualisierung eines Amazon Cognito Cognito-Tokens zu zählen, verwenden Sie den CloudWatch `Math` Ausdruck und subtrahieren Sie die `Sum` Statistik von der Statistik. `SampleCount` Diese Metrik wird pro Benutzerpool-Client veröffentlicht. Wenn ein ungültiger Benutzerpool-Client in einer Anforderung enthalten ist, enthält der Benutzerpool-Client-Wert einen festen Wert von `Invalid`.

Metrik	Beschreibung
	Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl
`TokenRefreshThrottles`	Stellt die Gesamtzahl der gedrosselten Anforderungen zum Aktualisieren eines Amazon-Cognito-Tokens bereit, die an den Amazon-Cognito-Benutzerpool gestellt wurden. Der Wert 1 wird veröffentlicht, wann immer eine Anforderung zur Aktualisierung eines Amazon-Cognito-Tokens gedrosselt wird. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl der gedrosselten Anforderungen zum Aktualisieren eines Amazon-Cognito-Tokens zu ermitteln. Diese Metrik wird für jeden Benutzerpool für jeden Client veröffentlicht. Falls ein ungültiger Benutzerpool-Client mit einer Anfrage versorgt wird, enthält der entsprechende Benutzerpool-Client-Wert in der Metrik einen festen Wert `Invalid` anstelle des tatsächlich in der Anfrage gesendeten ungültigen Werts. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl

Metrik	Beschreibung
FederationSuccesses	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten erfolgreichen Identitätsverbundanforderungen an. Ein Identitätsverbund gilt als erfolgreich, wenn Amazon Cognito Authentifizierungs-Token an den Benutzer ausgibt. Eine erfolgreiche Identitätsverbundanforderung liefert den Wert 1, eine fehlgeschlagene Anforderung den Wert 0. Gedrosselte Anfragen und Anfragen, die einen Autorisierungscode, aber keine Token generieren, ergeben den Wert 0. Verwenden Sie den statistischen Wert `Average` für diese Metrik, um den Prozentsatz erfolgreicher Identitätsverbundanforderungen zu ermitteln. Verwenden Sie den statistischen Wert `Sample Count` für diese Metrik, um die Gesamtzahl der Identitätsverbundanforderungen zu ermitteln. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl erfolgreicher Identitätsverbundanforderungen zu ermitteln. Um die Gesamtzahl der fehlgeschlagenen Identity Federation-Anfragen zu zählen, verwenden Sie den CloudWatch `Math` Ausdruck und subtrahieren Sie die Statistik von der `Sum` Statistik. `Sample Count` Metrikdimension: `UserPool`, `UserPoolClient` , `IdentityProvider` Einheiten: Anzahl

Metrik	Beschreibung
`FederationThrottles`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten gedrosselten Identitätsverbundanforderungen an. Sobald eine Identitätsverbundanforderung gedrosselt wird, wird der Wert 1 veröffentlicht. Verwenden Sie den statistischen Wert `Sum` für diese Metrik, um die Gesamtzahl der gedrosselten Identitätsverbündungsanforderungen zu ermitteln. Metrikdimension: `UserPool`, `UserPoolClient` , `IdentityProvider` Einheiten: Anzahl
`CallCount`	Gibt die Gesamtzahl der Anrufe an, die Kunden in Bezug auf eine Kategorie getätigt haben. Diese Metrik umfasst alle Anrufe, wie gedrosselte Anrufe, fehlgeschlagene Anrufe und erfolgreiche Anrufe. Diese Metrik ist unter Verwendung `nameSpace` verfügbar. Das Kategoriekontingent wird für jedes AWS Konto in allen Benutzerpools eines Kontos und einer Region durchgesetzt. Mithilfe der `Sum`-Statistik für diese Metrik können Sie die Gesamtzahl der Anrufe in einer Kategorie zählen. Metrik-Dimension: Service, Typ, Ressource, Klasse Einheiten: Anzahl

Metrik	Beschreibung
`ThrottleCount`	Gibt die Gesamtzahl der gedrosselten Anrufe in Bezug auf eine Kategorie an. Diese Metrik ist unter Verwendung `nameSpace` verfügbar. Diese Metrik wird auf Kontoebene veröffent licht. Mithilfe der `Sum`-Statistik für diese Metrik können Sie die Gesamtzahl der Anrufe in einer Kategorie zählen. Metrik-Dimension: Service, Typ, Ressource, Klasse Einheiten: Anzahl

Dimensionen für Amazon-Cognito-Benutzerpools

Die folgenden Dimensionen werden verwendet, um die Nutzungsmetriken zu verfeinern, die von Amazon Cognito veröffentlicht werden. Die Dimensionen gelten nur für `CallCount`- und `ThrottleCount` -Metriken.

Dimension	Beschreibung
Service	Der Name des AWS Dienstes, der die Ressource enthält. Für Amazon-Cognito-Nutzungsmetriken lautet der Wert für diese Dimension `Cognito user pool`.
Typ	Der Typ von Entität, die gemeldet wird. Der einzige gültige Wert für Amazon-Cognito-Nut zungsmetriken ist API.

Dimension	Beschreibung
Ressource	Der Typ der Ressource, die ausgeführt wird. Der einzige gültige Wert ist Kategoriename.
Klasse	Die Klasse der nachverfolgten Ressource. Amazon Cognito verwendet nicht die Klassendimension.

Verwenden der Service-Quotas-Konsole zum Nachverfolgen von Metriken

Sie können Ihre Kontingente für Amazon-Cognito-Benutzerpools und -Identitätspools von einem zentralen Ort aus mit Service Quotas anzeigen und verwalten. Sie können die Service-Quotas-Konsole verwenden, um Details zu einem bestimmten Kontingent anzuzeigen, die Kontingentnutzung zu überwachen und eine Kontingenterhöhung anzufordern. Für einige Kontingenttypen können Sie einen CloudWatch Alarm einrichten, um Ihre Kontingentauslastung nachzuverfolgen. Weitere Informationen zu den Amazon-Cognito-Metriken, die Sie verfolgen können, finden Sie unter Verfolgen der Kontingentnutzung.

Führen Sie die folgenden Schritte aus, um die Auslastung der Service-Kontingente von Amazon-Cognito-Benutzer- und Identitätspools anzuzeigen.

1. Öffnen Sie die Service Quotas-Konsole.
2. Wählen Sie im Navigationsbereich AWS -Services.
3. Suchen Sie in der Liste AWS -Services nach Amazon Cognito-Benutzerpools oder Amazon-Cognito-Verbundidentitäten und wählen Sie sie aus. Die Seite Service-Kontingent wird angezeigt.
4. Wählen Sie ein Kontingent aus, das die CloudWatch Überwachung unterstützt. Wählen Sie beispielsweise `Rate of UserAuthentication requests` in Amazon-Cognito-Benutzerpools.
5. Scrollen Sie nach unten zu Überwachung. Dieser Abschnitt wird nur für Kontingente angezeigt, die die CloudWatch Überwachung unterstützen.
6. In Überwachung können Sie die aktuelle Auslastung des Servicekontingents im Diagramm anzeigen.
7. Wählen Sie unter Überwachung entweder eine Stunde, drei Stunden, zwölf Stunden, einen Tag, drei Tage oder eine Woche aus.

8. Wählen Sie einen beliebigen Bereich innerhalb des Diagramms aus, um den Prozentsatz der Auslastung des Servicekontingents anzuzeigen. Von hier aus können Sie das Diagramm zu Ihrem Dashboard hinzufügen oder das Aktionsmenü verwenden, um in Metriken anzeigen auszuwählen. Dadurch gelangen Sie zu den entsprechenden Metriken in der CloudWatch Konsole.

Verwenden Sie die CloudWatch Konsole, um Metriken zu verfolgen

Sie können mithilfe CloudWatch von Amazon Cognito-Benutzerpool-Metriken verfolgen und sammeln. Das CloudWatch Dashboard zeigt Kennzahlen zu jedem AWS Service an, den Sie verwenden. Sie können es verwenden CloudWatch , um metrische Alarme zu erstellen. Die Alarme können so eingerichtet werden, dass sie Ihnen Benachrichtigungen senden oder eine Änderung an einer bestimmten Ressource vornehmen, die Sie überwachen. Gehen Sie wie folgt vor CloudWatch, um die Kennzahlen zu Servicekontingenten in anzuzeigen.

1. Öffnen Sie die CloudWatch-Konsole.
2. Wählen Sie im Navigationsbereich Metriken aus.
3. Wählen Sie unter Alle Metriken eine Metrik und eine Dimension aus.
4. Aktivieren Sie das Kontrollkästchen neben der Metrik. Die Metriken werden im Diagramm angezeigt.

> **Note**
>
> Metriken, für die in den letzten zwei Wochen keine neuen Datenpunkte vorlagen, werden nicht in der Konsole angezeigt. Sie werden auch nicht angezeigt, wenn Sie den Metriknamen oder die Dimensionsnamen in der Konsole in das Suchfeld auf der Registerkarte Alle Metriken eingeben, und sie werden nicht in den Ergebnissen eines Befehls vom Typ list-metrics zurückgegeben. Die beste Möglichkeit, diese Metriken abzurufen, ist mit den `get-metric-data`- oder `get-metric-statistics`-Befehlen in der AWS CLI.

Erstellen Sie einen CloudWatch Alarm für ein Kontingent

Amazon Cognito stellt CloudWatch Nutzungsmetriken bereit, die den AWS Servicekontingenten für `CallCount` und `ThrottleCount` APIs entsprechen. Weitere Informationen zur Nachverfolgung der Nutzung finden Sie CloudWatch unterVerfolgen der Kontingentnutzung.

In der Service-Quotas-Konsole können Sie Alarme erstellen, die Sie benachrichtigen, wenn sich Ihre Nutzung einem Servicekontingent nähert. Informationen zum Einrichten eines CloudWatch Alarms mithilfe der Service Quotas Console finden Sie unter Servicequotas und CloudWatch Alarme.

Protokollieren Amazon Cognito Cognito-API-Aufrufen mit AWS CloudTrail

Amazon Cognito ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon Cognito ausgeführt wurden. CloudTrail erfasst eine Teilmenge von API-Aufrufen für Amazon Cognito als Ereignisse, einschließlich Aufrufe von der Amazon Cognito Cognito-Konsole und von Codeaufrufen an die Amazon Cognito Cognito-API-Operationen. Wenn Sie einen Trail erstellen, können Sie wählen, ob CloudTrail Ereignisse an einen Amazon S3-Bucket gesendet werden sollen, einschließlich Ereignisse für Amazon Cognito. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Amazon Cognito gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen darüber CloudTrail, einschließlich der Konfiguration und Aktivierung, finden Sie im AWS CloudTrail Benutzerhandbuch.

Sie können auch CloudWatch Amazon-Alarme für bestimmte CloudTrail Ereignisse erstellen. Sie können beispielsweise einrichten, dass ein Alarm ausgelöst wird CloudWatch , wenn die Konfiguration eines Identitätspools geändert wird. Weitere Informationen finden Sie unter CloudWatch Alarme für CloudTrail Ereignisse erstellen: Beispiele.

Themen

- Informationen zu Amazon Cognito in CloudTrail
- Grundlegendes zu Amazon-Cognito-Anmeldeereignissen
- Analysieren von Amazon Cognito CloudTrail Cognito-Ereignissen mit Amazon CloudWatch Logs Insights

Informationen zu Amazon Cognito in CloudTrail

CloudTrail ist aktiviert, wenn Sie Ihre AWS-Konto erstellen. Wenn unterstützte Ereignisaktivitäten in Amazon Cognito auftreten, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen im CloudTrail Ereignisverlauf in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem

AWS Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung von Ereignissen in Ihrem AWS Konto, einschließlich Ereignissen für Amazon Cognito, erstellen Sie einen Trail. Ein CloudTrail Trail liefert Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie ein Trail in der Konsole anlegen, gilt dieser für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie hier:

- Übersicht zum Erstellen eines Trails
- CloudTrail unterstützte Dienste und Integrationen
- Konfiguration von Amazon SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien von mehreren Konten

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Gibt an, ob die Anforderung mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen finden Sie unter dem CloudTrail UserIdentity-Element.

Vertrauliche Daten in AWS CloudTrail

Da Benutzerpools und Identitätspools Benutzerdaten verarbeiten, verdeckt Amazon Cognito einige private Felder in Ihren CloudTrail Veranstaltungen mit dem Wert. `HIDDEN_FOR_SECURITY_REASONS` Beispiele für Felder, die Amazon Cognito bei Ereignissen nicht ausfüllt, finden Sie unter Grundlegendes zu Amazon-Cognito-Anmeldeereignissen. Amazon Cognito verdeckt nur einige Felder, die üblicherweise Benutzerinformationen enthalten, wie Passwörter und Token. Amazon Cognito führt keine automatische Erkennung oder Maskierung von personenbezogenen Daten durch, die Sie in Ihren API-Anforderungen in nicht private Felder eingeben.

Amazon Cognito-Benutzerpools

Amazon Cognito unterstützt die Protokollierung aller Aktionen, die auf der Seite Benutzerpool-Aktionen aufgeführt sind, als Ereignisse in CloudTrail Protokolldateien. Amazon Cognito protokolliert Benutzerpool-Ereignisse CloudTrail als Verwaltungsereignisse.

Das `eventType` Feld in einem Amazon Cognito CloudTrail Cognito-Benutzerpool-Eintrag gibt an, ob Ihre App die Anfrage an die Amazon Cognito Cognito-Benutzerpools-API oder an einen Endpunkt gestellt hat, der Ressourcen für OpenID Connect, SAML 2.0 oder die gehostete Benutzeroberfläche bereitstellt. API-Anfragen haben den `eventType` `AwsApiCall` und Endpunktanfragen den `eventType` `AwsServiceEvent`.

Amazon Cognito protokolliert die folgenden gehosteten UI-Anfragen auf Ihrer gehosteten Benutzeroberfläche als Ereignisse in CloudTrail.

Gehostete UI-Operationen in CloudTrail

Operation	Beschreibung
`Login_GET` , `CognitoAuthentication`	Ein Benutzer sieht Ihre Anmeldeinformationen an oder sendet sie an Ihren Login-Endpunkt.
`OAuth2_Authorize_GET` , `Beta_Authorize_GET`	Ein Benutzer sieht Ihren Autorisieren des Endpunkts an.
`OAuth2Response_GET` , `OAuth2Response_POST`	Ein Benutzer sendet ein IdP-Token an Ihren `/oauth2/idpresponse` -Endpunkt.
`SAML2Response_POST` , `Beta_SAML2Response_POST`	Ein Benutzer sendet eine IdP-SAML-Zusicherung an Ihren `/saml2/idpresponse` -Endpunkt.
`Login_OIDC_SAML_POST`	Ein Benutzer gibt einen Benutzernamen in Ihrem Login-Endpunkt ein und stimmt mit einer IdP-Kennung überein.
`Token_POST` , `Beta_Token_POST`	Ein Benutzer sendet einen Autorisierungscode an Ihren Token-Endpunkt.

Operation	Beschreibung
`Signup_GET` , `Signup_POST`	Ein Benutzer sendet Anmeldeinformationen an Ihren `/signup`-Endpunkt.
`Confirm_GET` , `Confirm_POST`	Ein Benutzer sendet in der gehosteten UI einen Bestätigungscode.
`ResendCode_POST`	Ein Benutzer sendet eine Anfrage zum erneuten Senden eines Bestätigungscodes in der gehosteten UI.
`ForgotPassword_GET` , `ForgotPassword_POST`	Ein Benutzer sendet eine Anfrage zum Zurücksetzen seines Passworts an Ihren `/forgotPassword` -Endpunkt.
`ConfirmForgotPassword_GET` , `ConfirmForgotPassword_POST`	Ein Benutzer sendet einen Code an Ihren `/confirmForgotPassword` -Endpunkt, der seine `ForgotPassword` -Anfrage bestätigt.
`ResetPassword_GET` , `ResetPassword_POST`	Ein Benutzer sendet in der gehosteten UI ein neues Passwort.
`Mfa_GET`, `Mfa_POST`	Ein Benutzer sendet einen Code zur Multi-Faktor-Authentifizierung (MFA) in der gehosteten UI.
`MfaOption_GET` , `MfaOption_POST`	Ein Benutzer wählt seine bevorzugte Methode für MFA in der gehosteten UI.
`MfaRegister_GET` , `MfaRegister_POST`	Ein Benutzer sendet bei der Registrierung der MFA einen Code zur Multi-Faktor-Authentifizierung (MFA) in der gehosteten UI.
`Logout`	Ein Benutzer meldet sich bei Ihrem `/logout`-Endpunkt ab.
`SAML2Logout_POST`	Ein Benutzer meldet sich bei Ihrem `/saml2/logout` -Endpunkt ab.

Operation	Beschreibung
`Error_GET`	Ein Benutzer sieht eine Fehlerseite in der gehosteten UI.
`UserInfo_GET` , `UserInfo_POST`	Ein Benutzer oder IdP tauscht Informationen mit Ihrem UserInfo-Endpunkt aus.
`Confirm_With_Link_GET`	Ein Benutzer sendet eine Bestätigung, die auf einem Link basiert, den Amazon Cognito in einer E-Mail-Nachricht gesendet hat.
`Event_Feedback_GET`	Ein Benutzer gibt Feedback an Amazon Cognito zu einem Ereignis mit erweiterten Sicherheitsfunktionen.

> **Note**
>
> Amazon Cognito zeichnet Anfragen auf, die für einen Benutzer spezifisch sind, `UserSub` jedoch nicht `UserName` in CloudTrail Protokollen. Sie können einen Benutzer für ein bestimmtes `UserSub` finden, indem Sie die `ListUsers`-API aufrufen und einen Filter für sub verwenden.

Amazon-Cognito-Identitätspools

Datenereignisse

Amazon Cognito protokolliert die folgenden Amazon Cognito Identity-Ereignisse CloudTrail als Datenereignisse. Datenereignisse sind API-Operationen mit hohem Volumen auf Datenebene, die standardmäßig CloudTrail nicht protokolliert werden. Für Datenereignisse werden zusätzliche Gebühren fällig.

- GetCredentialsForIdentity
- GetId
- GetOpenIdToken
- GetOpenIdTokenForDeveloperIdentity

- UnlinkIdentity

Um CloudTrail Protokolle für diese API-Operationen zu generieren, müssen Sie Datenereignisse in Ihrem Trail aktivieren und Event-Selektoren für Cognito-Identitätspools auswählen. Weitere Informationen finden Sie unter Protokollieren von Datenereignissen für Trails im AWS CloudTrail - Benutzerhandbuch.

Mit dem folgenden CLI-Befehl können Sie Ihrem Trail auch Ereignisselektoren für Identitätspools hinzufügen.

```
aws cloudtrail put-event-selectors --trail-name <trail name> --advanced-event-selectors
 \
"{\
   \"Name\": \"Cognito Selector\",\
   \"FieldSelectors\": [\
      {\
         \"Field\": \"eventCategory\",\
         \"Equals\": [\
            \"Data\"\
         ]\
      },\
      {\
         \"Field\": \"resources.type\",\
         \"Equals\": [\
            \"AWS::Cognito::IdentityPool\"\
         ]\
      }\
   ]\
}"
```

Verwaltungsereignisse

Amazon Cognito protokolliert die restlichen API-Operationen von Amazon Cognito Identity Pools als Verwaltungsereignisse. CloudTrail protokolliert standardmäßig API-Operationen für Verwaltungsereignisse.

Eine Liste der Amazon Cognito Identity Pools API-Operationen, bei denen Amazon Cognito sich anmeldet CloudTrail, finden Sie in der Amazon Cognito Identity Pools API-Referenz.

Amazon Cognito Sync

Amazon Cognito protokolliert alle Amazon Cognito Sync-API-Operationen als Verwaltungsereignisse. Eine Liste der Amazon Cognito Sync API-Operationen, bei denen Amazon Cognito sich anmeldet CloudTrail, finden Sie in der Amazon Cognito Sync API-Referenz.

Grundlegendes zu Amazon-Cognito-Anmeldeereignissen

Ein Trail kann Ereignisse als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket übermitteln. 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.

Themen

CloudTrail Beispielereignisse für eine gehostete UI-Anmeldung

Die folgenden CloudTrail Beispielereignisse veranschaulichen die Informationen, die Amazon Cognito protokolliert, wenn sich ein Benutzer über die gehostete Benutzeroberfläche anmeldet.

Amazon Cognito protokolliert das folgende Ereignis, wenn ein neuer Benutzer zur Anmeldeseite Ihrer App navigiert.


```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
```

```
    },
    "eventTime": "2022-04-06T05:38:12Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Login_GET",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "errorCode": "",
    "errorMessage": "",
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200.0
        },
        "requestParameters":
        {
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "response_type":
            [
                "token"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        }
    },
    "eventID": "382ae09a-151d-4116-8f2b-6ac0a804a38c",
    "readOnly": true,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Amazon Cognito protokolliert das folgende Ereignis, wenn ein neuer Benutzer Sign-up (Anmelden) auf der Anmeldeseite Ihrer App auswählt.


```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:21:43Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Signup_GET",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "response_type":
            [
                "code"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "7a63e7c2-b057-4f3d-a171-9d9113264fff",
    "eventID": "5e7b27a0-6870-4226-adb4-f86cd51ac5d8",
```

```
    "readOnly": true,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Amazon Cognito protokolliert das folgende Ereignis, wenn ein neuer Benutzer einen Benutzernamen auswählt, eine E-Mail-Adresse eingibt und auf der Anmeldeseite Ihrer App ein Passwort auswählt. Amazon Cognito protokolliert keine identifizierenden Informationen über die Identität des Benutzers. CloudTrail

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:22:05Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Signup_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 302
        },
        "requestParameters":
        {
            "password":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "requiredAttributes[email]":
```

```
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "response_type":
            [
                "code"
            ],
            "_csrf":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ],
            "username":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ]
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "9ad58dd8-3517-4aa8-96a5-d17a01df9eb4",
    "eventID": "c75eb7a5-eb8c-43d1-8331-f4412e756e69",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Amazon Cognito protokolliert das folgende Ereignis, wenn ein neuer Benutzer nach der Anmeldung auf die Benutzerbestätigungsseite in der gehosteten UI zugreift.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:22:06Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Confirm_GET",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "response_type":
            [
                "code"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "58a5b170-3127-45bb-88cc-3e652d779e0b",
    "eventID": "7f87291a-6d50-409a-822f-e3a5ec7e60da",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
```

```
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Amazon Cognito protokolliert das folgende Ereignis, wenn ein Benutzer auf der Benutzerbestätigungsseite in der gehosteten UI einen Code eingibt, den Amazon Cognito ihm in einer E-Mail-Nachricht gesendet hat.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:23:32Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Confirm_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 302
        },
        "requestParameters":
        {
            "confirm":
            [
                ""
            ],
            "deliveryMedium":
            [
                "EMAIL"
            ],
            "sub":
```

```
            [
                "704b1e47-34fe-40e9-8c41-504997494531"
            ],
            "code":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "destination":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "response_type":
            [
                "code"
            ],
            "_csrf":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "cognitoAsfData":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ],
            "username":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ]
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "9764300a-ed35-4f87-8a0f-b18b3fe2b11e",
    "eventID": "e24ac6e5-2f70-4c6e-ad4e-2f08a547bb36",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
```

```
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

CloudTrail Beispielereignis für eine SAML-Anfrage

Amazon Cognito protokolliert das folgende Ereignis, wenn ein Benutzer, der sich bei Ihrem SAML-IdP authentifiziert hat, die SAML-Zusicherung an Ihren `/saml2/idpresponse`-Endpunkt gesendet hat.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-06T00:50:57Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "SAML2Response_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 302
        },
        "requestParameters":
        {
            "RelayState":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "SAMLResponse":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ]
```

```
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "4f6f15d1-c370-4a57-87f0-aac4817803f7",
    "eventID": "9824b50f-d9d1-4fb8-a2c1-6aa78ca5902a",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "625647942648",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

CloudTrail Beispielereignisse für Anfragen an den Token-Endpunkt

Das Folgende sind Beispielereignisse von Anfragen an den Token-Endpunkt.

Amazon Cognito protokolliert das folgende Ereignis, wenn ein Benutzer, der sich authentifiziert und einen Autorisierungscode erhalten hat, den Code an Ihren `/oauth2/token`-Endpunkt gesendet hat.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-12T22:12:30Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Token_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
```

```
        },
        "requestParameters":
        {
            "code":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "grant_type":
            [
                "authorization_code"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "f257f752-cc14-4c52-ad5b-152a46915238",
    "eventID": "0bd1586d-cd3e-4d7a-abaf-fd8bfc3912fd",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Amazon Cognito protokolliert das folgende Ereignis, wenn Ihr Backend-System eine `client_credentials`-Anfrage für ein Zugriffs-Token an Ihren `/oauth2/token`-Endpunkt sendet.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
```

```
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-12T21:07:05Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Token_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "grant_type":
            [
                "client_credentials"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "4f871256-6825-488a-871b-c2d9f55caff2",
    "eventID": "473e5cbc-a5b3-4578-9ad6-3dfdcb8a6d34",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Amazon Cognito protokolliert das folgende Ereignis, wenn Ihre App ein Aktualisierungstoken gegen eine neue ID und ein Zugriffstoken mit Ihrem `/oauth2/token`-Endpunkt austauscht.


```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-12T22:16:40Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Token_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "refresh_token":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "grant_type":
            [
                "refresh_token"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "requestID": "2829f0c6-a3a9-4584-b046-11756dfe8a81",
    "eventID": "12bd3464-59c7-44fa-b8ff-67e1cf092018",
```

```
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

CloudTrail Beispielereignis für CreateIdentityPool

Das folgende Beispiel zeigt einen Protokolleintrag für eine Anforderung für die Aktion `CreateIdentityPool`. Die Anforderung wurde von dem IAM-Benutzer Alice erstellt.

```
{
    "eventVersion": "1.03",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "['EXAMPLE_KEY_ID']",
        "userName": "Alice"
    },
    "eventTime": "2016-01-07T02:04:30Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "CreateIdentityPool",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "USER_AGENT",
    "requestParameters": {
        "identityPoolName": "TestPool",
        "allowUnauthenticatedIdentities": true,
        "supportedLoginProviders": {
            "graph.facebook.com": "000000000000000"
        }
    },
    "responseElements": {
        "identityPoolName": "TestPool",
        "identityPoolId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
        "allowUnauthenticatedIdentities": true,
```

```
        "supportedLoginProviders": {
            "graph.facebook.com": "000000000000000"
        }
    },
    "requestID": "15cc73a1-0780-460c-91e8-e12ef034e116",
    "eventID": "f1d47f93-c708-495b-bff1-cb935a6064b2",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

CloudTrail Beispielereignis für GetCredentialsForIdentity

Das folgende Beispiel zeigt einen Protokolleintrag für eine Anforderung für die Aktion `GetCredentialsForIdentity`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetCredentialsForIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-credentials-for-identity",
    "requestParameters": {
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "aAaAaAaAaAaAab1111111111EXAMPLE",
            "expiration": "Jan 19, 2023 5:55:08 PM"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "659dfc23-7c4e-4e7c-858a-1abce884d645",
```

```
    "eventID": "6ad1c766-5a41-4b28-b5ca-e223ccb00f0d",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

CloudTrail Beispielereignis für GetId

Das folgende Beispiel zeigt einen Protokolleintrag für eine Anforderung für die Aktion `GetId`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:05Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetId",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-id",
    "requestParameters": {
        "identityPoolId": "us-east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE",
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
    },
    "responseElements": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "dc28def9-07c8-460a-a8f3-3816229e6664",
    "eventID": "c5c459d9-40ec-41fd-8f6b-57865d5a9975",
    "readOnly": false,
    "resources": [{
```

```
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

CloudTrail Beispielereignis für GetOpenIdToken

Das folgende Beispiel zeigt einen Protokolleintrag für eine Anforderung für die Aktion `GetOpenIdToken`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetOpenIdToken",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-open-id-token",
    "requestParameters": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
    },
    "responseElements": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "a506ba18-10d7-4fdb-9548-a8187b2e38bb",
    "eventID": "19ffc1a6-6ed8-4580-a4e1-3062c5ce6457",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
```

```
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

CloudTrail Beispielereignis für GetOpenIdTokenForDeveloperIdentity

Das folgende Beispiel zeigt einen Protokolleintrag für eine Anforderung für die Aktion `GetOpenIdTokenForDeveloperIdentity`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA1EXAMPLE:johns-AssumedRoleSession",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/johns-AssumedRoleSession",
        "accountId": "111122223333",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA1EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2023-01-19T16:53:14Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetOpenIdTokenForDeveloperIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "27.0.3.154",
```

```
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-open-id-token-for-developer-identity",
    "requestParameters": {
        "tokenDuration": 900,
        "identityPoolId": "us-east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE",
        "logins": {
            "JohnsDeveloperProvider": "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
    },
    "responseElements": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "b807df87-57e7-4dd6-b90c-b06f46a61c21",
    "eventID": "f26fed91-3340-4d70-91ae-cdf555547b76",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```


CloudTrail Beispielereignis für UnlinkIdentity

Das folgende Beispiel zeigt einen Protokolleintrag für eine Anforderung für die Aktion `UnlinkIdentity`.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "UnlinkIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
```

```
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.unlink-identity",
    "requestParameters": {
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
        "loginsToRemove": ["cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa"]
    },
    "responseElements": null,
    "requestID": "99c2c8e2-9c29-416f-bb17-b650a5cbada9",
    "eventID": "d8e26126-202a-43c2-b458-3f225efaedc7",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

Analysieren von Amazon Cognito CloudTrail Cognito-Ereignissen mit Amazon CloudWatch Logs Insights

Sie können Ihre Amazon CloudTrail Cognito-Ereignisse mit Amazon CloudWatch Logs Insights suchen und analysieren. Wenn Sie Ihren Trail so konfigurieren, dass Ereignisse an CloudWatch Logs gesendet werden, werden nur die Ereignisse CloudTrail gesendet, die Ihren Trail-Einstellungen entsprechen.

Um Ihre Amazon Cognito CloudTrail Cognito-Ereignisse abzufragen oder zu recherchieren, stellen Sie in der CloudTrail Konsole sicher, dass Sie in Ihren Trail-Einstellungen die Option Verwaltungsereignisse auswählen, damit Sie die auf Ihren AWS Ressourcen ausgeführten Verwaltungsvorgänge überwachen können. Sie können in Ihren Trail-Einstellungen optional die Option Insights–Ereignisse auswählen, wenn Sie Fehler, ungewöhnliche Aktivitäten oder ungewöhnliches Benutzerverhalten in Ihrem Konto identifizieren möchten.

Beispiele für Amazon-Cognito-Abfragen

Sie können die folgenden Abfragen in der CloudWatch Amazon-Konsole verwenden.

Allgemeine Abfragen

Findet die 25 zuletzt hinzugefügten Protokollereignisse.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com"
```

Rufen Sie eine Liste der 25 zuletzt hinzugefügten Protokollereignisse ab, die Ausnahmen enthalten.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com" and @message like /Exception/
```

Ausnahme- und Fehlerabfragen

Suchen Sie die 25 zuletzt hinzugefügten Protokollereignisse mit Fehlercode `NotAuthorizedException` zusammen mit dem Amazon-Cognito-Benutzerpool `sub`.

```
fields @timestamp, additionalEventData.sub as user | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com" and errorCode=
 "NotAuthorizedException"
```

Finden Sie die Anzahl der Datensätze mit `sourceIPAddress` und entsprechendem `eventName`.

```
filter eventSource = "cognito-idp.amazonaws.com"
| stats count(*) by sourceIPAddress, eventName
```

Finden Sie die Top 25 IP-Adressen, die einen `NotAuthorizedException`-Fehler ausgelöst haben.

```
filter eventSource = "cognito-idp.amazonaws.com" and errorCode=
 "NotAuthorizedException"
| stats count(*) as count by sourceIPAddress, eventName
| sort count desc | limit 25
```

Finden Sie die 25 wichtigsten IP-Adressen, die die `ForgotPassword`-API abgerufen haben.

```
filter eventSource = "cognito-idp.amazonaws.com" and eventName = 'ForgotPassword'
| stats count(*) as count by sourceIPAddress
```

```
| sort count desc | limit 25
```

Compliance-Validierung für Amazon Cognito

Externe Prüfer bewerten die Sicherheit und Konformität von Amazon Cognito im Rahmen mehrerer AWS Compliance-Programme. Hierzu zählen unter anderem SOC, PCI, FedRAMP und HIPAA.

Eine Liste der AWS Services im Rahmen bestimmter Compliance-Programme finden Sie unter AWS Services im Umfang nach Compliance-Programmen AWS . Allgemeine Informationen finden Sie unter AWS -Compliance-Programme.

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Compliance-Verantwortung bei der Verwendung von Amazon Cognito ist von der Sensibilität Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften abhängig. AWS stellt die folgenden Ressourcen zur Unterstützung der Compliance bereit:

- Schnellstartanleitungen für Sicherheit und Compliance - In diesen Bereitstellungsanleitungen werden architektonische Überlegungen erörtert und Schritte für die Bereitstellung von sicherheits- und konformitätsorientierten Basisumgebungen auf AWS.
- Whitepaper „Architecting for HIPAA Security and Compliance“ — In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-konforme Anwendungen entwickeln können AWS .
- AWS Compliance-Ressourcen — Diese Sammlung von Arbeitsmappen und Leitfäden kann auf Ihre Branche und Ihren Standort zutreffen.
- Bewertung von Ressourcen anhand von Regeln im AWS Config Entwicklerhandbuch — AWS Config; bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- AWS Security Hub— Dieser AWS Service bietet einen umfassenden Überblick über Ihren Sicherheitsstatus, sodass Sie überprüfen können AWS , ob Sie die Sicherheitsstandards und Best Practices der Branche einhalten.

Ausfallsicherheit bei Amazon Cognito

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die über hoch redundante

Netzwerke mit niedriger Latenz und hohen Durchsätzen 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.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Themen

Überlegungen zu regionenbezogenen Daten

Amazon Cognito Cognito-Benutzerpools werden jeweils in einer AWS Region erstellt und speichern die Benutzerprofildaten nur in dieser Region. Benutzerpools können Benutzerdaten in eine andere AWS Region senden, je nachdem, wie optionale Funktionen konfiguriert sind.

- Wenn die Standardeinstellung für die `no-reply@verificationemail.com`-E-Mail-Adresse für die Weiterleitungsüberprüfung von E-Mail-Adressen mit Amazon-Cognito-Benutzerpools verwendet wird, werden E-Mails über dieselbe Region wie der zugehörige Benutzerpool geleitet.
- Wenn eine andere E-Mail-Adresse verwendet wird, um Amazon Simple Email Service (Amazon SES) mit Amazon Cognito Cognito-Benutzerpools zu konfigurieren, wird diese E-Mail-Adresse über die AWS Region weitergeleitet, die der E-Mail-Adresse in Amazon SES zugeordnet ist.
- SMS-Nachrichten von Amazon-Cognito-Benutzerpools werden über Amazon SNS zu derselben Region geleitet, sofern unter Configuring email or phone verification (Konfigurieren der E-Mail- oder Telefonverifizierung nicht anders angegeben).
- Wenn Amazon-Pinpoint-Analysen mit Amazon-Cognito-Benutzerpools verwendet werden, werden die Ereignisdaten zur Region USA Ost (Nord-Virginia) geleitet.

> **Note**
>
> Amazon Pinpoint ist in mehreren AWS Regionen in Nordamerika, Europa, Asien und Ozeanien verfügbar. Zu den Amazon-Pinpoint-Regionen gehört die Amazon-Pinpoint-API. Wenn eine Amazon-Pinpoint-Region von Amazon Cognito unterstützt wird, sendet Amazon Cognito Ereignisse an Amazon-Pinpoint-Projekte innerhalb der gleichen Amazon-

Pinpoint-Region. Wenn eine Region nicht von Amazon Pinpoint unterstützt wird, unterstützt Amazon Cognito nur das Senden von Ereignissen in us-east-1. Detaillierte Informationen zu Amazon Pinpoint finden Sie unter Amazon Pinpoint endpoints and quotas (Amazon-Pinpoint-Endpunkte und -Kontingente) und Using Amazon Pinpoint analytics with Amazon Cognito user pools (Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools).

Sicherheit der Infrastruktur in Amazon Cognito

Als verwalteter Service ist Amazon Cognito durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon Cognito zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Konfigurations- und Schwachstellenanalyse in Amazon-Cognito-Benutzerpools

AWS erledigt grundlegende Sicherheitsaufgaben wie das Patchen von Gastbetriebssystemen (OS) und Datenbanken, die Firewall-Konfiguration und die Notfallwiederherstellung. Diese Verfahren wurden von qualifizierten Dritten überprüft und zertifiziert. Weitere Informationen finden Sie in den folgenden Ressourcen:

- Compliance-Validierung für Amazon Cognito

- Modell der übergreifenden Verantwortlichkeit

AWS verwaltete Richtlinien für Amazon Cognito

Um Benutzern, Gruppen und Rollen Berechtigungen hinzuzufügen, ist es einfacher, AWS verwaltete Richtlinien zu verwenden, als Richtlinien selbst zu schreiben. Es erfordert Zeit und Fachwissen, um von Kunden verwaltete IAM-Richtlinien zu erstellen, die Ihrem Team nur die benötigten Berechtigungen bieten. Um schnell loszulegen, können Sie unsere AWS verwalteten Richtlinien verwenden. Diese Richtlinien decken allgemeine Anwendungsfälle ab und sind in Ihrem AWS Konto verfügbar. Weitere Informationen zu AWS verwalteten Richtlinien finden Sie unter AWS Verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS Dienste verwalten und aktualisieren AWS verwaltete Richtlinien. Sie können die Berechtigungen in AWS verwalteten Richtlinien nicht ändern. 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 Beschreibungen der Richtlinien für Auftragsfunktionen finden Sie in Verwaltete AWS -Richtlinien für Auftragsfunktionen im IAM-Leitfaden.

Über die IAM-Konsole ist eine Reihe von Richtlinien verfügbar, mit denen Sie Kunden Zugriff auf Amazon Cognito gewähren können:

- `AmazonCognitoPowerUser` – Berechtigungen für den Zugriff und die Verwaltung sämtlicher Aspekte Ihres Identitäten-Pools und Benutzerpools Informationen zu den Berechtigungen für diese Richtlinie finden Sie unter. AmazonCognitoPowerUser

- `AmazonCognitoReadOnly`- Berechtigungen für den schreibgeschützten Zugriff auf Identitäten-Pools und Benutzerpools Informationen zu den Berechtigungen für diese Richtlinie finden Sie unter AmazonCognitoReadOnly.
- `AmazonCognitoDeveloperAuthenticatedIdentities` – Berechtigungen für Ihr Authentifizierungssystem zur Integration in Amazon Cognito Informationen zu den Berechtigungen für diese Richtlinie finden Sie unter AmazonCognitoDeveloperAuthenticatedIdentities.

Diese Richtlinien werden vom Amazon-Cognito-Team gepflegt, sodass Ihre Benutzer auch bei Hinzukommen neuer APIs weiterhin denselben Zugriff haben.

> **Note**
>
> Wenn Sie einen neuen Identitätspool erstellen, können Sie automatisch neue Rollen für den Zugriff von authentifizierten Benutzern und Gastbenutzern erstellen. Der Administrator, der Ihren Identitätspool mit neuen IAM-Rollen erstellt, muss auch über IAM-Berechtigungen verfügen, um Rollen erstellen zu können.

Identitätspools mit nicht authentifiziertem Gastzugriff wenden eine zusätzliche AWS verwaltete Richtlinie als Sitzungsrichtlinie für nicht authentifizierte Benutzer an. `AmazonCognitoUnAuthedIdentitiesSessionPolicy` Diese AWS verwaltete Richtlinie ist nicht für administrative Zwecke vorgesehen. Stattdessen schränkt sie den Umfang der Berechtigungen ein, die Sie im erweiterten Authentifizierungsablauf der Benutzerpools auf Gastbenutzer anwenden können. Weitere Informationen finden Sie unter IAM-Rollen.

Amazon Cognito aktualisiert AWS verwaltete Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für Amazon Cognito an, seit dieser Service begonnen hat, diese Änderungen zu verfolgen. Um automatische Warnungen über Änderungen an dieser Seite erhalten, abonnieren Sie den RSS-Feed auf der Dokumentverlauf-Seite zu Amazon Cognito.

Änderung	Beschreibung	Datum
`AmazonCognitoUnAuthedIdentitiesSessionPolicy` – neue Richtlinie	Es wurde eine AWS verwaltete Richtlinie für die Einschränkung des Rechtebereichs von Gastbenutzern in Identität spools hinzugefügt.	14. Juli 2023
`AmazonCognitoPowerUser` und `AmazonCognitoReadOnly` – Updates von vorhandenen Richtlinien	Neue Berechtigungen wurden hinzugefügt, um Power-Usern das Anzeigen und Verwalten von Zuordnungen von AWS WAF Web-ACLs zu Amazon Cognito Cognito-Benutzerpools zu ermöglichen. Neue Berechtigungen wurden hinzugefügt, um Benutzern mit Lesezugriff das Anzeigen von Zuordnungen von AWS WAF Web-ACLs zu Amazon Cognito Cognito-Benutzerpools zu ermöglichen.	19. Juli 2022
`AmazonCognitoPowerUser` – Aktualisierung auf eine bestehende Richtlinie	Es wurde eine neue Berechtigung hinzugefügt, damit Amazon Cognito die `PutIdentityPolicy` - und `ListConfigurationSets` -Operationen von Amazon Simple Notification Service aufrufen kann. Diese Änderung ermöglicht es Amazon-Cognito-Benutzerpools, Amazon SES zu aktualisieren und Autorisierungsrich	17. November 2021

Änderung	Beschreibung	Datum
	tlinien zu senden. Darüber hinaus können Amazon-SES-Konfigurationssets angewendet werden, wenn Sie das Versenden von E-Mails in Ihrem Benutzerpool konfigurieren.	
`AmazonCognitoPowerUser` – Aktualisierung auf eine bestehende Richtlinie	Es wurde eine neue Berechtigung hinzugefügt, damit Amazon Cognito die `GetSMSSandboxAccountStatus` -Operation von Amazon Simple Notification Service aufrufen kann. Durch diese Änderung können Benutzerpools von Amazon-Cognito-Benutzerpools entscheiden, ob Sie die Amazon-Simple-Notification-Service-Sandbox verlassen müssen, um Nachrichten über Benutzerpools an alle Endbenutzer zu senden.	1. Juni 2021
Amazon Cognito verfolgt Änderungen nach.	Amazon Cognito hat damit begonnen, Änderungen an seinen AWS verwalteten Richtlinien nachzuverfolgen.	1. März 2021

Markieren von Amazon-Cognito-Ressourcen

Ein Tag ist ein Metadaten-Label, das Sie oder AWS einer AWS-Ressource zuweisen. Jedes Tag besteht aus einem Schlüssel und einem Wert. Für Tags, die Sie zuweisen, definieren Sie einen Schlüssel und einen Wert. So können Sie beispielsweise den Schlüssel als `stage` und den Wert für eine Ressource als `test` definieren.

Tags sind für folgende Aktivitäten nützlich:

- Identifizieren und Organisieren Ihrer AWS-Ressourcen. Viele AWS-Services unterstützen das Markieren mit Tags. So können Sie Ressourcen aus verschiedenen Services dasselbe Tag zuweisen. Dies hilft Ihnen, anzugeben, welche Ressourcen dazugehören. Sie können beispielsweise dasselbe Tag einem Amazon–Cognito-Benutzerpool und einer Amazon-DynamoDB-Tabelle zuweisen.
- Überwachen von AWS-Kosten. Sie können diese Tags im AWS Billing and Cost Management-Dashboard aktivieren. AWS verwendet die Kostenzuordnungs-Tags zur Kategorisierung Ihrer Kosten und zur Bereitstellung eines monatlichen Kostenzuordnungsberichts. Weitere Informationen finden Sie unter Use cost allocation tags (Verwendung von Kostenzuordnungs-Tags) im AWS Billing-Benutzerhandbuch.
- Steuern des Zugriffs auf Ihre Ressourcen basierend auf den ihnen zugeordneten Tags. Sie können den Zugriff steuern, indem Sie in den Bedingungen für eine AWS Identity and Access Management-IAM-Richtlinie Tag-Schlüssel und -Werte angeben. So könnten Sie beispielsweise einem Benutzer die Aktualisierung eines Benutzerpools nur gestatten, wenn der Benutzerpool ein `owner`-Tag enthält, dessen Wert dem Namen des Benutzers entspricht. Weitere Informationen finden Sie unter Controlling access using tags (Zugriffssteuerung mit Tags) im IAM-Benutzerhandbuch.

Sie können die AWS Command Line Interface- oder die Amazon-Cognito-API zum Hinzufügen, Bearbeiten oder Löschen von Tags für Benutzerpools und Identitäten-Pools verwenden. Für Benutzerpools können Sie die Tags zudem mit der Amazon-Cognito-Konsole verwalten.

Tipps zur Verwendung von Tags finden Sie unter AWSMarkierungsstrategien im AWS-Antworten-Blog.

In den folgenden Abschnitten erhalten Sie weitere Informationen zu Tags für Amazon Cognito.

Unterstützte Ressourcen in Amazon Cognito

Die folgenden Ressourcen in Amazon Cognito unterstützen die Markierung.

- Benutzerpools
- Identitäten-Pools

Tag (Markierung)-Einschränkungen

Für Tags in Amazon-Cognito-Ressourcen gelten die folgenden Einschränkungen:

- Die maximale Anzahl der Tags, die Sie einer Ressource zuweisen können – 50.
- Maximale Schlüssellänge – 128 Unicode-Zeichen.
- Maximale Wertlänge – 256 Unicode-Zeichen.
- Gültige Zeichen für Schlüssel und Wert – a-z, A-Z, 0-9, Leerzeichen und die folgenden Zeichen: _ . : / = + - @
- Schlüssel und Werte unterscheiden zwischen Groß- und Kleinschreibung.
- Verwenden Sie nicht `aws:` als Präfix für Schlüssel. Dieses Präfix ist für AWS reserviert.

Verwalten von Tags über die Amazon-Cognito-Konsole

Mit der Amazon-Cognito-Konsole können Sie Tags verwalten, die Ihren eigenen Benutzerpools zugewiesen sind.

So fügen Sie einem Benutzerpool Tags hinzu

1. Navigieren Sie zur Amazon-Cognito-Konsole. Geben Sie bei Aufforderung Ihre AWS-Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
4. Wählen Sie die Registerkarte User pool properties (Benutzerpool-Eigenschaften) und dort Tags aus.

5. Wählen Sie Add tags (Tags hinzufügen) aus und fügen Sie Ihr erstes Tag hinzu. Wenn Sie diesem Benutzerpool zuvor bereits Tags zugewiesen haben, wählen Sie unter Manage tags (Tags verwalten) Add another (Ein weiteres hinzufügen) aus.
6. Machen Sie Angaben unter Tag Key (Tag-Schlüssel) und Tag Value (Tag-Wert).
7. Wählen Sie für jedes weitere Tag, das Sie hinzufügen möchten, Add another (Ein weiteres hinzufügen) aus.
8. Wenn Sie fertig mit dem Hinzufügen der Tags sind, klicken Sie auf Save changes (Änderungen speichern).

Auf der Seite Manage tags (Tags verwalten) können Sie auch die Schlüssel und Werte von vorhandenen Tags bearbeiten. Klicken Sie zum Entfernen eines Tags auf Remove (Entfernen).

AWS CLIBeispiele für

Die AWS CLI stellt Befehle zur Verwaltung der Tags bereit, die Sie Ihren Amazon-Cognito-Benutzerpools und -Identitäten-Pools hinzufügen.

Zuweisen von Tags

Mit den folgenden Befehlen können Sie Ihren vorhandenen Benutzer- und Identitäten-Pools Tags zuweisen.

Example **`tag-resource`**-Befehl für Benutzerpools

Verwenden Sie `tag-resource` in folgenden `cognito-idp`-Befehlen, um einem Benutzerpool Tags zuzuweisen:

```
$ aws cognito-idp tag-resource \
> --resource-arn user-pool-arn \
> --tags Stage=Test
```

Dieser Befehl enthält die folgenden Parameter:

- `resource-arn` – Der Amazon-Ressourcenname (ARN) des Benutzerpools, den Sie mit Tags versehen. Wählen Sie für die Suche des ARN den Benutzerpool in der Amazon-Cognito-Konsole aus; der Wert des Pool-ARN wird dann auf der Registerkarte Allgemeine Einstellungen angezeigt.
- `tags` – Die Schlüssel-Wert-Paare der Tags im Format *`key`*=*`value`*.

Wenn Sie mehrere Tags auf einmal zuweisen möchten, geben Sie sie in eine durch Kommata getrennte Liste ein:

```
$ aws cognito-idp tag-resource \
> --resource-arn user-pool-arn \
> --tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Example **`tag-resource`**-Befehl für Identitäten-Pools

Verwenden Sie `tag-resource` in folgenden `cognito-identity`-Befehlen, um einem Identitäten-Pool Tags zuzuweisen:

```
$ aws cognito-identity tag-resource \
> --resource-arn identity-pool-arn \
> --tags Stage=Test
```

Dieser Befehl enthält die folgenden Parameter:

- `resource-arn` – Der Amazon-Ressourcenname (ARN) des Identitäten-Pools, den Sie mit Tags versehen. Wählen Sie für die Suche des ARN den Identitäten-Pool in der Amazon-Cognito-Konsole aus, und klicken Sie auf Identitäten-Pool bearbeiten. Klicken Sie dann unter Identity pool ID (Identitäten-Pool-ID) auf Show ARN (ARN anzeigen).
- `tags` – Die Schlüssel-Wert-Paare der Tags im Format *`key`*=*`value`*.

Wenn Sie mehrere Tags auf einmal zuweisen möchten, geben Sie sie in eine durch Kommata getrennte Liste ein:

```
$ aws cognito-identity tag-resource \
> --resource-arn identity-pool-arn \
> --tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Anzeigen von Tags

Mit den folgenden Befehlen können Sie die Tags anzeigen, die Sie Ihren Benutzer- und Identitäten-Pools zugewiesen haben.

Example **`list-tags-for-resource`**-Befehl für Benutzerpools

Verwenden Sie `list-tags-for-resource` in folgenden `cognito-idp`-Befehlen, um die einem Benutzerpool zugewiesenen Tags anzuzeigen:

```
$ aws cognito-idp list-tags-for-resource --resource-arn user-pool-arn
```

Example **`list-tags-for-resource`**-Befehl für Identitäten-Pools

Verwenden Sie `list-tags-for-resource` in folgenden `cognito-identity`-Befehlen, um die einem Identitäten-Pool zugewiesenen Tags anzuzeigen:

```
$ aws cognito-identity list-tags-for-resource --resource-arn identity-pool-arn
```

Entfernen von Tags

Mit den folgenden Befehlen können Sie Tags aus Ihren Benutzer- und Identitäten-Pools entfernen.

Example **`untag-resource`**-Befehl für Benutzerpools

Verwenden Sie `untag-resource` in folgenden `cognito-idp`-Befehlen, um Tags aus einem Benutzerpool zu entfernen:

```
$ aws cognito-idp untag-resource \
> --resource-arn user-pool-arn \
> --tag-keys Stage CostCenter Owner
```

Geben Sie für den `--tag-keys`-Parameter mindestens einen Tag-Schlüssel an. Schließen Sie keine Tag-Werte ein. Separate Schlüssel mit Leerzeichen.

Example **`untag-resource`**-Befehl für Identitäten-Pools

Verwenden Sie `untag-resource` in folgenden `cognito-identity`-Befehlen, um Tags aus einem Identitäten-Pool zu entfernen:

```
$ aws cognito-identity untag-resource \
> --resource-arn identity-pool-arn \
> --tag-keys Stage CostCenter Owner
```

Geben Sie für den `--tag-keys`-Parameter mindestens einen Tag-Schlüssel an. Schließen Sie keine Tag-Werte ein.

> **Important**
>
> Nachdem Sie einen Benutzer oder einen Identitäten-Pool gelöscht haben, können Tags, die sich auf den gelöschten Pool beziehen, noch bis zu 30 Tage nach dem Löschen in der Konsole oder in API-Aufrufen angezeigt werden.

Anwenden von Tags beim Erstellen von Ressourcen

Mit den folgenden Befehlen können Sie Tags bei der Erstellung eines Benutzer- oder Identitäten-Pools zuweisen.

Example **`create-user-pool`**-Befehl mit Tags

Wenn Sie einen Benutzerpool mit dem `create-user-pool`-Befehl erstellen, können Sie Tags mit dem `--user-pool-tags`-Parameter angeben:

```
$ aws cognito-idp create-user-pool \
> --pool-name user-pool-name \
> --user-pool-tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Schlüssel-Wert-Paare für Tags müssen im Format *`key`*=*`value`* vorliegen. Wenn Sie mehrere Tags hinzufügen, geben Sie diese in einer durch Kommata getrennten Liste an.

Example **`create-identity-pool`**-Befehl mit Tags

Wenn Sie einen Identitäten-Pool mit dem `create-identity-pool`-Befehl erstellen, können Sie Tags mit dem `--identity-pool-tags`-Parameter angeben:

```
$ aws cognito-identity create-identity-pool \
> --identity-pool-name identity-pool-name \
> --allow-unauthenticated-identities \
> --identity-pool-tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Schlüssel-Wert-Paare für Tags müssen im Format *`key`*=*`value`* vorliegen. Wenn Sie mehrere Tags hinzufügen, geben Sie diese in einer durch Kommata getrennten Liste an.

Verwalten von Tags über die Amazon-Cognito-API

Mit den folgenden Aktionen können Sie in der Amazon-Cognito-API die Tags für Ihre Benutzer- und Identitäten-Pools verwalten.

API-Aktionen für Tags für Benutzerpools

Mit den folgenden API-Aktionen können Sie Tags für Benutzerpools zuweisen, anzeigen und entfernen.

- TagResource
- ListTagsForResource
- UntagResource
- CreateUserPool

API-Aktionen für Tags für Identitäten-Pools

Mit den folgenden API-Aktionen können Sie Tags für Identitäten-Pools zuweisen, anzeigen und entfernen.

- TagResource
- ListTagsForResource
- UntagResource
- CreateIdentityPool

Kontingente in Amazon Cognito

Amazon Cognito verfügt über Standardkontingente (früher als Limits bezeichnet) für die maximale Anzahl von Operationen, die Sie in Ihrem Konto ausführen können. Amazon Cognito umfasst außerdem Kontingente für die maximale Anzahl und Größe der Amazon-Cognito-Ressourcen.

Jedes Amazon Cognito Cognito-Kontingent entspricht einem maximalen Volumen von Anfragen AWS-Region in einer zu eins AWS-Konto. Ihre Apps können beispielsweise API-Anforderungen mit einer maximal dem Standardkontingent (RPS) entsprechenden Rate für `UserAuthentication`-Operationen in Bezug auf alle Ihre Benutzerpools in USA Ost (Nord-Virginia) stellen. Ihre Apps im asiatisch-pazifischen Raum (Tokio) können das gleiche Volumen an Anfragen für alle Ihre Benutzerpools in ihrer eigenen Region generieren. AWS kann einer Anfrage zur Erhöhung des Kontingents jeweils nur in einer Region stattgeben. Eine erfolgreiche Kontingenterhöhung in USA Ost (Nord-Virginia) hat keine Auswirkungen auf Ihre maximale Anforderungsrate in Asien-Pazifik (Tokio).

Themen

Informationen zu API-Anforderungsratenkontingenten

Kategorisierung von Kontingenten

Amazon Cognito erzwingt eine maximale Anforderungsrate für API-Operationen. Weitere Informationen zu den mit Amazon Cognito verfügbaren API-Operationen finden Sie unter Amazon-Cognito-API- und Endpunkt-Referenzen. Für Benutzerpools sind diese Operationen in Kategorien gängiger Anwendungsfälle wie `UserAuthentication` oder `UserCreation` gruppiert. Eine Liste der API-Operationen für Benutzerpools nach Kategorien finden Sie unter. API-Betriebskategorien und Anforderungsratenkontingente von Amazon-Cognito-Benutzerpools

In der Service Quotas Quotas-Konsole können Sie Ihre Kontingentnutzung nach Kategorien von Benutzerpools und Identitätspools verfolgen. Wenn die Anforderungsrate Ihrer Amazon Cognito

Cognito-Benutzerpools ein Kontingent bündelt oder überschreitet, können Sie zusätzliche Kapazität erwerben. In der Konsole Service Quotas können Sie die Kontingentnutzung Ihres Benutzerpools nach Kategorien verfolgen und Kontingenterhöhungen erwerben.

Operationskontingente sind definiert als die maximale Anzahl der Anforderungen pro Sekunde (RPS) für alle Operationen innerhalb einer Kategorie. Der Benutzerpool-Service von Amazon-Cognito wendet Kontingente für alle Operationen in jeder Kategorie an. Zum Beispiel umfasst die Kategorie `UserCreation` vier Operationen: `SignUp`, `ConfirmSignUp`, `AdminCreateUser` und `AdminConfirmSignUp`. Sie wird mit einem kombinierten Kontingent von 50 RPS zugewiesen. Wenn mehrere Operationen gleichzeitig stattfinden, kann jede Operation innerhalb dieser Kategorie bis zu 50 RPS separat oder kombiniert abrufen.

> **Note**
>
> Kategoriekontingente gelten nur für Benutzerpools. Amazon Cognito wendet jedes Identitätspoolkontingent auf eine einzelne Operation an. Misst sowohl bei Quoten pro Kategorie als auch pro Vorgang die Gesamtrate aller Anfragen aus allen Benutzerpools oder Identitätspools AWS-Konto in Ihrer Region. AWS

API-Operationen von Amazon-Cognito-Benutzerpools mit spezieller Handhabung der Anforderungsrate

Die Operationskontingente werden für die kombinierten Gesamtanforderungen auf Kategorieebene gemessen und durchgesetzt, mit Ausnahme der `AdminRespondToAuthChallenge`- und `RespondToAuthChallenge`-Operationen, bei denen besondere Handhabungsregeln gelten.

Die `UserAuthentication` Kategorie umfasst vier Operationen in der Amazon Cognito Cognito-Benutzerpools-API: `AdminInitiateAuthInitiateAuth`,`AdminRespondToAuthChallenge`, und`RespondToAuthChallenge`. Darüber hinaus trägt die Benutzerauthentifizierung in der gehosteten Benutzeroberfläche zu diesem Kontingent bei. Die Operationen `InitiateAuth` und `AdminInitiateAuth` werden pro Kategoriekontingent gemessen und durchgesetzt. Die Abgleichsvorgänge `RespondToAuthChallenge` und `AdminRespondToAuthChallenge` unterliegen einem separaten Kontingent, das das Dreifache des `UserAuthentication`-Kategorielimits beträgt. Dieses erhöhte Kontingent trägt mehreren Authentifizierungsherausforderungen Rechnung, die in Ihren Apps eingerichtet wurden. Das Kontingent reicht aus, um die überwiegende Mehrheit der Anwendungsfälle abzudecken. Nachdem Ihre App bis zu drei Antworten auf Authentifizierungsherausforderungen gegeben hat, werden

zusätzliche Anfragen auf das `UserAuthentication` Kategorienkontingent angerechnet. Multi-Faktor-Authentifizierung (MFA), Geräteauthentifizierung und benutzerdefinierte Authentifizierung sind Beispiele für Challenge-Prompts, die Sie in Ihren Benutzerpool integrieren können.

Wenn Ihr Kontingent für die `UserAuthentication` Kategorie beispielsweise 80 RPS beträgt, können Sie anrufen `RespondToAuthChallenge` oder mit einer `AdminRespondToAuthChallenge` Rate von bis zu 240 RPS (3 * 80 RPS). Wenn Ihr Benutzerpool zu vier Challenge-Runden pro Authentifizierung auffordert und sich 70 Benutzer pro Sekunde anmelden, `RespondToAuthChallenge` beträgt die Summe 280 RPS (70 x 4), was 40 RPS über dem Kontingent liegt. Die zusätzlichen 40 RPS werden zu 70 `InitiateAuth`-Anrufen addiert, wodurch die Gesamtnutzung der `UserAuthentication`-Kategorie 110 RPS (40 + 70) ergibt. Da dieser Wert das auf 80 RPS festgelegte Kategoriekontingent um 30 RPS überschreitet, drosselt Amazon Cognito Anfragen von Ihrer App.

Monthly active users (Aktive Benutzer pro Monat)

Wenn Amazon Cognito die Abrechnung für den Benutzerpool berechnet, berechnet es Ihnen einen Tarif für jeden monatlich aktiven Benutzer (MAU). Berücksichtigen Sie bei der Planung von Anträgen zur Erhöhung der Kontingente Ihre aktuelle und voraussichtliche MAU-Anzahl. Ein Benutzer wird als MAU gezählt, wenn innerhalb eines Kalendermonats eine Identitätsoperation im Zusammenhang mit diesem Benutzer stattfindet. Zu den Aktivitäten, die einen Benutzer aktiv machen, gehören die folgenden.

- Registrierung und administrative Erstellung eines Benutzers
- Anmelden
- Abmelden
- Bestätigung des Benutzerkontos oder Überprüfung der Attribute
- Zurücksetzen des Passworts
- Ändern der Benutzerattribute, der Gruppenmitgliedschaft oder der MFA-Einstellungen
- Abfragen der detaillierten Attribute eines Benutzers
- Aktivierung, Deaktivierung oder Löschung eines Benutzers

> **Note**
>
> Die Kategorie Detaillierte Attribute eines Benutzers abfragen beinhaltet die API-Operation AdminGetUser, aber nicht ListUsers. Eine detaillierte user-by-user Abfrage in einem großen

Benutzerpool kann erhebliche Auswirkungen auf Ihre AWS Rechnung haben. Um zusätzliche Gebühren zu vermeiden, sollten Sie Benutzerdaten mit einer externen Datenbank sammeln `ListUsers` oder Benutzerinformationen in einer externen Datenbank speichern.

Verwalten von API-Anforderungsratenkontingenten

Kontingentanforderungen identifizieren

Important

Wenn Sie die Amazon Cognito Cognito-Kontingente für Kategorien wie `UserAuthentication`, `UserCreation`, oder erhöhen`AccountRecovery`, müssen Sie möglicherweise die Kontingente für andere AWS-Services erhöhen. Beispielsweise können Nachrichten, die Amazon Cognito mit Amazon Simple Notification Service (Amazon SNS) oder Amazon Simple Email Service (Amazon SES) sendet, fehlschlagen, wenn die Kontingente für die Anforderungsrate in diesen Diensten nicht ausreichen.

Um die Kontingentanforderungen zu berechnen, bestimmen Sie, wie viele aktive Benutzer in einem bestimmten Zeitraum mit Ihrer Anwendung interagieren. Wenn Sie beispielsweise davon ausgehen, dass Ihre Anwendung innerhalb von acht Stunden durchschnittlich eine Million aktive Benutzer anmeldet, müssen Sie in der Lage sein, durchschnittlich 35 Benutzer pro Sekunde zu authentifizieren.

Wenn Sie außerdem davon ausgehen, dass die durchschnittliche Benutzersitzung zwei Stunden dauert, und Sie Token so konfigurieren, dass sie nach einer Stunde ablaufen, muss jeder Benutzer seine Token einmal während seiner Sitzung aktualisieren. Dann beträgt das erforderliche durchschnittliche Kontingent für die Kategorie `UserAuthentication` zur Unterstützung dieser Last 70 RPS.

Wenn Sie ein peak-to-average Verhältnis von 3:1 annehmen und dabei die Varianz der Benutzeranmeldefrequenz während des Zeitraums von acht Stunden berücksichtigen, benötigen Sie das gewünschte `UserAuthentication` Kontingent von 200 RPS.

> **Note**
>
> Wenn Sie für jede Benutzeraktion mehrere Operationen aufrufen, müssen Sie die Aufrufraten der einzelnen Operationen auf Kategorieebene summieren.

Optimieren Sie die Anforderungsraten für Kontingentgrenzen

Da die Erhöhung der API-Ratenlimits Ihre AWS Rechnung zusätzlich belastet, sollten Sie Anpassungen Ihres Nutzungsmodells in Betracht ziehen, bevor Sie eine Erhöhung des Kontingents beantragen. Im Folgenden finden Sie einige Beispiele für eine App-Architektur, die die Anforderungsraten optimiert.

Wiederholen Sie den Versuch nach einer Back-off-Wartezeit

Sie können Fehler bei jedem API-Aufruf abfangen und den Versuch nach einer Wartezeit erneut versuchen. Sie können den Back-off-Algorithmus entsprechend den geschäftlichen Anforderungen und Belastungen anpassen. Amazon SDKs verfügen über eine integrierte Wiederholungslogik. Weitere Informationen finden Sie unter Tools, auf AWS denen Sie aufbauen können.

Verwenden einer externen Datenbank für häufig aktualisierte Attribute

Wenn Ihre Anwendung mehrere Aufrufe an einen Benutzerpool erfordert, um benutzerdefinierte Attribute zu lesen oder zu schreiben, verwenden Sie externen Speicher. Sie können Ihre bevorzugte Datenbank verwenden, um benutzerdefinierte Attribute zu speichern oder eine Cache-Ebene verwenden, um während der Anmeldung ein Benutzerprofil zu laden. Sie können dieses Profil bei Bedarf aus dem Cache referenzieren, anstatt das Benutzerprofil aus einem Benutzerpool neu zu laden.

Validieren Sie JSON-Webtoken (JWTs) auf der Clientseite

Anwendungen müssen JWT-Token validieren, bevor sie ihnen vertrauen. Sie können die Signatur und Gültigkeit von Token auf der Clientseite überprüfen, ohne API-Anfragen an einen Benutzerpool zu senden. Nachdem das Token validiert wurde, können Sie den Ansprüchen im Token vertrauen und die Ansprüche verwenden, anstatt mehr `getUser`-API-Aufrufe zu tätigen. Weitere Informationen finden Sie unter Verifizieren eines JSON-Webtokens.

Drosseln Sie den Datenverkehr zu Ihrer Webanwendung mit einem Wartezimmer

Wenn Sie Datenverkehr von einer großen Anzahl von Benutzern erwarten, die sich während eines zeitgebundenen Ereignisses anmelden, z. B. wenn Sie an einer Prüfung teilnehmen oder an einer Live-Veranstaltung teilnehmen, können Sie den Anforderungsdatenverkehr mit Mechanismen zur Selbstdrosselung optimieren. Sie können beispielsweise einen Warteraum einrichten, in dem Benutzer bereit stehen können, bis eine Sitzung verfügbar ist, sodass Sie Anfragen bearbeiten können, wenn Sie über verfügbare Kapazität verfügen. Sehen Sie die AWS - Virtuelle-Wartezimmer-Lösung für eine Referenzarchitektur eines Wartezimmers.

JWTs zwischenspeichern

Verwenden Sie Zugriffstoken wieder, bis sie ablaufen. Ein Beispiel-Framework mit Token-Caching in einem API Gateway finden Sie unterZwischenspeicherung von Token. Anstatt API-Anfragen zur Abfrage von Benutzerinformationen zu generieren, können Sie ID-Token zwischenspeichern, bis sie ablaufen, und Benutzerattribute aus dem Cache lesen.

Weitere Informationen zur Arbeit mit API-Anforderungsraten finden Sie unter API-Drosselung in Ihren Workloads verwalten und überwachen. AWS Informationen zur Optimierung von Amazon Cognito Cognito-Vorgängen, die Ihre AWS Rechnung mit zusätzlichen Kosten belasten, finden Sie unterVerwalten von Kosten.

Verfolgen der Kontingentnutzung

Amazon Cognito generiert `CallCount` in Amazon `ThrottleCount` Metriken CloudWatch für jede API-Betriebskategorie auf Kontoebene. Sie können `CallCount` nutzen, um die Gesamtzahl der Anrufe zu verfolgen, die Kunden im Zusammenhang mit einer Kategorie getätigt haben. Sie können `ThrottleCount` verwenden, um die Gesamtzahl der gedrosselten Anrufe in Bezug auf eine Kategorie zu verfolgen. Sie können die `CallCount`- und `ThrottleCount`-Metriken mit der Statistik `Sum` verwenden, um die Gesamtzahl der Anrufe in einer Kategorie zu zählen. Weitere Informationen finden Sie unter CloudWatch Nutzungsmetriken.

Bei der Überwachung von Service-Quotas ist die Auslastung der Prozentsatz eines verwendeten Servicekontingents. Wenn der Kontingentwert beispielsweise 200 Ressourcen beträgt und 150 Ressourcen verwendet werden, beträgt die Auslastung 75 %. Verwendung ist die Anzahl der Ressourcen oder Operationen, die für ein Servicekontingent verwendet werden.

Nachverfolgung der Nutzung anhand von CloudWatch Metriken

Sie können Nutzungsmetriken für Amazon Cognito Cognito-Benutzerpools mit CloudWatch verfolgen und sammeln. Das CloudWatch Dashboard zeigt Metriken zu allen Geräten an AWS-Service , die Sie verwenden. Mit können Sie Metrikalarme erstellen CloudWatch, um Sie zu benachrichtigen oder eine bestimmte Ressource, die Sie überwachen, zu ändern. Weitere Informationen zu CloudWatch Messwerten finden Sie unter Verfolgen Sie Ihre CloudWatch Nutzungsmetriken.

Verfolgen der Auslastung durch Service-Quotas-Metriken

Amazon Cognito Cognito-Benutzerpools sind in Service Quotas integriert, eine Konsolenoberfläche zur Anzeige und Verwaltung Ihrer Service-Kontingentnutzung. In der Service-Kontingents-Konsole können Sie den Wert eines bestimmten Kontingents nachschlagen, Überwachungsinformationen anzeigen, eine Erhöhung des Kontingents beantragen oder CloudWatch Alarme einrichten. Nachdem Ihr Konto eine Zeit lang aktiv war, können Sie sich ein Diagramm Ihrer Ressourcennutzung anzeigen lassen.

In der Spalte Angewendeter Kontingentwert auf Kontoebene in der Konsole Service Quotas für Amazon Cognito-Benutzerpools und Amazon Cognito Cognito-Identitätspools wird Ihr aktuelles Kontingent angezeigt. In der Spalte Auslastung wird Ihre aktuelle Quotennutzungsrate angezeigt. Einstellbare Amazon Cognito Cognito-Benutzerpools requests-per-second (RPS) -Kontingente zeigen ihre aktuelle Nutzung an. Die Service-Kontingents-Konsole kann Sie auch zu CloudWatch Metriken weiterleiten, um sich eine ausgewählte Kontingentmetrik genauer anzusehen. Weitere Informationen zum Anzeigen von Kontingenten in der Service-Quotas-Konsole finden Sie unter Anzeigen von Service Quotas.

Verfolgen Sie monatlich aktive Benutzer (MAUs)

Die Anzahl der monatlich aktiven Benutzer (MAUs) in Ihrem Benutzerpool liefert wichtige Daten für Ihre Planung von Anforderungsquoten. Sie können Ihre API-Anforderungsraten mit der Anzahl der Benutzer vergleichen, die Sie in einem bestimmten Zeitraum aktiv hatten. Mit diesem Wissen können Sie berechnen, wie sich eine Zunahme der aktiven Nutzer Ihrer Anwendungen auf Ihre Kontingente in Ihrem Nutzungsmodell auswirkt. Stellen Sie sich beispielsweise vor, dass Ihre kombinierten Anwendungen in den USA West (Oregon) zu 2 Millionen aktiven Benutzern in einem Monat geführt haben und dass in Ihrer `UserAuthentication` Kategorie bei der Standardquote von 120 Anfragen pro Sekunde (RPS) gelegentlich Drosselungsfehler aufgetreten sind. Im Vormonat, vor Ihrer erfolgreichen Werbekampagne, hatten Sie 1 Million MAUs, und Ihre Anwendungen überstiegen nie 80 RPS. Wenn Sie aufgrund eines neuen TV-Spots einen ähnlichen Anstieg erwarten, könnten Sie weitere 40 RPS erwerben, um die nächsten Millionen Nutzer mit einem angepassten Kontingent von 160 RPS unterzubringen.

Um Ihre MAUs zu überprüfen

Rufen Sie die AWS Billing Konsole auf und überprüfen Sie eine aktuelle Rechnung. Unter Gebühren nach Service können Sie nach Cognito filtern, um eine Aufschlüsselung Ihrer MAUs für diesen Abrechnungszeitraum anzuzeigen.

Beantragen einer Kontingenterhöhung

Amazon Cognito hat ein Kontingent für die maximale Anzahl von Vorgängen pro Sekunde, die Sie jeweils AWS-Region in Ihren Benutzerpools und Identitätspools ausführen können. Sie können eine Erhöhung der einstellbaren API-Anforderungsquoten für Amazon Cognito Cognito-Benutzerpools erwerben. Überprüfen Sie Ihr aktuelles Kontingent und erwerben Sie eine Erhöhung über die Service Quotas Console oder über die Service Quotas API-Operationen `ListAWSDefaultServiceQuotas` und`RequestServiceQuotaIncrease`.

- Informationen zum Kauf einer Kontingenterhöhung über die Service Quotas Quota-Konsole finden Sie unter Beantragen einer API-Kontingentserhöhung im Service Quotas Quota-Benutzerhandbuch.
- AWS zielt darauf ab, Anfragen zur Erhöhung des Kontingents innerhalb von 10 Tagen abzuschließen. Verschiedene Überlegungen können jedoch dazu führen, dass die Bearbeitungszeit für Anfragen mehr als 10 Tage beträgt. Bei einigen Anfragen muss Amazon Cognito beispielsweise zusätzliche Hardwarekapazität bereitstellen, und saisonale Erhöhungen des Anforderungsvolumens können zu Verzögerungen führen.
- Wenn das Kontingent in Service Quotas noch nicht in verfügbar ist, verwenden Sie das Formular zur Erhöhung des Service-Limits.

> **Important**
>
> Es können nur anpassbare Kontingente erhöht werden. Sie müssen eine höhere Kontingentkapazität erwerben. Die Preise für die Erhöhung der Kontingente finden Sie unter Amazon Cognito Cognito-Preise.

API-Betriebskategorien und Anforderungsratenkontingente von Amazon-Cognito-Benutzerpools

Da es in Amazon Cognito sich überschneidende Klassen von API-Operationen mit unterschiedlichen Autorisierungsmodellen gibt, gehört jede Operation einer Kategorie an. Für jede Kategorie gibt es ein eigenes gepooltes Kontingent für alle API-Operationen der Mitglieder, und zwar für alle Benutzerpools in einer AWS-Region in Ihrem Konto. Sie können nur eine Erhöhung der einstellbaren Kategoriekontingente beantragen. Weitere Informationen finden Sie unter Beantragen einer Kontingenterhöhung. Kontingentanpassungen gelten für die Benutzerpools in Ihrem Konto in einer einzelnen Region. Amazon Cognito schränkt die Vorgänge in einigen Kategorien[3] auf 5 Anforderungen pro Sekunde (RPS) pro Benutzerpool ein. Das Standardkontingent (RPS) gilt zusätzlich für alle Benutzerpools in einem. AWS-Konto

> **Note**
>
> Das Kontingent für die einzelnen Kategorien wird in Form der aktiven Benutzer pro Monat (Monthy Active Users, MAUs) gemessen. AWS-Konten mit weniger als zwei Millionen MAUs können mit dem Standardkontingent auskommen. Wenn Sie weniger als eine Million MAUs haben und Amazon Cognito Anfragen drosselt, sollten Sie eine Optimierung Ihrer App in Betracht ziehen. Weitere Informationen finden Sie unter Optimieren Sie die Anforderungsraten für Kontingentgrenzen.

Kategorieoperationskontingente werden für alle Benutzer in allen Benutzerpools innerhalb einer AWS-Region angewendet. Amazon Cognito verwaltet auch ein Kontingent für die Anzahl der Anforderungen, die Ihre App für einen Benutzer generieren kann. Sie müssen die API-Anforderungen pro Benutzer begrenzen, wie in der folgenden Tabelle dargestellt.

Anforderungsratenkontingente pro Benutzer in Amazon-Cognito-Benutzerpools

Operation	Operationen pro Benutzer und Sekunde
Lesen eines Benutzerprofils Beispiele: `GetUser`, `GetDevice`	10
Schreiben eines Benutzerprofils	10

Operation	Operationen pro Benutzer und Sekunde
Beispiele: `UpdateUserAttributes` , `SetUserSettings`	

Sie müssen die API-Anforderungen pro Kategorie begrenzen, wie in der folgenden Tabelle dargestellt.

Anforderungsratenkontingente pro Kategorie in Amazon-Cognito-Benutzerpools

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserAuthentication` • InitiateAuth • Token-Aktualisierung mit `InitiateAuth` oder Token-Endpunkt • RespondToAuthChallenge[1] • AdminInitiateAuth • AdminRespondToAuthChallenge[1] • Gehostete UI-Anmeldung und MFA im Autorisierungscode oder implizite Erteilungen[2]	Operationen, die einen Benutzer authentifizieren (anmelden). Diese Operationen unterliegen API-Operationen von Amazon-Cognito-Benutzerpools mit spezieller Handhabung der Anforderungsrate .	120	Ja
`UserCreation`	Operationen, die einen lokalen	50	Ja

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
• SignUp • ConfirmSignUp • AdminCreateUser • AdminConfirmSignUp	Amazon-Cognito-Benutzer erstellen oder bestätigen. Dies ist ein Benutzer, der direkt von Ihren Amazon-Cognito-Benutzerpools erstellt und überprüft wird.		
`UserFederation` Operationen, die Benutzer mit einem Drittanbieter-Identitätsanbieter in Ihren Amazon-Cognito-Benutzerpools verbinden (authentifizieren).	Operationen, die eine IDP-Antwort an einen Endpunkt eines Benutzerpoolverbunds senden. OIDC- oder Social-Provider-Operationen, die zu einem IDP-Token führen, sowie alle SAML-Anfragen tragen zu diesem Kontingent bei.	25	Ja

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserAccountRecovery` • ChangePassword • ConfirmForgotPassword • ForgotPassword • AdminResetUserPassword • AdminSetUserPassword • RespondToAuthChallenge[1] • AdminRespondToAuthChallenge[1] • Passwortzurücksetzung für die gehostete Benutzeroberfläche	Operationen, die das Benutzerkonto wiederherstellen oder das Kennwort eines Benutzers ändern oder aktualisieren.	30	Nein
`UserRead` • AdminGetUser • GetUser	Operationen, die einen Benutzer aus Ihren Benutzerpools abrufen.	120	Ja

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserUpdate` • AdminAddUserToGroup • AdminDeleteUserAttributes • AdminUpdateUserAttributes • AdminDeleteUser • AdminDisableUser • AdminEnableUser • AdminLinkProviderForUser • AdminDisableProviderForUser • VerifyUserAttribute • DeleteUser • DeleteUserAttributes • UpdateUserAttributes • AdminUserGlobalSignOut • GlobalSignOut • AdminRemoveUserFromGroup	Vorgänge, die Sie zum Verwalten von Benutzern und Benutzerattributen verwenden.	25	Nein
`UserToken` • RevokeToken	Operationen für das Token-Management	120	Ja

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserResourceRead` • AdminGetDevice • AdminListGroupsForUser • AdminListDevices • GetDevice • ListDevices • GetUserAttributeVerificationCode • ResendConfirmationCode • AdminListUserAuthEvents	Vorgänge, die Benutzerressourcen informationen von Amazon Cognito abrufen, wie z. B. ein gespeichertes Gerät oder eine Gruppenmitgliedschaft.	50	Ja

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserResourceUpdate` • AdminForgetDevice • AdminUpdateAuthEventFeedback • AdminSetUserMFA-Präferenz • AdminSetUserSettings • AdminUpdateDeviceStatus • UpdateDeviceStatus • UpdateAuthEventFeedback • ConfirmDevice • SetUserMFAP-Referenz • SetUserSettings • VerifySoftwareToken • AssociateSoftwareToken • ForgetDevice	Vorgänge, die Ressourceninformationen für einen Benutzer aktualisieren, z. B. ein gespeichertes Gerät oder eine Gruppenmitgliedschaft.	25	Nein
`UserList` • ListUsers • ListUsersInGroup	Operationen, die eine Liste von Benutzern zurückgeben.	30	Nein

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserPoolRead` • DescribeUserPool • ListUserPools	Operationen, die Ihre Benutzerpools lesen.	15	Nein
`UserPoolUpdate` • CreateUserPool • UpdateUserPool • DeleteUserPool	Operationen zum Erstellen, Aktualisieren oder Löschen Ihrer Benutzerpools.	15	Nein

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserPoolResourceRead` • DescribeIdentityProvider • DescribeResourceServer • DescribeUserImportJob • DescribeUserPoolDomain • GetCSVHeader • GetGroup • GetSigningCertificate • GetIdentityProviderByIdentifier • GetUserPoolMfaConfig • ListGroups • ListIdentityProviders • ListResourceServers • ListTagsForResource • ListUserImportJobs • DescribeRiskConfiguration • GetUICustomization	Vorgänge, die Informationen über Ressourcen wie Gruppen oder Ressourcenserver aus einem Benutzerpool abrufen.[3]	20	Nein

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`UserPoolResourceUpdate` • AddCustomAttributes • CreateGroup • CreateIdentityProvider • CreateResourceServer • CreateUserImportJob • CreateUserPoolDomain • DeleteGroup • DeleteIdentityProvider • DeleteResourceServer • DeleteUserPoolDomain • SetUserPoolMfaConfig • StartUserImportJob • StopUserImportJob • UpdateGroup • UpdateIdentityProvider • UpdateResourceServer	Vorgänge, die Ressourcen wie Gruppen oder Ressourcenserver in einem Benutzerpool ändern.[3]	15	Nein

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
• UpdateUserPoolDomain • SetRiskConfiguration • SetUICustomization • TagResource • UntagResource			
`UserPoolClientRead` • DescribeUserPoolClient • ListUserPoolClients	Vorgänge, die Informationen über Ihre Benutzerpool-Clients abrufen.[3]	15	Nein
`UserPoolClientUpdate` • CreateUserPoolClient • DeleteUserPoolClient • UpdateUserPoolClient	Vorgänge zum Erstellen, Aktualisieren und Löschen Ihrer Benutzerpool-Clients.[3]	15	Nein
`ClientAuthentication` Anfragen des Erteilungstyps `client_credentials` an den Token-Endpunkt.	Operationen, die Anmeldeinformationen generieren, die bei der Autorisierung von Anfragen verwendet werden machine-to-machine	150	Nein

[1] Eine `AdminRespondToAuthChallenge` Antwort `RespondToAuthChallenge` oder Antwort mit einem `ChallengeName` von `NEW_PASSWORD_REQUIRED` zählt zur `UserAccountRecovery` Kategorie. Alle anderen Challenge-Antworten werden der `UserAuthentication` Kategorie zugerechnet.

[2] Jeder Vorgang auf der gehosteten Benutzeroberfläche während der Anmeldung trägt mit einer Anfrage zum Kontingent bei. Beispielsweise trägt ein Benutzer, der sich anmeldet und einen MFA-Code angibt, zwei Anfragen bei. Die Einlösung von Tokens im Rahmen von Autorisierungscode-Zuweisungen unterliegt einer zusätzlichen Kontingentzuweisung in derselben Höhe wie Ihr Kontingent in der Kategorie. `UserAuthentication`

[3] Jeder einzelne Vorgang in dieser Kategorie hat eine Einschränkung, die verhindert, dass der Vorgang mit einer Rate von mehr als 5 RPS für einen einzelnen Benutzerpool aufgerufen wird.

Amazon Cognito-Identitätspools (verbundene Identitäten) API-Vorgangsanforderungsratenkontingente

Operation	Beschreibung	Standardkontingent (RPS)[1]	Einstellbar	Erhöhung der Berechtigung
`GetId`	Ruft eine Identitäten-ID aus einem Identitäten-Pool ab.	25	Ja	Wenden Sie sich an Ihr Kontoteam.
`GetOpenIdToken`	Rufen Sie ein OpenID-Token aus einem Identitätspool im klassischen Workflow ab.	200	Ja	Wenden Sie sich an Ihr Kontoteam.
`GetCredentialsForIdentity`	Rufen Sie im erweiterten Workflow AWS Anmeldein	200	Ja	Wenden Sie sich an Ihr Kontoteam.

Operation	Beschreibung	Standardkontingent (RPS)[1]	Einstellbar	Erhöhung der Berechtigung
	formationen aus einem Identität spool ab.			
`GetOpenIdTokenForDeveloperIdentity`	Rufen Sie ein OpenID-Token aus einem Identitätspool im Entwickler-Workflow ab.	50	Ja	Wenden Sie sich an Ihr Kontoteam.
`ListIdentities`	Ruft eine Liste von Identität skennungen in einem Identität spool ab.	5	Ja	Wenden Sie sich an Ihr Kontoteam.
`DeleteIdentities`	Löscht eine oder mehrere registrierte Identitäten aus einem Identitätspool.	10	Ja	Wenden Sie sich an Ihr Kontoteam.
`TagResource`	Wenden Sie ein Tag auf einen Identitätspool an.	5	Ja	Wenden Sie sich an Ihr Kontoteam.
`UntagResource`	Entfernen Sie ein Tag aus einem Identitätspool.	5	Ja	Wenden Sie sich an Ihr Kontoteam.

Operation	Beschreibung	Standardkontingent (RPS)[1]	Einstellbar	Erhöhung der Berechtigung
`ListTagsForResource`	Zeigt eine Liste der Tags an, die auf einen Identitätspool angewendet wurden.	10	Ja	Wenden Sie sich an Ihr Kontoteam.

[1] Das Standardkontingent ist das Mindestkontingent für die Anforderungsrate für die Identitätspools AWS-Region in einem Ihrer AWS-Konto. Ihr RPS-Kontingent ist in einigen Regionen möglicherweise höher.

Kontingente für die Anzahl und Größe der Ressourcen

Ressourcenkontingente beziehen sich auf die maximale Anzahl oder Größe von Ressourcen, Eingabefeldern, Zeitdauern und verschiedenen anderen Funktionen in Amazon Cognito.

Sie können eine Anpassung einiger Ressourcenkontingente in der Service-Quotas-Konsole oder über ein Formular zur Erhöhung des Servicelimits beantragen. Informationen zum Anfordern eines Kontingents über die Service-Quotas-Konsole finden Sie unter Anfordern einer Kontingenterhöhung im Service-Quotas-Benutzerhandbuch. Wenn das Kontingent in Service Quotas noch nicht in verfügbar ist, verwenden Sie das Formular zur Erhöhung des Service-Limits.

> **Note**
>
> Ressourcenkontingente auf der jeweiligen AWS-Konto Ebene, wie Benutzerpools pro Region, gelten jeweils AWS-Region für Amazon Cognito Cognito-Ressourcen. Sie können beispielsweise 1 000 Benutzerpools in USA Ost (Nord-Virginia) und weitere 1 000 in Europa (Stockholm) haben.

In den folgenden Tabellen sind die Standard-Ressourcenkontingente aufgeführt und es ist angegeben, ob diese einstellbar sind.

Ressourcenkontingente für Amazon-Cognito-Benutzerpools

Ressource	Kontingent	Einstellbar	Höchstkontingent
App-Clients pro Benutzerpool	1.000	Ja	10.000
Benutzerpools pro Region	1.000	Ja	10.000
Identitätsanbieter pro Benutzerpool	300	Ja	1.000
Ressourcenserver pro Benutzerpool	25	Ja	300
Benutzer pro Benutzerpool	40.000.000	Ja	Wenden Sie sich an Ihr Kontoteam.
Gesamtzahl aller Änderungen im Pre-Token-Generierungs-Lambda-Trigger [1]	5,000	Ja	Wenden Sie sich an Ihr Kontoteam.
Benutzerdefinierte Attribute pro Benutzerpool	50	Nein	N/A
Zeichenanzahl pro Attribut	2048 Bytes	Nein	N/A
Zeichen im benutzerdefinierten Attributnamen	20	Nein	N/A
Erforderliche Mindestkennwortzeichen in Passwortrichtlinie.	6–99	Nein	N/A

Ressource	Kontingent	Einstellbar	Höchstkontingent
Täglich gesendete E-Mail-Nachrichten pro AWS-Konto[2]	50	Nein	N/A
Zeichenanzahl in E-Mail-Betreff	140	Nein	N/A
Zeichenanzahl in E-Mail-Nachricht	20 000	Nein	N/A
Zeichen in der SMS-Bestätigungsnachricht	140	Nein	N/A
Zeichenanzahl in Passwort	256	Nein	N/A
Zeichenanzahl im Namen des Identität sanbieters	32	Nein	N/A
Identifikatoren pro Identitätsanbieter	50	Nein	N/A
Identitäten, die mit einem Benutzer verbunden sind	5	Nein	N/A
Rückruf-URLs pro App-Client	100	Nein	N/A
Abmelde-URLs pro App-Client	100	Nein	N/A
Bereiche pro Ressourcenserver	100	Nein	N/A

Ressource	Kontingent	Einstellbar	Höchstkontingent
Bereiche pro App-Client	50	Nein	N/A
Benutzerdefinierte Domänen pro Konto	4	Nein	N/A
Gruppen, denen jeder Benutzer angehören kann	100	Nein	N/A
Gruppen pro Benutzerpool	10.000	Nein	N/A

[1] Dieses Kontingent kann bei Token aus einem Lambda-Auslöser für die Vorab-Generierung von Token auftreten. Die Summe der vorhandenen und hinzugefügten Ansprüche sowie der Bereiche in Zugriffs- und Identitäts-Token muss kleiner oder gleich dem Wert dieses Kontingents sein. Unterdrückte Ansprüche und Bereiche tragen nicht zu diesem Kontingent bei.

[2] Dieses Kontingent gilt nur, wenn Sie das Standard-E-Mail-Feature für einen Amazon-Cognito-Benutzerpool verwenden. Wenn Sie das E-Mail-Zustellungsvolumen erhöhen möchten, richten Sie Ihren Benutzerpool so ein, dass er Ihre Amazon-SES-E-Mail-Konfiguration verwendet. Weitere Informationen finden Sie unter E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools.

Gültigkeitsparameter für Amazon-Cognito-Benutzerpools

Token	Quota
ID-Token	5 Minuten – 1 Tag
Aktualisierungs-Token	1 Stunde – 3.650 Tage
Zugriffstoken	5 Minuten – 1 Tag
Cookie der gehosteten Benutzeroberfläche	1 Stunde
Sitzungstokens für Authentifizierungen	3 Minuten – 15 Minuten

Kontingente für Code-Sicherheitsressourcen für Amazon-Cognito-Benutzerpools (nicht anpassbar)

Ressource	Kontingent
Gültigkeitszeitraum für Anmeldebestätigung scode	24 Stunden
Gültigkeitszeitraum des Verifizierungscodes für Benutzerattribute	24 Stunden
Gültigkeitszeitraum des Codes für die Multi-Faktor-Authentifizierung (MFA)	3–15 Minuten
Gültigkeitszeitraum des Codes für „Passwort vergessen“	1 Stunde
Maximale Anzahl der `ConfirmForgotPassword` - und `ForgotPassword` -Anfragen pro Benutzer und Stunde[1]	5–20
Maximale Anzahl der `ResendConfirmationCode` -Anfragen pro Benutzer und Stunde	5
Maximale Anzahl der `ConfirmSignUp` -Anfragen pro Benutzer und Stunde	15
Maximale Anzahl der `ChangePassword` -Anfragen pro Benutzer und Stunde	5
Maximale Anzahl der `GetUserAttributeVerificationCode` -Anfragen pro Benutzer und Stunde	5
Maximale Anzahl der `VerifyUserAttribute` -Anfragen pro Benutzer und Stunde	15

[1] Amazon Cognito bewertet Risikofaktoren in der Anfrage zur Aktualisierung von Passwörtern und weist ein Kontingent zu, das an das bewertete Risikoniveau gebunden ist. Weitere Informationen finden Sie unter Verhalten bei „Passwort vergessen“.

Kontingente für Auftragsressourcen für Benutzerimporte von Amazon-Cognito-Benutzerpools

Ressource	Kontingent	Einstellbar	Höchstkontingent
Benutzerimportaufträge pro Benutzerpool	1.000	Ja	Wenden Sie sich an Ihr Kontoteam.
Maximale Zeichen pro Benutzerimport-CSV-Zeile	16,000	Nein	N/A
Maximale CSV-Dateigröße	100 MB	Nein	N/A
Maximale Anzahl von Benutzern pro CSV-Datei	500 000	Nein	N/A

Ressourcenkontingente für Amazon-Cognito-Identitätspools (verbundene Identitäten)

Ressource	Kontingent	Einstellbar	Höchstkontingent
Identitäten-Pools pro Konto	1.000	Ja	N/A
Amazon-Cognito-Benutzerpoolanbieter pro Identitätspool	50	Ja	1000
Zeichenlänge für Identitäten-Pool-Name	128 Byte	Nein	N/A

Ressource	Kontingent	Einstellbar	Höchstkontingent
Zeichenlänge eines Anmeldeanbieternamens	2048 Bytes	Nein	N/A
Identitäten pro Identitäten-Pool	Unbegrenzt	Nein	N/A
Identitätsanbieter, für die Rollenmappings angegeben werden können	10	Nein	N/A
Ergebnisse aus einer einzelnen Liste oder einem Suchaufruf	60	Nein	N/A
Regeln für rollenbasierte Zugriffskontrolle (RBAC)	25	Nein	N/A

Ressourcenkontingente für Amazon Cognito Sync

Ressource	Kontingent	Einstellbar	Höchstkontingent
Datensätze pro Identität	20	Ja	Wenden Sie sich an Ihr Kontoteam.
Akten pro Datensatz	1,024	Ja	Wenden Sie sich an Ihr Kontoteam.
Größe eines Datensatzes	1 MB	Ja	Wenden Sie sich an Ihr Kontoteam.
Zeichenanzahl im Datensatz-Namen	128 Byte	Nein	N/A

Ressource	Kontingent	Einstellbar	Höchstkontingent
Wartezeit für eine Massen-Veröffentlichung nach einer erfolgreichen Anfrage	24 Stunden	Nein	N/A

Amazon-Cognito-API- und Endpunkt-Referenzen

Die folgenden Referenzen beschreiben die Service-Endpunkte für jede Funktion von Amazon Cognito. Amazon-Benutzerpools haben die folgenden Optionen: Benutzerpool-Endpunkte mit einer Benutzerpool-Domain und die Benutzerpool-API. Eine Aufschlüsselung der Klassen von API-Operationen mit der Amazon-Cognito-Benutzerpool-API finden Sie unter Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte.

Eine Liste der Service-Endpunkte für die Benutzerpool-API nach AWS-Region finden Sie unter Service-Endpunkte in der allgemeinen AWS-Referenz.

Themen

- Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI
- API-Referenz der Amazon-Cognito-Benutzerpools
- API-Referenz des Amazon-Cognito-Identitätspools (Verbundidentitäten)
- Amazon-Cognito-Sync-API-Referenz

Referenz für Benutzerpool-Verbund-Endpunkte und gehostete UI

Amazon Cognito aktiviert die hier aufgeführten öffentlichen Webseiten, wenn Sie Ihrem Benutzerpool eine Domain zuweisen. Ihre Domain dient als zentraler Zugangspunkt für alle Ihre App-Clients. Sie umfassen die gehostete Benutzeroberfläche, auf der sich Ihre Benutzer registrieren und anmelden (Login-Endpunkt) sowie abmelden (Logout-Endpunkt) können. Weitere Informationen zu diesen Ressourcen finden Sie unter Einrichtung und Verwendung der gehosteten Benutzeroberfläche und der Verbundendpunkte in Amazon Cognito.

Diese Seiten enthalten auch die öffentlichen Webressourcen, die es Ihrem Benutzerpool ermöglichen, mit SAML-, OpenID Connect- (OIDC) - und OAuth 2.0-Identitätsanbietern von Drittanbietern () zu kommunizieren. IdPs Zum Anmelden eines Benutzers bei einem Verbundidentitätsanbieter müssen Ihre Benutzer eine Anfrage an die interaktive UI Login-Endpunkt oder den OIDC-Autorisieren des Endpunkts senden. Der Authorize-Endpunkt leitet Ihre Benutzer entweder zu Ihrer gehosteten Benutzeroberfläche oder zu Ihrer IdP-Anmeldeseite weiter.

Ihre App kann auch lokale Benutzer mit der Amazon-Cognito-Benutzerpool-API anmelden. Ein lokaler Benutzer existiert ausschließlich in Ihrem Benutzerpool-Verzeichnis ohne Verbund über einen externen IdP.

Zusätzlich zur gehosteten Benutzeroberfläche und den Verbundendpunkten lässt sich Amazon Cognito in SDKs für Android JavaScript, iOS und mehr integrieren. Die SDKs bieten Tools zur Interaktion Durchführung von Benutzerpool-API-Operationen mit Amazon-Cognito-API-Serviceendpunkten. Weitere Hinweise zu Service-Endpunkten finden Sie unter Endpunkte und Kontingente von Amazon Cognito Identity.

> **Warning**
>
> Fixieren Sie die Endentitäts- oder Zwischenzertifikate für Transport Layer Security (TLS) nicht für Amazon Cognito Cognito-Domains. AWS verwaltet alle Zertifikate für all Ihre Benutzerpool-Endpunkte und Präfix-Domains. Die Zertifizierungsstellen (CAs) in der Vertrauenskette, die Amazon-Cognito-Zertifikate unterstützt, rotieren und erneuern sich dynamisch. Wenn Sie Ihre App an ein Zwischen- oder Leaf-Zertifikat anheften, kann es sein, dass Ihre App ohne Vorankündigung fehlschlägt, wenn Zertifikate AWS rotiert werden. Heften Sie Ihre Anwendung stattdessen an alle verfügbaren Amazon-Stammzertifikate an. Weitere Informationen finden Sie unter „Bewährte Verfahren und Empfehlungen“ unter Zertifikat-Pinning im AWS Certificate Manager -Benutzerhandbuch.

Themen

Referenz für gehostete UI-Endpunkte

Amazon Cognito aktiviert die gehosteten UI-Endpunkte in diesem Abschnitt, wenn Sie Ihrem Benutzerpool eine Domain hinzufügen. Dabei handelt es sich um Webseiten, auf denen Ihre Benutzer die wichtigsten Authentifizierungsvorgänge eines Benutzerpools durchführen können. Sie enthalten Seiten für Passwortverwaltung, Multi-Faktor-Authentifizierung (MFA) und Attributüberprüfung. Weitere Informationen zur Umgebung in der gehosteten Benutzeroberfläche finden Sie unter Registrieren und Anmelden mit der gehosteten Benutzeroberfläche.

Die Webseiten, aus denen sich die gehostete Benutzeroberfläche zusammensetzt, sind eine Front-End-Webanwendung für interaktive Benutzersitzungen mit Ihren Kunden. Ihre App muss die gehostete Benutzeroberfläche in den Browsern Ihrer Benutzer aufrufen. Amazon Cognito unterstützt keinen programmatischen Zugriff auf die Webseiten in diesem Kapitel. Die Verbund-Endpunkte in der Referenz für OAuth-2.0-, OpenID-Connect- und SAML-2.0-Verbund-Endpunkte, die eine JSON-Antwort zurückgeben, können direkt in Ihrem App-Code abgefragt werden. Der Autorisieren des Endpunkts leitet Benutzer entweder zur gehosteten Benutzeroberfläche oder zu einer IdP-Anmeldeseite weiter und muss auch in den Browsern der Benutzer geöffnet werden.

In den Themen dieses Handbuchs werden häufig verwendete gehostete UI-Endpunkte ausführlich beschrieben. Amazon Cognito stellt die folgenden Webseiten bereit, wenn Sie Ihrem Benutzerpool eine Domain zuweisen.

Gehostete UI-Endpunkte

Endpunkt-URL	Beschreibung	Zugriff darauf
https://*Ihre Benutzerpool-Domain* /login	Meldet lokale Benutzer und Verbundbenutzer im Benutzerpool an.	Weiterleitung von Endpunkten wie Autorisieren des Endpunkts, `/logout` und `/confirmforgotPassword` . Siehe Login-Endpunkt.
https://*Ihre Benutzerpool-Domain* /logout	Meldet Benutzer des Benutzerpools ab.	Direkt-Link. Siehe Logout-Endpunkt.
https://*Ihre Benutzerpool-Domäne* /confirmUser	Bestätigt Benutzer, die einen E-Mail-Link ausgewählt haben, um ihr Benutzerkonto zu verifizieren.	Der Benutzer hat einen Link in einer E-Mail-Nachricht ausgewählt.
https://*Ihre Benutzerpool-Domäne* /signup	Registriert einen neuen Benutzer. Die Seite `/login` leitet die Benutzer an `/signup` weiter, wenn sie Sign up (Registrieren) auswählen.	Direkter Link mit den gleichen Parametern wie `/oauth2/authorize` .
https://*Ihre Benutzerpool-Domain* /confirm	Nachdem Ihr Benutzerpool einen Bestätigungscode an	Nur Weiterleitung von `/signup`.

Endpunkt-URL	Beschreibung	Zugriff darauf
	einen Benutzer gesendet hat, der sich angemeldet hat, fordert er Ihren Benutzer auf, den Code einzugeben.	
https://*Ihre Benutzerpool-Domain* /forgotPassword	Fordert die Benutzer zur Eingabe des Benutzernamens auf und sendet einen Code zum Zurücksetzen des Passworts. Die Seite `/login` leitet die Benutzer an `/forgotPassword` weiter, wenn sie Forgot your password? (Passwort vergessen?) auswählen.	1. Über den Link Passwort vergessen unter `/login`. 2. Direkter Link mit den gleichen Parametern wie `/oauth2/authorize` .
https://*Ihre Benutzerpool-Domain* /confirmforgotPassword	Fordert die Benutzer auf, den Code zum Zurücksetzen des Passworts und anschließend ein neues Passwort einzugeben. Die Seite `/forgotPassword` leitet die Benutzer an `/confirmforgotPassword` weiter, wenn sie Reset your password (Passwort zurücksetzen) auswählen.	Nur Weiterleitung von `/forgotPassword` .
https://*Ihre Benutzerpool-Domain* /resendcode	Sendet einen neuen Bestätigungscode an einen Benutzer, der sich in Ihrem Benutzerpool angemeldet hat.	Nur Weiterleitung von Neuen Code senden unter `/confirm`.

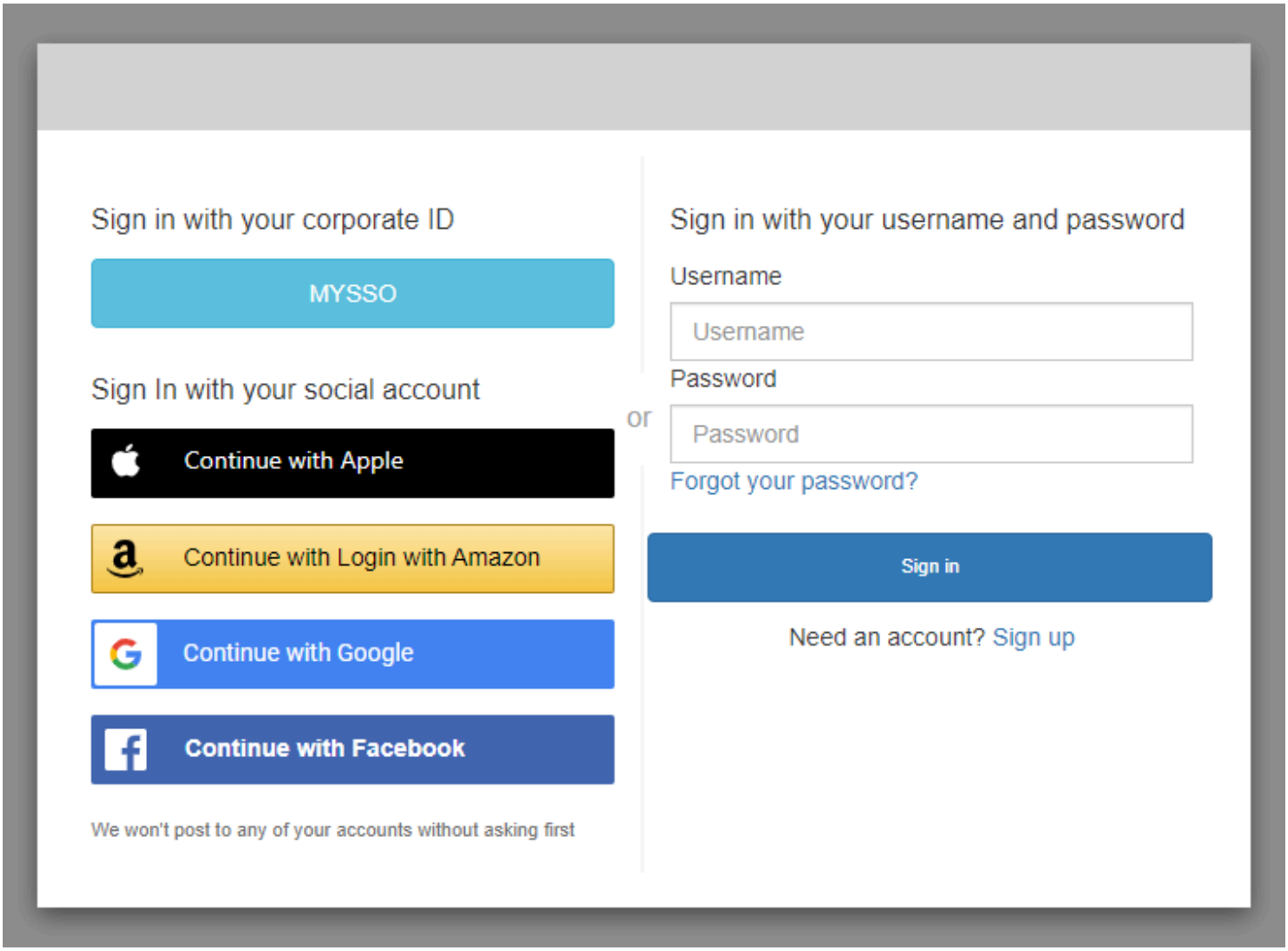
Themen

- Login-Endpunkt
- Logout-Endpunkt

Login-Endpunkt

Der Anmeldeendpunkt ist ein Authentifizierungsserver und ein Weiterleitungsziel von Autorisieren des Endpunkts. Es ist der Einstiegspunkt zur gehosteten Benutzeroberfläche, wenn Sie keinen Identitätsanbieter angeben. Wenn Sie eine Weiterleitung zum Anmeldeendpunkt generieren, wird die Anmeldeseite geladen und dem Benutzer werden die für den Client konfigurierten Authentifizierungsoptionen angezeigt.

> **Note**
>
> Der Anmeldeendpunkt ist eine Komponente der gehosteten Benutzeroberfläche. Rufen Sie in Ihrer App Verbundseiten und gehostete Benutzeroberflächenseiten auf, die Benutzer zum Anmeldeendpunkt weiterleiten. Der direkte Zugriff von Benutzern auf den Anmeldeendpunkt hat sich als Methode nicht bewährt.



GET /login

Der `/login`-Endpunkt unterstützt `HTTPS GET` nur für die erste Anfrage Ihres Benutzers. Ihre App ruft die Seite in einem Browser wie Chrome oder Firefox auf. Wenn Sie `/login` von der zu weiterleitenAutorisieren des Endpunkts, werden alle Parameter weitergegeben, die Sie in Ihrer ursprünglichen Anfrage angegeben haben. Der Anmeldeendpunkt unterstützt alle Anforderungsparameter des Autorisierungsendpunkts. Sie können auch direkt auf den Anmeldeendpunkt zugreifen. Es hat sich als Methode bewährt, die Sitzungen aller Benutzer mit `/oauth2/authorize` zu starten.

Beispiel — Fordere den Benutzer auf, sich anzumelden

Dieses Beispiel zeigt einen Anmeldebildschirm an.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/login?
```

```
                response_type=code&
                client_id=ad398u21ijw3s9w3939&
                redirect_uri=https://YOUR_APP/redirect_uri&
                state=STATE&
                scope=openid+profile+aws.cognito.signin.user.admin
```

Beispiel — Antwort

Der Authentifikationsserver wird mit Autorisierungs-Code und -Status zurück an Ihre App geleitet. Der Server muss den Code und den Status an die Abfragezeichenfolge-Parameter zurücksenden und nicht an das Fragment.

```
HTTP/1.1 302 Found
                    Location: https://YOUR_APP/redirect_uri?
code=AUTHORIZATION_CODE&state=STATE
```

Vom Benutzer initiierte Anmeldeanfrage

Nachdem Ihr Benutzer den `/login`-Endpunkt geladen hat, kann er einen Benutzernamen und ein Passwort eingeben und Anmelden wählen. Wenn er dies tut, generiert er eine `HTTPS POST`-Anfrage mit denselben Header-Anforderungsparametern wie die `GET`-Anfrage und einem Anforderungstext mit seinem Benutzernamen, Passwort und einem Gerätefingerabdruck.

Logout-Endpunkt

Der `/logout`-Endpunkt ist ein Umleitungsendpunkt. Es meldet den Benutzer ab und leitet entweder zu einer autorisierten Abmelde-URL für Ihren App-Client oder zum `/login` Endpunkt weiter. Die verfügbaren Parameter in einer GET-Anfrage an den `/logout` Endpunkt sind auf Anwendungsfälle der gehosteten Benutzeroberfläche von Amazon Cognito zugeschnitten.

Um Ihren Benutzer zur gehosteten Benutzeroberfläche umzuleiten, um sich erneut anzumelden, fügen Sie Ihrer Anfrage einen `redirect_uri`-Parameter hinzu. Eine `logout`-Anfrage mit einem `redirect_uri`-Parameter muss auch Parameter für Ihre nachfolgende Anfrage an den Login-Endpunkt enthalten, wie `client_id`, `response_type` und `scope`.

Der Abmelde-Endpunkt ist eine Front-End-Webanwendung für interaktive Benutzersitzungen mit Ihren Kunden. Ihre App muss diesen und andere gehostete UI-Endpunkte in den Browsern Ihrer Benutzer aufrufen.

Um Ihren Benutzer auf eine von Ihnen gewählte Seite umzuleiten, fügen Sie Ihrem App-Client die zulässigen Abmelde-URLs hinzu. Fügen Sie in den Anfragen Ihrer Benutzer an den `logout`-Endpunkt `logout_uri`- und `client_id`-Parameter hinzu. Wenn der Wert von `logout_uri` eine der zulässigen Abmelde-URLs für Ihren App-Client ist, leitet Amazon Cognito Benutzer zu dieser URL weiter.

Mit Single Logout (SLO) für SAML 2.0 IdPs leitet Amazon Cognito Ihren Benutzer zunächst an den SLO-Endpunkt weiter, den Sie in Ihrer IdP-Konfiguration definiert haben. Nachdem Ihr IdP Ihren Benutzer zurück zu geleitet hat`saml2/logout`, antwortet Amazon Cognito mit einer weiteren Weiterleitung auf die `redirect_uri` oder `logout_uri` von Ihrer Anfrage. Weitere Informationen finden Sie unter Ablauf der SAML-Abmeldung.

Der Abmeldeendpunkt meldet Benutzer nicht von OIDC oder Anbietern sozialer Identitäten ab (). IdPs Um Benutzer von ihrer Sitzung mit einem externen IdP abzumelden, leiten Sie sie zur Abmeldeseite für diesen Anbieter weiter.

GET/Abmeldung

Der `/logout` Endpunkt unterstützt ausschließlich `HTTPS GET`. Der Benutzerpool-Client übermittelt diese Anfrage in der Regel über den Systembrowser. Der Browser ist normalerweise ein Custom Chrome Tab bei Android oder Safari Control View bei iOS.

Anforderungsparameter

Client-ID

Die App-Client-ID für Ihre Anwendung. Um eine Client-ID für Ihre App abzurufen, müssen Sie die App im Benutzerpool registrieren. Weitere Informationen finden Sie unter App-Clients für Benutzerpools.

Erforderlich.

Abmelde-Uri

Leiten Sie Ihren Benutzer mit einem Parameter logout-uri auf eine benutzerdefinierte Abmeldeseite um. Legen Sie seinen Wert auf die sign-out URL (Abmelde-URL) des App-Clients fest, an die Sie Ihren Benutzer umleiten möchten, nachdem er sich abgemeldet hat. Verwenden Sie logout_uri nur mit einem client_id-Parameter. Weitere Informationen finden Sie unter App-Clients für Benutzerpools.

Sie können auch den logout_uri-Parameter verwenden, um Ihren Benutzer auf die Anmeldeseite eines anderen App-Clients umzuleiten. Legen Sie die Anmeldeseite für den anderen App-Client

als Allowed callback URL (Zulässige Rückruf-URL) in Ihrem App-Client fest. Legen Sie in Ihrer Anforderung an den `/logout`-Endpunkt den Wert des logout_uri-Parameters auf die URL-codierte Anmeldeseite fest.

Amazon Cognito benötigt entweder einen logout_uri- oder einen redirect_uri-Parameter in der Anforderung an den `/logout`-Endpunkt. Ein logout_uri-Parameter leitet Ihren Benutzer auf eine andere Website um. Wenn sowohl die Parameter logout_uri als auch redirect_uri in Ihrer Anfrage an den `/logout`-Endpunkt enthalten sind, verwendet Amazon Cognito ausschließlich den Parameter logout_uri und überschreibt damit den Parameter redirect_uri.

redirect_uri

Leiten Sie Ihren Benutzer auf Ihre Anmeldeseite um, um sich mit einem Parameter redirect_uri zu authentifizieren. Legen Sie seinen Wert auf die Allowed callback URL (Zulässige Rückruf-URL) des App-Clients fest, an die Sie Ihren Benutzer umleiten möchten, nachdem er sich wieder angemeldet hat. Fügen Sie die Parameter client_id, scope, state und response_type hinzu, die Sie an Ihren `/login`-Endpunkt übergeben möchten.

Amazon Cognito benötigt entweder einen logout_uri- oder einen redirect_uri-Parameter in der Anforderung an den `/logout`-Endpunkt. Um Ihren Benutzer an Ihren *`/login`* Endpunkt weiterzuleiten, um sich erneut zu authentifizieren und Token an Ihre App zu übergeben, fügen Sie einen redirect_uri-Parameter hinzu. Wenn sowohl die Parameter logout_uri als auch redirect_uri in Ihrer Anfrage an den *`/logout`* Endpunkt enthalten sind, überschreibt Amazon Cognito den Parameter redirect_uri und verarbeitet ausschließlich den Parameter logout_uri.

response_type

Die OAuth 2.0-Antwort, die Sie von Amazon Cognito erhalten möchten, nachdem sich Ihr Benutzer angemeldet hat. `code` und `token` sind die gültigen Werte für den response_type-Parameter.

Erforderlich, wenn Sie einen redirect_uri-Parameter verwenden.

state

Wenn Ihre Anwendung einer Anfrage einen State-Parameter hinzufügt, gibt Amazon Cognito seinen Wert an Ihre App zurück, wenn der `/oauth2/logout` Endpunkt Ihren Benutzer umleitet.

Fügen Sie diesen Wert Ihren Anfragen hinzu, um sich vor CSRF-Angriffen zu schützen.

Sie können den Wert eines Parameters `state` nicht auf eine URL-codierte JSON-Zeichenfolge festlegen. Um eine Zeichenfolge, die diesem Format entspricht, in einem `state` Parameter

zu übergeben, kodieren Sie die Zeichenfolge auf Base64 und dekodieren Sie sie dann in Ihrer Anwendung.

Dringend empfohlen, wenn Sie einen redirect_uri-Parameter verwenden.

scope

Die OAuth 2.0-Bereiche, die Sie von Amazon Cognito anfordern möchten, nachdem Sie sie mit einem redirect_uri-Parameter abgemeldet haben. Amazon Cognito leitet Ihren Benutzer an den `/login`-Endpunkt mit dem scope-Parameter in Ihrer Anforderung an den `/logout`-Endpunkt um.

Optional, wenn Sie einen redirect_uri-Parameter verwenden. Wenn Sie keinen scope-Parameter angeben, leitet Amazon Cognito Ihren Benutzer an den `/login`-Endpunkt mit einem scope-Parameter um. Wenn Amazon Cognito Ihren Benutzer umleitet und automatisch `scope` ausfüllt, enthält der Parameter alle autorisierten Bereiche für Ihren App-Client.

Beispielanfragen

Beispiel — Abmelden und Benutzer zum Client weiterleiten

Mit Ausnahme von `logout_uri` und `client_id` werden alle möglichen Abfrageparameter für diesen Endpunkt an den weitergegebenAutorisieren des Endpunkts. Amazon Cognito leitet Benutzersitzungen an die URL im Wert von `logout_uri` weiter und ignoriert dabei alle anderen Anforderungsparameter, wenn Anfragen `logout_uri` und `client_id` enthalten. Bei dieser URL muss es sich um eine autorisierte Abmelde-URL für den App-Client handeln.

Im Folgenden finden Sie ein Beispiel für eine Anfrage zur Abmeldung und Weiterleitung zu `https://www.example.com/welcome`.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/logout?
  client_id=1example23456789&
  logout_uri=https%3A%2F%2Fwww.example.com%2Fwelcome
```

Beispiel: Melden Sie sich ab und fordern Sie den Benutzer auf, sich als ein anderer Benutzer anzumelden

Wenn Anfragen `logout_uri` auslassen, aber ansonsten die Parameter angeben, die eine wohlgeformte Anfrage an den Autorisierungsendpunkt ausmachen, leitet Amazon Cognito Benutzer zur gehosteten UI-Anmeldung weiter. Der Abmelde-Endpunkt hängt die Parameter in Ihrer ursprünglichen Anfrage an das Weiterleitungsziel an. Der Parameter `redirect_uri` in einer

Anfrage an den Abmelde-Endpunkt ist keine Abmelde-URL, sondern eine Anmelde-URL, die Sie an den Autorisierungs-Endpunkt weiterleiten sollten.

Im Folgenden finden Sie eine Beispielanforderung, die einen Benutzer abmeldet, zur Anmeldeseite weiterleitet und `https://www.example.com` nach der Anmeldung einen Autorisierungscode bereitstellt.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/logout?
  response_type=code&
  client_id=1example23456789&
  redirect_uri=https%3A%2F%2Fwww.example.com&
  state=example-state-value&
  nonce=example-nonce-value&
  scope=openid+profile+aws.cognito.signin.user.admin
```

Referenz für OAuth-2.0-, OpenID-Connect- und SAML-2.0-Verbund-Endpunkte

Amazon Cognito aktiviert die Endpunkte in diesem Abschnitt, wenn Sie Ihrem Benutzerpool eine Domain hinzufügen. Die Verbund-Endpunkte sind nicht benutzerinteraktiv. Sie übernehmen eine Dienstfunktion für Ihre App, um mit externen OAuth 2.0-, OIDC- und SAML 2.0-Identitätsanbietern zu kommunizieren (). IdPs

Die Themen in dieser Anleitung beschreiben mehrere häufig verwendete OAuth-2.0- und OIDC-Endpunkte. Amazon Cognito erstellt die folgenden Endpunkte, wenn Sie Ihrem Benutzerpool eine Domain zuweisen.

Benutzerpool-Verbund-Endpunkte

Endpunkt-URL	Beschreibung	Zugriff darauf
https://*Ihre Benutzerpool-Domain* /oauth2/authorize	Leitet einen Benutzer entweder zur gehosteten UI weiter oder zur Anmeldung mit seinem IdP weiter.	Wird im Kundenbrowser aufgerufen, um mit der Benutzerauthentifizierung zu beginnen. Siehe Autorisieren des Endpunkts.
https://*Ihre Benutzerpool-Domain* /oauth2/token	Gibt Token auf der Grundlage eines Autorisierungscodes oder einer Anforderung von	Von der App zum Abrufen von Tokens aufgefordert. Siehe Token-Endpunkt.

Endpunkt-URL	Beschreibung	Zugriff darauf
	Kundenanmeldeinformationen zurück.	
https://*Ihre Benutzerpool-Domain* /oauth2/userInfo	Gibt Benutzerattribute auf der Grundlage von OAuth-2.0-Bereichen und der Benutzeridentität in einem Zugriffs-Token zurück.	Von der App zum Abrufen des Benutzerprofils aufgefordert. Siehe UserInfo-Endpunkt.
https://*Ihre Benutzerpool-Domain* /oauth2/revoke	Widerruft ein Aktualisierungs-Token und die zugehörigen Zugriffs-Token.	Von der App aufgefordert, ein Token zu widerrufen. Siehe Widerrufen des Endpunkts.
https://cognito-idp.*Region*.amazonaws.com/*Ihre Benutzerpool-ID* /.well-known/openid-configuration	Ein Verzeichnis der OIDC-Architektur Ihres Benutzerpools.	Von der App aufgefordert, Metadaten von Ausstellern des Benutzerpools zu finden.
https://cognito-idp.*Region*.amazonaws.com/*Ihre Benutzerpool-ID* /.well-known/jwks.json	Öffentliche Schlüssel, mit denen Sie Amazon-Cognito-Token validieren können.	Von der App zur Überprüfung von JWTs angefordert.
https://*Ihre Benutzerpool-Domäne* /oauth2/idpresponse	Social-Identitätsanbieter müssen Ihre Benutzer mit einem Autorisierungscode an diesen Endpunkt umleiten. Amazon Cognito löst den Code gegen ein Token ein, wenn es Ihren Verbundbenutzer authentifiziert.	Weitergeleitet von der OIDC-IdP-Anmeldung als IdP-Client-Callback-URL.

Endpunkt-URL	Beschreibung	Zugriff darauf
https://*Ihre Benutzerpool-Domäne* /saml2/idpresponse	Die Assertion Consumer Response (ACS) -URL für die Integration mit SAML 2.0-Identitätsanbietern.	Umgeleitet von SAML 2.0-IdP als ACS-URL oder Ausgangspunkt für die vom IdP initiierte Anmeldung. [1]
https://Ihre Benutzerpool-Domain /saml2/logout	Die Single Logout (SLO) -URL für die Integration mit SAML 2.0-Identitätsanbietern.	Umgeleitet von SAML 2.0 IdP als Single Logout (SLO) -URL. Akzeptiert nur POST-Bindung.

[1] Weitere Informationen zur IDP-initiierten SAML-Anmeldung finden Sie unter. Verwenden der IDP-initiierten SAML-Anmeldung

Weitere Informationen zu den OpenID-Connect- und OAuth-Standards finden Sie unter OpenID Connect 1.0 und OAuth 2.0.

Themen

- Autorisieren des Endpunkts
- Token-Endpunkt
- UserInfo-Endpunkt
- Widerrufen des Endpunkts
- saml2/idpresponse-Endpunkt

Autorisieren des Endpunkts

Der `/oauth2/authorize`-Endpunkt ist ein Umleitungsendpunkt, der zwei Umleitungsziele unterstützt. Wenn Sie einen `identity_provider`- oder `idp_identifier`-Parameter in der URL angeben, werden Ihre Benutzer im Hintergrund auf die Anmeldeseite für diesen Identitätsanbieter (IDP) umgeleitet. Andernfalls erfolgt die Umleitung an Login-Endpunkt mit denselben URL-Parametern, die Sie in Ihre Anfrage aufgenommen haben.

Der Autorisierungs-Endpunkt leitet Benutzer entweder zur gehosteten Benutzeroberfläche oder zu einer IdP-Anmeldeseite weiter. Das Ziel einer Benutzersitzung an diesem Endpunkt ist eine Webseite, mit der Ihr Benutzer direkt in seinem Browser interagieren muss.

Wenn Sie den Autorisierungsendpunkt verwenden möchten, rufen Sie den Browser Ihres Benutzers unter `/oauth2/authorize` mit Parametern auf, die Ihrem Benutzerpool Informationen zu den folgenden Benutzerpool-Details liefern.

- Der App-Client, bei dem Sie sich anmelden möchten.
- Die Rückruf-URL, zu der Sie gelangen möchten.
- Die OAuth-2.0-Bereiche, die Sie im Zugriffstoken Ihres Benutzers anfordern möchten.
- Optionaler Drittanbieter-IDP, den Sie für die Anmeldung verwenden möchten.

Sie können auch `state`- und `nonce`-Parameter angeben, die Amazon Cognito verwendet, um eingehende Ansprüche zu validieren.

GET **`/oauth2/authorize`**

Der `/oauth2/authorize` Endpunkt unterstützt ausschließlich `HTTPS GET`. Ihre App initiiert diese Anfrage normalerweise im Browser Ihres Benutzers. Sie können nur über HTTPS Anfragen an den `/oauth2/authorize`-Endpunkt stellen.

Weitere Informationen über die Definition des Autorisierungsendpunkts im OpenID Connect (OIDC)-Standard finden Sie unter Authorisierungsendpunkt.

Anforderungsparameter

`response_type`

(Erforderlich) Der Antworttyp. Es muss sich entweder um `code` oder `token` handeln.

Eine erfolgreiche Anfrage mit einem `response_type` von `code` gibt eine Autorisierungscode-Erteilung zurück. Eine Autorisierungscode-Erteilung ist ein `code`-Parameter, den Amazon Cognito an Ihre Umleitungs-URL anhängt. Ihre App kann den Code durch Zugriffs-, ID- und Aktualisierungstoken austauschen. Verwenden Sie als bewährte Sicherheitsmethode und zum Abrufen von Aktualisierungstoken für Ihre Benutzer eine Autorisierungscode-Erteilung in Ihrer App.

Eine erfolgreiche Anfrage mit dem `response_type token` gibt eine implizite Erteilung zurück. Eine implizite Erteilung besteht aus einer ID und einem Zugriffstoken, die Amazon Cognito an Ihre Umleitungs-URL anhängt. Eine implizite Erteilung ist weniger sicher, da sie Token und potenzielle identifizierende Informationen für Benutzer verfügbar macht. Sie können die Unterstützung für implizite Erteilungen in der Konfiguration Ihres App-Clients deaktivieren.

`client_id`

(Erforderlich) Die App-Client-ID.

Der Wert `client_id` muss die ID eines App-Clients in dem Benutzerpool sein, in dem Sie die Anfrage stellen. Ihr App-Client muss die Anmeldung durch lokale Benutzer von Amazon Cognito oder mindestens eines externen IdPs unterstützen.

`redirect_uri`

(Erforderlich) Die URL, an die der Authentifizierungsserver den Browser umleitet, nachdem Amazon Cognito den Benutzer autorisiert hat.

Ein Umleitungs-URI (Uniform Resource Identifier) muss die folgenden Attribute aufweisen:

- Es muss ein absoluter URI sein.
- Sie müssen die URI im Vorfeld mit einem Client registriert haben.
- Sie darf keine Fragmentkomponente enthalten.

Weitere Informationen finden Sie unter OAuth 2.0 – redirection endpoint (OAuth 2.0 – Umleitungsendpunkt).

Amazon Cognito erfordert, dass Ihr Umleitungs-URI HTTPS verwendet, mit Ausnahme von `http://localhost`, was Sie als Rückruf-URL für Testzwecke festlegen können.

Amazon Cognito unterstützt auch App-Callback-URLs wie `myapp://example`.

`state`

(Optional, empfohlen) Wenn Ihre App einer Anfrage einen State-Parameter hinzufügt, gibt Amazon Cognito seinen Wert an Ihre App zurück, wenn der `/oauth2/authorize` Endpunkt Ihren Benutzer weiterleitet.

Fügen Sie diesen Wert Ihren Anfragen hinzu, um sich vor CSRF-Angriffen zu schützen.

Sie können den Wert eines Parameters `state` nicht auf eine URL-codierte JSON-Zeichenfolge festlegen. Um eine Zeichenfolge, die diesem Format entspricht, in einem `state` Parameter zu übergeben, kodieren Sie die Zeichenfolge auf Base64 und dekodieren Sie sie dann in Ihrer App.

`identity_provider`

(Optional) Fügen Sie diesen Parameter hinzu, um die gehostete Benutzeroberfläche zu umgehen und Ihren Benutzer auf eine Anmeldeseite eines Anbieters umzuleiten. Der Wert

des identity_provider-Parameters ist der Name des Identitätsanbieters (IDP), wie er in Ihrem Benutzerpool angezeigt wird.

- Für soziale Anbieter können Sie die identity_provider-Werte`Facebook`,, und `Google` verwenden. `LoginWithAmazon` `SignInWithApple`
- Verwenden Sie für Amazon Cognito Cognito-Benutzerpools den Wert`COGNITO`.
- Verwenden Sie für SAML 2.0- und OpenID Connect (OIDC-) Identitätsanbieter (IdPs) den Namen, den Sie dem IdP in Ihrem Benutzerpool zugewiesen haben.

`idp_identifier`

(Optional) Fügen Sie diesen Parameter hinzu, um zu einem Anbieter mit einem alternativen Namen für den Namen identity_provider umzuleiten. Sie können Kennungen für Ihre SAML 2.0 und OIDC auf der Registerkarte Anmeldeerfahrung der Amazon IdPs Cognito Cognito-Konsole eingeben.

`scope`

(Optional) Kann eine Kombination aus beliebigen systemreservierten Bereichen oder benutzerdefinierten Bereichen sein, die einem Client zugeordnet sind. Bereiche müssen durch Leerzeichen getrennt werden. Für das System reservierte Bereiche sind `openidemail`, `phone`, `profile` und `aws.cognito.signin.user.admin`. Jeder Bereich muss dem Client zugeordnet werden, sonst wird der Client zur Laufzeit ignoriert.

Falls der Client keine Bereiche anfordert, verwendet der Authentifizierungsserver alle Bereiche im Zusammenhang mit dem Client.

Ein ID-Token wird nur zurückgegeben, wenn der `openid`-Bereich angefordert wird. Das Zugriffs-Token kann nur gegen Amazon-Cognito-Benutzerpools verwendet werden, wenn der Bereich `aws.cognito.signin.user.admin` angefordert wird. Die Bereiche `phone`, `email` und `profile` können nur angefordert werden, wenn der Bereich `openid` ebenfalls angefordert wird. Diese Bereiche bestimmen die Anträge, die im ID-Token eingesetzt werden.

`code_challenge_method`

(Optional) Das Hashing-Protokoll, mit dem Sie die Herausforderung generiert haben. Die PKCE RFC definiert zwei Methoden, die S256-Methode und eine einfache. Der Amazon-Cognito-Authentifikationsserver unterstützt jedoch nur die S256-Methode.

`code_challenge`

(Optional) Die Herausforderung, die Sie anhand der `code_verifier` generiert haben.

Nur erforderlich, wenn Sie einen `code_challenge_method`-Parameter angeben.

`nonce`

(Optional) Ein zufälliger Wert, den Sie der Anfrage hinzufügen können. Der von Ihnen bereitgestellte Nonce-Wert ist im ID-Token enthalten, das Amazon Cognito ausgibt. Zum Schutz vor Replay-Angriffen kann Ihre App den `nonce`-Anspruch im ID-Token untersuchen und mit dem vergleichen, den Sie generiert haben. Weitere Informationen zum `nonce`-Anspruch finden Sie unter ID-Token-Validierung im OpenID Connect-Standard.

Beispielanfragen mit positiven Antworten

Die folgenden Beispiele veranschaulichen das Format von HTTP-Anfragen an den `/oauth2/authorize` Endpunkt.

Erteilung des Autorisierungscodes

Dies ist ein Beispiel für eine Anfrage zur Erteilung eines Autorisierungscodes.

Beispiel — GET-Anfrage

Die folgende Anfrage initiiert eine Sitzung zum Abrufen eines Autorisierungscodes, den Ihr Benutzer am `redirect_uri` Ziel an Ihre App weitergibt. In dieser Sitzung werden Bereiche für Benutzerattribute und für den Zugriff auf Amazon Cognito-Self-Service-API-Operationen angefordert.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=openid+profile+aws.cognito.signin.user.admin
```

Beispiel — Antwort

Der Amazon-Cognito-Authentifikationsserver leitet Autorisierungs-Code und -Status zurück an Ihre App. Der Autorisierungscode ist fünf Minuten gültig.

```
HTTP/1.1 302 Found
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111&state=abcdefg
```

Erteilung des Autorisierungscodes mit PKCE

Dies ist ein Beispiel für einen Antrag auf Erteilung eines Autorisierungscodes bei PKCE.

Beispiel — GET-Anfrage

Die folgende Anfrage fügt der vorherigen Anfrage einen `code_challenge` Parameter hinzu. Um den Austausch eines Codes gegen ein Token abzuschließen, müssen Sie den `code_verifier` Parameter in Ihre Anfrage für den `/oauth2/token` Endpunkt aufnehmen.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin&
code_challenge_method=S256&
code_challenge=a1b2c3d4...
```

Beispiel — Antwort

Der Authentifizierungsserver leitet mit dem Autorisierungscode und dem Status zurück zu Ihrer Anwendung. Der Code und der Status müssen in den Parametern der Abfragezeichenfolge und nicht im Fragment zurückgegeben werden:

```
HTTP/1.1 302 Found
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111&state=abcdefg
```

Token gewähren, ohne **`openid`**-Umfang

Dies ist eine Beispielanforderung, die einen impliziten Grant generiert und JWTs direkt an die Sitzung des Benutzers zurückgibt.

Beispiel — GET-Anfrage

Die folgende Anfrage bezieht sich auf eine implizite Erteilung durch Ihren Autorisierungsserver. Das Zugriffstoken von Amazon Cognito autorisiert Self-Service-API-Operationen.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=token&
client_id=1example23456789&
```

```
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin
```

Beispiel — Antwort

Der Amazon-Cognito-Autorisierungsserver leitet das Zugriffs-Token zurück an Ihre App. Da der `openid`-Bereich nicht angefordert wurde, gibt Amazon Cognito kein ID-Token aus. Außerdem gibt Amazon Cognito in diesem Flow kein Aktualisierungs-Token aus. Amazon Cognito gibt das Zugriffstoken und den Status im Fragment und nicht in der Abfragezeichenfolge zurück:

```
HTTP/1.1 302 Found
Location: https://YOUR_APP/
redirect_uri#access_token=ACCESS_TOKEN&token_type=bearer&expires_in=3600&state=STATE
```

Token gewähren, mit **`openid`**-Umfang

Dies ist eine Beispielanforderung, die eine implizite Gewährung generiert und JWTs direkt an die Sitzung des Benutzers zurückgibt.

Beispiel — GET-Anfrage

Die folgende Anfrage bezieht sich auf eine implizite Erteilung durch Ihren Autorisierungsserver. Das Zugriffstoken von Amazon Cognito autorisiert den Zugriff auf Benutzerattribute und Self-Service-API-Operationen.

```
GET
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=token&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin+openid+profile
```

Beispiel — Antwort

Der Autorisierungsserver leitet mit Zugriffstoken und ID-Token zurück zu Ihrer App (da der `openid` Bereich enthalten war):

```
HTTP/1.1 302 Found
```

```
Location: https://
www.example.com#id_token=eyJra67890EXAMPLE&access_token=eyJra12345EXAMPLE&token_type=bearer&exp
```

Beispiele für negative Antworten

Amazon Cognito lehnt Ihre Anfrage möglicherweise ab. Negative Anfragen enthalten einen HTTP-Fehlercode und eine Beschreibung, anhand derer Sie Ihre Anforderungsparameter korrigieren können. Im Folgenden finden Sie Beispiele für negative Antworten.

- Wenn `client_id` und gültig `redirect_uri` sind, die Anforderungsparameter jedoch nicht korrekt formatiert sind, leitet der Authentifizierungsserver den Fehler an den des Clients weiter `redirect_uri` und fügt eine Fehlermeldung an einen URL-Parameter an. Im Folgenden finden Sie Beispiele für falsche Formatierungen.
  - Die Anfrage enthält keinen `response_type` Parameter.
  - Die Autorisierungsanfrage lieferte einen `code_challenge` Parameter, aber keinen `code_challenge_method` Parameter.
  - Der Wert des `code_challenge_method` Parameters ist nicht`S256`.

  Im Folgenden finden Sie die Antwort auf eine Beispielanfrage mit falscher Formatierung.

  ```
  HTTP 1.1 302 Found Location: https://client_redirect_uri?error=invalid_request
  ```

- Wenn der Client eine Anfrage `code` oder `token` eingibt`response_type`, aber keine Genehmigung für diese Anfragen hat, kehrt der Amazon Cognito Cognito-Autorisierungsserver wie folgt `unauthorized_client` zum Client-Autorisierungsserver zurück: `redirect_uri`

  ```
  HTTP 1.1 302 Found Location: https://client_redirect_uri?error=unauthorized_client
  ```

- Falls die Anforderung des Clients unbekannt, falsch formatiert oder ungültig ist, sollte der Amazon-Cognito-Autorisierungsserver `invalid_scope` folgendermaßen zur `redirect_uri` des Clients zurückgeben:

  ```
  HTTP 1.1 302 Found Location: https://client_redirect_uri?error=invalid_scope
  ```

- Wenn auf dem Server ein unerwarteter Fehler auftritt, kehrt der Authentifizierungsserver `server_error` zum Server des `redirect_uri` Clients zurück. Da der HTTP 500-Fehler nicht an den Client gesendet wird, wird der Fehler nicht im Browser des Benutzers angezeigt. Der Autorisierungsserver gibt den folgenden Fehler zurück.

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=server_error
```

- Wenn Amazon Cognito sich über einen Verbund mit einem Drittanbieter authentifiziert IdPs, kann es bei Amazon Cognito zu Verbindungsproblemen kommen, wie z. B. die folgenden:
  - Wenn es bei der Token-Anforderung vom IdP zu einem Verbindungs-Timeout kommt, leitet der Authentifizierungsserver den Fehler wie folgt an den `redirect_uri` des Clients weiter:

    ```
    HTTP 1.1 302 Found Location: https://client_redirect_uri?
    error=invalid_request&error_description=Timeout+occurred+in+calling+IdP+token
    +endpoint
    ```

  - Wenn beim Aufrufen des `jwks_uri` Endpunkts zur Überprüfung des ID-Tokens ein Verbindungs-Timeout auftritt, leitet der Authentifizierungsserver mit einem Fehler wie folgt an den des Clients weiter: `redirect_uri`

    ```
    HTTP 1.1 302 Found Location: https://client_redirect_uri?
    error=invalid_request&error_description=error_description=Timeout+in+calling+jwks
    +uri
    ```

- Bei der Authentifizierung über einen Verbund mit einem Drittanbieter IdPs geben die Anbieter möglicherweise Fehlerantworten zurück. Dies kann auf Konfigurationsfehler oder andere Gründe zurückzuführen sein, z. B. auf die folgenden:
  - Wenn eine Fehlermeldung von anderen Anbietern empfangen wird, leitet der Authentifizierungsserver den Fehler wie folgt an den `redirect_uri` des Clients weiter:

    ```
    HTTP 1.1 302 Found Location: https://client_redirect_uri?
    error=invalid_request&error_description=[IdP name]+Error+-+[status code]+error
     getting token
    ```

  - Wenn eine Fehlerantwort von Google empfangen wird, leitet der Authentifizierungsserver den Fehler wie folgt an den `redirect_uri` des Clients weiter:

    ```
    HTTP 1.1 302 Found Location: https://client_redirect_uri?
    error=invalid_request&error_description=Google+Error+-+[status code]+[Google-
    provided error code]
    ```

- Wenn Amazon Cognito bei der Verbindung zu einem externen IdP auf eine Kommunikationsausnahme stößt, leitet der Authentifizierungsserver mit einer der folgenden Meldungen `redirect_uri` mit einem Fehler an den des Kunden weiter:

- ```
  HTTP 1.1 302 Found Location: https://client_redirect_uri?
  error=invalid_request&error_description=Connection+reset
  ```
- ```
  HTTP 1.1 302 Found Location: https://client_redirect_uri?
  error=invalid_request&error_description=Read+timed+out
  ```

Token-Endpunkt

Der OAuth-2.0-Token-Endpunkt bei `/oauth2/token` gibt JSON-Webtoken (JWTs) aus.

Ihr OAuth 2.0-Autorisierungsserver für Benutzerpool gibt JSON-Webtoken (JWTs) vom Token-Endpunkt für die folgenden Sitzungstypen aus:

1. Benutzer, die eine Anfrage für die Gewährung eines Autorisierungscodes abgeschlossen haben. Wenn ein Code erfolgreich eingelöst wurde, werden ID-, Zugriffs- und Aktualisierungstoken zurückgegeben.
2. M achine-to-machine (M2M) -Sitzungen, für die eine Erteilung von Kundenanmeldedaten abgeschlossen wurde. Bei erfolgreicher Autorisierung mit dem geheimen Client-Schlüssel wird ein Zugriffstoken zurückgegeben.
3. Benutzer, die sich zuvor angemeldet und Aktualisierungstoken erhalten haben. Bei der Aktualisierungstoken-Authentifizierung werden neue ID- und Zugriffstoken zurückgegeben.

> **Note**
>
> Benutzer, die sich mit einem erteilten Autorisierungscode auf der gehosteten Benutzeroberfläche oder über einen Verbund anmelden, können ihre Token jederzeit vom Token-Endpunkt aus aktualisieren. Benutzer, die sich mit den API-Vorgängen anmelden `InitiateAuth` und ihre Token mit dem Token-Endpunkt aktualisieren `AdminInitiateAuth` können, wenn die gespeicherten Geräte in Ihrem Benutzerpool nicht aktiv sind. Wenn die gespeicherten Geräte aktiv sind, aktualisieren Sie die Tokens mit den Anfragen `AuthFlow` of `REFRESH_TOKEN_AUTH` in `InitiateAuth` oder `AdminInitiateAuth` API.

Der Token-Endpunkt wird öffentlich verfügbar, wenn Sie Ihrem Benutzerpool eine Domain hinzufügen. Er akzeptiert HTTP-POST-Anfragen. Verwenden Sie aus Gründen der Anwendungssicherheit PKCE mit Ihren Autorisierungscode-Anmeldeereignissen. PKCE überprüft, ob

der Benutzer, der einen Autorisierungscode übergibt, derselbe Benutzer ist, der sich authentifiziert hat. Weitere Informationen zu PKCE finden Sie unter IETF RFC 7636.

Weitere Informationen zu den App-Clients im Benutzerpool und ihren Erteilungstypen, geheimen Client-Schlüsseln, autorisierten Bereichen und Client-IDs finden Sie unter App-Clients für Benutzerpools. Weitere Informationen zur M2M-Autorisierung, zur Erteilung von Kundenanmeldedaten und zur Autorisierung mit Zugriffstoken-Gültigkeitsbereichen finden Sie unter. Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern

Um Informationen über einen Benutzer aus seinem Zugriffstoken abzurufen, geben Sie diese an Ihre UserInfo-Endpunkt oder eine GetUserAPI-Anfrage weiter.

POST /oauth2/token

Der `/oauth2/token` Endpunkt unterstützt ausschließlich `HTTPS POST`. Ihre Anwendung sendet Anfragen direkt an diesen Endpunkt und nicht über den Browser des Benutzers.

Der Token-Endpunkt unterstützt `client_secret_basic`- und `client_secret_post`-Authentifizierung. Weitere Informationen zur OpenID Connect-Spezifikation finden Sie unter Client Authentication. Weitere Informationen zum Token-Endpunkt aus der OpenID-Connect-Spezifikation finden Sie unter Token-Endpunkt.

Anfrageparameter im Header

`Authorization`

Falls dem Client ein Geheim-Schlüssel zugestellt wurde, kann der Client seine `client_id` und sein `client_secret` im Autorisierungs-Header als `client_secret_basic` HTTP-Autorisierung übergeben. Sie können auch die `client_id` und das `client_secret` im Anforderungstext als `client_secret_post`-Autorisierung aufnehmen.

Die Autorisierungs-Header-Stringl autet Basic `Base64Encode(client_id:client_secret)`. Das folgende Beispiel ist ein Autorisierungsheader für einen App-Client `djc98u3jiedmi283eu928` mit einem geheimen Clientschlüssel`abcdef01234567890`, der die Base64-kodierte Version der Zeichenfolge verwendet: `djc98u3jiedmi283eu928:abcdef01234567890`

```
Authorization: Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw
```

`Content-Type`

Stellen Sie den Wert dieses Parameters auf `'application/x-www-form-urlencoded'` ein.

Anfrageparameter im Fließtext

`grant_type`

(Erforderlich) Der Typ des OIDC-Zuschusses, den Sie beantragen möchten.

Es muss sich entweder um `authorization_code` oder `refresh_token` oder `client_credentials` handeln. Unter den folgenden Bedingungen können Sie vom Token-Endpunkt aus ein Zugriffstoken für einen benutzerdefinierten Bereich anfordern:

- Sie haben den angeforderten Bereich in Ihrer App-Client-Konfiguration aktiviert.
- Sie haben Ihren App-Client mit einem geheimen Clientschlüssel konfiguriert.
- Sie aktivieren die Gewährung von Kundenanmeldedaten in Ihrem App-Client.

`client_id`

(Optional) Die ID eines App-Clients in Ihrem Benutzerpool. Geben Sie denselben App-Client an, der Ihren Benutzer authentifiziert hat.

Sie müssen diesen Parameter angeben, wenn der Client öffentlich ist und kein Geheimnis hat, oder wenn er `client_secret_post` autorisiert ist. `client_secret`

`client_secret`

(Optional) Der geheime Client-Schlüssel für den App-Client, der Ihren Benutzer authentifiziert hat. Erforderlich, wenn Ihr App-Client über einen Client-Schlüssel verfügt und Sie keinen `Authorization`-Header gesendet haben.

`scope`

(Optional) Kann eine Kombination aus beliebigen benutzerdefinierten Bereichen sein, die einem App-Client zugeordnet sind. Jeder Bereich, den Sie anfordern, muss für den App-Client aktiviert sein. Wenn nicht, ignoriert Amazon Cognito es. Wenn der Client keine Bereiche anfordert, weist der Authentifizierungsserver alle benutzerdefinierten Bereiche zu, die Sie in Ihrer App-Client-Konfiguration autorisiert haben.

Wird nur verwendet, wenn `grant_type` `client_credentials`ist.

`redirect_uri`

(Optional) Muss derselbe sein`redirect_uri`, der für den Zugang verwendet wurde. `authorization_code` `/oauth2/authorize`

Wenn `grant_type` ja, müssen Sie diesen Parameter angeben`authorization_code`.

`refresh_token`

(Optional) Um neue Zugriffs- und ID-Token für die Sitzung eines Benutzers zu generieren, setzen Sie den Wert eines `refresh_token` Parameters in Ihrer `/oauth2/token` Anfrage auf ein zuvor ausgegebenes Aktualisierungstoken von demselben App-Client.

`code`

(Optional) Der Autorisierungscode aus einer Autorisierungscode-Erteilung. Sie müssen diesen Parameter angeben, wenn Ihre Autorisierungsanfrage ein `grant_type` of enthielt`authorization_code`.

`code_verifier`

(Optional) Der willkürliche Wert, den Sie zur Berechnung des `code_challenge` in einer Autorisierungscode-Erteilungsanforderung mit PKCE verwendeten.

Beispielanfragen mit positiven Antworten

Austausch eines Autorisierungscodes für Token

Beispiel — POST-Anfrage

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token&
                            Content-Type='application/x-www-form-urlencoded'&

 Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

                            grant_type=authorization_code&
                            client_id=1example23456789&
                            code=AUTHORIZATION_CODE&
                            redirect_uri=com.myclientapp://myclient/redirect
```

Beispiel — Antwort

```
HTTP/1.1 200 OK
```

```
                            Content-Type: application/json

                            {
                             "access_token":"eyJra1example",
                             "id_token":"eyJra2example",
                             "refresh_token":"eyJj3example",
                             "token_type":"Bearer",
                             "expires_in":3600
                            }
```

> **Note**
>
> Der Token-Endpunkt wird `refresh_token` nur zurückgegeben, wenn der `grant_type` `authorization_code` ist.

Austauschen von Client-Anmeldedaten für ein Zugriffs-Token: Client-Schlüssel im Authentifizierungs-Header

Beispiel — POST-Anfrage

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token >
                            Content-Type='application/x-www-form-urlencoded'&

 Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

                            grant_type=client_credentials&
                            client_id=1example23456789&

 scope=resourceServerIdentifier1/scope1 resourceServerIdentifier2/scope2
```

Beispiel — Antwort

```
HTTP/1.1 200 OK
                            Content-Type: application/json

                            {
                            "access_token":"eyJra1example",
                            "token_type":"Bearer",
                            "expires_in":3600
                            }
```

Austauschen von Client-Anmeldedaten für ein Zugriffs-Token: Client-Schlüssel im Anforderungstext

Beispiel — POST-Anfrage

```
POST /oauth2/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
X-Amz-Target: AWSCognitoIdentityProviderService.Client credentials request
User-Agent: USER_AGENT
Accept: /
Accept-Encoding: gzip, deflate, br
Content-Length: 177
Referer: http://auth.example.com/oauth2/token
Host: auth.example.com
Connection: keep-alive

grant_type=client_credentials&client_id=1example23456789&scope=my_resource_server_identifier%2F
```

Beispiel — Antwort

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Date: Tue, 05 Dec 2023 16:11:11 GMT
x-amz-cognito-request-id: 829f4fe2-a1ee-476e-b834-5cd85c03373b

{
"access_token": "eyJra12345EXAMPLE",
"expires_in": 3600,
"token_type": "Bearer"
}
```

Austausch einer Autorisierungscode-Erteilung mit PKCE für Token

Beispiel — POST-Anfrage

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token
                                Content-Type='application/x-www-form-urlencoded'&

 Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

                                grant_type=authorization_code&
                                client_id=1example23456789&
```

```
                                code=AUTHORIZATION_CODE&
                                code_verifier=CODE_VERIFIER&
                                redirect_uri=com.myclientapp://myclient/redirect
```

Beispiel — Antwort

```
HTTP/1.1 200 OK
                               Content-Type: application/json

                               {
                               "access_token":"eyJra1example",
                               "id_token":"eyJra2example",
                               "refresh_token":"eyJj3example",
                               "token_type":"Bearer",
                               "expires_in":3600
                               }
```

> **Note**
>
> Der Token-Endpunkt wird `refresh_token` nur zurückgegeben, wenn der `grant_type` `authorization_code` ist.

Austausch eines Refresh-Tokens für Token

Beispiel — POST-Anfrage

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token >
                              Content-Type='application/x-www-form-urlencoded'&

 Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

                              grant_type=refresh_token&
                              client_id=1example23456789&
                              refresh_token=eyJj3example
```

Beispiel — Antwort

```
HTTP/1.1 200 OK
```

```
                            Content-Type: application/json

                            {
                            "access_token":"eyJra1example",
                            "id_token":"eyJra2example",
                            "token_type":"Bearer",
                            "expires_in":3600
                                  }
```

> **Note**
>
> Der Token-Endpunkt wird `refresh_token` nur zurückgegeben, wenn der `grant_type` `authorization_code` ist.

Beispiele für negative Antworten

Beispiel — Fehlerantwort

```
HTTP/1.1 400 Bad Request
                            Content-Type: application/json;charset=UTF-8

                            {
                            "error":"invalid_request|invalid_client|invalid_grant|
unauthorized_client|unsupported_grant_type"
                            }
```

`invalid_request`

Der Anforderung fehlt ein erforderlicher Parameter, sie umfasst einen nicht unterstützten Parameter-Wert (außer `unsupported_grant_type`) oder sie ist aus anderen Gründen ungültig. Beispielsweise, `grant_type` ist `refresh_token` aber `refresh_token` ist nicht enthalten.

`invalid_client`

Client-Authentifizierung ist fehlgeschlagen. Wenn der Client zum Beispiel `client_id` und `client_secret` im Autorisierungs-Header enthält, aber kein Client mit dieser `client_id` und diesem `client_secret`existiert.

`invalid_grant`

Das Refresh-Token wurde widerrufen.

Der Autorisierungs-Code wurde bereits verwendet oder ist nicht vorhanden.

Der App-Client hat keinen Lesezugriff auf alle Attribute im angeforderten Bereich. Zum Beispiel fordert Ihre App den `email`-Bereich an und Ihr App-Client kann das Attribut `email`, aber nicht `email_verified` lesen.

`unauthorized_client`

Dem Client ist es nicht gestattet, einen Code zu gewähren oder Token zu aktualisieren.

`unsupported_grant_type`

Wird zurückgegeben, wenn `grant_type` ein anderer Wert ist als `authorization_code` oder `refresh_token` oder `client_credentials`.

UserInfo-Endpunkt

Der Endpunkt `userInfo` ist ein OIDC-UserInfo-Endpunkt (OpenID-Connect). Er antwortet mit Benutzerattributen, wenn Serviceanbieter von Ihrem Token-Endpunkt ausgegebene Zugriffstoken vorlegen. Die Bereiche im Zugriffstoken Ihres Benutzers definieren die Benutzerattribute, die der UserInfo-Endpunkt in seiner Antwort zurückgibt. Der `openid`-Gültigkeitsbereich muss einer der Zugriffstokenansprüche sein.

Amazon Cognito gibt Zugriffs-Token als Antwort auf Benutzerpool-API-Anfragen wie InitiateAuth aus. Da sie keine Bereiche enthalten, akzeptiert der userInfo-Endpunkt diese Zugriffstoken nicht. Stattdessen müssen Sie Zugriffstoken von Ihrem Token-Endpunkt vorlegen.

Ihr externer OAuth-2.0-Identitätsanbieter (IDP) hostet ebenfalls einen userInfo-Endpunkt. Wenn sich Ihr Benutzer bei diesem IdP authentifiziert, tauscht Amazon Cognito im Hintergrund einen Autorisierungscode mit dem IdP-Endpunkt aus. `token` Ihr Benutzerpool übergibt das IdP-Zugriffstoken, um den Abruf von Benutzerinformationen vom IdP-Endpunkt zu autorisieren. `userInfo`

GET /oauth2/userInfo

Ihre Anwendung sendet Anfragen direkt an diesen Endpunkt und nicht über einen Browser.

Weitere Informationen finden Sie unter UserInfo-Endpunkt in der Spezifikation zu OpenID Connect (OIDC).

Themen

- Anfrageparameter im Header

- Beispiel — Anfrage
- Beispiel — positive Antwort
- Beispiel für negative Antworten

Anfrageparameter im Header

`Authorization: Bearer <access_token>`

Übergeben Sie das Zugriffstoken im Autorisierungsheader-Feld.

Erforderlich

Beispiel — Anfrage

```
GET /oauth2/userInfo HTTP/1.1
Content-Type: application/x-amz-json-1.1
Authorization: Bearer eyJra12345EXAMPLE
User-Agent: [User agent]
Accept: */*
Host: auth.example.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

Beispiel — positive Antwort

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: [Integer]
Date: [Timestamp]
x-amz-cognito-request-id: [UUID]
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Frame-Options: DENY
Server: Server
Connection: keep-alive
{
```

```
    "sub": "[UUID]",
    "email_verified": "true",
    "custom:mycustom1": "CustomValue",
    "phone_number_verified": "true",
    "phone_number": "+12065551212",
    "email": "bob@example.com",
    "username": "bob"
}
```

Eine Liste der OIDC-Ansprüche finden Sie unter Standardansprüche. Derzeit gibt Amazon Cognito die Werte für `email_verified` und `phone_number_verified` als Zeichenfolgen zurück.

Beispiel für negative Antworten

Beispiel — schlechte Anfrage

```
HTTP/1.1 400 Bad Request
WWW-Authenticate: error="invalid_request",
error_description="Bad OAuth2 request at UserInfo Endpoint"
```

`invalid_request`

In der Anfrage fehlt ein erforderlicher Parameter, sie enthält einen nicht unterstützten Parameterwert oder sie ist anderweitig falsch formatiert.

Beispiel — schlechtes Token

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: error="invalid_token",
error_description="Access token is expired, disabled, or deleted, or the user has
 globally signed out."
```

`invalid_token`

Das Zugriffstoken ist abgelaufen, gesperrt, falsch formatiert oder ungültig.

Widerrufen des Endpunkts

Der `oauth2/revoke` Endpunkt/widerruft das Zugriffstoken eines Benutzers, das Amazon Cognito ursprünglich mit dem von Ihnen bereitgestellten Aktualisierungstoken ausgestellt hat.

Dieser Endpunkt widerruft auch alle nachfolgenden Zugriffs- und Identitätstoken desselben Aktualisierungstoken. Nachdem der Endpunkt die Token widerrufen hat, können Sie die widerrufenen Zugriffs-Token nicht verwenden, um auf APIs zuzugreifen, die Amazon-Cognito-Token authentifizieren.

POST /oauth2/revoke

Der `/oauth2/revoke` Endpunkt unterstützt ausschließlich `HTTPS POST`. Der Benutzer-Pool sendet Anforderungen direkt an diesen Endpunkt und nicht über den System-Browser.

Anfrageparameter im Header

`Authorization`

Wenn Ihr App-Client über einen geheimen Clientschlüssel verfügt, muss die Anwendung dessen `client_id` und `client_secret` im Autorisierungsheader über die Basic-HTTP-Autorisierung weitergeben. Der Geheim-Schlüssel ist Basic `Base64Encode(client_id:client_secret)`.

`Content-Type`

Es muss sich immer um handeln `'application/x-www-form-urlencoded'`.

Anfrageparameter im Fließtext

`token`

(Erforderlich) Das Aktualisierungstoken, das der Client widerrufen möchte. Die Anforderung widerruft auch alle Zugriffstoken, die Amazon Cognito mit diesem Aktualisierungstoken ausgegeben hat.

Erforderlich

`client_id`

(Optional) Die App-Client-ID für das Token, das Sie widerrufen möchten.

Erforderlich, wenn der Client öffentlich ist und keinen Geheim-Schlüssel hat.

Beispiel für einen Widerrufsanforderung

Beispiel 1: Widerrufen eines Token für einen App-Client ohne Client-Schlüssel

```
POST /oauth2/revoke HTTP/1.1
    Host: https://mydomain.auth.us-east-1.amazoncognito.com
    Accept: application/json
    Content-Type: application/x-www-form-urlencoded
    token=2YotnFZFEjr1zCsicMWpAA&
    client_id=djc98u3jiedmi283eu928
```

Beispiel 2: Widerrufen eines Token für einen App-Client mit Client-Schlüssel

```
POST /oauth2/revoke HTTP/1.1
    Host: https://mydomain.auth.us-east-1.amazoncognito.com
    Accept: application/json
    Content-Type: application/x-www-form-urlencoded
    Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
    token=2YotnFZFEjr1zCsicMWpAA
```

Antwort auf einen Fehler

Eine erfolgreiche Antwort enthält einen leeren Körper. Die Fehlerantwort ist ein JSON-Objekt mit einem `error`-Feld und in manchen Fällen ein `error_description`-Feld.

Endpunktfehler

- Ihnen werden HTTP 400 und der Fehler `invalid_request` angezeigt, wenn das Token in der Anforderung nicht vorhanden ist oder wenn die Funktion für den App-Client deaktiviert ist.
- Wenn das Token, das Amazon Cognito in der Widerrufsanfrage gesendet hat, kein Aktualisierungstoken ist, erhalten Sie ein HTTP 400 und den Fehler `unsupported_token_type`.
- Wenn die Client-Anmeldeinformationen nicht gültig sind, wird Ihnen HTTP 401 und der Fehler `invalid_client` angezeigt.
- Wenn das Token widerrufen wurde oder der Client ein ungültiges Token übermittelt hat, erhalten Sie ein HTTP 200-OK.

saml2/idpresponse-Endpunkt

Der `/saml2/idpresponse` empfängt SAML-Assertionen. Bei der service-provider-initiated (SP-initiierten) Anmeldung leitet Ihr SAML 2.0-Identitätsanbieter (IdP) Ihren Benutzer mit seiner SAML-Antwort an diesen Endpunkt weiter. Bei der SP-initiierten Anmeldung interagiert Ihre Anwendung

nicht mit diesem Endpunkt. Konfigurieren Sie Ihren IdP mit dem Pfad zu Ihrer URL `saml2/idpresponse` als Assertion Consumer Service (ACS). Weitere Informationen zur Sitzungsinitiierung finden Sie unter. Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools

Bei der vom IdP initiierten Anmeldung können sich Ihre Benutzer über Ihren eigenen Prozess bei Ihrem IdP anmelden und eine SAML-Assertion im Hauptteil einer Anfrage über HTTPS einreichen. `HTTP POST` Der Hauptteil Ihrer `POST` Anfrage muss aus einem Parameter und einem Parameter bestehen. `SAMLResponse` `Relaystate` Weitere Informationen finden Sie unter Verwenden der IDP-initiierten SAML-Anmeldung.

BEITRAG **`/saml2/idpresponse`**

Um den `/saml2/idpresponse` Endpunkt bei einer vom IDP initiierten Anmeldung zu verwenden, generieren Sie eine POST-Anforderung mit Parametern, die Ihrem Benutzerpool Informationen über die Sitzung Ihres Benutzers liefern.

- Der App-Client, bei dem sie sich anmelden möchten.
- Die Callback-URL, unter der sie landen möchten.
- Die OAuth 2.0-Bereiche, die sie im Zugriffstoken Ihres Benutzers anfordern möchten.
- Der IdP, der die Anmeldeanforderung initiiert hat.

Vom IDP initiierte Anfragetextparameter

SAML-Antwort

Eine Base64-kodierte SAML-Assertion von einem IdP, der einem gültigen App-Client und einer IdP-Konfiguration in Ihrem Benutzerpool zugeordnet ist.

RelayState

Ein `RelayState` Parameter enthält die Anforderungsparameter, die Sie andernfalls an den Endpunkt übergeben würden. `oauth2/authorize` Ausführliche Informationen zu diesen Parametern finden Sie unterAutorisieren des Endpunkts.

response_type

Der OAuth 2.0-Grant-Typ.

Client-ID

Die App-Client-ID).

redirect_uri

Die URL, an die der Authentifizierungsserver den Browser nach der Autorisierung des Benutzers durch Amazon Cognito weiterleitet.

Identitätsanbieter

Der Name des Identitätsanbieters, zu dem Sie Ihren Benutzer umleiten möchten.

Idp-Identifier

Die Kennung des Identitätsanbieters, zu dem Sie Ihren Benutzer weiterleiten möchten.

scope

Die OAuth 2.0-Bereiche, die Ihr Benutzer vom Autorisierungsserver anfordern soll.

Beispielanfragen mit positiven Antworten

Beispiel — POST-Anfrage

Die folgende Anfrage bezieht sich auf die Gewährung eines Autorisierungscodes für einen Benutzer von IdP `MySAMLIdP` im App-Client`1example23456789`. Der Benutzer leitet `https://www.example.com` mit seinem Autorisierungscode weiter, der gegen Token eingetauscht werden kann, die ein Zugriffstoken mit den OAuth 2.0-Bereichen `openid` enthalten, und. `email` `phone`

```
POST /saml2/idpresponse HTTP/1.1
User-Agent: USER_AGENT
Accept: */*
Host: example.auth.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded

SAMLResponse=[Base64-encoded SAML assertion]&RelayState=identity_provider
%3DMySAMLIdP%26client_id%3D1example23456789%26redirect_uri%3Dhttps%3A%2F
%2Fwww.example.com%26response_type%3Dcode%26scope%3Demail%2Bopenid%2Bphone
```

Beispiel — Antwort

Das Folgende ist die Antwort auf die vorherige Anfrage.

```
HTTP/1.1 302 Found
Date: Wed, 06 Dec 2023 00:15:29 GMT
```

```
Content-Length: 0
x-amz-cognito-request-id: 8aba6eb5-fb54-4bc6-9368-c3878434f0fb
Location: https://www.example.com?code=[Authorization code]
```

OAuth-2.0-Erteilungen

Der OAuth-2.0-Autorisierungsserver für den Amazon-Cognito-Benutzerpool gibt Token als Antwort auf drei Arten von OAuth-2.0-Autorisierungserteilungen aus. Sie können die unterstützten Erteilungstypen für jeden App-Client in Ihrem Benutzerpool festlegen. Sie können Erteilungen von Client-Anmeldeinformationen im selben App-Client nicht als implizite oder Autorisierungscode-Erteilungen aktivieren. Anforderungen für implizite und Autorisierungscode-Erteilungen beginnen bei Ihrem Autorisieren des Endpunkts und Anforderungen für Erteilungen von Client-Anmeldeinformationen bei Ihrem Token-Endpunkt.

Erteilung des Autorisierungscodes

Als Antwort auf Ihre erfolgreiche Authentifizierungsanforderung fügt der Autorisierungsserver Ihrer Callback-URL einen Autorisierungscode in einem `code`-Parameter an. Sie müssen dann den Code für ID-, Zugriffs- und Aktualisierungstoken durch den Token-Endpunkt ersetzen. Wenn Sie die Erteilung eines Autorisierungscodes anfordern möchten, legen Sie `response_type` in Ihrer Anforderung auf `code` fest. Eine Beispielanforderung finden Sie unter Erteilung des Autorisierungscodes.

Die Erteilung des Autorisierungscodes ist die sicherste Form der Autorisierungserteilung. Dabei werden Ihren Benutzern die Inhalte der Token nicht direkt angezeigt. Stattdessen ist Ihre App dafür zuständig, die Token Ihrer Benutzer abzurufen und sicher zu speichern. In Amazon Cognito ist die Erteilung eines Autorisierungscodes die einzige Möglichkeit, alle drei Tokentypen – ID-, Zugriffs- und Aktualisierungstoken – vom Autorisierungsserver abzurufen. Sie können alle drei Tokentypen auch durch Authentifizierung über die API von Amazon-Cognito-Benutzerpools abrufen. Die API gibt jedoch keine Zugriffstoken mit anderen Bereichen als `aws.cognito.signin.user.admin` aus.

Implizite Erteilung

Als Antwort auf Ihre erfolgreiche Authentifizierungsanforderung fügt der Autorisierungsserver Ihrer Callback-URL ein Zugriffstoken in einem `access_token`-Parameter und ein ID-Token in einem `id_token`-Parameter an. Eine implizite Erteilung erfordert keine zusätzliche Interaktion mit dem Token-Endpunkt. Wenn Sie eine implizite Erteilung anfordern möchten, legen Sie `response_type` in Ihrer Anforderung auf `token` fest. Die implizite Erteilung generiert nur eine ID

und ein Zugriffstoken. Eine Beispielanforderung finden Sie unter Token gewähren, ohne `openid`-Umfang.

Bei der impliziten Erteilung handelt es sich um eine ältere Autorisierungserteilung. Im Gegensatz zur Autorisierungscode-Erteilung können Benutzer Ihre Token abfangen und überprüfen. Wenn Sie die Tokenzustellung durch implizite Erteilung verhindern möchten, konfigurieren Sie Ihren App-Client so, dass er nur die Erteilung von Autorisierungscodes unterstützt.

Client-Anmeldeinformationen

Bei den Kundenanmeldedaten handelt es sich nur um eine Autorisierung für den Zugriff. machine-to-machine Wenn Sie eine Erteilung von Client-Anmeldeinformationen erhalten möchten, umgehen Sie den Autorisieren des Endpunkts und generieren Sie eine Anforderung direkt an den Token-Endpunkt. Ihr App-Client muss über ein Client-Geheimnis verfügen und darf nur Erteilungen von Client-Anmeldeinformationen unterstützen. Als Antwort auf Ihre erfolgreiche Anforderung gibt der Autorisierungsserver ein Zugriffstoken zurück.

Das Zugriffstoken aus der Erteilung von Client-Anmeldeinformationen ist ein Autorisierungsmechanismus, der OAuth-2.0-Bereiche enthält. In der Regel enthält das Token benutzerdefinierte Bereichsansprüche, die HTTP-Operationen für zugriffsgeschützte APIs autorisieren. Weitere Informationen finden Sie unter Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern.

Bei Zuschüssen mit Kundendaten fallen zusätzliche Kosten auf Ihre Rechnung an. AWS Weitere Informationen finden Sie unter Amazon Cognito – Preise.

Verwendung von PKCE in Autorisierungscode-Erteilungen mit Amazon Cognito Cognito-Benutzerpools

Amazon Cognito unterstützt die Proof Key for Code Exchange (PKCE) -Authentifizierung bei der Gewährung von Autorisierungscodes. PKCE ist eine Erweiterung der OAuth 2.0-Autorisierungscode-Gewährung für öffentliche Kunden. PKCE schützt vor der Rücknahme abgefangener Autorisierungscodes.

So verwendet Amazon Cognito PKCE

Um die Authentifizierung mit PKCE zu starten, muss Ihre Anwendung einen eindeutigen Zeichenkettenwert generieren. Diese Zeichenfolge ist der Code Verifier, ein geheimer Wert, den

Amazon Cognito verwendet, um den Client, der die ursprüngliche Autorisierung beantragt, mit dem Client zu vergleichen, der den Autorisierungscode gegen Token tauscht.

Ihre App muss einen SHA256-Hash auf die Codeverifizierer-Zeichenfolge anwenden und das Ergebnis mit Base64 codieren. Übergeben Sie die Hash-Zeichenfolge Autorisieren des Endpunkts als `code_challenge` Parameter im Anforderungstext an. Wenn Ihre App den Autorisierungscode gegen Token eintauscht, muss sie die Codeverifizierer-Zeichenfolge im Klartext als `code_verifier` Parameter im Anfragetext an die enthalten. Token-Endpunkt Amazon Cognito führt den gleichen hash-and-encode Vorgang mit dem Code-Verifier durch. Amazon Cognito gibt ID-, Zugriffs- und Aktualisierungstoken nur zurück, wenn es feststellt, dass der Codeverifier zu derselben Codeabfrage führt, die er in der Autorisierungsanfrage erhalten hat.

Um Authorization Grant Flow mit PKCE zu implementieren

1. Öffnen Sie die Amazon-Cognito-Konsole. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool. Wenn Sie einen Benutzerpool erstellen, werden Sie während des Assistenten aufgefordert, einen App-Client einzurichten und die gehostete Benutzeroberfläche zu konfigurieren.

   a. Wenn Sie einen neuen Benutzerpool erstellen, richten Sie einen App-Client ein und konfigurieren Sie die gehostete Benutzeroberfläche während der geführten Einrichtung.

   b. Wenn Sie einen vorhandenen Benutzerpool konfigurieren, fügen Sie eine Domäne und einen öffentlichen App-Client hinzu, falls Sie dies noch nicht getan haben.
4. Generieren Sie eine zufällige alphanumerische Zeichenfolge, in der Regel einen Universally Unique Identifier (UUID), um eine Code-Challenge für die PKCE zu erstellen. Diese Zeichenfolge ist der Wert des `code_verifier` Parameters, den Sie in Ihrer Anfrage an die senden. Token-Endpunkt
5. Hasht die `code_verifier` Zeichenfolge mit dem SHA256-Algorithmus. Kodieren Sie das Ergebnis des Hashing-Vorgangs auf Base64. Diese Zeichenfolge ist der Wert des `code_challenge` Parameters, den Sie in Ihrer Anfrage an die senden. Autorisieren des Endpunkts

   Das folgende Python Beispiel generiert einen `code_verifier` und berechnet dencode_challenge:

```
#!/usr/bin/env python3

import random
from base64 import urlsafe_b64encode
from hashlib import sha256
from string import ascii_letters
from string import digits

# use a cryptographically strong random number generator source
rand = random.SystemRandom()

code_verifier = ''.join(rand.choices(ascii_letters + digits, k=128))
code_verifier_hash = sha256(code_verifier.encode()).digest()
code_challenge = urlsafe_b64encode(code_verifier_hash).decode().rstrip('=')

print(f"code challenge: {code_challenge}")
print(f"code verifier: {code_verifier}")
```

Im Folgenden finden Sie ein Beispiel für die Ausgabe des Python Skripts:

```
code challenge: Eh0mg-OZv7BAyo-tdv_vYamx1boOYDulDklyXoMDtLg
code verifier: 9D-aW_iygXrgQcWJd0y0tNVMPSXSChIc2xceDhvYVdGLCBk-
JWFTmBNjvKSdOrjTTYazOFbUmrFERrjWx6oKtK2b6z_x4_gHBDlr4K1mRFGyE8yA-05-_v7Dxf3EIYJH
```

6. Schließen Sie die Anmeldung über die gehostete Benutzeroberfläche mit einer Anfrage zur Gewährung eines Autorisierungscodes bei PKCE ab. Im Folgenden finden Sie ein Beispiel für eine URL:

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com&code_challenge=Eh0mg-OZv7BAyo-
tdv_vYamx1boOYDulDklyXoMDtLg&code_challenge_method=S256
```

7. Sammeln Sie die Autorisierung `code` und lösen Sie sie gegen Token mit dem Token-Endpunkt ein. Im Folgenden finden Sie ein Beispiel für eine Anfrage:

```
POST /oauth2/token HTTP/1.1
Host: mydomain.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 296
```

```
redirect_uri=https%3A%2F%2Fwww.example.com&
client_id=1example23456789&
code=7378f445-c87f-400c-855e-0297d072ff03&
grant_type=authorization_code&
code_verifier=9D-aW_iygXrgQcWJd0y0tNVMPSXSChIc2xceDhvYVdGLCBk-
JWFTmBNjvKSdOrjTTYazOFbUmrFERrjWx6oKtK2b6z_x4_gHBDlr4K1mRFGyE8yA-05-_v7Dxf3EIYJH
```

8. Überprüfen Sie die Antwort. Sie wird ID-, Zugriffs- und Aktualisierungstoken enthalten. Weitere Informationen zur Verwendung von Amazon Cognito Cognito-Benutzerpool-Token finden Sie unterVerwenden von Token mit Benutzerpools.

Antworten auf gehostete UI- und Verbundfehler

Ein Anmeldevorgang in der gehosteten Benutzeroberfläche oder eine Verbundanmeldung kann zu einem Fehler führen. Im Folgenden sind einige Bedingungen aufgeführt, die dazu führen können, dass die Authentifizierung mit einem Fehler endet.

- Ein Benutzer führt einen Vorgang aus, den Ihr Benutzerpool nicht ausführen kann.
- Ein Lambda-Trigger reagiert nicht mit der erwarteten Syntax.
- Ihr Identitätsanbieter (IdP) gibt einen Fehler zurück.
- Amazon Cognito konnte die von Ihrem Benutzer bereitgestellten Attributinformationen nicht überprüfen.
- Ihr IdP hat keine Anforderungen gesendet, die den erforderlichen Attributen zugeordnet sind.

Wenn Amazon Cognito auf einen Fehler stößt, teilt es ihn auf eine der folgenden Arten mit.

1. Amazon Cognito sendet eine Weiterleitungs-URL mit dem Fehler in den Anforderungsparametern.
2. Amazon Cognito zeigt einen Fehler in der gehosteten UI an.

Fehler, die Amazon Cognito an Anforderungsparameter anfügt, haben das folgende Format.

```
https://<Callback URL>/?error_description=error+description&error=error+name
```

Wenn Sie Ihren Benutzern helfen, Fehlerinformationen zu übermitteln, obwohl sie einen Vorgang nicht ausführen können, fordern Sie sie auf, die URL und den Text oder einen Screenshot der Seite zu erfassen.

> **Note**
>
> Amazon Cognito-Fehlerbeschreibungen sind keine festen Zeichenketten, und Sie sollten keine Logik verwenden, die auf einem festen Muster oder Format basiert.

Fehlermeldungen von OIDC- und Social-Identity-Anbietern

Ihr Identitätsanbieter gibt möglicherweise einen Fehler zurück. Wenn ein OIDC- oder OAuth-2.0-IdP einen mit Standards konformen Fehler zurückgibt, leitet Amazon Cognito Ihren Benutzer zur Rückruf-URL weiter und fügt die Fehlerantwort des Anbieters zu den Fehleranforderungsparametern hinzu. Amazon Cognito fügt den Namen des Anbieters und den HTTP-Fehlercode zu den vorhandenen Fehlerzeichenfolgen hinzu.

Die folgende URL ist ein Beispiel für eine Weiterleitung von einem IdP, der einen Fehler an Amazon Cognito zurückgegeben hat.

```
https://www.amazon.com/?error_description=LoginWithAmazon+Error+-+400+invalid_request
+The+request+is+missing+a+required+parameter+%3A+client_secret&error=invalid_request
```

Da Amazon Cognito nur das zurückgibt, was es von einem Anbieter erhält, kann es sein, dass Ihr Benutzer nur einen Teil dieser Informationen sieht.

Wenn Ihr Benutzer bei der ersten Anmeldung über Ihren IdP auf ein Problem stößt, übermittelt der IdP alle Fehlermeldungen direkt an Ihren Benutzer. Amazon Cognito leitet eine Fehlermeldung an Ihren Benutzer weiter, wenn es eine Anfrage an Ihren IdP generiert, um die Sitzung Ihres Benutzers zu validieren. Amazon Cognito leitet OAuth- und OIDC-IdP-Fehlermeldungen von den folgenden Endpunkten weiter.

`/token`

Amazon Cognito tauscht den IdP-Autorisierungscode gegen ein Zugriffs-Token aus.

`/.well-known/openid-configuration`

Amazon Cognito ermittelt den Pfad zu Ihren Emittenten-Endpunkten.

`/.well-known/jwks.json`

Um die JSON Web Tokens (JWTs) Ihres Benutzers zu überprüfen, ermittelt Amazon Cognito die JSON Web Keys (JWKs), die Ihr IdP zum Signieren von Token verwendet.

Da Amazon Cognito keine ausgehenden Sitzungen an SAML-2.0-Anbieter initiiert, die möglicherweise HTTP-Fehler zurückgeben, enthalten die Fehler Ihrer Benutzer während einer Sitzung mit einem SAML-2.0-IdP diese Form von Anbieter-Fehlermeldungen nicht.

API-Referenz der Amazon-Cognito-Benutzerpools

Mithilfe von Amazon-Cognito-Benutzerpools können Sie Benutzer bei Ihrer Web- und Mobilanwendung anmelden und registrieren. Sie können Passwörter für authentifizierte Benutzer ändern und Abläufe für vergessene Passwörter für nicht authentifizierte Benutzer starten. Weitere Informationen erhalten Sie unter Ablauf der Authentifizierung in Benutzerpools und Verwenden von Token mit Benutzerpools.

Die Amazon-Cognito-Benutzerpool-API umfasst Operationen zum Anzeigen und Ändern Ihrer Benutzerpools und Benutzer sowie zur Benutzerauthentifizierung und -autorisierung. Eine Beschreibung der Klassen von API-Operationen, die in der Amazon-Cognito-Benutzerpool-API vereint sind, finden Sie unter Verwendung der Amazon-Cognito-Benutzerpool-API und der Benutzerpool-Endpunkte.

Eine detaillierte Liste der API-Operationen und -Syntax der Amazon-Cognito-Benutzerpools finden Sie in der API-Referenz der Amazon-Cognito-Benutzerpools. Jede Seite der API-Referenz der Amazon-Cognito-Benutzerpools enthält Links zu Referenzmaterial mit Angaben zur Syntax und Beispielen für eine Vielzahl von AWS-SDKs.

API-Referenz des Amazon-Cognito-Identitätspools (Verbundidentitäten)

Mit einem Amazon-Cognito-Identitäten-Pool können Ihre Web- und mobilen Anwendungs-Nutzer temporäre und begrenzte Berechtigungen erhalten AWS-Anmeldeinformationen, die ihnen den Zugriff auf andere AWS-Services ermöglichen.

Eine vollständige API-Referenz für Identitätspools (Verbundidentitäten) finden Sie in der Amazon-Cognito-API-Referenz.

Amazon-Cognito-Sync-API-Referenz

Amazon Cognito Sync ist ein AWS-Service und eine Client-Bibliothek, mit der die geräteübergreifende Synchronisierung von anwendungsbezogenen Benutzerdaten möglich ist.

Weitere Informationen zur Amazon-Cognito-Sync-API-Referenz finden Sie unter Amazon-Cognito-Sync-API-Referenz.

Dokumentverlauf für Amazon Cognito

In der folgenden Tabelle sind wichtige Ergänzungen zur Dokumentation zu Amazon Cognito enthalten. Darüber hinaus nehmen wir regelmäßig kleinere Änderungen an der Dokumentation vor, um auf das Feedback einzugehen, das Sie uns senden. Feedback können Sie über den Feedback-Link unten auf jeder Seite in der Dokumentation zu Amazon Cognito übermitteln.

Änderung	Beschreibung	Datum
Unterstützung für komplexe Objekte im Lambda-Trigger vor dem Token hinzugefügt	Sie können jetzt Arrays und JSON-Objekte zu ID- und Zugriffstoken-Ansprüchen hinzufügen.	30. Mai 2024
Die Informationen zu Verified Permissions und Amazon Cognito wurden aktualisiert.	Amazon Verified Permissions ist jetzt direkter mit Amazon Cognito integriert.	15. Mai 2024
Von Amazon SES verifizierte Identitäten für mehrere Regionen.	In einigen Fällen AWS-Regionen ohne Amazon SES bündelt Amazon Cognito Cognito-Benutzer den Lastenausgleich von E-Mails zwischen zwei entfernten Regionen.	10. Mai 2024
Es wurden Informationen zur M2M-Autorisierung und zur Kostenverwaltung hinzugefügt.	Erfahren Sie, wie Sie die Gewährung von Kundenanmeldedaten für machine-to-machine (M2M) -Anwendungsfälle mit Amazon Cognito Cognito-Benutzerpools verwenden können.	9. Mai 2024
Amazon Cognito ist jetzt in Europa (Spanien) und im asiatisch-pazifischen Raum	Sie können jetzt Amazon Cognito Cognito-Ressourcen in den Regionen Europa	15. April 2024

(Hyderabad) verfügbar. AWS-Regionen	(Spanien) und Asien-Pazifik (Hyderabad) erstellen.	
Amazon Cognito ist jetzt im asiatisch-pazifischen Raum (Melbourne) AWS-Region verfügbar.	Sie können jetzt Amazon Cognito Cognito-Ressourcen in der Region Asien-Pazifik (Melbourne) erstellen.	4. April 2024
Eine Android-Beispiel-App in Flutter für Amazon Cognito Cognito-Benutzerpools wurde hinzugefügt.	Sie können aus einer Flutter-Beispielanwendung eine mobile Starter-App für Amazon Cognito erstellen. GitHub	4. April 2024
Neue Inhalte für den Einstieg	Erweiterter Inhalt für die ersten Schritte, allgemeine Szenarien , bewährte Methoden für mehrere Mandanten und den Zugriff auf Ressourcen nach der Anmeldung.	1. April 2024
Amazon Cognito ist jetzt in Europa (Zürich) AWS-Region erhältlich.	Sie können jetzt Amazon Cognito Cognito-Ressourcen in der Region Europa (Zürich) erstellen.	14. März 2024
Amazon Cognito ist jetzt im Nahen Osten (VAE) AWS-Region verfügbar.	Sie können jetzt Amazon Cognito Cognito-Ressourcen in der Region Naher Osten (VAE) erstellen.	8. März 2024
Neue SAML-Funktionen und verbesserter Inhalt.	Sie können jetzt SAML-Anfr agen signieren, SAML-Antw orten verschlüsseln und IDP-initiiertes SAML-SSO einrichte n.	1. Februar 2024

Quotenerhöhungen verfügbar.	Sie können jetzt zusätzliche Kapazität für Amazon Cognito Cognito-Kontingente mit Anforderungsrate erwerben.	25. Januar 2024
Amazon Cognito Cognito-Identitätspools unterstützen Anforderungsraten in Form von Service Quotas.	Sie können jetzt requests-per-second (RPS) -Kontingente für Amazon Cognito Cognito-Identitätspools überwachen und eine Erhöhung in der Service Quotas Quotas-Konsole beantragen.	19. Dezember 2023
Es wurde eine neue Funktion zur Anpassung des Inhalts von Zugriffstoken hinzugefügt.	Ab sofort können Sie Ansprüche und Bereiche in Zugriffs-Token für Benutzerpools hinzufügen, ändern und entfernen.	12. Dezember 2023
Verbesserter Inhalt zu App-Clients und OAuth-Bereichen.	Clarity-Änderungen und -Korrekturen an App-Clients für Benutzerpools und Geltungsbereiche, M2M und API-Autorisierung mit Ressourcenservern. Entfernung älterer Konsolenanweisungen.	14. November 2023
Verbesserte Inhalte zu Geräten und Geräteauthentifizierung.	Neue Inhalte zur Verwendung von Geräteschlüsseln und zur Geräte-SRP-Authentifizierung.	18. Oktober 2023

Aktualisierte AWS Management Console Anleitung.	Die Referenz zur Benutzerpool-Konsole wurde entfernt und die Themen wurden auf verwandte Themen verteilt. Außerdem wurde eine Anleitung zur registerkartenbasierten Organisation in der Amazon-Cognito-Konsole hinzugefügt.	30. August 2023
Der direkte Zugriff auf den LOGIN-Endpunkt wurde weniger betont.	Es wurde eine visuelle Übersicht über den Login-Endpunkt des Benutzerpools hinzugefügt und hervorgehoben, die Authentifizierung mit Autorisieren des Endpunkts zu starten.	30. August 2023
Amazon Cognito ist jetzt im asiatisch-pazifischen Raum (Osaka) und in Israel (Tel Aviv) AWS-Regionen verfügbar.	Sie können jetzt Amazon Cognito Cognito-Ressourcen in den Regionen Asien-Pazifik (Osaka) und Israel (Tel Aviv) erstellen.	30. August 2023
Es wurden Informationen zur Autorisierung für Amazon Cognito mit Amazon Verified Permissions eingeführt.	In Ihrer App können Sie die Verified-Permissions-API aufrufen, um Zugriffsentscheidungen von einer zentralen Stelle aus zu treffen.	1. August 2023
Eine neue Funktion zur Protokollierung detaillierter Benutzeraktivitäten im Benutzerpool wurde in Amazon CloudWatch Logs hinzugefügt.	Sie können jetzt Fehler bei der Zustellung von E-Mail- und SMS-Nachrichten in CloudWatch Protokollgruppen protokollieren.	1. August 2023

Die Informationen zu AWS verwalteten Richtlinien für Identitätspool-Gastbenutzer wurden aktualisiert.	Der Umfang der Berechtigungen für Identitätspool-Gastbenutzer umfasst jetzt sowohl eine Inline-Sitzungsrichtlinie als auch eine AWS verwaltete Sitzungsrichtlinie.	16. Mai 2023
Inhaltsverbesserung und neue Konsolenanweisungen für Amazon Cognito Cognito-Identitätspools.	Ergänzung neuer Anleitungen zur Konsole entsprechend dem neuen Konsolenerlebnis, verbesserte Einzelheiten zur Code-Integration für Identitätspools.	16. Mai 2023
Ergänzungen und Verbesserungen der Service-Homepage und der Benutzerpool-Homepage.	Die Übersichtsseiten für Amazon Cognito und Benutzerpools wurden aktualisiert.	16. Mai 2023
Allgemeine Verbesserungen der Dokumentation zu Benutzerpool-Tokens.	Die Beispiel-Tokens wurden aktualisiert und es wurden neue Informationen zur Überprüfung von Tokens hinzugefügt.	16. Februar 2023
Sie können jetzt Datenereignisse in AWS CloudTrail Amazon Cognito Identity Pools protokollieren.	CloudTrail unterstützt die Auswahl von Amazon Cognito Cognito-Identitätspools für umfangreiche API-Operationen in Trails, die Datenereignisse protokollieren.	15. Februar 2023
Beispiele und Beschreibungen für Lambda-Trigger wurden aktualisiert.	Beispiele für Lambda-Trigger wurden auf JavaScript Version 3 aktualisiert. Sie können Lambda-Trigger jetzt direkt mit API-Aktionen korrelieren.	31. Januar 2023

Amazon Cognito Cognito-Identitätspools wenden eine AWS verwaltete Richtlinie auf nicht authentifizierte Sitzungen an.	Für Benutzer von Identitätspools, die sich mithilfe des erweiterten Ablaufs authentifizieren, wird jetzt eine zusätzliche AWS verwaltete Richtlinie auf ihre Sitzung angewendet.	31. Januar 2023
Codebeispiele wurden hinzugefügt.	Diese Anleitung enthält jetzt Beispielcode für Ihre Amazon-Cognito-App in einer Vielzahl von Programmiersprachen.	23. Januar 2023
Es wurden Informationen zu API-Modellen und zur Authentifizierung mit Amazon Cognito Cognito-Benutzerpools hinzugefügt.	Amazon-Cognito-Benutzerpools verfügen über mehrere API-Schnittstellen und Formate für die Autorisierung von Anforderungen.	15. Dezember 2022
Amazon Cognito ist jetzt in Europa (Mailand) AWS-Region erhältlich.	Sie können jetzt Amazon-Cognito-Benutzerpools in der Region Europa (Mailand) erstellen.	6. Dezember 2022
Es wurden Informationen zum Schutz vor dem Löschen von Benutzerpools hinzugefügt.	Wenn Sie mit dem einen neuen Benutzerpool erstellen AWS Management Console, ist dieser jetzt standardmäßig vor dem Löschen geschützt.	20. Oktober 2022
Es wurden ein Benutzerhandbuch für die gehostete Benutzeroberfläche und Informationen zu TOTP MFA in der gehosteten Benutzeroberfläche hinzugefügt.	Ihre Benutzer können jetzt ein TOTP-MFA-Gerät in der von Amazon Cognito gehosteten Benutzeroberfläche registrieren. Sie können jetzt eine Vorschau der standardmäßigen gehosteten Benutzeroberfläche anzeigen.	8. September 2022

Es wurden Informationen über AWS WAF und Amazon Cognito hinzugefügt.	Sie können jetzt eine AWS WAF Web-ACL mit einem Amazon Cognito Cognito-B enutzerpool verknüpfen.	03. August 2022
Weitere AWS CloudTrail Beispielereignisse wurden hinzugefügt.	Amazon Cognito protokolliert jetzt Verbund- und gehostete UI-Anfragen in Ihrem Trail.	15. Juni 2022
Es wurden Informationen zur zweistufigen Attributverifizierung hinzugefügt.	Sie können jetzt wählen, ob Ihr Benutzer eine neue E-Mail-Adresse oder Telefonnummer verifizieren muss, bevor er sich damit anmelden kann.	9. Juni 2022
Die Föderationsdokumentation wurde aktualisiert. Neue Funktion zur Weitergabe von IP-Adressen.	Aktualisierte exemplarische Vorgehensweisen für die Einrichtung eines Benutzerpools in sozialen Netzwerken. IdPs Es wurden Informationen über Verbundbenutzerprofile und Attributzuordnung hinzugefügt. Neue Informationen zu Gerätefingerabdrücken wurden hinzugefügt, um die Sicherheit zu erhöhen.	31. Mai 2022
Melden Sie Verbundbenutzer ohne Interaktion mit der gehosteten Benutzeroberfläche an	Es wurde eine neue Seite hinzugefügt, auf der erklärt wird, wie Anwendungen mit einem Lesezeichen versehen werden können, sodass Amazon Cognito Benutzer im Hintergrund zur Verbundanmeldung weiterleitet.	29. Mai 2022

SMS- und E-Mail-Nachrichten in der Region für Amazon Cognito Cognito-Benutzerpools	Sie können jetzt Amazon Simple Notification Service für SMS-Nachrichten und Amazon Simple Email Service für E-Mail-Nachrichten in demselben AWS-Region Benutzerpool verwenden.	14. März 2022
Aktualisierungen der Seite „Kontingente“	Ressourcen- und Anforderungsquoten wurden hinzugefügt und klarer formuliert.	10. Januar 2022
Neues Konsolenerlebnis für Amazon Cognito Cognito-Benutzerpools	Anweisungen zum Erstellen und Verwalten von Benutzerpools in der aktualisierten Amazon Cognito-Konsole wurden aktualisiert.	18. November 2021
RevokeToken API und Widerrufsendpunkt	Sie können den RevokeToken Vorgang verwenden, um ein Aktualisierungstoken für einen Benutzer zu widerrufen.	10. Juni 2021
Bewährte Methoden für mehrere Mandanten	Es wurden bewährte Methoden für Mehrmandantenanwendungen hinzugefügt.	4. März 2021

Attribute für die Zugriffskontrolle	Amazon Cognito Identity Pools bieten Attribute für die Zugriffskontrolle (AFAC), mit denen Kunden Benutzern Zugriff AWS auf Ressourcen gewähren können. Die Autorisierung kann basierend auf den Attributen der Benutzer des Identitätsanbieters erfolgen, mit dem sie mit Amazon Cognito verbunden waren.	15. Januar 2021
Benutzerdefinierter Lambda-Trigger für SMS-Absender und Lambda-Trigger für benutzerdefinierten E-Mail-Absender	Mit dem benutzerdefinierten Lambda-Auslöser für SMS-Sender und dem benutzerdefinierten Lambda-Auslöser für E-Mail-Sender können Sie einem Drittanbieter ermöglichen, E-Mail- und SMS-Benachrichtigungen an Ihre Benutzer aus Ihrem Lambda-Funktionscode zu senden.	30. November 2020
Aktualisierungen von Amazon Cognito-Tokens	Aktualisierte Ablaufinformationen wurden den Zugriffs-, ID- und Aktualisierungstoken hinzugefügt.	29. Oktober 2020

Amazon Cognito Service Quotas	Service Quotas ist für Amazon-Cognito-Kategoriekontingente verfügbar. Sie können die Konsole Service Quotas verwenden, um die Kontingentnutzung einzusehen, eine Kontingenterhöhung anzufordern und CloudWatch Alarme zur Überwachung Ihrer Kontingentnutzung zu erstellen. Im Rahmen dieser Änderung wurde der Abschnitt Verfügbare CloudWatch Metriken für Amazon Cognito Cognito-Benutzerpools aktualisiert, um die neuen Informationen widerzuspiegeln. Der neue Abschnittsname lautet: Tracking-Kontingente und Nutzung in CloudWatch Service Quotas	29. Oktober 2020
Amazon Cognito Cognito-Kontingentkategorisierung	Kontingentkategorien sind verfügbar, um die Kontingentnutzung zu überwachen und eine Erhöhung anzufordern. Die Kontingente werden nach allgemeinen Anwendungsfällen in Kategorien gruppiert.	17. August 2020
Amazon Cognito wird in US AWS GovCLoud unterstützt	Amazon Cognito wird jetzt in der Region AWS GovCloud (USA) unterstützt.	13. Mai 2020

Dokumentaktualisierungen von Amazon Cognito Pinpoint	Eine neue, mit einem Service verknüpfte Rolle wurde hinzugefügt. Die Anweisungen zum Thema „Verwenden von Amazon Pinpoint Analytics mit Amazon-Cognito-Benutzerpools“ wurden aktualisiert.	13. Mai 2020
Neues spezielles Sicherheitskapitel für Amazon Cognito	Das Kapitel Sicherheit kann Ihrem Unternehmen dabei helfen, detaillierte Informationen sowohl über die integrierte als auch über die konfigurierbare Sicherheit von AWS Diensten zu erhalten. Unsere neuen Kapitel informieren Sie über die Sicherheit der Cloud und in der Cloud.	30. April 2020
Amazon Cognito Identity Pools unterstützt jetzt die Anmeldung mit Apple	„Mit Apple anmelden“ ist in allen Regionen möglich, in denen Amazon Cognito angeboten wird, außer in der Region cn-north-1.	7. April 2020
Neue Facebook-API-Versionierung	Versionsauswahl zur Facebook-API hinzugefügt	3. April 2020
Aktualisierung ohne Berücksichtigung von Groß- und Kleinschreibung beim Benutzernamen	Eine Empfehlung zum Aktivieren der Nichtbeachtung der Groß-/Kleinschreibung vor dem Erstellen eines Benutzerpools wurde hinzugefügt.	11. Februar 2020

Neue Informationen über AWS Amplify	Es wurden Informationen zur Integration von Amazon Cognito mit Ihrer Web- oder mobilen App mithilfe von AWS Amplify SDKs und Bibliothe ken hinzugefügt. Informati onen zur Verwendung der Amazon-Cognito-SDKs, die AWS Amplify vorausgingen, wurden entfernt.	22. November 2019
Neues Attribut für Benutzerp ool-Trigger	Amazon Cognito nimmt jetzt einen `clientMet adata` Parameter in die Ereignisinformationen auf, die es für die meisten Benutzerpool-Trigger an die AWS Lambda Funktionen weitergibt. Sie können diesen Parameter verwenden, um Ihren benutzerdefinierten Authentifizierungs-Workflow mithilfe zusätzlicher Daten zu verbessern.	4. Oktober 2019
Das Limit wurde aktualisiert	Das Drosselungslimit für die ListUsers API-Aktion wurde aktualisiert.	25. Juni 2019
Neues Limit	Die weichen Limits für Benutzerpools enthalten jetzt ein Limit für die Anzahl der Benutzer.	17. Juni 2019

Amazon SES SES-E-Mail-Einstellungen für Amazon Cognito Cognito-Benutzerpools	Sie können einen Benutzerpool so konfigurieren, dass Amazon Cognito Ihren Benutzern E-Mails sendet, indem Sie Ihre Amazon-SES-Konfiguration verwenden. Mit dieser Einstellung ermöglicht Amazon Cognito mehr E-Mails zuzustellen als sonst.	8. April 2019
Unterstützung für Tagging	Es wurden Informationen zum Markieren von Amazon-Cognito-Ressourcen hinzugefügt.	26. März 2019
Ändern Sie das Zertifikat für eine benutzerdefinierte Domain	Wenn Sie eine benutzerdefinierte Domäne zum Hosten der Amazon-Cognito-Benutzeroberfläche verwenden, können Sie das SSL-Zertifikat für diese Domäne nach Bedarf ändern.	19. Dezember 2018
Neues Limit	Ein neues Limit wird hinzugefügt, das die maximale Anzahl von Gruppen begrenzt, denen ein Benutzer angehören kann.	14. Dezember 2018
Aktualisierte Grenzwerte	Die Soft Limits für Benutzerpools werden aktualisiert.	11. Dezember 2018
Aktualisierung der Dokumentation zur Überprüfung von E-Mail-Adressen und Telefonnummern	Es wurden Informationen zum Konfigurieren des Benutzerpools für eine erforderliche Verifizierung per E-Mail oder Telefon bei der Anmeldung eines Benutzers in der App hinzugefügt.	20. November 2018

Aktualisierung der Dokumentation zum Testen von E-Mails	Es wurden Anleitungen für das Initiieren von E-Mails von Amazon Cognito während des Testens der App hinzugefügt.	13. November 2018
Erweiterte Sicherheit von Amazon Cognito	Es wurden neue Sicherheitsfunktionen hinzugefügt, damit Ihre Entwickler ihre Apps und die Benutzer vor bösartigen Bots schützen können, damit Benutzerkonten gegen gefälschte Anmeldeinformationen geschützt werden können, und damit die Challenges, die für eine Signatur erforderlich sind, automatisch abhängig vom berechneten Risiko des Signierversuchs angepasst werden.	14. Juni 2018
Benutzerdefinierte Domains für Amazon Cognito Hosted UI	Ermöglichen Sie es Entwicklern, ihre eigene, vollständig benutzerdefinierte Domäne für die gehostete Benutzeroberfläche in Amazon-Cognito-Benutzerpools zu verwenden.	4. Juni 2018
Amazon Cognito Cognito-Benutzerpools OIDC-Identitätsanbieter	Benutzerpool-Anmeldung über einen OpenID Connect (OIDC)-Identitätsanbieter hinzugefügt, wie z. B. Salesforce oder Ping Identity.	17. Mai 2018
Amazon Cognito Lambda-Migrationstrigger	Hinzufügung von Seiten zur Lambda Migration-Auslöser-Funktion	8. April 2018

Aktualisierung des Amazon Cognito Cognito-Entwicklerhandbuchs	Hinzufügung von „Was ist Amazon Cognito“ und „Erste Schritte mit Amazon Cognito“ auf der obersten Ebene. Weiterhin Hinzufügung typischer Szenarien und Reorganisierung des Inhaltsverzeichnisses zu Benutzerpools. Hinzufügung eines neuen Abschnitts „Erste Schritte mit Amazon-Cognito-Benutzerpools“.	6. April 2018
Betaversion zur erweiterten Sicherheit von Amazon Cognito	Es wurden neue Sicherheitsfunktionen hinzugefügt, damit Ihre Entwickler ihre Apps und die Benutzer vor bösartigen Bots schützen können, damit Benutzerkonten gegen irgendwo im Internet kursierende, gefälschte Anmeldeinformationen geschützt werden können, und damit die Challenges, die für eine Signatur erforderlich sind, automatisch abhängig vom berechneten Risiko des Signierversuchs angepasst werden.	28. November 2017

Amazon Pinpoint Pinpoint-Integration	Es wurde die Möglichkeit hinzugefügt, Amazon Pinpoint zu verwenden, um Analysen für Ihre Amazon-Cognito-Benutzerpool-Apps bereitzustellen und die Benutzerdaten für Amazon-Pinpoint-Kampagnen zu ergänzen.	26. September 2017
Verbund und integrierte App-UI-Funktionen von Amazon Cognito Cognito-Benutzerpools	Es wurde die Möglichkeit hinzugefügt, dass sich Ihre Benutzer über Facebook, Google, Login with Amazon oder einen SAML-Identitätsanbieter bei Ihrem Benutzerpool anmelden. Es wurde eine anpassbare integrierte App-Benutzeroberfläche und OAuth-2.0-Unterstützung für benutzerdefinierte Ansprüche hinzugefügt.	10. August 2017
Änderungen an den Funktionen im Zusammenhang mit HIPAA und PCI-Compliance	Es wurde die Möglichkeit hinzugefügt, die Verwendung einer Telefonnummer oder E-Mail-Adresse als Benutzername durch Ihre Benutzer zuzulassen.	6. Juli 2017

Benutzergruppen und Funktionen zur rollenbasierten Zugriffskontrolle	Administrative Funktionen zum Erstellen und Verwalten von Benutzergruppen wurden hinzugefügt. Administratoren können Benutzern basierend auf den Gruppenmitgliedschaften und den von Administratoren erstellten Regeln IAM-Rollen zuweisen.	15. Dezember 2016
Aktualisierung der Dokumentation	Aktualisierte Beispiele, die zeigen, wie AWS Lambda Trigger mit Benutzerpools verwendet werden.	27. November 2016
Aktualisierung der Dokumentation	Aktualisierte iOS-Codebeispiele.	18. November 2016
Aktualisierung der Dokumentation	Informationen zum Bestätigungsablauf für Benutzerkonten wurden hinzugefügt.	9. November 2016
Funktion „Benutzerkonten erstellen“	Administrative Funktionen zum Erstellen von Benutzerkonten über die Amazon-Cognito-Konsole und die API wurden hinzugefügt.	6. Oktober 2016
Funktion zum Importieren von Benutzern	Es wurde eine Massen-Import-Funktion für Cognito User Pools hinzugefügt. Verwenden Sie diese Funktion zum Migrieren von Benutzern des vorhandenen Identitätsanbieters in einen Amazon-Cognito-Benutzerpool.	1. September 2016

Allgemeine Verfügbarkeit von Cognito User Pools	Die Funktion Cognito User Pools wurde hinzugefügt. Mit dieser Funktion können Sie in Ihrer mobilen App oder Webanwendung Benutzerv erzeichnisse erstellen und verwalten sowie Registrie rungs- und Anmeldein formationen hinzufügen.	28. Juli 2016
SAML-Unterstützung	Unterstützung der Authentif izierung mit Identitätsanbieter n über Security Assertion Markup Language 2.0 (SAML 2.0) wurde hinzugefügt.	23. Juni 2016
CloudTrail Integration	Integration mit hinzugefügt AWS CloudTrail.	18. Februar 2016
Integration von Ereignissen mit Lambda	Ermöglicht es Ihnen, eine AWS Lambda Funktion als Reaktion auf wichtige Ereignisse in Amazon Cognito auszuführen.	9. April 2015
Datenstream zu Amazon Kinesis	Ermöglicht die Steuerung der Daten-Streams und stellt zugehörige Informationen bereit.	4. März 2015
OpenID Connect-Unterstütz ung	Ermöglicht die Unterstüt zung von OpenID Connect-A nbietern.	23. November 2014
Synchronisation per Push	Ermöglicht die Unterstützung der automatischen Push-Sync hronisierung.	6. November 2014

Unterstützung für vom Entwickler authentifizierte Identitäten hinzugefügt	Entwickler mit eigenen Authentifizierungs- und Identitätsmanagementsysteme n können jetzt in Amazon Cognito als Identitätsanbieter behandelt werden.	29. September 2014
Allgemeine Verfügbarkeit von Amazon Cognito		10. Juli 2014

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.