Entwicklerhandbuch

Amazon Cognito

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Cognito: Entwicklerhandbuch

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, auf eine Art und Weise, dass Kunden irrgeführt werden könnten oder Amazon schlecht gemacht oder diskreditiert werden könnte. Alle anderen Marken, die nicht im Besitz von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist Amazon Cognito? ... 1
Benutzerpools ... 1
Identitäten-Pools ... 2
Funktionen von Amazon Cognito ... 3
Benutzerpools ... 3
Identitäten-Pools ... 5
Vergleich von Amazon-Cognito-Benutzerpools und -Identitätspools ... 8
Erste Schritte mit Amazon Cognito ... 13
Regionale Verfügbarkeit ... 14
Preise für Amazon Cognito ... 14
Begriffe und Konzepte ... 14
Allgemeines ... 15
Benutzerpools ... 18
Identitäten-Pools ... 23
Erste Schritte mit AWS ... 24
Melden Sie sich an für ein AWS-Konto ... 24
Erstellen eines Benutzers mit Administratorzugriff ... 25
Erste Schritte mit Benutzerpools ... 27
Ihre erste App und Ihr erster Benutzerpool ... 28
Andere Anwendungsoptionen ... 31
Beispiel für React SPA ... 31
Beispiel für eine mobile Flutter-App ... 35
Nächste Schritte ... 37
Hinzufügen eines Social-Identity-Anbieters ... 38
Einen SAML-IdP hinzufügen ... 47
Erste Schritte mit Identitätspools ... 51
Erstellen eines Identitätspools in Amazon Cognito ... 51
Einrichten eines SDK ... 54
Integrieren der Identitätsanbieter ... 54
Abrufen von Anmeldeinformationen ... 54
Zusätzliche Optionen für den Einstieg ... 56
Integration in Apps ... 58
Authentifizierung mit AWS Amplify ... 60
Erstellen einer Benutzeroberfläche (User Interface, UI) mit Amplify ... 60

Authentifizierung mit AWS SDKs ... 61
Wie funktioniert die Authentifizierung ... 62
Verwaltete Anmeldeauthentifizierung ... 62
SDK-Authentifizierung ... 64
Authentifizierung durch einen Identitätsanbieter eines Drittanbieters ... 66
Authentifizierung des Identitätspools ... 68
Arbeitet mit AWS SDKs ... 70
Autorisierung mit Amazon Verified Permissions ... 71
API-Autorisierung mit verifizierten Berechtigungen ... 73
Beispielrichtlinie für einen Amazon-Cognito-Benutzer ... 75
Codebeispiele ... 79
Amazon Cognito Identity ... 81
Grundlagen ... 81
Szenarien ... 107
Amazon Cognito Identity Provider ... 109
Grundlagen ... 120
Szenarien ... 276
Amazon Cognito Sync ... 442
Grundlagen ... 442
Bewährte Verfahren für mehrere Mandanten ... 445
Benutzerpools pro Mandant ... 447
App-Clients pro Mandant ... 448
Benutzerpoolgruppen pro Mandant ... 449
Benutzerdefinierte Attribute pro Mandant ... 450
Benutzerdefinierte Bereiche pro Mandant ... 451
Beispiel für eine Ressource ... 453
Sicherheitsempfehlungen für Mehrmandantenfähigkeit ... 454
Häufige Amazon-Cognito-Szenarien ... 456
Authentifizierung über einen Benutzerpool ... 456
Zugriff auf serverseitige Ressourcen ... 457
Zugriff auf Ressourcen mit API Gateway und Lambda ... 458
Greifen Sie mit einem Benutzerpool und einem Identitätspool auf AWS Dienste zu ... 458
Authentifizierung über einen Drittanbieter und Zugriff auf AWS -Services über einen Identitätspool ... 458
Greifen Sie mit Amazon Cognito auf AWS AppSync Ressourcen zu ... 459
Amazon-Cognito-Benutzerpools ... 460

Features ... 461
Registrieren ... 461
Anmelden ... 462
Verwaltete Anmeldung ... 463
Sicherheit ... 464
Benutzerdefinierte Nutzerumgebung ... 464
Überwachung und Analytik ... 465
Integration von Amazon-Cognito-Identitätspools ... 465
Funktionspläne für Benutzerpools ... 466
Wählen Sie einen Funktionsplan aus ... 468
Funktionen nach Tarif ... 469
Funktionen des Essentials-Plans ... 473
Funktionen des Plus-Plans ... 477
Schalten Sie Funktionen aus, die nicht in Frage kommen ... 481
Bewährte Methoden für die Gewährleistung der Sicherheit ... 482
Schützen Sie Ihren Benutzerpool auf Netzwerkebene ... 482
Schützen Sie sich vor Missbrauch von SMS-Nachrichten ... 482
Verstehen Sie öffentliche Authentifizierung ... 482
Schützen Sie vertrauliche Kunden mit vertraulichen Kundengeheimnissen ... 486
Schützen Sie andere Geheimnisse ... 487
Verwaltung des Benutzerpools, geringste Rechte ... 488
Token sichern und verifizieren ... 491
Ermitteln Sie die Identitätsanbieter, denen Sie vertrauen möchten ... 491
Machen Sie sich mit den Auswirkungen von Geltungsbereichen auf den Zugriff auf Benutzerprofile vertraut ... 492
Bereinigen Sie die Eingaben für Benutzerattribute ... 492
Authentifizierung ... 493
Implementieren Sie Authentifizierungsabläufe ... 494
Wissenswertes ... 496
Beispiel für einen Authentifizierungsablauf ... 500
Verwaltete Anmeldeauthentifizierung ... 502
SDK-Authentifizierung ... 505
Authentifizierungsabläufe ... 510
SDK-Autorisierungsmodelle ... 537
IdP-Anmeldung eines Drittanbieters ... 554
Funktionsweise der Verbundanmeldung in Amazon-Cognito-Benutzerpools ... 554

Die Verantwortlichkeiten einer App als SP für Amazon Cognito 555
Wissenswertes über die Anmeldung von Drittanbietern bei Amazon-Cognito-Benutzerpools 556
Identitätsanbieter 557
Anbieter sozialer Identitäten 564
SAML-Anbieter 574
OIDC-Anbieter 604
Zuordnung von IdP-Attributen 615
Verbundene Benutzer verknüpfen 622
Verwaltete Anmeldung 626
Lokalisierung verwalteter Logins 628
Verwaltete Anmeldung einrichten mit AWS Amplify 629
Verwaltete Anmeldung mit der Amazon Cognito Cognito-Konsole einrichten 630
Anzeigen Ihrer Anmeldeseite 630
Anpassen Ihrer Authentifizierungsseiten 632
Wissenswertes zur verwalteten Anmeldung und zur gehosteten Benutzeroberfläche 632
Konfigurieren einer Domäne 635
Branding und Anpassung 650
Verwenden von Lambda-Auslösern 670
Wissenswertes 673
Richten Sie Trigger ein 676
Lambda-Auslöserereignis für einen Benutzerpool 676
Allgemeine Parameter von Lambda-Auslösern für Benutzerpools 677
Client-Metadaten 678
Triggern Sie Quellen durch Operation 682
Triggerquellen nach Funktion 688
Voranmeldung 693
Nachbestätigung 701
Vorauthentifizierung 704
Nachauthentifizierung 707
Benutzerdefinierte Herausforderung 710
Generierung von Pre-Token 730
Benutzer migrieren 755
Benutzerdefinierte Nachricht 761
Benutzerdefinierte Absender 769
Verwalten von Benutzern 786

Zulassen der Benutzerregistrierung ... 787
Registrieren und Bestätigen von Benutzerkonten ... 790
Erstellen von Benutzern als Administrator ... 820
Hinzufügen von Gruppen zu einem Benutzerpool ... 828
Verwalten von und Suchen nach Benutzern ... 832
Passwörter ... 840
Importieren von Benutzern in einen Benutzerpool ... 846
Attribute ... 867
Benutzerpool-Token ... 885
ID-Token ... 887
Zugriffstoken ... 892
Tokens aktualisieren ... 896
Widerrufen von Token ... 903
Verifizieren eines JSON-Web-Tokens ... 905
Zwischenspeicherung von Token ... 912
Zugreifen auf Ressourcen nach der Anmeldung ... 916
Zugriff auf Ressourcen mit verifizierten Berechtigungen ... 457
Zugreifen auf API-Gateway-Ressourcen ... 918
Zugreifen auf AWS Ressourcen mithilfe eines Identitätspools ... 919
M2M und Bereiche ... 923
API-Autorisierung ... 924
Machine-to-machine (M2M) -Autorisierung ... 925
Grundlegendes zu Bereichen ... 926
Grundlegendes zu Ressourcenservern ... 928
Weitere Features ... 933
Aktualisierung eines Benutzerpools und eines App-Clients ... 933
App-Clients ... 938
Arbeiten mit Geräten ... 948
Verwenden von Amazon Pinpoint Analytics ... 955
E-Mail-Einstellungen ... 961
Einstellungen für SMS-Nachrichten ... 977
Verwendung der Sicherheitsfunktionen ... 987
Hinzufügen von MFA ... 989
Schutz vor Bedrohungen ... 1010
AWS WAF Web ACLs ... 1040
Groß-/Kleinschreibung ... 1045

Löschschutz ... 1047
Verwalten der Offenlegung von Benutzern ... 1048
Referenz für Benutzerpool-Endpunkte ... 1055
Verwaltete Anmeldeendpunkte ... 1056
Endpunkte im Verbund ... 1065
OAuth 2.0 Zuschüsse ... 1098
Verwendung von PKCE ... 1100
Antworten auf verwaltete Anmelde- und Verbundfehler ... 1102
Amazon-Cognito-Identitätspools ... 1105
Konfiguration von Identitätspools ... 1107
Erstellen eines Identitäten-Pools ... 1108
IAM-Rollen von Benutzern ... 1110
Authentifizierte und nicht authentifizierte Identitäten ... 1110
Gastzugang aktivieren oder deaktivieren ... 1110
Ändern der mit einem Identitätstyp verknüpften Rolle ... 1112
Identitätsanbieter bearbeiten ... 1113
Löschen eines Identitätspools ... 1115
Löschen einer Identität aus einem Identitätspool ... 1115
Verwenden von Amazon Cognito Sync mit Identitätspools ... 1116
Identitäten-Pools – Authentifzierungsablauf ... 1119
Der erweiterte (vereinfachte) Authentifizierungsablauf ... 1120
Der grundlegende (klassische) Authentifizierungsablauf ... 1120
Der vom Entwickler authentifizierte Authentifizierungsablauf ... 1122
Welchen Authentifizierungsablauf sollte ich implementieren? ... 1122
Überblick über die API-Operationen des Authentifizierungsflusses ... 1123
IAM-Rollen ... 1128
Einrichten einer Vertrauensrichtlinie ... 1129
Zugriffsrichtlinien ... 1133
Vertrauensstellungen und Berechtigungen für Rollen ... 1145
Bewährte Methoden für die Gewährleistung der Sicherheit ... 1147
Bewährte Methoden für die IAM-Konfiguration ... 1147
Bewährte Methoden zur Konfiguration des Identitätspo ... 1150
Verwenden von Attributen für Zugriffskontrolle ... 1152
Verwenden von Attributen für die Zugriffskontrolle mit Amazon-Cognito-Identitätspools ... 1153
Verwenden von Attributen für die Zugriffskontrollrichtlinie (Beispiel) ... 1154
Attribute für Zugriffskontrolle deaktivieren ... 1157

Standard-Anbietermappings ... 1157
Verwenden der rollenbasierten Zugriffskontrolle ... 1159
Erstellen von Rollen für das Rollen-Mapping ... 1160
Gewähren der Berechtigung zum Übergeben einer Rolle ... 1161
Zuweisen von Rollen zu Benutzern mit Token ... 1162
Verwendung des regelbasierten Mappings, um Benutzern Rollen zuzuweisen ... 1163
Token-Ansprüche zur Verwendung in regelbasiertem Mapping ... 1165
Bewährte Methoden für rollenbasierte Zugriffskontrolle ... 1166
Abrufen von Anmeldeinformationen ... 1167
Verwenden von -Anmeldeinformationen ... 1174
Identitätsanbieter von Drittanbietern ... 1177
Facebook ... 1178
Login with Amazon ... 1187
Google ... 1191
Mit Apple anmelden ... 1201
Open-ID-Connect-Anbieter ... 1208
SAML-Identitätsanbieter ... 1211
Entwicklerauthentifizierte Identitäten ... 1214
Erläuterungen zum Authentifizierungsfluss ... 1214
Definieren eines Entwickleranbieternamens und Zuordnen zu einem Identitäten-Pool ... 1215
Implementieren eines Identitätsanbieters ... 1216
Aktualisieren der Anmeldezuweisung (nur Android und iOS) ... 1224
Aufrufen eines Tokens (Serverseite) ... 1225
Verbinden mit einer vorhandenen Social Identity ... 1226
Unterstützen des Anbieterwechsels ... 1227
Wechseln von Identitäten ... 1231
Android ... 1231
iOS – Objective-C ... 1232
iOS – Swift ... 1232
JavaScript ... 1233
Unity ... 1234
Xamarin ... 1234
Amazon Cognito Sync ... 1236
Erste Schritte mit Amazon Cognito Sync ... 1237
Einrichten eines Identitätspools in Amazon Cognito ... 1237
Speichern und Synchronisieren von Daten ... 1237

Synchronisieren von Daten zwischen Clients ... 1237
Initialisieren des Amazon-Cognito-Sync-Clients ... 1238
Grundlegendes zu Datensätzen ... 1240
Lesen und Schreiben von Daten in Datensätze ... 1242
Synchronisieren lokaler Daten mit dem Sync Store ... 1244
Bearbeitung von Rückrufen bei Ereignissen ... 1248
Android ... 1249
iOS – Objective-C ... 1251
iOS – Swift ... 1254
JavaScript ... 1258
Unity ... 1261
Xamarin ... 1264
Push-Synchronisierung implementieren ... 1266
Erstellen einer Amazon-Simple-Notification-Service-(Amazon-SNS)-App ... 1267
Aktivieren der Push-Synchronisierung in der Amazon-Cognito-Konsole ... 1267
Push-Synchronisierung in Ihrer App verwenden: Android ... 1268
Push-Sync in Ihrer App verwenden: iOS – Objective-C ... 1271
Push-Sync in Ihrer App verwenden: iOS – Swift ... 1274
Implementierung von Amazon Cognito Sync-Streams ... 1277
Anpassen von Workflows mit Amazon Cognito Events ... 1279
Sicherheit ... 1285
Datenschutz ... 1286
Datenverschlüsselung ... 1286
Identity and Access Management ... 1287
Zielgruppe ... 1288
Authentifizierung mit Identitäten ... 1289
Verwalten des Zugriffs mit Richtlinien ... 1293
Funktionsweise von der Amazon Cognito mit IAM ... 1296
Beispiele für identitätsbasierte Richtlinien ... 1306
Fehlerbehebung ... 1311
Verwenden von servicegebundenen Rollen ... 1313
Protokollierung und Überwachung ... 1318
Überwachung der Kosten ... 1319
Exportieren von Benutzerpool-Protokollen ... 1322
Überwachung von Kontingenten und Nutzung ... 1334
CloudTrail protokolliert ... 1357

Compliance-Validierung ... 1386
Ausfallsicherheit ... 1387
Überlegungen zu regionenbezogenen Daten ... 1387
Sicherheit der Infrastruktur ... 1388
Konfigurations- und Schwachstellenanalyse ... 1389
AWS verwaltete Richtlinien ... 1389
Richtlinienaktualisierungen ... 1391
Markieren von Ressourcen ... 1395
Unterstützte Ressourcen ... 1396
Tag (Markierung)-Einschränkungen ... 1396
Verwalten von Tags mit der Konsole ... 1396
AWS CLI Beispiele ... 1397
Zuweisen von Tags ... 1397
Anzeigen von Tags ... 1398
Entfernen von Tags ... 1399
Anwenden von Tags beim Erstellen von Ressourcen ... 1400
API-Aktionen ... 1400
API-Aktionen für Tags für Benutzerpools ... 1400
API-Aktionen für Tags für Identitäten-Pools ... 1401
Kontingente ... 1402
Informationen zu API-Anforderungsratenkontingenten ... 1402
Kategorisierung von Kontingenten ... 1402
API-Operationen von Amazon-Cognito-Benutzerpools mit spezieller Handhabung der Anforderungsrate ... 1403
Monthly active users (Aktive Benutzer pro Monat) ... 1404
Verwalten von API-Anforderungsratenkontingenten ... 1406
Kontingentanforderungen identifizieren ... 1406
Optimieren Sie die Anfrageraten ... 1407
Verfolgen der Kontingentnutzung ... 1408
Verfolgen Sie die monatlich aktiven Benutzer (MAUs) ... 1409
Beantragen einer Kontingenterhöhung ... 1410
Anforderungsratenkontingente von Benutzerpools ... 1410
Grenzwerte für die Rate von Massenanfragen für Benutzerpool-Domänen ... 1423
Anforderungsratenkontingente von Identitätspools ... 1424
Kontingente für die Anzahl und Größe der Ressourcen ... 1426
Amazon Cognito Cognito-Benutzerpools, Ressourcenkontingente ... 1427

Gültigkeitsparameter für Amazon-Cognito-Benutzerpools 1430
Kontingente für Code-Sicherheitsressourcen für Amazon-Cognito-Benutzerpools (nicht anpassbar) 1430
Kontingente für Auftragsressourcen für Benutzerimporte von Amazon-Cognito-Benutzerpools 1431
Ressourcenkontingente für Amazon Cognito Cognito-Identitätspools (föderierte Identitäten) 1432
Ressourcenkontingente für Amazon Cognito Sync 1433
Dokumentverlauf 1435
..... mcdlx

Was ist Amazon Cognito?

Amazon Cognito ist eine Identitätsplattform für Web- und mobile Apps. Es ist ein Benutzerverzeichnis, ein Authentifizierungsserver und ein Autorisierungsdienst für OAuth 2.0-Zugriffstoken und AWS Anmeldeinformationen. 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

- Benutzerpools
- Identitäten-Pools
- Funktionen von Amazon Cognito
- Vergleich von Amazon-Cognito-Benutzerpools und -Identitätspools
- Erste Schritte mit Amazon Cognito
- Regionale Verfügbarkeit
- Preise für Amazon Cognito
- Allgemeine Begriffe und Konzepte von Amazon Cognito
- Erste Schritte mit AWS

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 in Ihrer App auch SSO für die 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-Webtoken (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 2.0-Token. OAuth
2. Ihre App tauscht ein Benutzerpool-Token mit einem Identitätspool gegen temporäre AWS Anmeldeinformationen aus, die Sie mit AWS APIs und AWS Command Line Interface (AWS CLI) verwenden können.
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 Sicherheit in der Cloud in Amazon Cognito so gestalten, dass sie SOC1 -3, ISO 27001 und HIPAA-BAA entspricht, jedoch nicht PCI DSS. 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, einem AWS SDK oder dem AWS Management Console() 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 einen Satz von standardisieren, JWTs während Amazon Cognito die Interaktionen mit IdPs abwickelt und deren Ansprüche einem zentralen Token-Format zuordnet.

Ein Amazon-Cognito-Benutzerpool kann ein eigenständiger IdP sein. Amazon Cognito stützt sich bei der Generierung JWTs für Authentifizierung und Autorisierung auf den OpenID Connect (OIDC) - Standard. 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) -Token, die ein Amazon Cognito 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-Identitätsanbieter	Geben Sie ID-Token aus, um Benutzer zu authentifizieren
Autorisierungsserver	Stellen Sie Zugriffstoken aus, um den Benutzerzugriff auf zu autorisieren APIs
SAML 2.0-Dienstanbieter	Verwandeln Sie SAML-Assertionen in ID- und Zugriffstoken
Vertrauende OIDC-Partei	Transformieren Sie OIDC-Token in ID- und Zugriffstoken
Vertrauende Partei des sozialen Anbieters	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 verwaltetem Login
API-Unterstützung für Ihre eigene Benutzero berfläche	Erstellen, verwalten und authentifizieren Sie Benutzer mithilfe von API-Authentifizier

	ungsanfragen in der unterstützten Version [1] AWS SDKs
Multifaktor-Authentifizierung	Verwenden Sie SMS-Nachrichten oder das Gerät Ihres Benutzers als zusätzlichen Authentifizierungsfaktor[1] TOTPs
Sicherheitsüberwachung und Reaktion	Schützt vor böswilligen Aktivitäten und unsicheren Passworten[1]
Passen Sie die Authentifizierungsabläufe an	Erstellen Sie Ihren eigenen Authentifizierungs mechanismus oder fügen Sie benutzerdefinierte Schritte zu bestehenden Abläufen hinzu[2]
Gruppen	Erstellen Sie logische Gruppierungen von Benutzern und eine Hierarchie von IAM-Rolle nansprüchen, wenn Sie Token an Identität spools übergeben
Passen Sie Token an	Passen Sie Ihre ID und Zugriffstoken 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 für Verbundbenutzer nicht verfügbar.

[2] Die Funktion steht Benutzern mit verbundener und verwalteter Anmeldung nicht zur Verfügung.

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 von

vertrauenswürdigen Ansprüchen eines SAML 2.0-, OpenID Connect (OIDC) oder OAuth 2.0 Social Identity Providers (IdP) vorlegen, verknüpfen Sie Ihren Benutzer mit einer Identität im Identitätspool. Das Token, das Ihr Identitätspool für die Identität erstellt, kann temporäre Sitzungsanmeldedaten von () abrufen. AWS Security Token Service AWS STS

Als Ergänzung zu authentifizierten Identitäten können Sie auch einen Identitätspool konfigurieren, um den AWS Zugriff ohne IdP-Authentifizierung zu autorisieren. Sie können einen benutzerdefinierten Authentifizierungsnachweis mit anbieten. Entwicklerauthentifizierte Identitäten Sie können Gastbenutzern mit nicht authentifizierten Identitäten auch temporäre AWS Anmeldeinformationen gewähren.

Mit Identitätspools haben Sie zwei Möglichkeiten, die IAM-Richtlinien in Ihren zu integrieren. 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, vertrauende Partei	Tauschen Sie ein ID-Token aus Ihrem Benutzerpool gegen Web-Identitätsanme ldedaten von AWS STS
SAML 2.0-Dienstanbieter	Tauschen Sie SAML-Assertionen gegen Web-Identitätsanmeldedaten von aus AWS STS
Vertrauende OIDC-Partei	Tauschen Sie OIDC-Token gegen Web-Ident itätsanmeldedaten von AWS STS
Vertrauende Partei des sozialen Anbieters	Tauschen Sie OAuth Tokens von Amazon, Facebook, Google, Apple und Twitter gegen Web-Identitätsanmeldedaten von AWS STS
Benutzerdefinierter vertrauender Partner	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 authentif izierten 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-Identitätsanbieter	Geben Sie OIDC-ID-Token aus, um App-Benutzer zu authentifizieren	✓	
Benutzerverzeichnis	Benutzerprofile für die Authentifizierung speichern	✓	
Autorisieren Sie den API-Zugriff	Stellen Sie Zugriffstoken aus, um den Benutzerzugriff auf APIs (einschließlich Self-Service-API-Operationen für Benutzerprofile), Datenbanken und andere Ressourcen zu autorisieren, die Bereiche akzeptieren OAuth	✓	
Autorisierung der IAM-Webidentität	Generieren Sie Token, die Sie gegen temporäre AWS Anmeldeinformationen AWS STS eintauschen können		✓
SAML 2.0-Dienstanbieter und OIDC-Identitätsanbieter	Stellen Sie maßgeschneiderte OIDC-Token auf	✓	

	der Grundlage von Ansprüchen eines SAML 2.0-Identitätsanbieters aus		
Vertrauende OIDC-Partei und OIDC-Identitätsanbieter	Stellen Sie maßgeschneiderte OIDC-Token auf der Grundlage von Ansprüchen eines OIDC-Identitätsanbieters aus	✓	
OAuth 2.0 Vertrauende Partei und OIDC-Identitätsanbieter	Stellen Sie maßgeschneiderte OIDC-Token aus, die auf den Geltungsbereichen von OAuth 2.0-Social-Anbietern wie Apple und Google basieren	✓	
SAML 2.0-Dienstanbieter und Broker für Anmeldeinformationen	Stellen Sie temporäre AWS Anmeldeinformationen auf der Grundlage von Ansprüchen eines SAML 2.0-Identitätsanbieters aus		✓
Vertrauende OIDC-Partei und Broker für Anmeldeinformationen	Stellen Sie temporäre AWS Anmeldeinformationen auf der Grundlage von Ansprüchen eines OIDC-Identitätsanbieters aus		✓

Vertrauensperson des sozialen Anbieters und Vermittler von Zugangsdaten	Stellen Sie temporäre AWS Anmeldeinformationen auf der Grundlage von JSON-Webtoken aus Entwickleranwendungen bei sozialen Anbietern wie Apple und Google aus		✓
Amazon Cognito Cognito-Benutzerpool, vertrauenswürdige Partei und Broker für Anmeldeinformationen	Geben Sie temporäre AWS Anmeldeinformationen auf der Grundlage von JSON-Webtoken aus Amazon Cognito Cognito-Benutzerpools aus		✓
Benutzerdefinierter Vertrauenspartner und Broker für Anmeldeinformationen	Geben Sie temporäre AWS Anmeldeinformationen für beliebige Identitäten aus, die durch die IAM-Anmeldeinformationen des Entwicklers autorisiert wurden		✓
Frontend-Dienst zur Authentifizierung	Registrieren, verwalten und authentifizieren Sie Benutzer mit verwaltetem Login	✓	

API-Unterstützung für Ihre eigene Authentif izierungs-Benutzer oberfläche	Erstellen, verwalten und authentifizieren Sie Benutzer mithilfe von API-Anfragen in der unterstützten Version [1] AWS SDKs	✓
MFA	Verwenden Sie SMS-Nachrichten oder das Gerät Ihres Benutzers als zusätzlichen Authentifizierungs faktor[1] TOTPs	✓
Sicherheitsüberwac hung und Reaktion	Schützen Sie sich vor böswilligen Aktivität en und unsicheren Passworten[1]	✓
Passen Sie die Authentifizierungs abläufe an	Erstellen Sie Ihren eigenen Authentif izierungsmechanism us oder fügen Sie benutzerdefinierte Schritte zu bestehend en Abläufen hinzu[1]	✓
Benutzergruppen	Erstellen Sie logische Gruppierungen von Benutzern und eine Hierarchie von IAM-Rollenansprüchen, wenn Sie Token an Identitätspools weitergeben	✓

Passen Sie Token an	Passen Sie Ihre ID und Zugriffstoken mit neuen, geänderten und unterdrückten Ansprüchen und Geltungsbereichen an	✓	
AWS WAF Web ACLs	Überwachen und kontrollieren Sie Anfragen an Ihr Authentifizierungs-Frontend 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 Rollenvertrauensstellung so, dass der Zugriff auf Web-Identity-Benutzer beschränkt wird	✓
Attributbasierte Zugriffskontrolle	Verwandeln Sie Benutzeransprüche in Prinzipal-Tags für Ihre AWS STS temporäre Sitzung und filtern Sie den Ressourcenzugriff mithilfe von IAM-Richtlinien anhand von Prinzipal-Tags	✓

[1] Die Funktion ist für Verbundbenutzer nicht 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.

Informationen zu den ersten Schritten mit einem AWS SDK finden Sie unter AWS Entwicklertools. Spezifische Entwicklerressourcen für Amazon Cognito 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.

Informationen darüber, ob Amazon Cognito derzeit in einer Region verfügbar ist AWS-Region, finden Sie unter AWS Services nach Regionen.

Weitere Informationen zu regionalen API-Dienstendpunkten 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.

Allgemeine Begriffe und Konzepte 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 Wissensbestand

- 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.

Zugriffstoken

Ein JSON-Web-Token (JWT), das Informationen über die Autorisierung einer Entität für den Zugriff auf Informationssysteme enthält.

App, Anwendung

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 (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.

Authentifizierung

Der Prozess der Etablierung einer authentischen Identität für den Zugriff auf ein Informationssystem. Amazon Cognito akzeptiert Authentifizierungsnachweise von externen Identitätsanbietern und dient auch als Authentifizierungsanbieter für Softwareanwendungen.

Autorisierung

Der Prozess der Erteilung von Berechtigungen für eine Ressource. Zugriffstoken für Benutzerpools enthalten Informationen, anhand derer Anwendungen Benutzern und Systemen den Zugriff auf Ressourcen ermöglichen können.

Autorisierungsserver

Ein OAuth oder OpenID Connect (OIDC) -System, das JSON-Webtoken generiert. Der von Amazon Cognito User Pools verwaltete Autorisierungsserver ist die Autorisierungsserver-Komponente der beiden Authentifizierungs- und Autorisierungsmethoden in Benutzerpools. Benutzerpools unterstützen auch API-Challenge-Response-Flows bei der SDK-Authentifizierung.

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.

Machine-to-machine (M2M) -Autorisierung

Der Prozess der Autorisierung von Anfragen an API-Endpunkte für non-user-interactive Maschinenentitäten, z. B. eine Webserver-Anwendungsebene. Benutzerpools dienen der M2M-Autorisierung bei der Gewährung von Kundenanmeldedaten mit OAuth einem Geltungsbereich von 2.0 in Form von Zugriffstoken.

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 (sozialer) Anbieter

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 OAuthSpezifikation um die Bereitstellung von ID-Token erweitert. Amazon Cognito Cognito-Benutzerpools automatisieren Interaktionen mit OIDC-Anbietern, nachdem sich Benutzer authentifiziert haben.

Hauptschlüssel, WebAuthn

Eine Form der Authentifizierung, bei der kryptografische Schlüssel oder Hauptschlüssel auf dem Gerät eines Benutzers als Authentifizierungsnachweis dienen. Benutzer verifizieren mit biometrischen oder PIN-Code-Mechanismen in einem Hardware- oder Software-Authentifikator, ob sie vorhanden sind. Passwörter sind vor Phishing geschützt und an bestimmte Webseiten/Apps gebunden, sodass sie ein sicheres, passwortloses Erlebnis bieten. Amazon Cognito Cognito-Benutzerpools unterstützen die Anmeldung mit Hauptschlüsseln.

Passwortlos

Eine Form der Authentifizierung, bei der ein Benutzer kein Passwort eingeben muss. Zu den Methoden der kennwortlosen Anmeldung gehören Einmalpasswörter (OTPs), die an E-Mail-Adressen und Telefonnummern gesendet werden, sowie Hauptschlüssel. Amazon Cognito Cognito-Benutzerpools unterstützen die Anmeldung mit OTPs und Hauptschlüsseln.

Öffentliche App

Eine eigenständige Anwendung auf einem Gerät mit lokal gespeichertem Code und ohne Zugriff auf geheime Daten. 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 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 agiert extern als SP und IdPs gegenüber Apps als IdP. SPs

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)

Ein 128-Bit-Etikett, das auf ein Objekt angewendet wird. Amazon Cognito UUIDs ist pro Benutzerpool oder Identitätspool einzigartig, entspricht jedoch keinem bestimmten UUID-Format.

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.

Adaptive Authentifizierung

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

App-Client

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

Rückruf-URL, Umleitungs-URI, Rückgabe-URL

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

Wahlbasierte Authentifizierung

Eine Form der API-Authentifizierung mit Benutzerpools, in denen jedem Benutzer eine Reihe von Anmeldemöglichkeiten zur Verfügung stehen. Sie können zwischen Benutzername und Passwort mit oder ohne MFA, Anmeldung mit Passkey oder passwortloser Anmeldung

mit Einmalpasswörtern per E-Mail oder SMS-Nachricht wählen. Ihre Anwendung kann den Auswahlprozess für Benutzer beeinflussen, indem sie eine Liste mit Authentifizierungsoptionen anfordert oder eine bevorzugte Option deklariert.

Vergleichen Sie mit der clientbasierten Authentifizierung.

Client-basierte Authentifizierung

Eine Form der Authentifizierung mit den Benutzerpools, auf der die API und die Anwendungs-Backends basieren. AWS SDKs Bei der deklarativen Authentifizierung bestimmt Ihre Anwendung unabhängig den Anmeldetyp, den ein Benutzer ausführen soll, und fordert diesen Typ im Voraus an.

Vergleichen Sie mit der wahlbasierten Authentifizierung.

Kompromittierte Anmeldeinformationen

Eine erweiterte Sicherheitsfunktion, die Benutzerkennwörter erkennt, die Angreifer möglicherweise kennen, und Benutzerprofile zusätzlich schützt.

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.

Domäne, Benutzerpool-Domäne

Eine Webdomäne, in der Ihre verwalteten Anmeldeseiten gehostet AWS werden. Sie können DNS in einer Domain einrichten, die Ihnen gehört, oder ein identifizierendes Subdomain-Präfix in einer Domain verwenden, die Ihnen AWS gehört.

Grundlegendes Paket

Der Feature-Plan mit den neuesten Entwicklungen bei Benutzerpools. Der Essentials-Plan beinhaltet nicht die Sicherheitsfunktionen für automatisiertes Lernen im Plus-Plan.

Externer Anbieter, Drittanbieter

Ein IdP, der eine Vertrauensbeziehung zu einem Benutzerpool unterhält. Benutzerpools dienen als Zwischeneinheit zwischen externen Anbietern und Ihrer Anwendung und verwalten Authentifizierungsprozesse mit SAML 2.0, OIDC und sozialen Anbietern. Benutzerpools konsolidieren die Authentifizierungsergebnisse externer Anbieter in einem einzigen IdP, sodass Ihre Anwendungen viele Benutzer mit einer einzigen OIDC-Rely-Party-Bibliothek verarbeiten können.

Funktionsplan

Die Gruppe von Funktionen, die Sie für einen Benutzerpool auswählen können. Bei Funktionsplänen fallen in Ihrer AWS Rechnung unterschiedliche Kosten an. Für neue Benutzerpools wird standardmäßig der Essentials-Plan verwendet.

Aktuelle Pläne

- Lite-Plan
- Grundlegendes Paket
- Plus-Abo

Verbundbenutzer, externer Benutzer

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

Gehostete Benutzeroberfläche (klassisch), gehostete UI-Seiten

Die frühe Version der Authentifizierungs-Front-End-, Relying Party- und Identity Provider-Dienste in Ihrer Benutzerpool-Domain. Die gehostete Benutzeroberfläche verfügt über grundlegende Funktionen und ein vereinfachtes Erscheinungsbild. Sie können das Branding der gehosteten Benutzeroberfläche anwenden, indem Sie eine Logo-Image-Datei und eine Datei mit einem vordefinierten Satz von CSS-Stilen hochladen. Vergleiche mit verwalteter Anmeldung.

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.

Lite-Paket

Der Funktionsplan mit den Funktionen, die ursprünglich mit Benutzerpools eingeführt wurden. Der Lite-Plan beinhaltet weder die neuen Funktionen des Essentials-Plans noch die Sicherheitsfunktionen für automatisiertes Lernen im Plus-Plan.

Verwalteter Autorisierungsserver, gehosteter UI-Autorisierungsserver, Autorisierungsserver

Eine Komponente der verwalteten Anmeldung, die Dienste für die Interaktion mit IdPs und Apps in Ihrer Benutzerpool-Domain hostet. Die gehostete Benutzeroberfläche unterscheidet sich von der verwalteten Anmeldung durch die von ihr angebotenen benutzerinteraktiven Funktionen, verfügt jedoch über dieselben Funktionen für den Autorisierungsserver.

Verwaltete Anmeldung, verwaltete Anmeldeseiten

Eine Reihe von Webseiten in Ihrer Benutzerpool-Domain, auf denen Dienste für die Benutzerauthentifizierung gehostet werden. Diese Dienste umfassen Funktionen für den Betrieb als IdP, als vertrauende Partei für Dritte IdPs und als Server einer benutzerinteraktiven Authentifizierungsschnittstelle. Wenn Sie eine Domain für Ihren Benutzerpool einrichten, stellt Amazon Cognito alle verwalteten Anmeldeseiten online.

Ihre Anwendung importiert OIDC-Bibliotheken, die die Browser der Benutzer aufrufen und sie für die Registrierung, Anmeldung, Passwortverwaltung und andere Authentifizierungsvorgänge zur verwalteten Anmelde-Benutzeroberfläche weiterleiten. Nach der Authentifizierung können die OIDC-Bibliotheken das Ergebnis der Authentifizierungsanfrage verarbeiten.

Verwaltete Anmeldeauthentifizierung

Melden Sie sich mit den Diensten in Ihrer Benutzerpool-Domain an, indem Sie benutzerinteraktive Browserseiten oder HTTPS-API-Anfragen verwenden. Anwendungen verwalten die verwaltete Anmeldeauthentifizierung mit OpenID Connect (OIDC) -Bibliotheken. Dieser Prozess umfasst die Anmeldung bei externen Anbietern, die Anmeldung für lokale Benutzer mit interaktiven verwalteten Anmeldeseiten und die M2M-Autorisierung. Die Authentifizierung mit der klassischen gehosteten Benutzeroberfläche fällt ebenfalls unter diesen Begriff.

Vergleiche mit der AWS SDK-Authentifizierung.

Plus-Plan

Der Funktionsplan mit den neuesten Entwicklungen und erweiterten Sicherheitsfunktionen in Benutzerpools.

SDK-Authentifizierung, AWS SDK-Authentifizierung

Eine Reihe von API-Vorgängen für Authentifizierung und Autorisierung, die Sie Ihrem Anwendungs-Backend mit einem AWS SDK hinzufügen können. Dieses Authentifizierungsmodell erfordert Ihren eigenen, maßgeschneiderten Anmeldemechanismus. Die API kann lokale Benutzer und verknüpfte Benutzer anmelden.

Vergleichen Sie mit der verwalteten Anmeldeauthentifizierung.

Schutz vor Bedrohungen, erweiterte Sicherheitsfunktionen

In Benutzerpools bezieht sich der Begriff Bedrohungsschutz auf Technologien, die darauf ausgelegt sind, Bedrohungen für Ihre Authentifizierungs- und Autorisierungsmechanismen zu minimieren. Adaptive Authentifizierung, Erkennung kompromittierter Anmeldeinformationen und Blocklisten für IP-Adressen gehören zur Kategorie Bedrohungsschutz.

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

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, auch solche von Drittanbietern IdPs, 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.

Erste Schritte mit AWS

Bevor Sie mit der Arbeit mit Amazon Cognito beginnen, sollten Sie sich mit einigen erforderlichen AWS Ressourcen vertraut machen. Wenn Sie sich bereits bei einem anmelden können AWS-Konto, können Sie diesen Abschnitt überspringen. Lesen Sie weiter, wenn Sie nach Informationen zur Registrierung und Anmeldung mit AWS Anmeldeinformationen suchen. Sobald Sie über Anmeldeinformationen mit ausreichenden AWS Identity and Access Management (IAM-) Berechtigungen verfügen, können Sie mit Benutzerpools und Identitätspools beginnen.

Melden Sie sich an für ein AWS-Konto

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

Um sich für eine anzumelden AWS-Konto

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

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

   Wenn Sie sich für eine anmelden AWS-Konto, wird eine Root-Benutzer des AWS-Kontoserstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Als bewährte Sicherheitsmethode weisen Sie einem Administratorbenutzer Administratorzugriff zu und verwenden Sie nur den Root-Benutzer, um Aufgaben auszuführen, die Root-Benutzerzugriff erfordern.

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

Erstellen eines Benutzers mit Administratorzugriff

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

Sichern Sie Ihre Root-Benutzer des AWS-Kontos

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

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

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

Erstellen eines Benutzers mit Administratorzugriff

1. Aktivieren Sie das IAM Identity Center.

   Anweisungen finden Sie unter Aktivieren AWS IAM Identity Center im AWS IAM Identity Center Benutzerhandbuch.
2. Gewähren Sie einem Administratorbenutzer im IAM Identity Center Benutzerzugriff.

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

Anmelden als Administratorbenutzer

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

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

Weiteren Benutzern Zugriff zuweisen

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

   Anweisungen hierzu finden Sie unter Berechtigungssatz erstellen im AWS IAM Identity Center Benutzerhandbuch.

2. Weisen Sie Benutzer einer Gruppe zu und weisen Sie der Gruppe dann Single Sign-On-Zugriff zu.

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

Erste Schritte mit Benutzerpools

Sie haben eine Anwendung, die Authentifizierung und Zugriffskontrolle erfordert. Sie können innerhalb des OpenID Connect (OIDC) -Frameworks für Single Sign-On (SSO) arbeiten. Amazon Cognito verfügt über Tools zur Handhabung der Authentifizierungslogik im Anwendungs-Backend mit einem AWS SDK und zum Aufrufen eines Browsers in Ihrem Client für den Zugriff auf einen verwalteten Autorisierungsserver.

Die Amazon Cognito Cognito-Konsole führt Sie anhand Ihres bevorzugten Anwendungs-Frameworks durch die Erstellung eines Benutzerpools. Von dort aus können Sie weitere Funktionen wie die föderierte Anmeldung mit externen sozialen Anbietern oder SAML 2.0-Identitätsanbietern hinzufügen (). IdPs Die Anwendungsmodelle in der Amazon Cognito Cognito-Konsole basieren auf dem Hinzufügen von OIDC-Bibliotheken zu Ihrem Projekt und dem Aufrufen eines Browsers.

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.

Die Beispiele in diesem Kapitel und in der Amazon Cognito Cognito-Konsole zeigen 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 Funktionen zu übernehmen und mit ihnen zu interagieren IdPs.

Wenn Sie die verwalteten Anmeldeseiten nicht verwenden möchten, können Sie mithilfe eines AWS SDK oder AWS Amplify eine Anwendung mit benutzerdefinierten Authentifizierungsschnittstellen erstellen. Anwendungen, die Sie auf diese Weise erstellen, interagieren mit der Benutzerpools-API und eignen sich nur für die Authentifizierung lokaler Benutzer. Weitere Informationen zu diesem Authentifizierungsmodell finden Sie unterAndere Anwendungsoptionen.

Themen

- Erstellen Sie eine neue Anwendung in der Amazon Cognito Cognito-Konsole
- Andere Anwendungsoptionen
- Fügen Sie Ihrem Benutzerpool weitere Funktionen und Sicherheitsoptionen hinzu

Erstellen Sie eine neue Anwendung in der Amazon Cognito Cognito-Konsole

Benutzerpools fügen Authentifizierungsoptionen zu Softwareanwendungen hinzu. Um den Einstieg zu erleichtern, rufen Sie die Amazon Cognito Cognito-Konsole auf und folgen Sie den dortigen Anweisungen. Der dortige Erstellungsprozess führt Sie nicht nur durch die Einrichtung der Benutzerpool-Ressourcen, sondern auch durch die Einrichtung der ersten Teile Ihrer Anwendung.

Wenn Sie bereit sind, zu beginnen, navigieren Sie zur Amazon Cognito Cognito-Konsole und wählen Sie die Schaltfläche, um einen neuen Benutzerpool zu erstellen. Der Einrichtungsprozess führt Sie durch Ihre Konfiguration und die Optionen für die Programmiersprache.

Zusätzliche Ressourcen für Authentifizierungskonzepte

- Authentifizierung mit Amazon Cognito Cognito-Benutzerpools
- Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten
- So funktioniert die Authentifizierung mit Amazon Cognito
- Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps

Um Amazon Cognito Cognito-Ressourcen für Ihre Anwendung zu erstellen

1. Navigieren Sie zur Amazon-Cognito-Konsole. Informationen zum Zuweisen von Berechtigungen für Ihren IAM-Prinzipal, sodass er Amazon Cognito Cognito-Ressourcen erstellen und verwalten kann, finden Sie unter. AWS verwaltete Richtlinien für Amazon Cognito Die `AmazonCognitoPowerUser` Richtlinie ist ausreichend für die Erstellung von Benutzerpools.
2. Wählen Sie im Menü Benutzerpools die Option Benutzerpool erstellen aus, oder wählen Sie In weniger als fünf Minuten kostenlos loslegen aus.
3. Wählen Sie unter Anwendung definieren den Anwendungstyp aus, der am besten zu dem Anwendungsszenario passt, für das Sie Authentifizierungs- und Autorisierungsdienste erstellen möchten.
4. Geben Sie im Feld Benennen Sie Ihre Anwendung einen aussagekräftigen Namen ein, oder fahren Sie mit dem Standardnamen fort.
5. Sie müssen unter Optionen konfigurieren, die Einstellungen unterstützen, die Sie nach der Erstellung Ihres Benutzerpools nicht mehr ändern können, einige grundlegende Optionen treffen.

a. Teilen Sie uns unter Optionen für Anmeldekennungen mit, wie Sie Benutzer identifizieren möchten, wenn sie sich anmelden. Sie können benutzergenerierte Benutzernamen, E-Mail-Adressen oder Telefonnummern bevorzugen. Sie können auch eine Kombination mehrerer Optionen zulassen. Amazon Cognito akzeptiert die Optionen, die Sie hier im Feld Benutzername der verwalteten Anmeldeformulare konfigurieren.

b. Teilen Sie uns unter Erforderliche Attribute für die Registrierung mit, welche Benutzerinformationen Sie sammeln möchten, wenn sich Benutzer für ein neues Konto registrieren. Auf verwalteten Anmeldeseiten zeigt Amazon Cognito Eingabeaufforderungen für alle erforderlichen Attribute an.

   Die Optionen für Anmelde-Identifikatoren beeinflussen Ihre erforderlichen Attribute. Der Benutzername erfordert E-Mail- oder Telefonattribute für jeden Benutzer, damit dieser einen Code zum Zurücksetzen des Passworts in einer E-Mail- oder SMS-Nachricht erhalten kann. E-Mail erfordert das E-Mail-Attribut und Telefonnummer erfordert das Telefonnummer-Attribut.

6. Geben Sie unter Rückgabe-URL hinzufügen einen Weiterleitungspfad zu Ihrer Anwendung ein, der nach Abschluss der Benutzerauthentifizierung verfügbar ist. Dieser Speicherort sollte eine Route in Ihrer Anwendung sein, die OpenID Connect (OIDC) -Bibliotheken zur Verarbeitung von Benutzerauthentifizierungsergebnissen verwendet. Ein Beispiel für eine Rückgabe-URL für eine Testanwendung ist. `https://localhost:3000/callback` In der NodeJS-Beispielanwendung in der Amazon Cognito Cognito-Konsole verwendet diese Route openid-client, um das Zugriffstoken zu sammeln und es gegen Benutzerinformationen einzulösen. Nachdem Sie Ihre Ressourcen erstellt haben, können Sie nach Beispielen für Ihre Entwicklungsplattform suchen.

7. Wählen Sie Create your application aus. Amazon Cognito erstellt einen Benutzerpool und einen App-Client mit Standardeinstellungen für Ihren Anwendungstyp. Sie können zusätzliche Optionen wie externe Identitätsanbieter und Multi-Faktor-Authentifizierung (MFA) konfigurieren, nachdem Sie Ihre ersten Ressourcen erstellt haben.

8. Auf der Seite „Anwendung einrichten“ finden Sie sofort Codebeispiele für Ihre Anwendung. Um Ihren neuen Benutzerpool zu erkunden, scrollen Sie nach unten und wählen Sie Gehe zur Übersicht aus.

9. Um weitere Anwendungen zu demselben Benutzerpool hinzuzufügen, navigieren Sie zum Menü App-Clients und fügen Sie einen neuen App-Client hinzu. Dadurch wird der Prozess der anwendungsorientierten Erstellung wiederholt, dem vorhandenen Benutzerpool wird jedoch nur ein neuer App-Client hinzugefügt.

Nachdem Sie mit diesem Prozess einen Benutzerpool und einen oder mehrere App-Clients erstellt haben, können Sie mit dem Testen von Authentifizierungsvorgängen mit verwalteter Anmeldung beginnen. Diese Schnellstartoptionen stehen für die öffentliche Selbstregistrierung zur Verfügung. Wir empfehlen, dass Sie eine Testumgebung mit dem Konsolenprozess erstellen und Ihr fertiges Design dann in die Produktion überführen. Nehmen Sie sich Zeit, um sich mit den Funktionen von Amazon Cognito vertraut zu machen. Um dann zu Produktionsworkloads überzugehen, benutzerdefinierte Konfigurationen zu erstellen und diese mit Automatisierungstools wie AWS CloudFormation und dem bereitzustellen. AWS Cloud Development Kit (AWS CDK)

Amazon Cognito nimmt in diesem Prozess einige Standardkonfigurationen vor, die Sie nicht rückgängig machen können. Weitere Informationen zu Benutzerpool-Einstellungen, die Sie nicht ändern können, und zu den Optionen, die Sie in der Konsole auswählen können, finden Aktualisierung der Benutzerpool- und App-Client-Konfiguration Sie unter.

Einstellung	Auswirkung	Wie kann ich das ändern	Weitere Informationen
Clientschlüssel	Erfordert einen geheimen Client-Hash für Authentifizierungsanfragen.	Erstellen Sie einen neuen App-Client mit einer herkömmlichen Webanwendung oder einem herkömmlichen Machine-to-machine Anwendungsprofil.	Anwendungsspezifische Einstellungen mit App-Clients
Bevorzugter Benutzername	Der Benutzerpool akzeptiert das `preferred_username` Attribut nicht als Alias.	Erstellen Sie programmgesteuert einen Benutzerpool mit einem AWS SDK.	Anpassen von Anmeldeattributen
Groß-/Kleinschreibung	Bei Benutzerpool-Benutzernamen wird nicht zwischen Groß- und Kleinschreibung unterschieden, `JohnDe` s wird beispielsweise davon ausgegang	Erstellen Sie einen Benutzerpool programmgesteuert mit einem SDK. AWS	Berücksichtigung der Groß-/Kleinschreibung im Benutzerpool

Einstellung	Auswirkung	Wie kann ich das ändern	Weitere Informationen
	en, dass es sich um denselben Benutzer handelt wie. `johnd`		

Andere Anwendungsoptionen

Möglicherweise verfügen Sie über eine bestehende Anwendungsbenutzeroberfläche, die Sie in die Amazon Cognito Cognito-Authentifizierung integrieren möchten. Möglicherweise haben Sie sogar Ihre eigenen vorhandenen Authentifizierungsseiten mit einer weniger funktionalen Verzeichniseinrichtung als Amazon Cognito Cognito-Benutzerpools. Sie können einer Anwendung dieses Typs mit Amazon Cognito Cognito-Integrationen AWS SDKs für eine Vielzahl von Programmiersprachen eine Authentifizierungskomponente hinzufügen oder diese ersetzen. Im Folgenden sind einige Beispiele aufgeführt.

Wenn Sie zu diesem Zweck in der Amazon Cognito Cognito-Konsole einen Benutzerpool erstellen, ist es möglicherweise nicht erforderlich, eine Benutzerpool-Domain zu haben, die interaktive Anmeldeseiten und OpenID Connect (OIDC) -Dienste hostet. Bei der Erstellung des Benutzerpools in der Konsole wird automatisch eine Domain für Sie generiert. Sie können diese Domäne auf der Registerkarte „Domäne“ Ihres Benutzerpools löschen. Zu den weiteren Optionen gehört die programmatische Erstellung von Amazon Cognito Cognito-Ressourcen für Ihre Anwendung mit API-Anfragen in AWS SDKs und mit den Optionen für die automatische Einrichtung in der CLI. AWS Amplify Weitere Informationen finden Sie unter Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps.

Themen

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

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.

Um diese Anwendung einzurichten, muss Ihr Benutzerpool die folgenden Anforderungen erfüllen:

- 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: Die Option 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 einer Anwendung

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

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/
```

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

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

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

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

   b. `YOUR_COGNITO_USER_POOL_ID`Ersetzen Sie es durch die ID des Benutzerpools, den Sie zum Testen bestimmt haben. 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. 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 unter Plattform auswählen die Option Linux/Unix.
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 `node -v` das Programm aus, `npm -v` um zu bestätigen, 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.

Um diese Anwendung einzurichten, muss Ihr Benutzerpool die folgenden Anforderungen erfüllen:

- 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: Die Option 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 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 den IDs Ihres Benutzerpools und App-Clients.
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.

Fügen Sie Ihrem Benutzerpool weitere Funktionen und Sicherheitsoptionen hinzu

Nachdem Sie die Anleitungen zur Fertigstellung der Beispielanwendungen befolgt haben, können Sie den Umfang Ihrer Benutzerpool-Implementierung erweitern. Oder, falls Sie keine Testanwendung erstellt haben, erstellen Sie einen neuen Benutzerpool nach Ihren Wünschen. Sie können die Funktionen des Benutzerpools 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.

Wenn Ihre nächste Priorität darin besteht, die Sicherheitsoptionen für Anwendungen in Ihren Benutzerpools zu untersuchen und anzuwenden, finden Sie weitere Informationen unterBewährte Sicherheitsmethoden für Amazon-Cognito-Benutzerpools.

Amazon Cognito bietet Funktionspläne, die Funktions- und Sicherheitsoptionen hinzufügen, wenn Sie sich für höhere Stufen entscheiden. Sie können mit dem Lite-Plan beginnen, erweiterte Authentifizierungs- und Autorisierungsoptionen mit dem Essentials-Plan hinzufügen und mit dem Plus-Plan Sicherheitsvorkehrungen für automatisiertes Denken hinzufügen. Weitere Informationen finden Sie unter Funktionspläne für Benutzerpools.

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

- Branding auf verwaltete Anmeldeseiten anwenden
- Hinzufügen der MFA zu einem Benutzerpool
- Erweiterte Sicherheit mit Bedrohungsschutz
- Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern
- Verwendung von Amazon Pinpoint für Benutzerpool-Analysen

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

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 den AWS Management Console und den Benutzerpool verwenden SDKs, sondern auch Ihre Benutzerpools verwalten, indem Sie den verwenden AWS Command Line Interface.

Themen

- Fügen Sie Ihrem Benutzerpool Social Sign-In hinzu
- Fügen Sie einen SAML 2.0-Identitätsanbieter hinzu

Fügen Sie Ihrem Benutzerpool Social Sign-In hinzu

Wenn Sie Benutzern die Möglichkeit geben, sich über ihre bestehenden öffentlichen oder sozialen Identitätsanbieter bei Ihrer Anwendung anzumelden, können Sie ihre Authentifizierungserfahrung verbessern. Amazon Cognito Cognito-Benutzerpools lassen sich in beliebte Anbieter für soziale Identitäten (IdPs) wie Facebook, Google, Amazon und Apple integrieren und bieten Ihren Benutzern bequeme Anmeldeoptionen, mit denen sie bereits vertraut sind.

Wenn Sie die Anmeldung über soziale Netzwerke einrichten, bieten Sie Ihren Benutzern eine Alternative zur Erstellung eines speziellen Kontos nur für Ihre Anwendung. Dies kann die Konversionsraten verbessern und den Anmeldevorgang reibungsloser gestalten. Aus Sicht des Benutzers können sie ihre vorhandenen sozialen Anmeldeinformationen verwenden, um sich schnell zu authentifizieren, ohne sich mühsam einen anderen Benutzernamen und ein anderes Passwort merken zu müssen.

Die Konfiguration eines sozialen IdP in Ihrem Benutzerpool umfasst einige wichtige Schritte. Sie müssen Ihre Anwendung beim sozialen Anbieter registrieren, um eine Client-ID und ein geheimes Passwort zu erhalten. Anschließend können Sie Ihrem Benutzerpool die Konfiguration für soziale IdP hinzufügen und dabei die Bereiche angeben, die Sie anfordern möchten, und die Benutzerpool-Attribute, die Sie anhand von IdP-Attributen zuordnen möchten. Zur Laufzeit wickelt Amazon Cognito den Token-Austausch mit dem Anbieter ab, ordnet Benutzerattribute zu und gibt Tokens für Ihre Anwendung im Format des gemeinsamen Benutzerpools aus.

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 Sie noch keine Facebook-App haben, wird Ihnen 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 Änderungen speichern aus.
11. Geben Sie für App Domains (App-Domänen) Ihre Benutzerpool-Domäne ein.

    ```
    https://your_user_pool_domain
    ```

12. Wählen Sie Änderungen speichern aus.
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.

    Gib deine Weiterleitungs-URL in das Feld Gültige OAuth Weiterleitung ein URIs. Die Weiterleitungs-URL besteht aus Ihrer Benutzerpool-Domain mit dem `/oauth2/idpresponse` Endpunkt.

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

15. Wählen Sie Änderungen speichern aus.

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-Domain mit dem `/oauth2/idpresponse` Endpunkt in das Feld Allowed Return ein URLs.

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

11. Wählen Sie Speichern.

Eine App bei Google registrieren

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

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 Dienste APIs und dann 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. 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 aus.

10. Erweitern Sie die linke Navigationsleiste erneut, wählen Sie Services APIs 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 Autorisierte Weiterleitung URIs die Option 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 Konfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP ausgewählt haben.
   3. Wählen Sie unter Funktionen die Option Mit Apple anmelden und dann Bearbeiten aus.
   4. Wählen Sie auf der Seite „Mit Apple anmelden: App-ID-Konfiguration“ aus, ob Sie die App entweder als primäre App oder als Gruppe mit einer anderen App einrichten möchten IDs, und wählen Sie dann Speichern aus.
   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 ID registrieren“ die Option Dienste 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 habenKonfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP.
   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 das Pluszeichen neben Website URLs.
   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 URLs unter Zurück den Pfad zum `/oauth2/idpresponse` Endpunkt Ihrer Benutzerpool-Domain ein.

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

   6. Wählen Sie Weiter und dann 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 „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 Konfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP 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 das Menü Soziale Netzwerke und externe Anbieter. 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 Erstellen aus.
10. Wählen Sie im Menü App-Clients einen der App-Clients in der Liste aus und bearbeiten Sie die Einstellungen für die gehostete Benutzeroberfläche. Fügen Sie unter Identity providers (Identitätsanbieter) den neuen Social-Identity-Anbieter zum App-Client hinzu.
11. Wählen Sie Änderungen speichern aus.

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.auth.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 storniert Authentifizierungsanfragen, die nicht innerhalb von 5 Minuten abgeschlossen werden, und leitet den Benutzer zur verwalteten Anmeldung weiter. Für die Seite wird eine `Something went wrong`-Fehlermeldung angezeigt.

Fügen Sie einen SAML 2.0-Identitätsanbieter hinzu

Ihre App-Benutzer können sich mit einem SAML 2.0-Identitätsanbieter (IdP) anmelden. IdPs Wenn es sich bei Ihren Kunden um interne Kunden oder verbundene Unternehmen Ihrer Organisation handelt, können Sie SAML 2.0 den sozialen Netzwerken IdPs vorziehen. Wenn ein sozialer IdP es allen Benutzern ermöglicht, sich für ein Konto zu registrieren, ist es wahrscheinlicher, dass ein SAML-IdP mit einem Benutzerverzeichnis verknüpft wird, das von Ihrer Organisation kontrolliert wird. 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 Verwendung 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 im Domain-Menü 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 im Übersichts-Dashboard für Ihren Benutzerpool in der Amazon Cognito 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 das Menü Soziale Netzwerke und externe Anbieter. 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 bei der

Konfiguration der verwalteten Anmeldung erstellt wird. 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 Erstellen aus.

Erste Schritte mit Amazon Cognito Cognito-Identitätspools

Mit Amazon-Cognito-Identitätspools können Sie eindeutige Identitäten erstellen und Benutzern Berechtigungen zuweisen. Ihr Identitätspool kann Identitäten aus den folgenden Arten von Authentifizierungsdiensten einbeziehen:

- 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

Nachdem sich Benutzer bei ihrem Anbieter authentifiziert und die Autorisierung für einen Identitätspool vorgelegt haben, erhalten sie temporäre AWS Anmeldeinformationen. Die Anmeldeinformationen von Benutzern verfügen über Berechtigungen, die Sie für den Zugriff auf andere AWS-Services Benutzer definieren.

Themen

- Erstellen eines Identitätspools in Amazon Cognito
- Einrichten eines SDK
- Integrieren der Identitätsanbieter
- Abrufen von Anmeldeinformationen

Erstellen eines Identitätspools in Amazon Cognito

Sie können einen Identitätspool über die Amazon Cognito-Konsole erstellen, oder Sie können die AWS Command Line Interface (CLI) oder Amazon APIs Cognito verwenden. Das folgende Verfahren ist eine allgemeine Anleitung zum Erstellen eines neuen Identitätspools in der Konsole. Sie können auch direkt zur Konsole springen und den Anleitungen und den Inhalten der Inline-Hilfe folgen.

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. Informationen zum Zuweisen von Berechtigungen für Ihren IAM-Prinzipal, sodass er Amazon Cognito Cognito-Ressourcen erstellen und verwalten kann, finden Sie unter. AWS verwaltete

Richtlinien für Amazon Cognito Die `AmazonCognitoPowerUser` Richtlinie ist ausreichend für die Erstellung von Identitätspools.

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 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 und direkt IdPs anrufen. AssumeRoleWithWebIdentity Weitere Informationen finden Sie unter Identitäten-Pools – Authentifzierungsablauf.

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 für Java, den oder den SDK für .NET ein. Weitere Informationen finden Sie unter den folgenden Themen.

- Die Einrichtung des SDK für finden Sie JavaScript im AWS SDK für JavaScript Entwicklerhandbuch
- Amplify-Dokumentation im Amplify Dev Center
- Anbieter von Amazon-Cognito-Anmeldeinformationen im SDK für .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, 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 Identitätspools, Identitätsanbieter von Drittanbietern.

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

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
- Amazon Cognito Cognito-Workshop
- Workshop zur Fehlerbehebung bei Amazon Cognito
- Authentifizierung und Autorisierung mit Amazon Cognito und verifizierten Berechtigungen
- Amazon Cognito JWT im Detail

Blog-Beiträge

- Schützen Sie öffentliche Kunden für Amazon Cognito mithilfe eines Amazon-Proxys CloudFront
- So richten Sie Amazon Cognito für die Verbundauthentifizierung mit Azure AD ein
- Vereinfachen Sie die Web-App-Authentifizierung: Ein Leitfaden für den AD FS-Verbund mit Amazon Cognito Cognito-Benutzerpools

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

Die Implementierung von Amazon Cognito ist eine Mischung aus AWS Management Console AWS SDK-Verwaltungstools und SDK-Bibliotheken in Anwendungen. Die Amazon-Cognito-Konsole ist die visuelle Oberfläche für die Einrichtung und Verwaltung Ihrer Amazon-Cognito-Benutzerpools und -Identitätspools.

Die Integration mit Amazon Cognito Cognito-Benutzerpools mit dem geringsten Aufwand ist die verwaltete Anmeldung. Managed Login ist eine ready-to-use webbasierte Anmeldeanwendung zum schnellen Testen und Bereitstellen von Amazon Cognito Cognito-Benutzerpools. Die Benutzerpoolauthentifizierung mit verwalteter Anmeldung erfordert OpenID Connect (OIDC) -Bibliotheken, die Benutzer zu gehosteten Anmeldeseiten weiterleiten. In dieser Reihe von benutzerinteraktiven und umgeleiteten Web-Endpunkten kümmert sich Amazon Cognito um den Authentifizierungsfluss, einschließlich der Anmeldung durch Dritte, der Multi-Faktor-Authentifizierung (MFA) und der Auswahl eines Authentifizierungsablaufs. Ihre Anwendung muss nur das Authentifizierungsergebnis verarbeiten, das Amazon Cognito in der Antwort zurückgibt.

Sie können Ihrer Anwendung auch ein AWS SDK hinzufügen, benutzerdefinierte Authentifizierungsschnittstellen erstellen und API-Operationen zur Authentifizierung und Autorisierung Ihrer Benutzer aufrufen. AWS Amplifyist AWS-Service für die Erstellung von Full-Stack-Anwendungen mit Amazon Cognito Cognito-Authentifizierung im Backend vorgesehen.

Beispielsweise könnte Ihre App die verwaltete Anmeldung 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. Identitätspools verfügen nicht über dieselben verwalteten Authentifizierungsoptionen wie Benutzerpools. Implementieren Sie für den Zugriff auf AWS Anmeldeinformationen in Ihren Anwendungen Identitätspools-Operationen in importierten SDK-Modulen. Um mit Ihrem eigenen benutzerdefinierten Anwendungscode zu beginnen, besuchen Sie die Amazon Cognito Cognito-Codebeispiele für. AWS SDKs 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. Informationen zu den verfügbaren Plattformen für finden Sie AWS SDKs unter. Authentifizierung mit AWS SDKs Das AWS CLI ist ein Befehlszeilen-SDK für Amazon Cognito und andere. Es ist ein wertvoller Ausgangspunkt AWS-Services, um sich mit den Amazon Cognito Cognito-API-Operationen und deren Syntax 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 Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten.

Themen

- Authentifizierung mit AWS Amplify
- Authentifizierung mit AWS SDKs
- So funktioniert die Authentifizierung mit Amazon Cognito
- Verwenden Sie diesen Dienst mit einem AWS SDK
- Autorisierung mit Amazon Verified Permissions

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 React
- Amplify Sie die Authentifizierung für React Native
- Amplify Sie die Authentifizierung für Swift (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

Vom Benutzerpool verwaltete Anmeldungkann die grundlegenden Anforderungen eines Authentifizierungs-Frontends für eine Web- oder mobile App erfüllen. Um Ihre Benutzeroberfläche (UI) über die Parameter hinaus anzupassen, die die verwaltete Anmeldung unterstützt, sollten Sie eine Anwendung nach Ihren Wünschen erstellen. 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 auch Abschnitte mit Ressourcen zur Verwendung verschiedener unterstützter Plattformen. SDKs

- 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

So funktioniert die Authentifizierung mit Amazon Cognito

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

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

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 (verwaltete Anmeldung) 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

- Benutzerpool-Authentifizierung mit verwalteter Anmeldung
- API-Authentifizierung und Autorisierung des Benutzerpools mit einem SDK AWS
- Benutzerpool-Authentifizierung mit einem externen Identitätsanbieter
- Authentifizierung des Identitätspools

Benutzerpool-Authentifizierung mit verwalteter Anmeldung

Managed Login 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 durchführen. Die Implementierung einer Anwendung mit einer verwalteten Login-Komponente zur Authentifizierung kann weniger Aufwand durch Entwickler erfordern. Eine Anwendung kann Benutzeroberflächenkomponenten für die Authentifizierung überspringen und verwaltete Anmeldewebseiten im Browser des Benutzers aufrufen.

Anwendungen erfassen Benutzer JWTs mit einem Web- oder App-Weiterleitungsstandort. Anwendungen, die verwaltete Anmeldung implementieren, können sich zur Authentifizierung mit Benutzerpools verbinden, als wären sie ein OpenID Connect (OIDC) IdP.

Die verwaltete Anmeldung eignet sich für das Modell, bei dem Anwendungen die Authentifizierungsdienste eines OIDC-Autorisierungsservers benötigen, aber nicht sofort 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 Benutzerpool-Komponente für ein SDK implementieren.

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

Das folgende Diagramm zeigt eine typische Anmeldesitzung für die verwaltete Anmeldeauthentifizierung.

Ablauf der verwalteten Authentifizierung bei der Anmeldung

1. Ein Benutzer greift auf Ihre Anwendung zu.
2. Er wählt einen Link „Anmelden“ aus.
3. Die Anwendung leitet den Benutzer zu einer Anmeldeaufforderung auf den verwalteten Anmeldeseiten Ihrer Benutzerpool-Domain 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. Auf der verwalteten Anmeldeseite wird der Benutzer aufgefordert, einen MFA-Code einzugeben.
7. Der Benutzer gibt seinen MFA-Code ein.
8. Ihr Benutzerpool leitet den Benutzer zur Anwendungs-URL weiter.
9. Die Anwendung erfasst den Autorisierungscode aus dem URL-Anforderungsparameter, der die verwaltete Anmeldung an die Callback-URL angehängt hat.
10. Die Anwendung fordert Token mit dem Autorisierungscode an.
11. Der Token-Endpunkt kehrt JWTs zur Anwendung zurück.
12. Die Anwendung dekodiert, validiert und speichert oder zwischenspeichert die Daten des Benutzers. JWTs
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 Ihrer verwalteten Anmeldeseiten mit dem Branding-Editor für Ihren gesamten Benutzerpool oder auf der Ebene eines beliebigen App-Clients anpassen. Sie können App-Clients auch mit eigenen Identitätsanbietern, Bereichen, Zugriff auf Benutzerattributen und erweiterter Sicherheitskonfiguration konfigurieren.

Zugehörige Ressourcen

- Vom Benutzerpool verwaltete Anmeldung
- Bereiche, M2M und APIs mit Ressourcenservern
- Benutzerpool-Endpunkte und Referenz für verwaltete Logins

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 darin integrierten Methoden SDKs rufen die Amazon Cognito-Benutzerpools-API auf. Derselbe API-Namespace für Benutzerpools enthält Operationen zur Konfiguration von Benutzerpools und zur Benutzerauthentifizierung. Eine ausführlichere Übersicht finden Sie unterGrundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten.

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 kann sie dekodiert und verwendet werden. JWTs 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.

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 dem des Benutzers. JWTs
11. Die Anwendung dekodiert, validiert und speichert oder zwischenspeichert die Daten des Benutzers. JWTs
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
- Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten

Benutzerpool-Authentifizierung mit einem externen Identitätsanbieter

Die Anmeldung mit einem externen Identitätsanbieter (IdP) oder die Verbundauthentifizierung ist ein ähnliches Modell wie die verwaltete Anmeldung. 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 die Anmeldung im Browser des Benutzers zu verwalten, 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 verwalteten Anmeldeauthentifizierung sammelt eine Anwendung Daten 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 eine verwaltete Anmeldeauthentifizierung implementiert hat. Tatsächlich IdPs führen verwaltete Anmeldedaten und Drittanbieter zu einem konsistenten Authentifizierungsergebnis, wenn nur geringfügige Abweichungen in den Browsern der Benutzer aufgerufen werden.

Wie die verwaltete Anmeldeauthentifizierung eignet sich auch die Verbundauthentifizierung am besten für erweiterte Autorisierungsmodelle mit Geltungsbereich OAuth 2.0.

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 kehrt JWTs zur Anwendung zurück.
13. Die Anwendung dekodiert, validiert und speichert oder zwischenspeichert die Daten des Benutzers. JWTs
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 über die verwaltete Anmeldung initiieren, bei der Benutzer aus einer Liste auswählen können IdPs , die Sie Ihrem App-Client zugewiesen haben. Die verwaltete Anmeldung kann auch nach einer E-Mail-Adresse fragen und die Anfrage eines Benutzers automatisch an den entsprechenden SAML-IdP weiterleiten. Die Authentifizierung mit einem externen Identitätsanbieter erfordert keine Benutzerinteraktion bei der verwalteten Anmeldung. Ihre Anwendung kann der Autorisierungsserveranforderung eines Benutzers einen Anforderungsparameter hinzufügen und den Benutzer veranlassen, unbemerkt auf seine IdP-Anmeldeseite umzuleiten.

Zugehörige Ressourcen

- Benutzerpool-Anmeldung mit externen Identitätsanbietern
- Bereiche, M2M und APIs mit Ressourcenservern
- Benutzerpool-Endpunkte und Referenz für verwaltete Logins

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 Sie 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.

Ablauf der Authentifizierung im Identitätspool

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 Assertion des Benutzers. JWTs
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) antwortet auf 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“ übersprungen, 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-Pools – Authentifzierungsablauf

Verwenden Sie diesen Dienst mit einem AWS SDK

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 für C++	AWS SDK für C++ Codebeispiele
AWS CLI	AWS CLI Code-Beispiele
AWS SDK für Go	AWS SDK für Go Code-Beispiele

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

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 haben`IsAuthorizedWithToken`. 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 Anzeige JWKs einsehen`https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/.well-known/jwks.json`.

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 Ressourcen

- Zuordnen von Amazon Cognito Cognito-Token zum Schema „Verified Permissions“
- Autorisieren Sie API Gateway APIs mit Amazon Verified Permissions und Amazon Cognito
- Workshop: Authentifizierung und Autorisierung mit Amazon Cognito und verifizierte Berechtigungen

API-Autorisierung mit verifizierten Berechtigungen

Ihre ID oder Zugriffstoken können Anfragen an Backend-Backend-Amazon-API-Gateway-REST APIs mit verifizierten Berechtigungen autorisieren. Sie können einen Richtlinienspeicher mit direkten Links zu Ihrem Benutzerpool und Ihrer API erstellen. Mit der Startoption Setup with API Gateway und einer Identitätsquelle fügt Verified Permissions dem Richtlinienspeicher eine Benutzerpool-Identitätsquelle

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 Sie 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
);
```

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	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.

Service	Prozess	Ergebnis
die Zugriffsk ontrolle)		

Codebeispiele für Amazon Cognito mit AWS SDKs

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

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

Codebeispiele

- Codebeispiele für Amazon Cognito Identity mit AWS SDKs
  - Grundlegende Beispiele für die Verwendung von Amazon Cognito Identity AWS SDKs
    - Aktionen für Amazon Cognito Identity mit AWS SDKs
      - Verwendung CreateIdentityPool mit einem AWS SDK oder CLI
      - Verwendung DeleteIdentityPool mit einem AWS SDK oder CLI
      - Verwendung von DescribeIdentityPool mit einer CLI
      - GetCredentialsForIdentityMit einem AWS SDK verwenden
      - Verwendung von GetIdentityPoolRoles mit einer CLI
      - Verwendung ListIdentityPools mit einem AWS SDK oder CLI
      - Verwendung von SetIdentityPoolRoles mit einer CLI
      - Verwendung von UpdateIdentityPool mit einer CLI
  - Szenarien für die Verwendung von Amazon Cognito Identity AWS SDKs
    - Erstellen Sie eine Amazon-Textract-Explorer-Anwendung
- Codebeispiele für Amazon Cognito Identity Provider mit AWS SDKs
  - Grundlegende Beispiele für die Verwendung von Amazon Cognito Identity Provider AWS SDKs
    - Hello Amazon Cognito
    - Aktionen für Amazon Cognito Identity Provider mit 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 die Verwendung von Amazon Cognito Identity Provider 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

- Codebeispiele für Amazon Cognito Sync mit AWS SDKs
  - Grundlegende Beispiele für die Verwendung von Amazon Cognito Sync AWS SDKs
    - Aktionen für Amazon Cognito Sync mit AWS SDKs
      - Verwenden Sie es ListIdentityPoolUsage mit einem AWS SDK

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 Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarios anzeigen.

Szenarien sind Code-Beispiele, die Ihnen zeigen, wie Sie bestimmte Aufgaben ausführen, indem Sie mehrere Funktionen innerhalb eines Services aufrufen oder mit anderen AWS-Services kombinieren.

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

Codebeispiele

- Grundlegende Beispiele für die Verwendung von Amazon Cognito Identity AWS SDKs
  - Aktionen für Amazon Cognito Identity mit AWS SDKs
    - Verwendung CreateIdentityPool mit einem AWS SDK oder CLI
    - Verwendung DeleteIdentityPool mit einem AWS SDK oder CLI
    - Verwendung von DescribeIdentityPool mit einer CLI
    - GetCredentialsForIdentityMit einem AWS SDK verwenden
    - Verwendung von GetIdentityPoolRoles mit einer CLI
    - Verwendung ListIdentityPools mit einem AWS SDK oder CLI
    - Verwendung von SetIdentityPoolRoles mit einer CLI
    - Verwendung von UpdateIdentityPool mit einer CLI
- Szenarien für die Verwendung von Amazon Cognito Identity AWS SDKs
  - Erstellen Sie eine Amazon-Textract-Explorer-Anwendung

Grundlegende Beispiele für die Verwendung von Amazon Cognito Identity AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie die Grundlagen von Amazon Cognito Identity mit AWS SDKs verwenden können.

Beispiele

- Aktionen für Amazon Cognito Identity mit AWS SDKs
  - Verwendung CreateIdentityPool mit einem AWS SDK oder CLI
  - Verwendung DeleteIdentityPool mit einem AWS SDK oder CLI
  - Verwendung von DescribeIdentityPool mit einer CLI
  - GetCredentialsForIdentityMit einem AWS SDK verwenden
  - Verwendung von GetIdentityPoolRoles mit einer CLI
  - Verwendung ListIdentityPools mit einem AWS SDK oder CLI
  - Verwendung von SetIdentityPoolRoles mit einer CLI
  - Verwendung von UpdateIdentityPool mit einer CLI

Aktionen für Amazon Cognito Identity mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie einzelne Amazon Cognito Identity-Aktionen mit AWS SDKs durchführen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Diese Auszüge rufen die Amazon-Cognito–Identity-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Sie können Aktionen im Kontext unter Szenarien für die Verwendung von Amazon Cognito Identity AWS SDKs anzeigen.

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 von DescribeIdentityPool mit einer CLI
- GetCredentialsForIdentityMit einem AWS SDK verwenden
- Verwendung von GetIdentityPoolRoles mit einer CLI
- Verwendung ListIdentityPools mit einem AWS SDK oder CLI
- Verwendung von SetIdentityPoolRoles mit einer CLI
- Verwendung von UpdateIdentityPool mit einer CLI

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

Die folgenden Code-Beispiele zeigen, wie `CreateIdentityPool` verwendet wird.

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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 V4

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 CreateIdentityPool AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 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 CreateIdentityPool AWS -Tools für PowerShellCmdlet-Referenz (V5).

Swift

SDK für Swift

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

```
import AWSCognitoIdentity


    /// 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? {
        do {
            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
        } catch {
            print("ERROR: createIdentityPool:", dump(error))
            throw error
        }
    }
```

- 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 Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `DeleteIdentityPool` verwendet wird.

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 DeleteIdentityPoolin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 V4

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 für PowerShellCmdlet-Referenz (V4).

Tools für V5 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 für PowerShellCmdlet-Referenz (V5).

Swift

SDK für Swift

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

```
import AWSCognitoIdentity


    /// Delete the specified identity pool.
    ///
    /// - Parameters:
    ///   - id: The ID of the identity pool to delete.
    ///
    func deleteIdentityPool(id: String) async throws {
        do {
            let input = DeleteIdentityPoolInput(
                identityPoolId: id
            )

            _ = try await cognitoIdentityClient.deleteIdentityPool(input: input)
        } catch {
            print("ERROR: deleteIdentityPool:", dump(error))
            throw error
        }
    }
```

- 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 Dienst mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`DescribeIdentityPool`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeIdentityPool` verwendet wird.

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 V4

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 für PowerShellCmdlet-Referenz (V4).

Tools für V5 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 für PowerShellCmdlet-Referenz (V5).

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

`GetCredentialsForIdentity`Mit einem AWS SDK verwenden

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

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 Dienst mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`GetIdentityPoolRoles`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `GetIdentityPoolRoles` verwendet wird.

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 V4

Beispiel 1: Ruft die Informationen über 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 für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Ruft die Informationen über 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 für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK 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 Code-Beispiele zeigen, wie `ListIdentityPools` verwendet wird.

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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 V4

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 für PowerShellCmdlet-Referenz (V4).

Tools für V5 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 für PowerShellCmdlet-Referenz (V5).

Swift

SDK für Swift

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

```
import AWSCognitoIdentity


    /// 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? {
        let listPoolsInput = ListIdentityPoolsInput(maxResults: 25)
        // Use "Paginated" to get all the objects.
        // This lets the SDK handle the 'nextToken' field in
 "ListIdentityPoolsOutput".
        let pages = cognitoIdentityClient.listIdentityPoolsPaginated(input:
 listPoolsInput)

        do {
            for try await page in pages {
                guard let identityPools = page.identityPools else {
                    print("ERROR: listIdentityPoolsPaginated returned nil
 contents.")
                    continue
                }

                /// 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.

                for pool in identityPools {
                    if pool.identityPoolName == name {
                        return pool.identityPoolId!
                    }
                }
            }
        } catch {
            print("ERROR: getIdentityPoolID:", dump(error))
            throw error
        }

        return nil
    }
```

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

```
import AWSCognitoIdentity


    /// 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.

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

            return poolId
        } catch {
            print("ERROR: getOrCreateIdentityPoolID:", dump(error))
            throw error
        }
    }
```

- 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 Dienst mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`SetIdentityPoolRoles`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `SetIdentityPoolRoles` verwendet wird.

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 V4

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 AWS -Tools für PowerShell Cmdlet-Referenz (SetIdentityPoolRolesV4).

Tools für V5 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 AWS -Tools für PowerShell Cmdlet-Referenz (SetIdentityPoolRolesV5).

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

Verwendung von **`UpdateIdentityPool`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `UpdateIdentityPool` verwendet wird.

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 V4

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 für PowerShellCmdlet-Referenz (V4).

Tools für V5 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 für PowerShellCmdlet-Referenz (V5).

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

Szenarien für die Verwendung von Amazon Cognito Identity AWS SDKs

Die folgenden Codebeispiele zeigen Ihnen, wie Sie allgemeine Szenarien in Amazon Cognito Identity mit AWS SDKs implementieren. Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben erledigen können, indem Sie mehrere Funktionen in Amazon Cognito Identity oder in Kombination mit anderen AWS-Services aufrufen. Jedes Szenario enthält einen Link zum vollständigen Quell-Code, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Szenarien zielen auf eine mittlere Erfahrungsebene ab, um Ihnen zu helfen, Service-Aktionen im Kontext zu verstehen.

Beispiele

- Erstellen Sie eine Amazon-Textract-Explorer-Anwendung

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 für 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

Python

SDK für Python (Boto3)

Zeigt, wie Sie AWS SDK für Python (Boto3) mit Amazon Textract Text-, Formular- und Tabellenelemente in einem Dokumentbild erkennen können. Das Eingabe-Image und die Amazon-Textract-Ausgabe werden in einer Tkinter-Anwendung angezeigt, mit der Sie die erkannten Elemente untersuchen können.

- Senden Sie ein Dokument-Image an Amazon Textract und untersuchen Sie die Ausgabe erkannter Elemente.
- Senden Sie Images direkt an Amazon Textract oder über einen Amazon Simple Storage Service (Amazon S3)-Bucket.
- Verwenden Sie asynchron APIs , um einen Job zu starten, der nach Abschluss des Jobs eine Benachrichtigung in einem Amazon Simple Notification Service (Amazon SNS) -Thema veröffentlicht.
- Stellen Sie eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange ab, um eine Meldung zum Abschluss des Auftrags zu erhalten.

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 Dienst mit einem AWS SDK. 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 Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarios anzeigen.

Szenarien sind Code-Beispiele, die Ihnen zeigen, wie Sie bestimmte Aufgaben ausführen, indem Sie mehrere Funktionen innerhalb eines Services aufrufen oder mit anderen AWS-Services kombinieren.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Code für die CMake Datei CMake Lists.txt.

```
# 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 ListUserPoolsunter AWS SDK für C++ API-Referenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 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(ctx)
  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 für Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Python

SDK für Python (Boto3)

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

```
import boto3

# Create a Cognito Identity Provider client
cognitoidp = boto3.client("cognito-idp")

# Initialize a paginator for the list_user_pools operation
paginator = cognitoidp.get_paginator("list_user_pools")

# Create a PageIterator from the paginator
page_iterator = paginator.paginate(MaxResults=10)

# Initialize variables for pagination
user_pools = []

# Handle pagination
for page in page_iterator:
    user_pools.extend(page.get("UserPools", []))

# Print the list of user pools
print("User Pools for the account:")
if user_pools:
    for pool in user_pools:
        print(f"Name: {pool['Name']}, ID: {pool['Id']}")
else:
    print("No user pools found.")
```

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

Ruby

SDK für Ruby

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

```
require 'aws-sdk-cognitoidentityprovider'
require 'logger'

# CognitoManager is a class responsible for managing AWS Cognito operations
# such as listing all user pools in the current AWS account.
class CognitoManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all user pools associated with the AWS account.
  def list_user_pools
    paginator = @client.list_user_pools(max_results: 10)
    user_pools = []
    paginator.each_page do |page|
      user_pools.concat(page.user_pools)
    end

    if user_pools.empty?
      @logger.info('No Cognito user pools found.')
    else
      user_pools.each do |user_pool|
        @logger.info("User pool ID: #{user_pool.id}")
        @logger.info("User pool name: #{user_pool.name}")
        @logger.info("User pool status: #{user_pool.status}")
        @logger.info('---')
      end
    end
  end
end
```

```
if $PROGRAM_NAME == __FILE__
  cognito_client = Aws::CognitoIdentityProvider::Client.new
  manager = CognitoManager.new(cognito_client)
  manager.list_user_pools
end
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK für Ruby API-Referenz.

Codebeispiele

- Grundlegende Beispiele für die Verwendung von Amazon Cognito Identity Provider AWS SDKs
  - Hello Amazon Cognito
  - Aktionen für Amazon Cognito Identity Provider mit 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 die Verwendung von Amazon Cognito Identity Provider 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

Grundlegende Beispiele für die Verwendung von Amazon Cognito Identity Provider AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie die Grundlagen von Amazon Cognito Identity Provider mit AWS SDKs verwenden können.

Beispiele

- Hello Amazon Cognito
- Aktionen für Amazon Cognito Identity Provider mit 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

Hello Amazon Cognito

Die folgenden Codebeispiele veranschaulichen die ersten Schritte mit Amazon Cognito.

C++

SDK für C++

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

Code für die CMake Datei CMake Lists.txt.

```
# 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 ListUserPoolsunter AWS SDK für C++ API-Referenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 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(ctx)
  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 für Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Python

SDK für Python (Boto3)

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

```
import boto3

# Create a Cognito Identity Provider client
cognitoidp = boto3.client("cognito-idp")

# Initialize a paginator for the list_user_pools operation
paginator = cognitoidp.get_paginator("list_user_pools")

# Create a PageIterator from the paginator
page_iterator = paginator.paginate(MaxResults=10)
```

```
# Initialize variables for pagination
user_pools = []

# Handle pagination
for page in page_iterator:
    user_pools.extend(page.get("UserPools", []))

# Print the list of user pools
print("User Pools for the account:")
if user_pools:
    for pool in user_pools:
        print(f"Name: {pool['Name']}, ID: {pool['Id']}")
else:
    print("No user pools found.")
```

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

Ruby

SDK für Ruby

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

```
require 'aws-sdk-cognitoidentityprovider'
require 'logger'

# CognitoManager is a class responsible for managing AWS Cognito operations
# such as listing all user pools in the current AWS account.
class CognitoManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end
```

```
  # Lists and prints all user pools associated with the AWS account.
  def list_user_pools
    paginator = @client.list_user_pools(max_results: 10)
    user_pools = []
    paginator.each_page do |page|
      user_pools.concat(page.user_pools)
    end

    if user_pools.empty?
      @logger.info('No Cognito user pools found.')
    else
      user_pools.each do |user_pool|
        @logger.info("User pool ID: #{user_pool.id}")
        @logger.info("User pool name: #{user_pool.name}")
        @logger.info("User pool status: #{user_pool.status}")
        @logger.info('---')
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  cognito_client = Aws::CognitoIdentityProvider::Client.new
  manager = CognitoManager.new(cognito_client)
  manager.list_user_pools
end
```

- Einzelheiten zur API finden Sie ListUserPoolsin der AWS SDK für Ruby API-Referenz.

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

Aktionen für Amazon Cognito Identity Provider mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie einzelne Amazon Cognito Identity Provider-Aktionen mit AWS SDKs durchführen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Diese Auszüge rufen die Amazon-Cognito–Identity-Provider-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Sie können Aktionen im Kontext unter Szenarien für die Verwendung von Amazon Cognito Identity Provider AWS SDKs anzeigen.

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 Code-Beispiele zeigen, wie `AdminCreateUser` verwendet wird.

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 von benutzerdefinierten Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &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 für Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `AdminGetUser` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

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

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

```
            username = userNameVal
            userPoolId = poolIdVal
        }

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

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

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Get information about a specific user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The Amazon Cognito Identity Provider client to use.
    ///   - userName: The user to retrieve information about.
    ///   - userPoolId: The user pool to search for the specified user.
    ///
    /// - Returns: `true` if the user's information was successfully
    ///   retrieved. Otherwise returns `false`.
    func adminGetUser(cipClient: CognitoIdentityProviderClient, userName: String,
                      userPoolId: String) async -> Bool {
        do {
            let output = try await cipClient.adminGetUser(
                input: AdminGetUserInput(
                    userPoolId: userPoolId,
                    username: userName
                )
```

```
            )

            guard let userStatus = output.userStatus else {
                print("*** Unable to get the user's status.")
                return false
            }

            print("User status: \(userStatus)")
            return true
        } catch {
            return false
        }
    }
```

- Einzelheiten zur API finden Sie AdminGetUserin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `AdminInitiateAuth` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für C++ API-Referenz.

CLI

AWS CLI

Um einen Benutzer als Administrator anzumelden

Im folgenden `admin-initiate-auth` Beispiel wird der Benutzer diego@example.com angemeldet. Dieses Beispiel enthält auch Metadaten für den Bedrohungsschutz und ClientMetadata für Lambda-Trigger. Der Benutzer ist für TOTP MFA konfiguriert und erhält eine Aufforderung, einen Code von seiner Authenticator-App einzugeben, bevor er die Authentifizierung abschließen kann.

```
aws cognito-idp admin-initiate-auth \
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789 \
    --auth-flow ADMIN_USER_PASSWORD_AUTH \
    --auth-parameters USERNAME=diego@example.com,PASSWORD="My@Example
$Password3!",SECRET_HASH=ExampleEncodedClientIdSecretAndUsername= \
    --context-data="{\"EncodedData\":\"abc123example\",\"HttpHeaders\":
[{\"headerName\":\"UserAgent\",\"headerValue\":\"Mozilla/5.0 (Windows NT
 6.1; Win64; x64; rv:47.0) Gecko/20100101 Firefox/47.0\"}],\"IpAddress\":
\"192.0.2.1\",\"ServerName\":\"example.com\",\"ServerPath\":\"/login\"}" \
    --client-metadata="{\"MyExampleKey\": \"MyExampleValue\"}"
```

Ausgabe:

```
{
    "ChallengeName": "SOFTWARE_TOKEN_MFA",
    "Session": "AYABeExample...",
    "ChallengeParameters": {
        "FRIENDLY_DEVICE_NAME": "MyAuthenticatorApp",
        "USER_ID_FOR_SRP": "diego@example.com"
    }
}
```

Weitere Informationen finden Sie unter Ablauf der Administratorauthentifizierung im Amazon Cognito Developer Guide.

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

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminInitiateAuth(authRequest)
        println("Result Challenge is ${response.challengeName}")
        return response
    }
}
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Begin an authentication session.
    ///
    /// - Parameters:
    ///   - cipClient: The `CongitoIdentityProviderClient` to use.
```

```
    ///   - clientId: The app client ID to use.
    ///   - userName: The username to check.
    ///   - password: The user's password.
    ///   - userPoolId: The user pool to use.
    ///
    /// - Returns: The session token associated with this authentication
    ///   session.
    func initiateAuth(cipClient: CognitoIdentityProviderClient, clientId: String,
                      userName: String, password: String,
                      userPoolId: String) async -> String? {
        var authParams: [String: String] = [:]

        authParams["USERNAME"] = userName
        authParams["PASSWORD"] = password

        do {
            let output = try await cipClient.adminInitiateAuth(
                input: AdminInitiateAuthInput(
                    authFlow:
CognitoIdentityProviderClientTypes.AuthFlowType.adminUserPasswordAuth,
                    authParameters: authParams,
                    clientId: clientId,
                    userPoolId: userPoolId
                )
            )

            guard let challengeName = output.challengeName else {
                print("*** Invalid response from the auth service.")
                return nil
            }

            print("=====> Response challenge is \(challengeName)")

            return output.session
        } catch _ as UserNotFoundException {
            print("*** The specified username, \(userName), doesn't exist.")
            return nil
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return nil
        } catch {
            print("*** An unexpected error occurred.")
            return nil
        }
```

```
    }
```

- Einzelheiten zur API finden Sie AdminInitiateAuthin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `AdminRespondToAuthChallenge` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für C++ API-Referenz.

CLI

AWS CLI

Um auf eine Authentifizierungsherausforderung zu antworten

Abhängig von Ihrem Authentifizierungsablauf, der Konfiguration des Benutzerpools und den Benutzereinstellungen gibt es viele Möglichkeiten, auf unterschiedliche Authentifizierungsherausforderungen zu reagieren. Das folgende `admin-respond-to-auth-challenge` Beispiel stellt einen TOTP-MFA-Code für diego@example.com bereit und schließt die Anmeldung ab. In diesem Benutzerpool ist die Gerätespeicherung aktiviert, sodass das Authentifizierungsergebnis auch einen neuen Geräteschlüssel zurückgibt.

```
aws cognito-idp admin-respond-to-auth-challenge \
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789 \
    --challenge-name SOFTWARE_TOKEN_MFA \
    --challenge-
responses USERNAME=diego@example.com,SOFTWARE_TOKEN_MFA_CODE=000000 \
    --session AYABeExample...
```

Ausgabe:

```
{
    "ChallengeParameters": {},
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-ExAmPlE1"
        }
    }
}
```

Weitere Informationen finden Sie unter Ablauf der Administratorauthentifizierung im Amazon Cognito Developer Guide.

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

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val respondToAuthChallengeResult =
 identityProviderClient.adminRespondToAuthChallenge(adminRespondToAuthChallengeRequest)
        println("respondToAuthChallengeResult.getAuthenticationResult()
 ${respondToAuthChallengeResult.authenticationResult}")
    }
}
```

- Einzelheiten zur API finden Sie AdminRespondToAuthChallengein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Respond to the authentication challenge received from Cognito after
    /// initiating an authentication session. This involves sending a current
    /// MFA code to the service.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - userName: The user's username.
    ///   - clientId: The app client ID.
    ///   - userPoolId: The user pool to sign into.
    ///   - mfaCode: The 6-digit MFA code currently displayed by the user's
    ///     authenticator.
    ///   - session: The authentication session to continue processing.
    func adminRespondToAuthChallenge(cipClient: CognitoIdentityProviderClient,
 userName: String,
                                     clientId: String, userPoolId: String,
 mfaCode: String,
                                     session: String) async {
        print("=====> SOFTWARE_TOKEN_MFA challenge is generated...")

        var challengeResponsesOb: [String: String] = [:]
        challengeResponsesOb["USERNAME"] = userName
        challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode
```

```
        do {
            let output = try await cipClient.adminRespondToAuthChallenge(
                input: AdminRespondToAuthChallengeInput(
                    challengeName:
CognitoIdentityProviderClientTypes.ChallengeNameType.softwareTokenMfa,
                    challengeResponses: challengeResponsesOb,
                    clientId: clientId,
                    session: session,
                    userPoolId: userPoolId
                )
            )

            guard let authenticationResult = output.authenticationResult else {
                print("*** Unable to get authentication result.")
                return
            }

            print("=====> Authentication result (JWTs are redacted):")
            print(authenticationResult)
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return
        } catch _ as CodeMismatchException {
            print("*** The specified MFA code doesn't match the expected value.")
            return
        } catch _ as UserNotFoundException {
            print("*** The specified username, \(userName), doesn't exist.")
            return
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return
        } catch let error as NotAuthorizedException {
            print("*** Unauthorized access. Reason: \(error.properties.message ??
"<unknown>")")
        } catch {
            print("*** Error responding to the MFA challenge.")
            return
        }
    }
```

- Einzelheiten zur API finden Sie AdminRespondToAuthChallengein der API-Referenz zum AWS SDK für Swift.

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

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

Die folgenden Code-Beispiele zeigen, wie `AdminSetUserPassword` verwendet wird.

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 von benutzerdefinierten Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito-Benutzerauthentifizierung

CLI

AWS CLI

Um ein Benutzerpasswort als Administrator festzulegen

Im folgenden `admin-set-user-password` Beispiel wird das Passwort für diego@example.com dauerhaft festgelegt.

```
aws cognito-idp admin-set-user-password \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego@example.com \
    --password MyExamplePassword1! \
    --permanent
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Passwörter, Passwortwiederherstellung und Passwortrichtlinien im Amazon Cognito Developer Guide.

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

Go

SDK für Go V2

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

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &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 für Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `AssociateSoftwareToken` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für C++ API-Referenz.

CLI

AWS CLI

Um einen geheimen Schlüssel für eine MFA-Authenticator-App zu generieren

Im folgenden `associate-software-token` Beispiel wird ein privater TOTP-Schlüssel für einen Benutzer generiert, der sich angemeldet und ein Zugriffstoken erhalten hat. Der resultierende private Schlüssel kann manuell in eine Authentifizierungs-App eingegeben werden, oder Anwendungen können ihn als QR-Code rendern, den der Benutzer scannen kann.

```
aws cognito-idp associate-software-token \
    --access-token eyJra456defEXAMPLE
```

Ausgabe:

```
{
    "SecretCode": "QWERTYUIOP123456EXAMPLE"
}
```

Weitere Informationen finden Sie unter TOTP-Softwaretoken-MFA im Amazon Cognito Developer Guide.

- Einzelheiten zur API finden Sie unter Befehlsreferenz AssociateSoftwareToken.AWS CLI

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

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

```
suspend fun getSecretForAppMFA(sessionVal: String?): String? {
    val softwareTokenRequest =
        AssociateSoftwareTokenRequest {
            session = sessionVal
        }

    CognitoIdentityProviderClient.fromEnvironment { 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
    }
}
```

- Einzelheiten zur API finden Sie AssociateSoftwareTokenin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Request and display an MFA secret token that the user should enter
    /// into their authenticator to set it up for the user account.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - authSession: The authentication session to request an MFA secret
    ///     for.
    ///
    /// - Returns: A string containing the MFA secret token that should be
    ///   entered into the authenticator software.
    func getSecretForAppMFA(cipClient: CognitoIdentityProviderClient,
 authSession: String?) async -> String? {
        do {
            let output = try await cipClient.associateSoftwareToken(
                input: AssociateSoftwareTokenInput(
                    session: authSession
                )
            )

            guard let secretCode = output.secretCode else {
                print("*** Unable to get the secret code")
                return nil
            }
```

```
            print("=====> Enter this token into Google Authenticator:
\(secretCode)")
            return output.session
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return nil
        } catch {
            print("*** An unexpected error occurred getting the secret for the
app's MFA.")
            return nil
        }
    }
```

- Einzelheiten zur API finden Sie AssociateSoftwareTokenin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `ConfirmDevice` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 ConfirmDeviceein der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Um ein Benutzergerät zu bestätigen

Im folgenden `confirm-device` Beispiel wird dem aktuellen Benutzer ein neues Gerät hinzugefügt, an das man sich erinnern kann.

```
aws cognito-idp confirm-device \
   --access-token eyJra456defEXAMPLE \
   --device-key us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
   --device-secret-verifier-
config PasswordVerifier=TXlWZXJpZmllclN0cmluZw,Salt=TXlTUlBTYWx0
```

Ausgabe:

```
{
     "UserConfirmationNecessary": false
}
```

Weitere Informationen finden Sie unter Arbeiten mit Benutzergeräten in Ihrem Benutzerpool im Amazon Cognito Developer Guide.

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

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 Dienst mit einem AWS SDK 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 Code-Beispiele zeigen, wie `ConfirmForgotPassword` verwendet wird.

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

- Migrieren Sie automatisch bekannte 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import (
 "context"
 "errors"
 "log"

 "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"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &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 für Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `ConfirmSignUp` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

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

```
suspend fun confirmSignUp(
    clientIdVal: String?,
    codeVal: String?,
    userNameVal: String?,
) {
    val signUpRequest =
        ConfirmSignUpRequest {
            clientId = clientIdVal
            confirmationCode = codeVal
            username = userNameVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.confirmSignUp(signUpRequest)
```

```
        println("$userNameVal  was confirmed")
    }
}
```

- Einzelheiten zur API finden Sie ConfirmSignUpin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Submit a confirmation code for the specified user. This is the code as
    /// entered by the user after they've received it by email or text
    /// message.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The app client ID the user is signing up for.
    ///   - userName: The username of the user whose code is being sent.
    ///   - code: The user's confirmation code.
    ///
    /// - Returns: `true` if the code was successfully confirmed; otherwise
 `false`.
    func confirmSignUp(cipClient: CognitoIdentityProviderClient, clientId:
 String,
                       userName: String, code: String) async -> Bool {
        do {
            _ = try await cipClient.confirmSignUp(
                input: ConfirmSignUpInput(
                    clientId: clientId,
                    confirmationCode: code,
                    username: userName
                )
            )

            print("=====> \(userName) has been confirmed.")
            return true
        } catch {
            print("=====> \(userName)'s code was entered incorrectly.")
            return false
        }
    }
```

- Einzelheiten zur API finden Sie ConfirmSignUpin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `CreateUserPool` verwendet wird.

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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `CreateUserPoolClient` verwendet wird.

CLI

AWS CLI

Um einen Benutzerpool-Client zu erstellen

Im folgenden `create-user-pool-client` Beispiel wird ein neuer Benutzerpool-Client mit einem geheimen Client-Schlüssel, expliziten Lese- und Schreibattributen, Anmeldung mit Benutzername-Passwort und SRP-Flows, Anmeldung mit drei, Zugriff auf eine Teilmenge von OAuth Bereichen IdPs, PinPoint Analysen und einer erweiterten Gültigkeitsdauer der Authentifizierungssitzung erstellt.

```
aws cognito-idp create-user-pool-client \
    --user-pool-id us-west-2_EXAMPLE \
    --client-name MyTestClient \
    --generate-secret \
    --refresh-token-validity 10 \
    --access-token-validity 60 \
    --id-token-validity 60 \
    --token-validity-units AccessToken=minutes,IdToken=minutes,RefreshToken=days
 \
    --read-attributes email phone_number email_verified phone_number_verified \
    --write-attributes email phone_number \
    --explicit-auth-
flows ALLOW_USER_PASSWORD_AUTH ALLOW_USER_SRP_AUTH ALLOW_REFRESH_TOKEN_AUTH \
    --supported-identity-providers Google Facebook MyOIDC \
    --callback-urls https://www.amazon.com https://example.com http://
localhost:8001 myapp://example \
    --allowed-o-auth-flows code implicit \
    --allowed-o-auth-scopes openid profile aws.cognito.signin.user.admin solar-
system-data/asteroids.add \
    --allowed-o-auth-flows-user-pool-client \
    --analytics-configuration ApplicationArn=arn:aws:mobiletargeting:us-
west-2:767671399759:apps/thisisanexamplepinpointapplicationid,UserDataShared=TRUE
 \
    --prevent-user-existence-errors ENABLED \
    --enable-token-revocation \
    --enable-propagate-additional-user-context-data \
    --auth-session-validity 4
```

Ausgabe:

```
{
    "UserPoolClient": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ClientName": "MyTestClient",
        "ClientId": "123abc456defEXAMPLE",
        "ClientSecret": "this1234is5678my91011example1213client1415secret",
        "LastModifiedDate": 1726788459.464,
        "CreationDate": 1726788459.464,
        "RefreshTokenValidity": 10,
        "AccessTokenValidity": 60,
        "IdTokenValidity": 60,
        "TokenValidityUnits": {
            "AccessToken": "minutes",
            "IdToken": "minutes",
            "RefreshToken": "days"
        },
        "ReadAttributes": [
            "email_verified",
            "phone_number_verified",
            "phone_number",
            "email"
        ],
        "WriteAttributes": [
            "phone_number",
            "email"
        ],
        "ExplicitAuthFlows": [
            "ALLOW_USER_PASSWORD_AUTH",
            "ALLOW_USER_SRP_AUTH",
            "ALLOW_REFRESH_TOKEN_AUTH"
        ],
        "SupportedIdentityProviders": [
            "Google",
            "MyOIDC",
            "Facebook"
        ],
        "CallbackURLs": [
            "https://example.com",
            "https://www.amazon.com",
            "myapp://example",
            "http://localhost:8001"
        ],
```

```
        "AllowedOAuthFlows": [
            "implicit",
            "code"
        ],
        "AllowedOAuthScopes": [
            "aws.cognito.signin.user.admin",
            "openid",
            "profile",
            "solar-system-data/asteroids.add"
        ],
        "AllowedOAuthFlowsUserPoolClient": true,
        "AnalyticsConfiguration": {
            "ApplicationArn": "arn:aws:mobiletargeting:us-
west-2:123456789012:apps/thisisanexamplepinpointapplicationid",
            "RoleArn": "arn:aws:iam::123456789012:role/aws-service-role/cognito-
idp.amazonaws.com/AWSServiceRoleForAmazonCognitoIdp",
            "UserDataShared": true
        },
        "PreventUserExistenceErrors": "ENABLED",
        "EnableTokenRevocation": true,
        "EnablePropagateAdditionalUserContextData": true,
        "AuthSessionValidity": 4
    }
}
```

Weitere Informationen finden Sie unter Anwendungsspezifische Einstellungen mit App-Clients im Amazon Cognito Developer Guide.

- Einzelheiten zur API finden Sie unter Befehlsreferenz CreateUserPoolClient.AWS CLI

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `DeleteUser` verwendet wird.

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 automatisch bekannte Benutzer mit einer Lambda-Funktion
- Migrieren Sie automatisch bekannte Benutzer mit einer Lambda-Funktion
- Schreiben von benutzerdefinierten Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito-Benutzerauthentifizierung

C++

SDK für C++

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

```
import (
 "context"
 "errors"
 "log"

 "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"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &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 für Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

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

```
/**
 * Delete the signed-in user. Useful for allowing a user to delete their
 * own profile.
 * @param {{ region: string, accessToken: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").DeleteUserCommandOutput | null, unknown]>}
```

```
 */
export const deleteUser = async ({ region, accessToken }) => {
  try {
    const client = new CognitoIdentityProviderClient({ region });
    const response = await client.send(
      new DeleteUserCommand({ AccessToken: accessToken }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};
```

- Einzelheiten zur API finden Sie DeleteUserin der AWS SDK für JavaScript API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `ForgotPassword` verwendet wird.

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

- Migrieren Sie automatisch bekannte 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &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 für Go API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `InitiateAuth` verwendet wird.

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 automatisch bekannte Benutzer mit einer Lambda-Funktion
- Migrieren Sie automatisch bekannte Benutzer mit einer Lambda-Funktion
- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert
- Schreiben von benutzerdefinierten Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito-Benutzerauthentifizierung

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

CLI

AWS CLI

Um einen Benutzer anzumelden

Im folgenden `initiate-auth` Beispiel wird ein Benutzer mit dem einfachen Benutzername-Passwort-Ablauf und ohne zusätzliche Herausforderungen angemeldet.

```
aws cognito-idp initiate-auth \
    --auth-flow USER_PASSWORD_AUTH \
    --client-id 1example23456789 \
    --analytics-metadata AnalyticsEndpointId=d70b2ba36a8c4dc5a04a0451aEXAMPLE \
    --auth-parameters USERNAME=testuser,PASSWORD=[Password] --user-context-
data EncodedData=mycontextdata --client-metadata MyTestKey=MyTestValue
```

Ausgabe:

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-v7w9UcY6"
        }
    }
}
```

Weitere Informationen finden Sie unter Authentifizierung im Amazon Cognito Developer Guide.

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

Go

SDK für Go V2

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

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &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 für Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 Dienst mit einem AWS SDK 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 Code-Beispiele zeigen, wie `ListUserPools` verwendet wird.

.NET

SDK für .NET (v4)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

CLI

AWS CLI

Benutzerpools auflisten

Das folgende `list-user-pools` Beispiel listet 3 der verfügbaren Benutzerpools im AWS Konto mit den aktuellen CLI-Anmeldeinformationen auf.

```
aws cognito-idp list-user-pools \
    --max-results 3
```

Ausgabe:

```
{
    "NextToken": "[Pagination token]",
    "UserPools": [
        {
            "CreationDate": 1681502497.741,
            "Id": "us-west-2_EXAMPLE1",
            "LambdaConfig": {
                "CustomMessage": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreSignUp": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreTokenGeneration": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreTokenGenerationConfig": {
                    "LambdaArn": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                    "LambdaVersion": "V1_0"
                }
            },
            "LastModifiedDate": 1681502497.741,
            "Name": "user pool 1"
        },
        {
            "CreationDate": 1686064178.717,
```

```
            "Id": "us-west-2_EXAMPLE2",
            "LambdaConfig": {
            },
            "LastModifiedDate": 1686064178.873,
            "Name": "user pool 2"
        },
        {
            "CreationDate": 1627681712.237,
            "Id": "us-west-2_EXAMPLE3",
            "LambdaConfig": {
                "UserMigration": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction"
            },
            "LastModifiedDate": 1678486942.479,
            "Name": "user pool 3"
        }
    ]
}
```

Weitere Informationen finden Sie unter Amazon-Cognito-Benutzerpools im Amazon-Cognito-Entwicklerhandbuch.

- Einzelheiten zur API finden Sie ListUserPoolsunter AWS CLI Befehlsreferenz.

Go

SDK für Go V2

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 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(ctx)
  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 für Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `ListUsers` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

CLI

AWS CLI

Beispiel 1: Um Benutzer mit einem serverseitigen Filter aufzulisten

Im folgenden `list-users` Beispiel werden 3 Benutzer im angeforderten Benutzerpool aufgeführt, deren E-Mail-Adressen mit `testuser` beginnen.

```
aws cognito-idp list-users \
    --user-pool-id us-west-2_EXAMPLE \
    --filter email^=\"testuser\" \
    --max-items 3
```

Ausgabe:

```
{
    "PaginationToken": "efgh5678EXAMPLE",
    "Users": [
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "eaad0219-2117-439f-8d46-4db20e59268f"
                },
                {
                    "Name": "email",
                    "Value": "testuser@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1682955829.578,
            "UserLastModifiedDate": 1689030181.63,
            "UserStatus": "CONFIRMED",
            "Username": "testuser"
        },
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "3b994cfd-0b07-4581-be46-3c82f9a70c90"
                },
                {
                    "Name": "email",
                    "Value": "testuser2@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1684427979.201,
            "UserLastModifiedDate": 1684427979.201,
            "UserStatus": "UNCONFIRMED",
            "Username": "testuser2"
        },
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "5929e0d1-4c34-42d1-9b79-a5ecacfe66f7"
                },
```

```
                {
                    "Name": "email",
                    "Value": "testuser3@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1684427823.641,
            "UserLastModifiedDate": 1684427823.641,
            "UserStatus": "UNCONFIRMED",
            "Username": "testuser3@example.com"
        }
    ]
}
```

Weitere Informationen finden Sie unter Benutzer verwalten und suchen im Amazon Cognito Developer Guide.

Beispiel 2: Um Benutzer mit einem clientseitigen Filter aufzulisten

Das folgende `list-users` Beispiel listet die Attribute von drei Benutzern auf, deren Attribut, in diesem Fall ihre E-Mail-Adresse, die E-Mail-Domäne "@example .com“ enthält. Wenn andere Attribute diese Zeichenfolge enthielten, würden sie ebenfalls angezeigt. Der zweite Benutzer hat keine Attribute, die der Abfrage entsprechen, und ist von der angezeigten Ausgabe ausgeschlossen, aber nicht von der Serverantwort.

```
aws cognito-idp list-users \
    --user-pool-id us-west-2_EXAMPLE \
    --max-items 3
    --query Users\[\*\].Attributes\[\?Value\.contains\(\@\,\'@example.com\'\)\]
```

Ausgabe:

```
[
    [
        {
            "Name": "email",
            "Value": "admin@example.com"
        }
    ],
    [],
    [
        {
```

```
            "Name": "email",
            "Value": "operator@example.com"
        }
    ]
]
```

Weitere Informationen finden Sie unter Benutzer verwalten und suchen im Amazon Cognito Developer Guide.

- Einzelheiten zur API finden Sie unter ListUsers AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

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

```
suspend fun listAllUsers(userPoolId: String) {
    val request =
        ListUsersRequest {
            this.userPoolId = userPoolId
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { cognitoClient ->
        val response = cognitoClient.listUsers(request)
        response.users?.forEach { user ->
            println("The user name is ${user.username}")
        }
    }
}
```

- Einzelheiten zur API finden Sie ListUsersin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
        do {
            let output = try await cognitoClient.listUsers(
                input: ListUsersInput(
                    userPoolId: poolId
                )
            )

            guard let users = output.users else {
                print("No users found.")
                return
            }

            print("\(users.count) user(s) found.")
            for user in users {
                print("  \(user.username ?? "<unknown>")")
            }
        } catch _ as NotAuthorizedException {
            print("*** Please authenticate with AWS before using this command.")
            return
        } catch _ as ResourceNotFoundException {
```

```
            print("*** The specified User Pool was not found.")
            return
        } catch {
            print("*** An unexpected type of error occurred.")
            return
        }
```

- Einzelheiten zur API finden Sie ListUsersin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `ResendConfirmationCode` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

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

```
suspend fun resendConfirmationCode(
    clientIdVal: String?,
    userNameVal: String?,
) {
    val codeRequest =
        ResendConfirmationCodeRequest {
            clientId = clientIdVal
            username = userNameVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.resendConfirmationCode(codeRequest)
        println("Method of delivery is " +
 (response.codeDeliveryDetails?.deliveryMedium))
    }
}
```

- Einzelheiten zur API finden Sie ResendConfirmationCodein der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Requests a new confirmation code be sent to the given user's contact
    /// method.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The application client ID.
    ///   - userName: The user to resend a code for.
    ///
    /// - Returns: `true` if a new code was sent successfully, otherwise
    ///   `false`.
    func resendConfirmationCode(cipClient: CognitoIdentityProviderClient,
 clientId: String,
                                userName: String) async -> Bool {
        do {
            let output = try await cipClient.resendConfirmationCode(
                input: ResendConfirmationCodeInput(
                    clientId: clientId,
                    username: userName
                )
            )

            guard let deliveryMedium = output.codeDeliveryDetails?.deliveryMedium
 else {
```

```
                print("*** Unable to get the delivery method for the resent
code.")
                return false
            }

            print("=====> A new code has been sent by \(deliveryMedium)")
            return true
        } catch {
            print("*** Unable to resend the confirmation code to user
\(userName).")
            return false
        }
    }
```

- Einzelheiten zur API finden Sie ResendConfirmationCodein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `RespondToAuthChallenge` verwendet wird.

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

Beispiel 1: Um auf eine NEW_PASSWORD_REQUIRED-Herausforderung zu antworten

Das folgende `respond-to-auth-challenge` Beispiel reagiert auf eine NEW_PASSWORD_REQUIRED-Abfrage, die initiate-auth zurückgegeben hat. Es legt ein Passwort für den Benutzer fest. `jane@example.com`

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789 \
    --challenge-name NEW_PASSWORD_REQUIRED \
    --challenge-responses USERNAME=jane@example.com,NEW_PASSWORD=[Password] \
    --session AYABeEv5HklEXAMPLE
```

Ausgabe:

```
{
    "ChallengeParameters": {},
    "AuthenticationResult": {
        "AccessToken": "ACCESS_TOKEN",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "REFRESH_TOKEN",
        "IdToken": "ID_TOKEN",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-wt2ha1Zd"
        }
    }
}
```

Weitere Informationen finden Sie unter Authentifizierung im Amazon Cognito Developer Guide.

Beispiel 2: Um auf eine SELECT_MFA_TYPE-Herausforderung zu antworten

Im folgenden `respond-to-auth-challenge` Beispiel wird TOTP MFA als MFA-Option für den aktuellen Benutzer ausgewählt. Der Benutzer wurde aufgefordert, einen MFA-Typ auszuwählen, und wird als Nächstes aufgefordert, seinen MFA-Code einzugeben.

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789
    --session AYABeEv5HklEXAMPLE
    --challenge-name SELECT_MFA_TYPE
    --challenge-responses USERNAME=testuser,ANSWER=SOFTWARE_TOKEN_MFA
```

Ausgabe:

```
{
```

```
    "ChallengeName": "SOFTWARE_TOKEN_MFA",
    "Session": "AYABeEv5HklEXAMPLE",
    "ChallengeParameters": {
        "FRIENDLY_DEVICE_NAME": "transparent"
    }
}
```

Weitere Informationen finden Sie unter Hinzufügen von MFA im Amazon Cognito Developer Guide.

Beispiel 3: Um auf eine SOFTWARE_TOKEN_MFA-Herausforderung zu antworten

Das folgende `respond-to-auth-challenge` Beispiel stellt einen TOTP-MFA-Code bereit und schließt die Anmeldung ab.

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789 \
    --session AYABeEv5HklEXAMPLE \
    --challenge-name SOFTWARE_TOKEN_MFA \
    --challenge-responses USERNAME=testuser,SOFTWARE_TOKEN_MFA_CODE=123456
```

Ausgabe:

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-v7w9UcY6"
        }
    }
}
```

Weitere Informationen finden Sie unter Hinzufügen von MFA im Amazon Cognito Developer Guide.

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

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 Dienst mit einem AWS SDK 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 Code-Beispiele zeigen, wie `SignUp` verwendet wird.

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 automatisch bekannte Benutzer mit einer Lambda-Funktion
- Migrieren Sie automatisch bekannte Benutzer mit einer Lambda-Funktion
- Registrieren eines Benutzers bei einem Benutzerpool, der MFA erfordert

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

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

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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

```
// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &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 für Go API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.signUp(signUpRequest)
        println("User has been signed up")
    }
}
```

- Einzelheiten zur API finden Sie SignUpin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Create a new user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The ID of the app client to create a user for.
    ///   - userName: The username for the new user.
    ///   - password: The new user's password.
    ///   - email: The new user's email address.
    ///
    /// - Returns: `true` if successful; otherwise `false`.
    func signUp(cipClient: CognitoIdentityProviderClient, clientId: String,
 userName: String, password: String, email: String) async -> Bool {
        let emailAttr = CognitoIdentityProviderClientTypes.AttributeType(
            name: "email",
            value: email
        )

        let userAttrsList = [emailAttr]
```

```
        do {
            _ = try await cipClient.signUp(
                input: SignUpInput(
                    clientId: clientId,
                    password: password,
                    userAttributes: userAttrsList,
                    username: userName
                )

            )

            print("=====> User \(userName) signed up.")
        } catch _ as AWSCognitoIdentityProvider.UsernameExistsException {
            print("*** The username \(userName) already exists. Please use a
different one.")
            return false
        } catch let error as AWSCognitoIdentityProvider.InvalidPasswordException
{
            print("*** Error: The specified password is invalid. Reason:
\(error.properties.message ?? "<none available>").")
            return false
        } catch _ as AWSCognitoIdentityProvider.ResourceNotFoundException {
            print("*** Error: The specified client ID (\(clientId)) doesn't
exist.")
            return false
        } catch {
            print("*** Unexpected error: \(error)")
            return false
        }

        return true
    }
```

- Einzelheiten zur API finden Sie SignUpin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `UpdateUserPool` verwendet wird.

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 automatisch bekannte Benutzer mit einer Lambda-Funktion
- Migrieren Sie automatisch bekannte Benutzer mit einer Lambda-Funktion
- Schreiben von benutzerdefinierten Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon Cognito-Benutzerauthentifizierung

CLI

AWS CLI

Um einen Benutzerpool zu aktualisieren

Im folgenden `update-user-pool` Beispiel wird ein Benutzerpool mit Beispielsyntax für jede der verfügbaren Konfigurationsoptionen geändert. Um einen Benutzerpool zu aktualisieren, müssen Sie alle zuvor konfigurierten Optionen angeben. Andernfalls werden sie auf einen Standardwert zurückgesetzt.

```
aws cognito-idp update-user-pool --user-pool-id us-west-2_EXAMPLE \
    --policies PasswordPolicy=
\{MinimumLength=6,RequireUppercase=true,RequireLowercase=true,RequireNumbers=true,Require
 \
    --deletion-protection ACTIVE \
    --lambda-config PreSignUp="arn:aws:lambda:us-
west-2:123456789012:function:cognito-test-presignup-
function",PreTokenGeneration="arn:aws:lambda:us-
west-2:123456789012:function:cognito-test-pretoken-function" \
    --auto-verified-attributes "phone_number" "email" \
    --verification-message-template \{\"SmsMessage\":\""Your code is
 {####}"\",\"EmailMessage\":\""Your code is {####}"\",\"EmailSubject\":\""Your
 verification code"\",\"EmailMessageByLink\":\""Click {##here##} to verify
 your email address."\",\"EmailSubjectByLink\":\""Your verification link"\",
\"DefaultEmailOption\":\"CONFIRM_WITH_LINK\"\} \
    --sms-authentication-message "Your code is {####}" \
    --user-attribute-update-settings
 AttributesRequireVerificationBeforeUpdate="email","phone_number" \
```

```
    --mfa-configuration "OPTIONAL" \
    --device-
configuration ChallengeRequiredOnNewDevice=true,DeviceOnlyRememberedOnUserPrompt=true
 \
    --email-configuration SourceArn="arn:aws:ses:us-
west-2:123456789012:identity/admin@example.com",ReplyToEmailAddress="amdin
+noreply@example.com",EmailSendingAccount=DEVELOPER,From="admin@amazon.com",Configuration
configuration-set" \
    --sms-configuration SnsCallerArn="arn:aws:iam::123456789012:role/service-
role/SNS-SMS-Role",ExternalId="12345",SnsRegion="us-west-2" \
    --admin-create-user-config
 AllowAdminCreateUserOnly=false,InviteMessageTemplate=\{SMSMessage=\""Welcome
 {username}. Your confirmation code is {####}"\",EmailMessage=\""Welcome
 {username}. Your confirmation code is {####}"\",EmailSubject=\""Welcome to
 MyMobileGame"\"\} \
    --user-pool-tags "Function"="MyMobileGame","Developers"="Berlin" \
    --admin-create-user-config
 AllowAdminCreateUserOnly=false,InviteMessageTemplate=\{SMSMessage=\""Welcome
 {username}. Your confirmation code is {####}"\",EmailMessage=\""Welcome
 {username}. Your confirmation code is {####}"\",EmailSubject=\""Welcome to
 MyMobileGame"\"\} \
    --user-pool-add-ons AdvancedSecurityMode="AUDIT" \
    --account-recovery-setting RecoveryMechanisms=
\[\{Priority=1,Name="verified_email"\},
\{Priority=2,Name="verified_phone_number"\}\]
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Aktualisieren der Benutzerpool-Konfiguration im Amazon Cognito Developer Guide.

- Einzelheiten zur API finden Sie unter UpdateUserPool AWS CLIBefehlsreferenz.

Go

SDK für Go V2

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

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &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(ctx,
 &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 für Go API-Referenz.

JavaScript

SDK für JavaScript (v3)

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

```
/**
 * Connect a Lambda function to the PreSignUp trigger for a Cognito user pool
```

```
 * @param {{ region: string, userPoolId: string, handlerArn: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").UpdateUserPoolCommandOutput | null, unknown]>}
 */
export const addPreSignUpHandler = async ({
  region,
  userPoolId,
  handlerArn,
}) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({
      region,
    });

    const command = new UpdateUserPoolCommand({
      UserPoolId: userPoolId,
      LambdaConfig: {
        PreSignUp: handlerArn,
      },
    });

    const response = await cognitoClient.send(command);
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};
```

- Einzelheiten zur API finden Sie UpdateUserPoolin der AWS SDK für JavaScript API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Code-Beispiele zeigen, wie `VerifySoftwareToken` verwendet wird.

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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für C++ API-Referenz.

CLI

AWS CLI

Um die Registrierung eines TOTP-Authentifikators zu bestätigen

Im folgenden `verify-software-token` Beispiel wird die TOTP-Registrierung für den aktuellen Benutzer abgeschlossen.

```
aws cognito-idp verify-software-token \
    --access-token eyJra456defEXAMPLE \
    --user-code 123456
```

Ausgabe:

```
{
    "Status": "SUCCESS"
}
```

Weitere Informationen finden Sie unter Hinzufügen von MFA zu einem Benutzerpool im Amazon Cognito Developer Guide.

- Einzelheiten zur API finden Sie unter VerifySoftwareToken AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val verifyResponse =
 identityProviderClient.verifySoftwareToken(tokenRequest)
        println("The status of the token is ${verifyResponse.status}")
    }
}
```

- Einzelheiten zur API finden Sie VerifySoftwareTokenin der API-Referenz zum AWS SDK für Kotlin.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.

Swift

SDK für Swift

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

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Confirm that the user's TOTP authenticator is configured correctly by
    /// sending a code to it to check that it matches successfully.
    ///
    /// - Parameters:
    ///   - cipClient: The `CongnitoIdentityProviderClient` to use.
    ///   - session: An authentication session previously returned by an
    ///     `associateSoftwareToken()` call.
```

```
    ///   - mfaCode: The 6-digit code currently displayed by the user's
    ///     authenticator, as provided by the user.
    func verifyTOTP(cipClient: CognitoIdentityProviderClient, session: String?,
 mfaCode: String?) async {
        do {
            let output = try await cipClient.verifySoftwareToken(
                input: VerifySoftwareTokenInput(
                    session: session,
                    userCode: mfaCode
                )
            )

            guard let tokenStatus = output.status else {
                print("*** Unable to get the token's status.")
                return
            }
            print("=====> The token's status is: \(tokenStatus)")
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return
        } catch _ as CodeMismatchException {
            print("*** The specified MFA code doesn't match the expected value.")
            return
        } catch _ as UserNotFoundException {
            print("*** The specified username doesn't exist.")
            return
        } catch _ as UserNotConfirmedException {
            print("*** The user has not been confirmed.")
            return
        } catch {
            print("*** Error verifying the MFA token!")
            return
        }
    }
```

- Einzelheiten zur API finden Sie VerifySoftwareTokenin der API-Referenz zum AWS SDK für Swift.

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

Szenarien für die Verwendung von Amazon Cognito Identity Provider 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 erledigen können, indem Sie mehrere Funktionen innerhalb von Amazon Cognito Identity Provider oder in Kombination mit anderen AWS-Services aufrufen. Jedes Szenario enthält einen Link zum vollständigen Quell-Code, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Szenarien zielen auf eine mittlere Erfahrungsebene ab, um Ihnen zu helfen, Service-Aktionen im Kontext zu verstehen.

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

Die folgenden Codebeispiele zeigen, wie bekannte Amazon Cognito-Benutzer mit einer Lambda-Funktion automatisch bestätigt werden können.

- Konfigurieren Sie einen Benutzerpool, um eine Lambda-Funktion für den `PreSignUp`-Trigger aufzurufen.
- Registrieren eines Benutzers bei Amazon Cognito.
- Die Lambda-Funktion scannt eine DynamoDB-Tabelle und bestätigt automatisch bekannte Benutzer.
- Melden Sie sich als neuer Benutzer an und bereinigen Sie dann die Ressourcen.

Go

SDK für Go V2

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

Führen Sie ein interaktives Szenario an einem Prompt aus.

```
import (
 "context"
 "errors"
 "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"
)

// 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(ctx context.Context, 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(
  ctx, 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(ctx context.Context, 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(ctx, 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(ctx, 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(ctx context.Context, 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(ctx, 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(ctx context.Context, stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup(ctx)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(ctx, stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(ctx, stackOutputs["TableName"])

 runner.AddPreSignUpTrigger(ctx, stackOutputs["UserPoolId"],
 stackOutputs["AutoConfirmFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers, actions.PreSignUp)
 userName, password := runner.SignUpUser(ctx, stackOutputs["UserPoolClientId"],
 stackOutputs["TableName"])
 runner.helper.ListRecentLogEvents(ctx, stackOutputs["AutoConfirmFunction"])
 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
  runner.SignInUser(ctx, stackOutputs["UserPoolClientId"], userName, password))

 runner.resources.Cleanup(ctx)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Behandeln Sie den `PreSignUp`-Trigger mit einer Lambda-Funktion.

```
import (
```

```
 "context"
 "log"
 "os"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 dynamodbtypes "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

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() {
```

```
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Erstellen Sie eine Struktur, die allgemeine Aufgaben ausführt.

```
import (
 "context"
 "log"
 "strings"
 "time"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(ctx context.Context, stackName string) (actions.StackOutputs,
 error)
 PopulateUserTable(ctx context.Context, tableName string)
 GetKnownUsers(ctx context.Context, tableName string) (actions.UserList, error)
 AddKnownUser(ctx context.Context, tableName string, user actions.User)
 ListRecentLogEvents(ctx context.Context, 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(ctx context.Context, stackName
 string) (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(ctx, stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(ctx context.Context, 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(ctx, tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(ctx context.Context, tableName string)
 (actions.UserList, error) {
 knownUsers, err := helper.dynamoActor.Scan(ctx, 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(ctx context.Context, 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(ctx, 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(ctx context.Context,
 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(ctx, functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
```

```
 events, err := helper.cwlActor.GetLogEvents(ctx, 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-Aktionen umschließt.

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &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(ctx,
 &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(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &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(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &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(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &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(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &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(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &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(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &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(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &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 umhüllt.

```
import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
```

```
 "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

// 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(ctx context.Context, 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(ctx, &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(ctx context.Context, tableName string) (UserList,
 error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(ctx, &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(ctx context.Context, 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(ctx, &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 CloudWatch Logs-Aktionen umschließt.

```
import (
 "context"
 "fmt"
 "log"

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

type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(ctx context.Context,
 functionName string) (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(ctx,
 &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(ctx context.Context, 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(ctx,
 &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

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
```

```
 "github.com/aws/aws-sdk-go-v2/service/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(ctx context.Context, stackName
 string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(ctx,
 &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
}
```

Bereinigen von Ressourcen.

```
import (
 "context"
 "log"
 "user_pools_and_lambda_triggers/actions"

 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// 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(ctx context.Context) {
 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(ctx, 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(ctx, 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 für Go -API-Referenz.
  - DeleteUser
  - InitiateAuth
  - SignUp
  - UpdateUserPool

JavaScript

SDK für JavaScript (v3)

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

Konfigurieren Sie einen interaktiven „Szenario“-Lauf. Die JavaScript (v3) -Beispiele verwenden gemeinsam einen Szenario-Runner, um komplexe Beispiele zu vereinfachen. Der komplette Quellcode ist aktiviert GitHub.

```
import { AutoConfirm } from "./scenario-auto-confirm.js";
```

```
/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {
  errors: [],
  users: [
    {
      UserName: "test_user_1",
      UserEmail: "test_email_1@example.com",
    },
    {
      UserName: "test_user_2",
      UserEmail: "test_email_2@example.com",
    },
    {
      UserName: "test_user_3",
      UserEmail: "test_email_3@example.com",
    },
  ],
};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Demonstrate automatically confirming known users in a database.
  "auto-confirm": AutoConfirm(context),
};

// Call function if run directly
import { fileURLToPath } from "node:url";
import { parseScenarioArgs } from "@aws-doc-sdk-examples/lib/scenario/index.js";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
    name: "Cognito user pools and triggers",
    description:
      "Demonstrate how to use the AWS SDKs to customize Amazon Cognito
 authentication behavior.",
  });
}
```

Dieses Szenario demonstriert die automatische Bestätigung eines bekannten Benutzers. Es orchestriert die Beispielschritte.

```
import { wait } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import {
  Scenario,
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";

import {
  getStackOutputs,
  logCleanUpReminder,
  promptForStackName,
  promptForStackRegion,
  skipWhenErrors,
} from "./steps-common.js";
import { populateTable } from "./actions/dynamodb-actions.js";
import {
  addPreSignUpHandler,
  deleteUser,
  getUser,
  signIn,
  signUpUser,
} from "./actions/cognito-actions.js";
import {
  getLatestLogStreamForLambda,
  getLogEvents,
} from "./actions/cloudwatch-logs-actions.js";

/**
 * @typedef {{
 *   errors: Error[],
 *   password: string,
 *   users: { UserName: string, UserEmail: string }[],
 *   selectedUser?: string,
 *   stackName?: string,
 *   stackRegion?: string,
 *   token?: string,
 *   confirmDeleteSignedInUser?: boolean,
```

```
 *   TableName?: string,
 *   UserPoolClientId?: string,
 *   UserPoolId?: string,
 *   UserPoolArn?: string,
 *   AutoConfirmHandlerArn?: string,
 *   AutoConfirmHandlerName?: string
 * }} State
 */

const greeting = new ScenarioOutput(
  "greeting",
  (/** @type {State} */ state) => `This demo will populate some users into the \
database created as part of the "${state.stackName}" stack. \
Then the AutoConfirmHandler will be linked to the PreSignUp \
trigger from Cognito. Finally, you will choose a user to sign up.`,
  { skipWhen: skipWhenErrors },
);

const logPopulatingUsers = new ScenarioOutput(
  "logPopulatingUsers",
  "Populating the DynamoDB table with some users.",
  { skipWhenErrors: skipWhenErrors },
);

const logPopulatingUsersComplete = new ScenarioOutput(
  "logPopulatingUsersComplete",
  "Done populating users.",
  { skipWhen: skipWhenErrors },
);

const populateUsers = new ScenarioAction(
  "populateUsers",
  async (/** @type {State} */ state) => {
    const [_, err] = await populateTable({
      region: state.stackRegion,
      tableName: state.TableName,
      items: state.users,
    });
    if (err) {
      state.errors.push(err);
    }
  },
  {
    skipWhen: skipWhenErrors,
```

```
  },
);

const logSetupSignUpTrigger = new ScenarioOutput(
  "logSetupSignUpTrigger",
  "Setting up the PreSignUp trigger for the Cognito User Pool.",
  { skipWhen: skipWhenErrors },
);

const setupSignUpTrigger = new ScenarioAction(
  "setupSignUpTrigger",
  async (/** @type {State} */ state) => {
    const [_, err] = await addPreSignUpHandler({
      region: state.stackRegion,
      userPoolId: state.UserPoolId,
      handlerArn: state.AutoConfirmHandlerArn,
    });
    if (err) {
      state.errors.push(err);
    }
  },
  {
    skipWhen: skipWhenErrors,
  },
);

const logSetupSignUpTriggerComplete = new ScenarioOutput(
  "logSetupSignUpTriggerComplete",
  (
    /** @type {State} */ state,
  ) => `The lambda function "${state.AutoConfirmHandlerName}" \
has been configured as the PreSignUp trigger handler for the user pool
 "${state.UserPoolId}".`,
  { skipWhen: skipWhenErrors },
);

const selectUser = new ScenarioInput(
  "selectedUser",
  "Select a user to sign up.",
  {
    type: "select",
    choices: (/** @type {State} */ state) => state.users.map((u) => u.UserName),
    skipWhen: skipWhenErrors,
    default: (/** @type {State} */ state) => state.users[0].UserName,
```

```
  },
);

const checkIfUserAlreadyExists = new ScenarioAction(
  "checkIfUserAlreadyExists",
  async (/** @type {State} */ state) => {
    const [user, err] = await getUser({
      region: state.stackRegion,
      userPoolId: state.UserPoolId,
      username: state.selectedUser,
    });

    if (err?.name === "UserNotFoundException") {
      // Do nothing. We're not expecting the user to exist before
      // sign up is complete.
      return;
    }

    if (err) {
      state.errors.push(err);
      return;
    }

    if (user) {
      state.errors.push(
        new Error(
          `The user "${state.selectedUser}" already exists in the user pool
 "${state.UserPoolId}".`,
        ),
      );
    }
  },
  {
    skipWhen: skipWhenErrors,
  },
);

const createPassword = new ScenarioInput(
  "password",
  "Enter a password that has at least eight characters, uppercase, lowercase,
 numbers and symbols.",
  { type: "password", skipWhen: skipWhenErrors, default: "Abcd1234!" },
);
```

```
const logSignUpExistingUser = new ScenarioOutput(
  "logSignUpExistingUser",
  (/** @type {State} */ state) => `Signing up user "${state.selectedUser}".`,
  { skipWhen: skipWhenErrors },
);

const signUpExistingUser = new ScenarioAction(
  "signUpExistingUser",
  async (/** @type {State} */ state) => {
    const signUp = (password) =>
      signUpUser({
        region: state.stackRegion,
        userPoolClientId: state.UserPoolClientId,
        username: state.selectedUser,
        email: state.users.find((u) => u.UserName === state.selectedUser)
          .UserEmail,
        password,
      });

    let [_, err] = await signUp(state.password);

    while (err?.name === "InvalidPasswordException") {
      console.warn("The password you entered was invalid.");
      await createPassword.handle(state);
      [_, err] = await signUp(state.password);
    }

    if (err) {
      state.errors.push(err);
    }
  },
  { skipWhen: skipWhenErrors },
);

const logSignUpExistingUserComplete = new ScenarioOutput(
  "logSignUpExistingUserComplete",
  (/** @type {State} */ state) =>
    `"${state.selectedUser} was signed up successfully.`,
  { skipWhen: skipWhenErrors },
);

const logLambdaLogs = new ScenarioAction(
  "logLambdaLogs",
  async (/** @type {State} */ state) => {
```

```
    console.log(
      "Waiting a few seconds to let Lambda write to CloudWatch Logs...\n",
    );
    await wait(10);

    const [logStream, logStreamErr] = await getLatestLogStreamForLambda({
      functionName: state.AutoConfirmHandlerName,
      region: state.stackRegion,
    });
    if (logStreamErr) {
      state.errors.push(logStreamErr);
      return;
    }

    console.log(
      `Getting some recent events from log stream "${logStream.logStreamName}"`,
    );
    const [logEvents, logEventsErr] = await getLogEvents({
      functionName: state.AutoConfirmHandlerName,
      region: state.stackRegion,
      eventCount: 10,
      logStreamName: logStream.logStreamName,
    });
    if (logEventsErr) {
      state.errors.push(logEventsErr);
      return;
    }

    console.log(logEvents.map((ev) => `\t${ev.message}`).join(""));
  },
  { skipWhen: skipWhenErrors },
);

const logSignInUser = new ScenarioOutput(
  "logSignInUser",
  (/** @type {State} */ state) => `Let's sign in as ${state.selectedUser}`,
  { skipWhen: skipWhenErrors },
);

const signInUser = new ScenarioAction(
  "signInUser",
  async (/** @type {State} */ state) => {
    const [response, err] = await signIn({
      region: state.stackRegion,
```

```
      clientId: state.UserPoolClientId,
      username: state.selectedUser,
      password: state.password,
    });

    if (err?.name === "PasswordResetRequiredException") {
      state.errors.push(new Error("Please reset your password."));
      return;
    }

    if (err) {
      state.errors.push(err);
      return;
    }

    state.token = response?.AuthenticationResult?.AccessToken;
  },
  { skipWhen: skipWhenErrors },
);

const logSignInUserComplete = new ScenarioOutput(
  "logSignInUserComplete",
  (/** @type {State} */ state) =>
    `Successfully signed in. Your access token starts with:
 ${state.token.slice(0, 11)}`,
  { skipWhen: skipWhenErrors },
);

const confirmDeleteSignedInUser = new ScenarioInput(
  "confirmDeleteSignedInUser",
  "Do you want to delete the currently signed in user?",
  { type: "confirm", skipWhen: skipWhenErrors },
);

const deleteSignedInUser = new ScenarioAction(
  "deleteSignedInUser",
  async (/** @type {State} */ state) => {
    const [_, err] = await deleteUser({
      region: state.stackRegion,
      accessToken: state.token,
    });

    if (err) {
      state.errors.push(err);
```

```
    }
  },
  {
    skipWhen: (/** @type {State} */ state) =>
      skipWhenErrors(state) || !state.confirmDeleteSignedInUser,
  },
);

const logErrors = new ScenarioOutput(
  "logErrors",
  (/** @type {State}*/ state) => {
    const errorList = state.errors
      .map((err) => ` - ${err.name}: ${err.message}`)
      .join("\n");
    return `Scenario errors found:\n${errorList}`;
  },
  {
    // Don't log errors when there aren't any!
    skipWhen: (/** @type {State} */ state) => state.errors.length === 0,
  },
);

export const AutoConfirm = (context) =>
  new Scenario(
    "AutoConfirm",
    [
      promptForStackName,
      promptForStackRegion,
      getStackOutputs,
      greeting,
      logPopulatingUsers,
      populateUsers,
      logPopulatingUsersComplete,
      logSetupSignUpTrigger,
      setupSignUpTrigger,
      logSetupSignUpTriggerComplete,
      selectUser,
      checkIfUserAlreadyExists,
      createPassword,
      logSignUpExistingUser,
      signUpExistingUser,
      logSignUpExistingUserComplete,
      logLambdaLogs,
      logSignInUser,
```

```
      signInUser,
      logSignInUserComplete,
      confirmDeleteSignedInUser,
      deleteSignedInUser,
      logCleanUpReminder,
      logErrors,
    ],
    context,
  );
```

Dies sind Schritte, die mit anderen Szenarien geteilt werden.

```
import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { getCfnOutputs } from "@aws-doc-sdk-examples/lib/sdk/cfn-outputs.js";

export const skipWhenErrors = (state) => state.errors.length > 0;

export const getStackOutputs = new ScenarioAction(
  "getStackOutputs",
  async (state) => {
    if (!state.stackName || !state.stackRegion) {
      state.errors.push(
        new Error(
          "No stack name or region provided. The stack name and \
region are required to fetch CFN outputs relevant to this example.",
        ),
      );
      return;
    }

    const outputs = await getCfnOutputs(state.stackName, state.stackRegion);
    Object.assign(state, outputs);
  },
);

export const promptForStackName = new ScenarioInput(
  "stackName",
  "Enter the name of the stack you deployed earlier.",
```

```
  { type: "input", default: "PoolsAndTriggersStack" },
);

export const promptForStackRegion = new ScenarioInput(
  "stackRegion",
  "Enter the region of the stack you deployed earlier.",
  { type: "input", default: "us-east-1" },
);

export const logCleanUpReminder = new ScenarioOutput(
  "logCleanUpReminder",
  "All done. Remember to run 'cdk destroy' to teardown the stack.",
  { skipWhen: skipWhenErrors },
);
```

Ein Handler für den `PreSignUp`-Trigger mit einer Lambda-Funktion.

```
import type { PreSignUpTriggerEvent, Handler } from "aws-lambda";
import type { UserRepository } from "./user-repository";
import { DynamoDBUserRepository } from "./user-repository";

export class PreSignUpHandler {
  private userRepository: UserRepository;

  constructor(userRepository: UserRepository) {
    this.userRepository = userRepository;
  }

  private isPreSignUpTriggerSource(event: PreSignUpTriggerEvent): boolean {
    return event.triggerSource === "PreSignUp_SignUp";
  }

  private getEventUserEmail(event: PreSignUpTriggerEvent): string {
    return event.request.userAttributes.email;
  }

  async handlePreSignUpTriggerEvent(
    event: PreSignUpTriggerEvent,
  ): Promise<PreSignUpTriggerEvent> {
    console.log(
      `Received presignup from ${event.triggerSource} for user
 '${event.userName}'`,
```

```
    );

    if (!this.isPreSignUpTriggerSource(event)) {
      return event;
    }

    const eventEmail = this.getEventUserEmail(event);
    console.log(`Looking up email ${eventEmail}.`);
    const storedUserInfo =
      await this.userRepository.getUserInfoByEmail(eventEmail);

    if (!storedUserInfo) {
      console.log(
        `Email ${eventEmail} not found. Email verification is required.`,
      );
      return event;
    }

    if (storedUserInfo.UserName !== event.userName) {
      console.log(
        `UserEmail ${eventEmail} found, but stored UserName
 '${storedUserInfo.UserName}' does not match supplied UserName
 '${event.userName}'. Verification is required.`,
      );
    } else {
      console.log(
        `UserEmail ${eventEmail} found with matching UserName
 ${storedUserInfo.UserName}. User is confirmed.`,
      );
      event.response.autoConfirmUser = true;
      event.response.autoVerifyEmail = true;
    }
    return event;
  }
}

const createPreSignUpHandler = (): PreSignUpHandler => {
  const tableName = process.env.TABLE_NAME;
  if (!tableName) {
    throw new Error("TABLE_NAME environment variable is not set");
  }

  const userRepository = new DynamoDBUserRepository(tableName);
  return new PreSignUpHandler(userRepository);
```

```
};

export const handler: Handler = async (event: PreSignUpTriggerEvent) => {
  const preSignUpHandler = createPreSignUpHandler();
  return preSignUpHandler.handlePreSignUpTriggerEvent(event);
};
```

Modul für CloudWatch Logs-Aktionen.

```
import {
  CloudWatchLogsClient,
  GetLogEventsCommand,
  OrderBy,
  paginateDescribeLogStreams,
} from "@aws-sdk/client-cloudwatch-logs";

/**
 * Get the latest log stream for a Lambda function.
 * @param {{ functionName: string, region: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cloudwatch-logs").LogStream | null,
 unknown]>}
 */
export const getLatestLogStreamForLambda = async ({ functionName, region }) => {
  try {
    const logGroupName = `/aws/lambda/${functionName}`;
    const cwlClient = new CloudWatchLogsClient({ region });
    const paginator = paginateDescribeLogStreams(
      { client: cwlClient },
      {
        descending: true,
        limit: 1,
        orderBy: OrderBy.LastEventTime,
        logGroupName,
      },
    );

    for await (const page of paginator) {
      return [page.logStreams[0], null];
    }
  } catch (err) {
    return [null, err];
```

```
  }
};

/**
 * Get the log events for a Lambda function's log stream.
 * @param {{
 *   functionName: string,
 *   logStreamName: string,
 *   eventCount: number,
 *   region: string
 * }} config
 * @returns {Promise<[import("@aws-sdk/client-cloudwatch-logs").OutputLogEvent[]
 | null, unknown]>}
 */
export const getLogEvents = async ({
  functionName,
  logStreamName,
  eventCount,
  region,
}) => {
  try {
    const cwlClient = new CloudWatchLogsClient({ region });
    const logGroupName = `/aws/lambda/${functionName}`;
    const response = await cwlClient.send(
      new GetLogEventsCommand({
        logStreamName: logStreamName,
        limit: eventCount,
        logGroupName: logGroupName,
      }),
    );

    return [response.events, null];
  } catch (err) {
    return [null, err];
  }
};
```

Modul für Amazon-Cognito-Aktionen.

```
import {
  AdminGetUserCommand,
```

```
  CognitoIdentityProviderClient,
  DeleteUserCommand,
  InitiateAuthCommand,
  SignUpCommand,
  UpdateUserPoolCommand,
} from "@aws-sdk/client-cognito-identity-provider";

/**
 * Connect a Lambda function to the PreSignUp trigger for a Cognito user pool
 * @param {{ region: string, userPoolId: string, handlerArn: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").UpdateUserPoolCommandOutput | null, unknown]>}
 */
export const addPreSignUpHandler = async ({
  region,
  userPoolId,
  handlerArn,
}) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({
      region,
    });

    const command = new UpdateUserPoolCommand({
      UserPoolId: userPoolId,
      LambdaConfig: {
        PreSignUp: handlerArn,
      },
    });

    const response = await cognitoClient.send(command);
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Attempt to register a user to a user pool with a given username and password.
 * @param {{
 *   region: string,
 *   userPoolClientId: string,
 *   username: string,
 *   email: string,
```

```
 *   password: string
 * }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").SignUpCommandOutput | null, unknown]>}
 */
export const signUpUser = async ({
  region,
  userPoolClientId,
  username,
  email,
  password,
}) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({
      region,
    });

    const response = await cognitoClient.send(
      new SignUpCommand({
        ClientId: userPoolClientId,
        Username: username,
        Password: password,
        UserAttributes: [{ Name: "email", Value: email }],
      }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Sign in a user to Amazon Cognito using a username and password authentication
 flow.
 * @param {{ region: string, clientId: string, username: string, password:
 string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").InitiateAuthCommandOutput | null, unknown]>}
 */
export const signIn = async ({ region, clientId, username, password }) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({ region });
    const response = await cognitoClient.send(
      new InitiateAuthCommand({
```

```
        AuthFlow: "USER_PASSWORD_AUTH",
        ClientId: clientId,
        AuthParameters: { USERNAME: username, PASSWORD: password },
      }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Retrieve an existing user from a user pool.
 * @param {{ region: string, userPoolId: string, username: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").AdminGetUserCommandOutput | null, unknown]>}
 */
export const getUser = async ({ region, userPoolId, username }) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({ region });
    const response = await cognitoClient.send(
      new AdminGetUserCommand({
        UserPoolId: userPoolId,
        Username: username,
      }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Delete the signed-in user. Useful for allowing a user to delete their
 * own profile.
 * @param {{ region: string, accessToken: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").DeleteUserCommandOutput | null, unknown]>}
 */
export const deleteUser = async ({ region, accessToken }) => {
  try {
    const client = new CognitoIdentityProviderClient({ region });
    const response = await client.send(
      new DeleteUserCommand({ AccessToken: accessToken }),
```

```
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};
```

Modul für DynamoDB-Aktionen.

```
import { DynamoDBClient } from "@aws-sdk/client-dynamodb";
import {
  BatchWriteCommand,
  DynamoDBDocumentClient,
} from "@aws-sdk/lib-dynamodb";

/**
 * Populate a DynamoDB table with provide items.
 * @param {{ region: string, tableName: string, items: Record<string,
 unknown>[] }} config
 * @returns {Promise<[import("@aws-sdk/lib-dynamodb").BatchWriteCommandOutput |
 null, unknown]>}
 */
export const populateTable = async ({ region, tableName, items }) => {
  try {
    const ddbClient = new DynamoDBClient({ region });
    const docClient = DynamoDBDocumentClient.from(ddbClient);
    const response = await docClient.send(
      new BatchWriteCommand({
        RequestItems: {
          [tableName]: items.map((item) => ({
            PutRequest: {
              Item: item,
            },
          })),
        },
      }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
```

```
};
```

- API-Details finden Sie in den folgenden Themen der AWS SDK für JavaScript -API-Referenz.
  - DeleteUser
  - InitiateAuth
  - SignUp
  - UpdateUserPool

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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-Benutzer mit einer Lambda-Funktion automatisch migriert werden können.

- 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 durchsucht eine DynamoDB-Tabelle und migriert bekannte Benutzer automatisch in den Benutzerpool.
- Führen Sie den Ablauf „Passwort vergessen“ aus, um das Passwort für den migrierten Benutzer zurückzusetzen.
- Melden Sie sich als neuer Benutzer an und bereinigen Sie dann die Ressourcen.

Go

SDK für Go V2

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

Führen Sie ein interaktives Szenario an einem Prompt aus.

```
import (
 "context"
 "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(ctx context.Context, 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(
  ctx, 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(ctx context.Context, 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(ctx, 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(ctx, 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(ctx context.Context, 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(ctx, 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(ctx, 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(ctx, 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(ctx context.Context, stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup(ctx)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(ctx, stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]

 runner.AddMigrateUserTrigger(ctx, stackOutputs["UserPoolId"],
 stackOutputs["MigrateUserFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers,
 actions.UserMigration)
 resetNeeded, user := runner.SignInUser(ctx, stackOutputs["TableName"],
 stackOutputs["UserPoolClientId"])
 if resetNeeded {
  runner.helper.ListRecentLogEvents(ctx, stackOutputs["MigrateUserFunction"])
  runner.ResetPassword(ctx, stackOutputs["UserPoolClientId"], user)
 }

 runner.resources.Cleanup(ctx)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Behandeln Sie den `MigrateUser`-Trigger mit einer Lambda-Funktion.

```
import (
 "context"
 "log"
 "os"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/expression"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
)

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 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() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Erstellen Sie eine Struktur, die allgemeine Aufgaben ausführt.

```
import (
 "context"
 "log"
 "strings"
 "time"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
```

```
 GetStackOutputs(ctx context.Context, stackName string) (actions.StackOutputs,
 error)
 PopulateUserTable(ctx context.Context, tableName string)
 GetKnownUsers(ctx context.Context, tableName string) (actions.UserList, error)
 AddKnownUser(ctx context.Context, tableName string, user actions.User)
 ListRecentLogEvents(ctx context.Context, 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(ctx context.Context, stackName
 string) (actions.StackOutputs, error) {
```

```
 return helper.cfnActor.GetOutputs(ctx, stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(ctx context.Context, 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(ctx, tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(ctx context.Context, tableName string)
 (actions.UserList, error) {
 knownUsers, err := helper.dynamoActor.Scan(ctx, 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(ctx context.Context, 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(ctx, 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(ctx context.Context,
 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(ctx, functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(ctx, 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-Aktionen umschließt.

```
import (
 "context"
 "errors"
 "log"

 "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"
)

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(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &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(ctx,
 &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(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &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(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &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(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &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(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &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(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &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(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &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(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &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 umhüllt.

```
import (
```

```
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

// 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(ctx context.Context, 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(ctx, &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(ctx context.Context, tableName string) (UserList,
 error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(ctx, &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(ctx context.Context, 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(ctx, &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 CloudWatch Logs-Aktionen umschließt.

```
import (
 "context"
 "fmt"
 "log"

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

type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(ctx context.Context,
 functionName string) (types.LogStream, error) {
```

```
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(ctx,
 &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(ctx context.Context, 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(ctx,
 &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

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/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(ctx context.Context, stackName
 string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(ctx,
 &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
}
```

Bereinigen von Ressourcen.

```
import (
 "context"
 "log"
```

```
 "user_pools_and_lambda_triggers/actions"

 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// 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(ctx context.Context) {
 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(ctx, 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(ctx, 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 für Go -API-Referenz.
  - ConfirmForgotPassword
  - DeleteUser
  - ForgotPassword
  - InitiateAuth
  - SignUp
  - UpdateUserPool

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für .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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 für 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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 { logger } 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);
    logger.log("Signing up.");
    await signUp({ clientId, username, password, email });
    logger.log(`Signed up. A confirmation email has been sent to: ${email}.`);
    logger.log(
      `Run 'confirm-sign-up ${username} <code>' to confirm your account.`,
    );
  } catch (err) {
    logger.error(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 { logger } 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);
    logger.log("Confirming user.");
    await confirmSignUp({ clientId, username, code });
    logger.log(
      `User confirmed. Run 'admin-initiate-auth ${username} <password>' to sign
 in.`,
    );
  } catch (err) {
    logger.error(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 { logger } 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);

    logger.log("Signing in.");
    const { ChallengeName, Session } = await adminInitiateAuth({
      clientId,
      userPoolId,
      username,
      password,
    });
```

```
    if (ChallengeName === "MFA_SETUP") {
      logger.log("MFA setup is required.");
      return handleMfaSetup(Session, username);
    }

    if (ChallengeName === "SOFTWARE_TOKEN_MFA") {
      handleSoftwareTokenMfa(Session);
      logger.log(`Run 'admin-respond-to-auth-challenge ${username} <totp>'`);
    }
  } catch (err) {
    logger.error(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 { logger } 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);

    logger.log("Successfully authenticated.");
  } catch (err) {
    logger.error(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 { logger } 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);

    logger.log("Verifying TOTP.");
    await verifySoftwareToken(totp);
    logger.log("TOTP Verified. Run 'admin-initiate-auth' again to sign-in.");
  } catch (err) {
    logger.error(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 für 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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.fromEnvironment { 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. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- 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

Swift

SDK für Swift

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

Die Datei `Package.swift`

```
// swift-tools-version: 5.9
//
// The swift-tools-version declares the minimum version of Swift required to
// build this package.

import PackageDescription

let package = Package(
    name: "cognito-scenario",
    // Let Xcode know the minimum Apple platforms supported.
    platforms: [
        .macOS(.v13),
        .iOS(.v15)
    ],
    dependencies: [
        // Dependencies declare other packages that this package depends on.
        .package(
            url: "https://github.com/awslabs/aws-sdk-swift",
            from: "1.0.0"),
        .package(
            url: "https://github.com/apple/swift-argument-parser.git",
            branch: "main"
        )
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module
 or a test suite.
        // Targets can depend on other targets in this package and products
        // from dependencies.
        .executableTarget(
            name: "cognito-scenario",
```

```
            dependencies: [
                .product(name: "AWSCognitoIdentityProvider", package: "aws-sdk-
swift"),
                .product(name: "ArgumentParser", package: "swift-argument-
parser")
            ],
            path: "Sources")

    ]
)
```

Die Swift-Codedatei.

```
// An example demonstrating various features of Amazon Cognito. Before running
// this Swift 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 example performs the following functions:
//
// 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.
```

```
import ArgumentParser
import Foundation

import AWSClientRuntime
import AWSCognitoIdentityProvider

struct ExampleCommand: ParsableCommand {
    @Argument(help: "The application clientId.")
    var clientId: String
    @Argument(help: "The user pool ID to use.")
    var poolId: String
    @Option(help: "Name of the Amazon Region to use")
    var region = "us-east-1"

    static var configuration = CommandConfiguration(
        commandName: "cognito-scenario",
        abstract: """
        Demonstrates various features of Amazon Cognito.
        """,
        discussion: """
        """
    )

    /// Prompt for an input string of at least a minimum length.
    ///
    /// - Parameters:
    ///   - prompt: The prompt string to display.
    ///   - minLength: The minimum number of characters to allow in the
    ///     response. Default value is 0.
    ///
    /// - Returns: The entered string.
    func stringRequest(_ prompt: String, minLength: Int = 1) -> String {
        while true {
            print(prompt, terminator: "")
            let str = readLine()

            guard let str else {
                continue
            }
            if str.count >= minLength {
                return str
            } else {
```

```
                print("*** Response must be at least \(minLength) character(s)
long.")
            }
        }
    }

    /// Ask a yes/no question.
    ///
    /// - Parameter prompt: A prompt string to print.
    ///
    /// - Returns: `true` if the user answered "Y", otherwise `false`.
    func yesNoRequest(_ prompt: String) -> Bool {
        while true {
            let answer = stringRequest(prompt).lowercased()
            if answer == "y" || answer == "n" {
                return answer == "y"
            }
        }
    }

    /// Get information about a specific user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The Amazon Cognito Identity Provider client to use.
    ///   - userName: The user to retrieve information about.
    ///   - userPoolId: The user pool to search for the specified user.
    ///
    /// - Returns: `true` if the user's information was successfully
    ///   retrieved. Otherwise returns `false`.
    func adminGetUser(cipClient: CognitoIdentityProviderClient, userName: String,
                      userPoolId: String) async -> Bool {
        do {
            let output = try await cipClient.adminGetUser(
                input: AdminGetUserInput(
                    userPoolId: userPoolId,
                    username: userName
                )
            )

            guard let userStatus = output.userStatus else {
                print("*** Unable to get the user's status.")
                return false
            }
```

```
            print("User status: \(userStatus)")
            return true
        } catch {
            return false
        }
    }

    /// Create a new user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The ID of the app client to create a user for.
    ///   - userName: The username for the new user.
    ///   - password: The new user's password.
    ///   - email: The new user's email address.
    ///
    /// - Returns: `true` if successful; otherwise `false`.
    func signUp(cipClient: CognitoIdentityProviderClient, clientId: String,
userName: String, password: String, email: String) async -> Bool {
        let emailAttr = CognitoIdentityProviderClientTypes.AttributeType(
            name: "email",
            value: email
        )

        let userAttrsList = [emailAttr]

        do {
            _ = try await cipClient.signUp(
                input: SignUpInput(
                    clientId: clientId,
                    password: password,
                    userAttributes: userAttrsList,
                    username: userName
                )

            )

            print("=====> User \(userName) signed up.")
        } catch _ as AWSCognitoIdentityProvider.UsernameExistsException {
            print("*** The username \(userName) already exists. Please use a
different one.")
            return false
        } catch let error as AWSCognitoIdentityProvider.InvalidPasswordException
{
```

```
            print("*** Error: The specified password is invalid. Reason:
\(error.properties.message ?? "<none available>").")
            return false
        } catch _ as AWSCognitoIdentityProvider.ResourceNotFoundException {
            print("*** Error: The specified client ID (\(clientId)) doesn't
exist.")
            return false
        } catch {
            print("*** Unexpected error: \(error)")
            return false
        }

        return true
    }

    /// Requests a new confirmation code be sent to the given user's contact
    /// method.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The application client ID.
    ///   - userName: The user to resend a code for.
    ///
    /// - Returns: `true` if a new code was sent successfully, otherwise
    ///   `false`.
    func resendConfirmationCode(cipClient: CognitoIdentityProviderClient,
clientId: String,
                                userName: String) async -> Bool {
        do {
            let output = try await cipClient.resendConfirmationCode(
                input: ResendConfirmationCodeInput(
                    clientId: clientId,
                    username: userName
                )
            )

            guard let deliveryMedium = output.codeDeliveryDetails?.deliveryMedium
else {
                print("*** Unable to get the delivery method for the resent
code.")
                return false
            }

            print("=====> A new code has been sent by \(deliveryMedium)")
```

```
            return true
        } catch {
            print("*** Unable to resend the confirmation code to user
\(userName).")
            return false
        }
    }

    /// Submit a confirmation code for the specified user. This is the code as
    /// entered by the user after they've received it by email or text
    /// message.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The app client ID the user is signing up for.
    ///   - userName: The username of the user whose code is being sent.
    ///   - code: The user's confirmation code.
    ///
    /// - Returns: `true` if the code was successfully confirmed; otherwise
`false`.
    func confirmSignUp(cipClient: CognitoIdentityProviderClient, clientId:
String,
                       userName: String, code: String) async -> Bool {
        do {
            _ = try await cipClient.confirmSignUp(
                input: ConfirmSignUpInput(
                    clientId: clientId,
                    confirmationCode: code,
                    username: userName
                )
            )

            print("=====> \(userName) has been confirmed.")
            return true
        } catch {
            print("=====> \(userName)'s code was entered incorrectly.")
            return false
        }
    }

    /// Begin an authentication session.
    ///
    /// - Parameters:
    ///   - cipClient: The `CongitoIdentityProviderClient` to use.
```

```
    ///   - clientId: The app client ID to use.
    ///   - userName: The username to check.
    ///   - password: The user's password.
    ///   - userPoolId: The user pool to use.
    ///
    /// - Returns: The session token associated with this authentication
    ///   session.
    func initiateAuth(cipClient: CognitoIdentityProviderClient, clientId: String,
                      userName: String, password: String,
                      userPoolId: String) async -> String? {
        var authParams: [String: String] = [:]

        authParams["USERNAME"] = userName
        authParams["PASSWORD"] = password

        do {
            let output = try await cipClient.adminInitiateAuth(
                input: AdminInitiateAuthInput(
                    authFlow:
CognitoIdentityProviderClientTypes.AuthFlowType.adminUserPasswordAuth,
                    authParameters: authParams,
                    clientId: clientId,
                    userPoolId: userPoolId
                )
            )

            guard let challengeName = output.challengeName else {
                print("*** Invalid response from the auth service.")
                return nil
            }

            print("=====> Response challenge is \(challengeName)")

            return output.session
        } catch _ as UserNotFoundException {
            print("*** The specified username, \(userName), doesn't exist.")
            return nil
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return nil
        } catch {
            print("*** An unexpected error occurred.")
            return nil
        }
```

```
    }

    /// Request and display an MFA secret token that the user should enter
    /// into their authenticator to set it up for the user account.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - authSession: The authentication session to request an MFA secret
    ///     for.
    ///
    /// - Returns: A string containing the MFA secret token that should be
    ///   entered into the authenticator software.
    func getSecretForAppMFA(cipClient: CognitoIdentityProviderClient,
authSession: String?) async -> String? {
        do {
            let output = try await cipClient.associateSoftwareToken(
                input: AssociateSoftwareTokenInput(
                    session: authSession
                )
            )

            guard let secretCode = output.secretCode else {
                print("*** Unable to get the secret code")
                return nil
            }

            print("=====> Enter this token into Google Authenticator:
\(secretCode)")
            return output.session
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return nil
        } catch {
            print("*** An unexpected error occurred getting the secret for the
app's MFA.")
            return nil
        }
    }

    /// Confirm that the user's TOTP authenticator is configured correctly by
    /// sending a code to it to check that it matches successfully.
    ///
    /// - Parameters:
    ///   - cipClient: The `CongnitoIdentityProviderClient` to use.
```

```
    ///   - session: An authentication session previously returned by an
    ///     `associateSoftwareToken()` call.
    ///   - mfaCode: The 6-digit code currently displayed by the user's
    ///     authenticator, as provided by the user.
    func verifyTOTP(cipClient: CognitoIdentityProviderClient, session: String?,
 mfaCode: String?) async {
        do {
            let output = try await cipClient.verifySoftwareToken(
                input: VerifySoftwareTokenInput(
                    session: session,
                    userCode: mfaCode
                )
            )

            guard let tokenStatus = output.status else {
                print("*** Unable to get the token's status.")
                return
            }
            print("=====> The token's status is: \(tokenStatus)")
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return
        } catch _ as CodeMismatchException {
            print("*** The specified MFA code doesn't match the expected value.")
            return
        } catch _ as UserNotFoundException {
            print("*** The specified username doesn't exist.")
            return
        } catch _ as UserNotConfirmedException {
            print("*** The user has not been confirmed.")
            return
        } catch {
            print("*** Error verifying the MFA token!")
            return
        }
    }

    /// Respond to the authentication challenge received from Cognito after
    /// initiating an authentication session. This involves sending a current
    /// MFA code to the service.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - userName: The user's username.
```

```
   ///   - clientId: The app client ID.
   ///   - userPoolId: The user pool to sign into.
   ///   - mfaCode: The 6-digit MFA code currently displayed by the user's
   ///     authenticator.
   ///   - session: The authentication session to continue processing.
   func adminRespondToAuthChallenge(cipClient: CognitoIdentityProviderClient,
userName: String,
                                    clientId: String, userPoolId: String,
mfaCode: String,
                                    session: String) async {
       print("=====> SOFTWARE_TOKEN_MFA challenge is generated...")

       var challengeResponsesOb: [String: String] = [:]
       challengeResponsesOb["USERNAME"] = userName
       challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

       do {
           let output = try await cipClient.adminRespondToAuthChallenge(
               input: AdminRespondToAuthChallengeInput(
                   challengeName:
CognitoIdentityProviderClientTypes.ChallengeNameType.softwareTokenMfa,
                   challengeResponses: challengeResponsesOb,
                   clientId: clientId,
                   session: session,
                   userPoolId: userPoolId
               )
           )

           guard let authenticationResult = output.authenticationResult else {
               print("*** Unable to get authentication result.")
               return
           }

           print("=====> Authentication result (JWTs are redacted):")
           print(authenticationResult)
       } catch _ as SoftwareTokenMFANotFoundException {
           print("*** The specified user pool isn't configured for MFA.")
           return
       } catch _ as CodeMismatchException {
           print("*** The specified MFA code doesn't match the expected value.")
           return
       } catch _ as UserNotFoundException {
           print("*** The specified username, \(userName), doesn't exist.")
           return
```

```
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return
        } catch let error as NotAuthorizedException {
            print("*** Unauthorized access. Reason: \(error.properties.message ??
"<unknown>")")
        } catch {
            print("*** Error responding to the MFA challenge.")
            return
        }
    }

    /// Called by ``main()`` to run the bulk of the example.
    func runAsync() async throws {
        let config = try await
CognitoIdentityProviderClient.CognitoIdentityProviderClientConfiguration(region:
region)
        let cipClient = CognitoIdentityProviderClient(config: config)

        print("""
              This example collects information about a user, then creates that
user in the
              specified user pool. Then, it enables Multi-Factor Authentication
(MFA) for that
              user by associating an authenticator application (such as Google
Authenticator
              or a password manager that supports TOTP). Then, the user uses a
code from their
              authenticator application to sign in.

              """)

        let userName = stringRequest("Please enter a new username: ")
        let password = stringRequest("Enter a password: ")
        let email = stringRequest("Enter your email address: ", minLength: 5)

        // Submit the sign-up request to AWS.

        print("==> Signing up user \(userName)...")
        if await signUp(cipClient: cipClient, clientId: clientId,
                        userName: userName, password: password,
                        email: email) == false {
            return
        }
```

```
        // Check the user's status. This time, it should come back "unconfirmed".

        print("==> Getting the status of user \(userName) from the user pool
(should be 'unconfirmed')...")
        if await adminGetUser(cipClient: cipClient, userName: userName,
userPoolId: poolId) == false {
            return
        }

        // Ask the user if they want a replacement code sent, such as if the
        // code hasn't arrived yet. If the user responds with a "yes," send a
        // new code.

        if yesNoRequest("==> A confirmation code was sent to \(userName). Would
you like to send a new code (Y/N)? ") {
            print("==> Sending a new confirmation code...")
            if await resendConfirmationCode(cipClient: cipClient, clientId:
clientId, userName: userName) == false {
                return
            }
        }

        // Ask the user to enter the confirmation code, then send it to Amazon
        // Cognito to verify it.

        let code = stringRequest("==> Enter the confirmation code sent to
\(userName): ")
        if await confirmSignUp(cipClient: cipClient, clientId: clientId,
userName: userName, code: code) == false {
            // The code didn't match. Your application may wish to offer to
            // re-send the confirmation code here and try again.
            return
        }

        // Check the user's status again. This time it should come back
        // "confirmed".

        print("==> Rechecking status of user \(userName) in the user pool (should
be 'confirmed')...")
        if await adminGetUser(cipClient: cipClient, userName: userName,
userPoolId: poolId) == false {
            return
        }
```

```
        // Check the challenge mode. Here, it should be "mfaSetup", indicating
        // that the user needs to add MFA before using it. This returns a
        // session that can be used to register MFA, or nil if an error occurs.

        let authSession = await initiateAuth(cipClient: cipClient, clientId:
clientId,
                                             userName: userName, password:
password,
                                             userPoolId: poolId)
        if authSession == nil {
            return
        }

        // Ask Cognito for an MFA secret token that the user should enter into
        // their authenticator software (such as Google Authenticator) or
        // password manager to configure it for this user account. This
        // returns a new session that should be used for the new stage of the
        // authentication process.

        let newSession = await getSecretForAppMFA(cipClient: cipClient,
authSession: authSession)
        if newSession == nil {
            return
        }

        // Ask the user to enter the current 6-digit code displayed by their
        // authenticator. Then verify that it matches the value expected for
        // the session.

        let mfaCode1 = stringRequest("==> Enter the 6-digit code displayed in
your authenticator: ",
                                     minLength: 6)
        await verifyTOTP(cipClient: cipClient, session: newSession, mfaCode:
mfaCode1)

        // Ask the user to authenticate now that the authenticator has been
        // configured. This creates a new session using the user's username
        // and password as already entered.

        print("\nNow starting the sign-in process for user \(userName)...\n")

        let session2 = await initiateAuth(cipClient: cipClient, clientId:
clientId,
```

```
                                    userName: userName, password: password,
 userPoolId: poolId)
        guard let session2 else {
            return
        }

        // Now that we have a new auth session, `session2`, ask the user for a
        // new 6-digit code from their authenticator, and send it to the auth
        // session.

        let mfaCode2 = stringRequest("==> Wait for your authenticator to show a
 new 6-digit code, then enter it: ",
                                    minLength: 6)
        await adminRespondToAuthChallenge(cipClient: cipClient, userName:
 userName,
                                          clientId: clientId, userPoolId: poolId,
                                          mfaCode: mfaCode2, session: session2)
    }
}

/// The program's asynchronous entry point.
@main
struct Main {
    static func main() async {
        let args = Array(CommandLine.arguments.dropFirst())

        do {
            let command = try ExampleCommand.parse(args)
            try await command.runAsync()
        } catch {
            ExampleCommand.exit(withError: error)
        }
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS SDK für Swift.
  - 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 Dienst mit einem AWS SDK. 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 veranschaulicht, wie benutzerdefinierte Aktivitätsdaten mit einer Lambda-Funktion nach der Amazon-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.
- Ermitteln Sie benutzerdefinierte Daten aus der DynamoDB-Tabelle, zeigen Sie diese an und bereinigen Sie dann die Ressourcen.

Go

SDK für Go V2

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

Führen Sie ein interaktives Szenario an einem Prompt aus.

```
import (
 "context"
 "errors"
 "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"
)

// 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(ctx context.Context, 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(ctx, 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(ctx, 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(ctx, 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(ctx context.Context, 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(
  ctx, 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(ctx context.Context, 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(ctx, 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(ctx context.Context, 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(ctx, 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(ctx context.Context, stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup(ctx)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(ctx, stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(ctx, stackOutputs["TableName"])
```

```
 userName, password := runner.AddUserToPool(ctx, stackOutputs["UserPoolId"],
 stackOutputs["TableName"])

 runner.AddActivityLogTrigger(ctx, stackOutputs["UserPoolId"],
 stackOutputs["ActivityLogFunctionArn"])
 runner.SignInUser(ctx, stackOutputs["UserPoolClientId"], userName, password)
 runner.helper.ListRecentLogEvents(ctx, stackOutputs["ActivityLogFunction"])
 runner.GetKnownUserLastLogin(ctx, stackOutputs["TableName"], userName)

 runner.resources.Cleanup(ctx)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Behandeln Sie den `PostAuthentication`-Trigger mit einer Lambda-Funktion.

```
import (
 "context"
 "fmt"
 "log"
 "os"
 "time"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 dynamodbtypes "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

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() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Erstellen Sie eine Struktur, die allgemeine Aufgaben ausführt.

```
import (
 "context"
 "log"
```

```
 "strings"
 "time"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(ctx context.Context, stackName string) (actions.StackOutputs,
 error)
 PopulateUserTable(ctx context.Context, tableName string)
 GetKnownUsers(ctx context.Context, tableName string) (actions.UserList, error)
 AddKnownUser(ctx context.Context, tableName string, user actions.User)
 ListRecentLogEvents(ctx context.Context, 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(ctx context.Context, stackName
 string) (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(ctx, stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(ctx context.Context, 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(ctx, tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(ctx context.Context, tableName string)
 (actions.UserList, error) {
 knownUsers, err := helper.dynamoActor.Scan(ctx, 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(ctx context.Context, 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(ctx, 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(ctx context.Context,
 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(ctx, functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(ctx, 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-Aktionen umschließt.

```
import (
 "context"
 "errors"
 "log"
```

```
 "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"
)

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(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &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(ctx,
 &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(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &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(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &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(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &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(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &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(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &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(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &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(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &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 umhüllt.

```
import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

// 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(ctx context.Context, 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(ctx, &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(ctx context.Context, tableName string) (UserList,
 error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(ctx, &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(ctx context.Context, 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(ctx, &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 CloudWatch Logs-Aktionen umschließt.

```
import (
 "context"
```

```
 "fmt"
 "log"

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

type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(ctx context.Context,
 functionName string) (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(ctx,
 &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(ctx context.Context, 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(ctx,
 &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

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/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(ctx context.Context, stackName
 string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(ctx,
 &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
}
```

Bereinigen von Ressourcen.

```
import (
 "context"
 "log"
 "user_pools_and_lambda_triggers/actions"

 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// 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(ctx context.Context) {
 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(ctx, 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(ctx, 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 für Go -API-Referenz.
  - AdminCreateUser
  - AdminSetUserPassword

- DeleteUser
- InitiateAuth
- UpdateUserPool

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. 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 Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarios anzeigen.

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

Codebeispiele

- Grundlegende Beispiele für die Verwendung von Amazon Cognito Sync AWS SDKs
  - Aktionen für Amazon Cognito Sync mit AWS SDKs
    - Verwenden Sie es ListIdentityPoolUsage mit einem AWS SDK

Grundlegende Beispiele für die Verwendung von Amazon Cognito Sync AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie die Grundlagen von Amazon Cognito Sync mit AWS SDKs verwenden können.

Beispiele

- Aktionen für Amazon Cognito Sync mit AWS SDKs
  - Verwenden Sie es ListIdentityPoolUsage mit einem AWS SDK

Aktionen für Amazon Cognito Sync mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie einzelne Amazon Cognito Sync-Aktionen mit AWS SDKs ausführen. Jedes Beispiel enthält einen Link zu GitHub, über den 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

- Verwenden Sie es ListIdentityPoolUsage mit einem AWS SDK

Verwenden Sie es **`ListIdentityPoolUsage`** mit einem AWS SDK

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 Dienst mit einem AWS SDK. 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 denselben Bedrohungsschutz, dieselbe 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.

Bei der verwalteten Anmeldung wird ein Sitzungscookie im Browser gesetzt, 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
- Bewährte Methoden für die Nutzung mehrerer Mandanten zwischen Apps und Clients
- Bewährte Methoden für die Mehrmandantenfähigkeit von Benutzergruppen
- Bewährte Methoden für Mehrmandantenfähigkeit mit benutzerdefinierten Attributen
- Bewährte Methoden für Mehrmandantenfähigkeit mit benutzerdefiniertem Umfang
- Sicherheitsempfehlungen für Mehrmandantenfähigkeit

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 bei mehreren App-Clients in Benutzerpools mit verwalteter Anmeldung anmelden können, ist die Mehrmandantenfähigkeit für Benutzerpools auch ideal, wenn Sie möchten, dass sich mehr als einer Ihrer Mandanten mit verwalteter Anmeldung 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 Mehrmandantenfähigkeit von Benutzergruppen

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.

Wann sollte Mehrmandantenfähigkeit für Gruppen implementiert werden

Wenn der Zugriff auf AWS Ressourcen Ihr Hauptanliegen ist. Gruppen in Amazon Cognito Cognito-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.

Bewährte Methoden für Mehrmandantenfähigkeit mit benutzerdefiniertem Umfang

Amazon Cognito unterstützt benutzerdefinierte OAuth 2.0-Bereiche für Ressourcenserver. Sie können die Mehrmandantenfähigkeit von App-Clients in Benutzerpools für machine-to-machine (M2M-) Autorisierungsmodelle mit benutzerdefinierten Bereichen implementieren. Die bereichsbasierte Mehrmandantenfähigkeit reduziert den Aufwand für die Implementierung von M2M-Multi-Tenancy, indem der Zugriff in Ihrem App-Client oder Ihrer Anwendungskonfiguration definiert wird.

Das folgende Diagramm zeigt eine Option für Mehrmandantenfähigkeit mit benutzerdefiniertem Umfang. Es zeigt jeden Mandanten mit einem dedizierten App-Client, der Zugriff auf relevante Bereiche in einem Benutzerpool hat.

Wann sollte eine Mehrmandantenfähigkeit mit benutzerdefiniertem Geltungsbereich implementiert werden

Wenn Sie die M2M-Autorisierung mit Kundenanmeldedaten in einem vertraulichen Client verwenden. Es hat sich bewährt, Ressourcenserver zu erstellen, die ausschließlich für einen App-Client bestimmt sind. Mehrmandantenfähigkeit mit benutzerdefiniertem Umfang kann anforderungs- oder clientabhängig sein.

Abhängig von der Anfrage

Implementieren Sie die Anwendungslogik, um nur die Bereiche anzufordern, die den Anforderungen Ihres Mandanten entsprechen. Beispielsweise kann ein App-Client Lese- und Schreibzugriff auf API A und API B gewähren, aber Mandantenanwendung A fordert nur den Lesebereich für API A und den Bereich an, der die Mieterschaft angibt. Dieses Modell ermöglicht komplexere Kombinationen von gemeinsam genutzten Bereichen zwischen Mandanten.

Abhängig vom Kunden

Fordern Sie in Ihren Autorisierungsanfragen alle Bereiche an, die einem App-Client zugewiesen sind. Lassen Sie dazu den `scope` Anforderungsparameter in Ihrer Anfrage an die weg. Token-Endpunkt Dieses Modell ermöglicht es App-Clients, die Zugriffsindikatoren zu speichern, die Sie Ihren benutzerdefinierten Bereichen hinzufügen möchten.

In beiden Fällen erhalten Ihre Anwendungen Zugriffstoken mit Bereichen, die ihre Rechte für Datenquellen angeben, von denen sie abhängig sind. Bereiche können Ihrer Anwendung auch andere Informationen bieten:

- Benennen Sie das Mietverhältnis
- Tragen Sie zur Protokollierung von Anfragen bei
- Geben Sie an APIs , dass die Anwendung für Abfragen autorisiert ist
- Informieren Sie über die ersten Checks für aktive Kunden.

Grad des Aufwands

Ein kundenspezifischer Mehrmandantenbetrieb erfordert je nach Umfang Ihrer Anwendung einen unterschiedlichen Aufwand. Sie müssen eine Anwendungslogik entwickeln, die es Ihren Anwendungen ermöglicht, Zugriffstoken zu analysieren und die entsprechenden API-Anfragen zu stellen.

Ein Ressourcenserverbereich hat beispielsweise das Format. `[resource server identifier]/[name]` Es ist unwahrscheinlich, dass die ID des Ressourcenservers für die Autorisierungsentscheidung aus dem Mandantenbereich relevant ist, weshalb der Bereichsname konsistent analysiert werden muss.

Beispiel für eine Ressource

Die folgende AWS CloudFormation Vorlage erstellt einen Benutzerpool für Mehrmandantenfähigkeit in benutzerdefiniertem Umfang mit einem Ressourcenserver und einem App-Client.

```
AWSTemplateFormatVersion: "2010-09-09"
Description: A sample template illustrating scope-based multi-tenancy
Resources:
  MyUserPool:
    Type: "AWS::Cognito::UserPool"
  MyUserPoolDomain:
    Type: AWS::Cognito::UserPoolDomain
    Properties:
      UserPoolId: !Ref MyUserPool
      # Note that the value for "Domain" must be unique across all of AWS.
      # In production, you may want to consider using a custom domain.
      # See: https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-
pools-add-custom-domain.html#cognito-user-pools-add-custom-domain-adding
      Domain: !Sub "example-userpool-domain-${AWS::AccountId}"
  MyUserPoolResourceServer:
    Type: "AWS::Cognito::UserPoolResourceServer"
    Properties:
      Identifier: resource1
      Name: resource1
      Scopes:
        - ScopeDescription: Read-only access
          ScopeName: readScope
      UserPoolId: !Ref MyUserPool
  MyUserPoolTenantBatch1ResourceServer:
    Type: "AWS::Cognito::UserPoolResourceServer"
    Properties:
      Identifier: TenantBatch1
```

```
      Name: TenantBatch1
      Scopes:
        - ScopeDescription: tenant1 identifier
          ScopeName: tenant1
        - ScopeDescription: tenant2 identifier
          ScopeName: tenant2
      UserPoolId: !Ref MyUserPool
  MyUserPoolClientTenant1:
    Type: "AWS::Cognito::UserPoolClient"
    Properties:
      AllowedOAuthFlows:
        - client_credentials
      AllowedOAuthFlowsUserPoolClient: true
      AllowedOAuthScopes:
        - !Sub "${MyUserPoolTenantBatch1ResourceServer}/tenant1"
        - !Sub "${MyUserPoolResourceServer}/readScope"
      GenerateSecret: true
      UserPoolId: !Ref MyUserPool
Outputs:
  UserPoolClientId:
    Description: User pool client ID
    Value: !Ref MyUserPoolClientTenant1
  UserPoolDomain:
    Description: User pool domain
    Value: !Sub "https://${MyUserPoolDomain}.auth.${AWS::Region}.amazoncognito.com"
```

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
- Greifen Sie mit Benutzerpool-Token auf Back-End-Ressourcen zu
- Zugriff auf Ressourcen mit API Gateway und Lambda mit einem Benutzerpool
- Greifen Sie mit einem Benutzerpool und einem Identitätspool auf AWS Dienste zu
- Authentifizierung über einen Drittanbieter und Zugriff auf AWS -Services über einen Identitätspool
- Greifen Sie mit Amazon Cognito auf AWS AppSync Ressourcen zu

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 Ein Beispiel für eine Authentifizierungssitzung und JSON-Webtoken (JWTs) für den Benutzerpool verstehen.

Greifen Sie mit Benutzerpool-Token auf Back-End-Ressourcen zu

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. Mithilfe dieser OAuth 2.0-Grundlage können Sie Ihren eigenen Ressourcenserver erstellen, um Ihren Benutzern den Zugriff auf geschützte Ressourcen zu ermöglichen. Weitere Informationen finden Sie unter Bereiche, M2M und APIs mit Ressourcenservern.

Weitere Informationen über die Benutzerpool-Authentifizierung finden Sie unter Ein Beispiel für eine Authentifizierungssitzung und JSON-Webtoken (JWTs) für den Benutzerpool verstehen.

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 erhalten 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. Siehe AutorisierungAWS_IAM.

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
- Funktionspläne für Benutzerpools
- Bewährte Sicherheitsmethoden für Amazon-Cognito-Benutzerpools
- Authentifizierung mit Amazon Cognito Cognito-Benutzerpools
- Benutzerpool-Anmeldung mit externen Identitätsanbietern
- Vom Benutzerpool verwaltete Anmeldung
- Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern
- Verwalten von Benutzern in Ihrem Benutzerpool
- JSON-Webtoken (JWTs) für den Benutzerpool verstehen
- Zugriff auf Ressourcen nach erfolgreicher Anmeldung
- Bereiche, M2M und APIs mit Ressourcenservern
- Benutzerpoolfunktionen konfigurieren
- Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools
- Benutzerpool-Endpunkte und Referenz für verwaltete Logins

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-Anmeldung im Anmeldemenü unter Authentifizierung in der Benutzerpool-Konsole oder aktualisieren Sie den Wert von in einer oder API-Anfrage. AllowAdminCreateUserOnly CreateUserPool UpdateUserPool
> 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-Token, OAuth `userInfo` 2.0-Daten und SAML 2.0-Assertionen zu 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
- Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten
- 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 mit verwalteten Anmeldeseiten, die von Amazon Cognito gehostet werden, oder mit einem maßgeschneiderten Benutzerauthentifizierungsservice über die Amazon Cognito Cognito-Benutzerpools-API anmelden. Die Anwendungsebene hinter Ihrem maßgeschneiderten Frontend kann Anfragen am Backend mit einer von mehreren Methoden autorisieren, um legitime Anfragen zu bestätigen.

Benutzer können Benutzernamen und Passwörter, Hauptschlüssel sowie Einmalpasswörter für E-Mail- und SMS-Nachrichten einrichten und mit ihnen signieren. Sie können eine konsolidierte Anmeldung mit externen Benutzerverzeichnissen, Multi-Faktor-Authentifizierung (MFA) nach der Anmeldung, vertrauenswürdigen Geräten und von Ihnen entworfene benutzerdefinierte Authentifizierungsabläufe anbieten.

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 Kundenbenutzerdaten mit Social Sign-In OAuth 2.0. Amazon Cognito unterstützt die Anmeldung bei Google, Facebook, Amazon und Apple bis OAuth 2.0.
2. Melden Sie sich mit SAML- und OIDC-Anmeldung an und importieren Sie Benutzerdaten für Arbeit und Schule. 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

- Benutzerpool-Anmeldung mit externen Identitätsanbietern
- Verknüpfen von Verbundbenutzern mit einem vorhandenen Benutzerprofil

Machine-to-machine Autorisierung

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

- Bereiche, M2M und APIs mit Ressourcenservern

Verwaltete Anmeldung

Wenn Sie keine Benutzeroberfläche erstellen möchten, können Sie Ihren Benutzern benutzerdefinierte verwaltete Anmeldeseiten anbieten. Bei der verwalteten Anmeldung handelt es sich um eine Reihe von Webseiten für die Registrierung, Anmeldung, Multi-Faktor-Authentifizierung (MFA) und das Zurücksetzen von Passwörtern. Sie können Ihrer bestehenden Domain eine verwaltete Anmeldung hinzufügen oder eine Präfix-ID in einer Subdomain verwenden. AWS

Verwandte Themen

- Vom Benutzerpool verwaltete Anmeldung
- Konfigurieren einer Benutzerpool-Domäne

Sicherheit

Ihre lokalen Benutzer können einen zusätzlichen Authentifizierungsfaktor mit einem Code aus einer SMS- oder E-Mail-Nachricht oder einer App bereitstellen, die Multi-Faktor-Authentifizierungscodes (MFA) generiert. Sie können Mechanismen erstellen, um MFA in Ihrer Anwendung einzurichten und zu verarbeiten, oder Sie können sie von Managed Login 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 der adaptiven Authentifizierung kann Amazon Cognito potenzielle böswillige Aktivitäten erkennen und Ihren Benutzer auffordern, MFA einzurichten oder die Anmeldung zu blockieren.

Wenn der Netzwerkverkehr zu Ihrem Benutzerpool bösartig sein könnte, können Sie ihn überwachen und über das Internet Maßnahmen ergreifen. AWS WAF ACLs

Verwandte Themen

- Hinzufügen der MFA zu einem Benutzerpool
- Erweiterte Sicherheit mit Bedrohungsschutz
- Ordnen Sie einem AWS WAF Benutzerpool eine Web-ACL zu

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 das vertraute Erscheinungsbild der verwalteten Anmeldung zu verleihen.

Verwandte Themen

- Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern
- Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen
- Branding auf verwaltete Anmeldeseiten anwenden

Überwachung und Analytik

Amazon Cognito Cognito-Benutzerpools protokollieren API-Anfragen, einschließlich Anfragen zur verwalteten Anmeldung, an AWS CloudTrail. In der Service Quotas Quotas-Konsole können Sie Leistungskennzahlen in Amazon CloudWatch Logs überprüfen, benutzerdefinierte Protokolle CloudWatch mit Lambda-Triggern per Push übertragen, die E-Mail- und SMS-Nachrichtenzustellung überwachen und das Volumen der API-Anfragen überwachen.

Mit dem Plus-Funktionsplan können Sie mithilfe von Technologie für automatisiertes Lernen die Versuche zur Benutzerauthentifizierung auf Anzeichen von Beeinträchtigungen hin überwachen und Risiken sofort beheben. Diese erweiterten Sicherheitsfunktionen protokollieren auch Benutzeraktivitäten in Ihrem Benutzerpool und optional in Amazon S3, CloudWatch Logs oder Amazon Data Firehose.

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

- Amazon Cognito anmelden AWS CloudTrail
- Kontingente und Nutzung in CloudWatch und Service Quotas verfolgen
- Exportieren von Protokollen aus Amazon Cognito Cognito-Benutzerpools
- Verwendung von Amazon Pinpoint für Benutzerpool-Analysen

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

Funktionspläne für Benutzerpools

Das Verständnis der Kosten ist ein entscheidender Schritt bei der Vorbereitung der Implementierung der Amazon Cognito Cognito-Benutzerpool-Authentifizierung. Amazon Cognito bietet Funktionspläne für Benutzerpools. Jeder Plan hat eine Reihe von Funktionen und monatliche Kosten pro aktivem Benutzer. Jeder Funktionsplan ermöglicht den Zugriff auf mehr Funktionen als der vorherige.

Benutzerpools verfügen über eine Vielzahl von Funktionen, die Sie ein- und ausschalten können. Sie können beispielsweise die Multi-Faktor-Authentifizierung (MFA) aktivieren und die Anmeldung bei externen Identitätsanbietern () deaktivieren. IdPs Bei einigen Änderungen müssen Sie Ihren Funktionsplan wechseln. Die folgenden Merkmale Ihres Benutzerpools bestimmen die Kosten, die Ihnen monatlich für die Nutzung in AWS Rechnung gestellt werden.

- Die Funktionen, die Sie wählen
- Die Anfragen pro Sekunde, die Ihre Anwendung an die Benutzerpools-API stellt
- Die Anzahl der Benutzer mit Authentifizierungs-, Aktualisierungs- oder Abfrageaktivitäten in einem Monat, auch aktive Benutzer pro Monat genannt oder MAUs
- Die Anzahl der monatlich aktiven Benutzer von SAML 2.0 oder OpenID Connect (OIDC) eines Drittanbieters IdPs
- Die Anzahl der App-Clients und Benutzerpools, die Client-Anmeldeinformationen für die Autorisierung gewähren machine-to-machine

Die aktuellsten Informationen zu den Preisen für Benutzerpools finden Sie unter Amazon Cognito Cognito-Preise.

Die ausgewählten Funktionspläne gelten für einen Benutzerpool. Verschiedene Benutzerpools im selben Pool AWS-Konto können unterschiedliche Planauswahlen haben. Sie können keine separaten Featurepläne auf App-Clients innerhalb eines Benutzerpools anwenden. Die Standardplanauswahl für neue Benutzerpools ist Essentials.

Sie können jederzeit zwischen den Funktionsplänen wechseln, um sie an die Anforderungen Ihrer Anwendungen anzupassen. Bei einigen Änderungen zwischen den Plänen müssen Sie aktive Funktionen deaktivieren. Weitere Informationen finden Sie unter Funktionen ausschalten, um Funktionspläne zu ändern.

Funktionspläne für Benutzerpools

Leicht

Lite ist ein kostengünstiger Funktionsplan für Benutzerpools mit einer geringeren Anzahl von monatlich aktiven Benutzern. Dieser Plan ist ausreichend für Benutzerverzeichnisse mit grundlegenden Authentifizierungsfunktionen. Er umfasst Anmeldefunktionen und die klassische gehostete Benutzeroberfläche, einen schlankeren, weniger anpassbaren Vorgänger der verwalteten Anmeldung. Viele neuere Funktionen, wie die Anpassung von Zugriffstoken und die Authentifizierung mit dem Hauptschlüssel, sind nicht im Lite-Plan enthalten.

Grundlegendes

Essentials verfügt über die neuesten Authentifizierungsfunktionen für Benutzerpools. Dieser Plan erweitert Ihre Anwendungen um neue Optionen, unabhängig davon, ob es sich bei Ihren Anmeldeseiten um verwaltete Anmeldeseiten oder um benutzerdefinierte Anmeldeseiten handelt. Essentials bietet erweiterte Authentifizierungsfunktionen wie wahlbasierte Anmeldung und E-Mail-MFA.

Plus

Plus beinhaltet alles, was im Essentials-Plan enthalten ist, und bietet erweiterte Sicherheitsfunktionen, die Ihre Benutzer schützen. Überwachen Sie Benutzeranmeldungen, Registrierungsanfragen und Anfragen zur Passwortverwaltung auf Hinweise auf Sicherheitslücken. Benutzerpools können beispielsweise erkennen, ob sich Benutzer von einem unerwarteten Ort aus anmelden oder ein Passwort verwenden, das Teil einer öffentlichen Sicherheitsverletzung war.

Benutzerpools mit dem Plus-Plan generieren Protokolle mit Details zu Benutzeraktivitäten und Risikobewertungen. Sie können Ihre eigenen Nutzungs- und Sicherheitsanalysen auf diese Protokolle anwenden, wenn Sie sie in externe Dienste exportieren.

> **Note**
>
> Bisher waren einige Funktionen des Benutzerpools in der Preisstruktur für erweiterte Sicherheitsfunktionen enthalten. Die Funktionen, die in dieser Struktur enthalten waren, sind jetzt entweder im Essentials- oder Plus-Plan enthalten.

Themen

- Wählen Sie einen Funktionsplan aus
- Funktionen nach Tarif
- Funktionen des Essentials-Plans
- Funktionen des Plus-Plans
- Funktionen ausschalten, um Funktionspläne zu ändern

Wählen Sie einen Funktionsplan aus

AWS Management Console

Um einen Feature-Plan auszuwählen

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 das Menü Einstellungen und überprüfen Sie den Tab Funktionspläne.
5. Sehen Sie sich die Funktionen an, die Ihnen in den Lite-, Essentials- und Plus-Plänen zur Verfügung stehen.
6. Um Ihren Plan zu ändern, wählen Sie Zu Essentials wechseln oder Zu Plus wechseln aus. Um zum Lite-Plan zu wechseln, wähle Andere Tarife und dann Mit Lite vergleichen aus.
7. Überprüfen Sie auf dem nächsten Bildschirm Ihre Auswahl und wählen Sie Bestätigen aus.

CLI/API/SDK

Die UpdateUserPoolOperationen CreateUserPoolund legen Ihren Funktionsplan im `UserPoolTier` Parameter fest. Wenn Sie keinen Wert für angeben`UserPoolTier`, ist Ihr Benutzerpool standardmäßig auf`Essentials`. Wenn Sie `AdvancedSecurityMode` auf `AUDIT` oder setzen`ENFORCED`, muss Ihre Benutzerpool-Stufe lauten `PLUS` und die Standardeinstellung ist, `PLUS` wenn nichts angegeben ist.

Informationen CreateUserPool zur Syntax finden Sie in den Beispielen unter. Links zu dieser Funktion in oder CreateUserPool AWS SDKs für eine Vielzahl von Programmiersprachen finden Sie unter Auch in.

```
"UserPoolTier": "PLUS"
```

In der AWS CLI ist diese Option ein `--user-pool-tier` Argument.

```
--user-pool-tier PLUS
```

Weitere Informationen finden Sie unter create-user-poolund update-user-poolin der AWS CLI Befehlsreferenz.

Funktionen nach Tarif

Funktionen und Pläne in Benutzerpools

Funktion	Beschreibung	Funktionsplan
Schützen Sie sich vor unsicheren Passwörtern	Überprüfen Sie Klartext-Passwörter zur Laufzeit auf Anzeichen für kompromittierte Passwörter	Plus
Schützen Sie sich vor böswilligen Anmeldeversuchen	Überprüfen Sie die Sitzungseigenschaften zur Laufzeit auf Anzeichen einer Beeinträchtigung	Plus
Benutzeraktivitäten protokollieren und analysieren	Generieren Sie Protokolle mit den Eigenschaften und Risikobewertungen von Benutzerauthentifizierungssitzungen	Plus
Exportieren Sie Benutzeraktivitätsprotokolle	Senden Sie Benutzersitzungs- und Risikoprotokolle an ein externes AWS-Service	Plus
Passen Sie verwaltete Anmeldeseiten mit einem visuellen Editor an	Verwenden Sie einen visuellen Editor in der Amazon Cognito Cognito-Konsole, um Branding	Essentials + Plus

Funktion	Beschreibung	Funktionsplan
	und Stil auf Ihre verwalteten Anmeldeseiten anzuwenden	
MFA mit E-Mail-Einmalcodes	Fordere lokale Benutzer an oder fordere sie auf, nach der Benutzernamenauthentifizierung einen zusätzlichen Anmeldefaktor für E-Mail-Nachrichten anzugeben	Essentials + Plus
Passen Sie die Gültigkeitsbereiche und Ansprüche von Zugriffstoken zur Laufzeit an	Verwenden Sie einen Lambda-Trigger, um die Autorisierungsfunktionen von Zugriffstoken für Benutzerpools zu erweitern	Essentials + Plus
Passwortlose Anmeldung mit Einmalcodes	Erlauben Sie Benutzern, als ersten Authentifizierungsfaktor ein Einmalpasswort per E-Mail oder SMS zu erhalten	Essentials + Plus
Passkey-Anmeldung mit Hardware- oder Software-Authentifikatoren FIDO2	Erlauben Sie Benutzern, einen auf einem FIDO2 Authentifikator gespeicherten kryptografischen Schlüssel als ersten Authentifizierungsfaktor zu verwenden	Essentials + Plus
Melde dich an und melde dich an	Führen Sie Authentifizierungsvorgänge durch und lassen Sie neue Benutzer sich für ein Konto in Ihrer Anwendung registrieren.	Lite + Essentials + Plus

Funktion	Beschreibung	Funktionsplan
Benutzergruppen	Erstellen Sie logische Gruppierungen von Benutzern und weisen Sie Standard-IAM-Rollen für Identitätspool-Operationen zu.	Lite + Essentials + Plus
Melden Sie sich bei Social-, SAML- und OIDC-Anbietern an	Bieten Sie Benutzern die Möglichkeit, sich direkt oder bei ihrem bevorzugten Anbieter anzumelden.	Lite + Essentials + Plus
OAuth 2.0/OIDC-Autorisierungsserver	Als OIDC-Emittent agieren.	Lite + Essentials + Plus
Anmeldeseiten	Eine gehostete Sammlung von Webseiten zur Authentifizierung. Die verwaltete Anmeldung ist in den Stufen Essentials und Plus verfügbar. Die klassische gehostete Benutzeroberfläche ist in allen Funktionsstufen verfügbar.	Lite + Essentials + Plus
Passwort, benutzerdefinierte Authentifizierung, Aktualisierungstoken und SRP-Authentifizierung	Fordert Benutzer in Ihrer Anwendung zur Eingabe eines Benutzernamens und Kennworts auf.	Lite + Essentials + Plus
Machine-to-machine (M2M) mit Kundenanmeldedaten	Stellen Sie Zugriffstoken für die Autorisierung nichtmenschlicher Entitäten aus.	Lite + Essentials + Plus
API-Autorisierung mit Ressourcenservern	Stellen Sie Zugriffstoken mit benutzerdefinierten Bereichen aus, die den Zugriff auf externe Systeme autorisieren.	Lite + Essentials + Plus

Funktion	Beschreibung	Funktionsplan
Benutzer importieren	Richten Sie Importaufträge aus CSV-Dateien ein und führen Sie die just-in-time Migration der Benutzer bei der Anmeldung durch.	Lite + Essentials + Plus
MFA mit Authenticator-Apps und SMS-Einmalcodes	Fordere lokale Benutzer an oder fordere sie auf, nach der Benutzernamen-Auth entifizierung eine zusätzlic he SMS-Nachricht oder einen Anmeldefaktor für die Authenticator-App anzugeben	Lite + Essentials + Plus
Passen Sie die Gültigkei tsbereiche und Ansprüche von ID-Tokens zur Laufzeit an	Verwenden Sie einen Lambda-Trigger, um die Authentifizierungsfunktione n von Benutzerpool-Ident itätstoken (ID) zu erweitern	Lite + Essentials + Plus
Benutzerdefinierte Laufzeita ktionen mit Lambda-Triggern	Passen Sie den Anmeldevo rgang zur Laufzeit mit Lambda-Funktionen an, die externe Aktionen ausführen und die Authentifizierung beeinflussen	Lite + Essentials + Plus
Passen Sie verwaltete Anmeldeseiten mit CSS an	Laden Sie eine CSS-Vorla ge herunter und ändern Sie einige Stile auf Ihren verwaltet en Anmeldeseiten	Lite + Essentials + Plus

Funktionen des Essentials-Plans

Der Feature-Plan Essentials bietet die meisten der besten und neuesten Funktionen der Amazon Cognito Cognito-Benutzerpools. Wenn Sie vom Lite- zum Essentials-Plan wechseln, erhalten Sie neue Funktionen für Ihre verwalteten Anmeldeseiten, Multi-Faktor-Authentifizierung mit Einmalpasswörtern für E-Mail-Nachrichten, eine erweiterte Passwortrichtlinie und benutzerdefinierte Zugriffstoken. Wählen Sie up-to-date den Essentials-Plan für Ihre Benutzerpools, um die neuen Funktionen des Benutzerpools nutzen zu können.

Die folgenden Abschnitte bieten einen kurzen Überblick über die Funktionen, die Sie Ihrer Anwendung mit dem Essentials-Plan hinzufügen können. Ausführliche Informationen finden Sie auf den folgenden Seiten.

Weitere Ressourcen

- Anpassung des Zugriffstokens: Lambda-Auslöser für die Vorab-Generierung von Token
- E-Mail MFA: SMS- und E-Mail-Nachricht MFA
- Verlauf des Passworts: Passwörter, Kontowiederherstellung und Passwortrichtlinien
- Verbesserte Benutzeroberfläche: Branding auf verwaltete Anmeldeseiten anwenden

Themen

- Anpassen von Zugriffs-Token
- E-Mail MFA
- Verhinderung der Wiederverwendung von Passwörtern
- Verwaltete Anmeldung, gehosteter Anmelde- und Autorisierungsserver
- Wahlbasierte Authentifizierung

Anpassen von Zugriffs-Token

Zugriffstoken für Benutzerpools gewähren Anwendungen Berechtigungen: zum Zugriff auf eine API, zum Abrufen von Benutzerattributen vom UserInfo-Endpunkt oder zum Einrichten einer Gruppenmitgliedschaft für ein externes System. In fortgeschrittenen Szenarien möchten Sie den standardmäßigen Zugriffstokendaten aus dem Benutzerpoolverzeichnis möglicherweise zusätzliche temporäre Parameter hinzufügen, die Ihre Anwendung zur Laufzeit bestimmt. Möglicherweise möchten Sie beispielsweise die API-Berechtigungen eines Benutzers mit Amazon Verified Permissions überprüfen und die Bereiche im Zugriffstoken entsprechend anpassen.

Der Essentials-Plan erweitert die bestehenden Funktionen eines Triggers vor der Token-Generierung. Bei Tarifen niedrigerer Stufen können Sie ID-Token mit zusätzlichen Ansprüchen, Rollen und Gruppenmitgliedschaften personalisieren. Essentials fügt neue Versionen des Trigger-Eingabeereignisses hinzu, mit denen Ansprüche, Rollen, Gruppenmitgliedschaft und Geltungsbereiche auf Zugriffstoken angepasst werden können. Die Anpassung des Zugriffstokens ist für machine-to-machine (M2M-) Kundenanmeldeberechtigungen mit Eventversion 3 verfügbar.

Um Zugriffstoken anzupassen

1. Wählen Sie den Featureplan Essentials oder Plus aus.
2. Erstellen Sie eine Lambda-Funktion für Ihren Trigger. Um unsere Beispielfunktion zu verwenden, konfigurieren Sie sie für Node.js.
3. Füllen Sie Ihre Lambda-Funktion mit unserem Beispielcode oder verfassen Sie Ihren eigenen. Ihre Funktion muss ein Anforderungsobjekt von Amazon Cognito verarbeiten und die Änderungen zurückgeben, die Sie einbeziehen möchten.
4. Weisen Sie Ihre neue Funktion als Trigger der Version zwei oder drei vor der Token-Generierung zu. Ereignisse der Version zwei passen Zugriffstoken an Benutzeridentitäten an. Version drei passt Zugriffstoken für Benutzer- und Maschinenidentitäten an.

Weitere Informationen

- Anpassen des Zugriffs-Token
- So passen Sie Zugriffstoken in Amazon Cognito Cognito-Benutzerpools an

E-Mail MFA

Amazon Cognito Cognito-Benutzerpools können so konfiguriert werden, dass sie E-Mail als zweiten Faktor bei der Multi-Faktor-Authentifizierung (MFA) verwenden. Mit E-Mail-MFA kann Amazon Cognito Benutzern eine E-Mail mit einem Bestätigungscode senden, den sie eingeben müssen, um den Authentifizierungsprozess abzuschließen. Dies fügt dem Benutzeranmeldeablauf eine wichtige zusätzliche Sicherheitsebene hinzu. Um E-Mail-basiertes MFA zu aktivieren, muss der Benutzerpool so konfiguriert sein, dass er die Amazon SES SES-E-Mail-Versandkonfiguration anstelle der Standard-E-Mail-Konfiguration verwendet.

Wenn Ihr Benutzer MFA per E-Mail-Nachricht auswählt, sendet Amazon Cognito bei jedem Anmeldeversuch einen einmaligen Bestätigungscode an die registrierte E-Mail-Adresse des Benutzers. Der Benutzer muss diesen Code dann an Ihren Benutzerpool zurückgeben, um den

Authentifizierungsvorgang abzuschließen und Zugriff zu erhalten. Dadurch wird sichergestellt, dass ein Benutzer, selbst wenn der Benutzername und das Passwort kompromittiert werden, einen zusätzlichen Faktor — den per E-Mail gesendeten Code — angeben muss, bevor er auf Ihre Anwendungsressourcen zugreifen kann.

Weitere Informationen finden Sie unter SMS- und E-Mail-Nachricht MFA. Im Folgenden finden Sie eine Übersicht darüber, wie Sie Ihren Benutzerpool und Benutzer für E-Mail-MFA einrichten.

So richten Sie E-Mail-MFA in der Amazon Cognito Cognito-Konsole ein

1. Wählen Sie den Featureplan Essentials oder Plus aus.
2. Bearbeiten Sie im Anmeldemenü Ihres Benutzerpools die Option Multi-Faktor-Authentifizierung.
3. Wählen Sie die Ebene der MFA-Durchsetzung, die Sie einrichten möchten. Mit Require MFA erhalten Benutzer in der API automatisch eine Aufforderung, MFA einzurichten, zu bestätigen und sich mit MFA anzumelden. In Benutzerpools, die MFA erfordern, werden sie bei der verwalteten Anmeldung aufgefordert, einen MFA-Faktor auszuwählen und einzurichten. Mit optionalem MFA muss Ihre Anwendung Benutzern die Möglichkeit bieten, MFA einzurichten und die Benutzerpräferenz für E-Mail-MFA festzulegen.
4. Wählen Sie unter MFA-Methoden die Option E-Mail-Nachricht als eine der Optionen aus.

Weitere Informationen

- SMS- und E-Mail-Nachricht MFA

Verhinderung der Wiederverwendung von Passwörtern

Standardmäßig legt eine Amazon Cognito Cognito-Benutzerpool-Passwortrichtlinie die Anforderungen an die Passwortlänge und den Zeichentyp sowie den temporären Ablauf von Passwörtern fest. Der Essentials-Plan bietet die Möglichkeit, den Passwortverlauf durchzusetzen. Wenn ein Benutzer versucht, sein Passwort zurückzusetzen, kann Ihr Benutzerpool verhindern, dass er es auf ein vorheriges Passwort setzt. Weitere Informationen zur Konfiguration der Passwortrichtlinie finden Sie unterHinzufügen von Benutzerpool-Passwortanforderungen. Im Folgenden finden Sie eine Übersicht darüber, wie Sie Ihren Benutzerpool mit einer Richtlinie zur Kennworthistorie einrichten.

So richten Sie den Passwortverlauf in der Amazon Cognito Cognito-Konsole ein

1. Wählen Sie den Featureplan Essentials oder Plus aus.

2. Suchen Sie im Menü Authentifizierungsmethoden Ihres Benutzerpools nach Passwortrichtlinie und wählen Sie Bearbeiten aus.
3. Konfigurieren Sie andere verfügbare Optionen und legen Sie einen Wert für „Verwendung früherer Passwörter verhindern“ fest.

Weitere Informationen

- Passwörter, Kontowiederherstellung und Passwortrichtlinien

Verwaltete Anmeldung, gehosteter Anmelde- und Autorisierungsserver

Amazon Cognito Cognito-Benutzerpools verfügen über optionale Webseiten, die die folgenden Funktionen unterstützen: einen OpenID Connect (OIDC) IdP, einen Dienstanbieter oder eine vertrauende Partei eines IdPs Drittanbieters und öffentliche benutzerinteraktive Seiten für die Registrierung und Anmeldung. Diese Seiten werden zusammenfassend als verwaltete Anmeldung bezeichnet. Wenn Sie eine Domain für Ihren Benutzerpool auswählen, aktiviert Amazon Cognito diese Seiten automatisch. Wenn der Lite-Plan über die gehostete Benutzeroberfläche verfügt, öffnet der Essentials-Plan diese erweiterte Version der Anmelde- und Anmeldeseiten.

Verwaltete Anmeldeseiten verfügen über eine übersichtliche up-to-date Oberfläche mit mehr Funktionen und Optionen zur Anpassung Ihres Brandings und Ihrer Stile. Der Essentials-Plan ist der niedrigste Tarif, der den Zugriff auf die verwaltete Anmeldung freischaltet.

So richten Sie die verwaltete Anmeldung in der Amazon Cognito Cognito-Konsole ein

1. Wählen Sie im Menü Einstellungen den Featureplan Essentials oder Plus aus.
2. Weisen Sie Ihrem Benutzerpool im Menü Domain eine Domain zu und wählen Sie eine Branding-Version von Managed Login aus.
3. Wählen Sie im Menü Verwaltete Anmeldung auf der Registerkarte Stile die Option Stil erstellen aus und weisen Sie den Stil einem App-Client zu, oder erstellen Sie einen neuen App-Client.

Weitere Informationen

- Vom Benutzerpool verwaltete Anmeldung

Wahlbasierte Authentifizierung

Mit der Stufe „Essentials“ wird ein neuer Authentifizierungsablauf für Authentifizierungsvorgänge in der erweiterten Benutzeroberfläche und SDK-basierte API-Operationen eingeführt. Bei diesem Ablauf handelt es sich um eine wahlbasierte Authentifizierung. Die wahlbasierte Authentifizierung ist eine Methode, bei der die Authentifizierung Ihrer Benutzer nicht mit einer anwendungsseitigen Deklaration einer Anmeldemethode beginnt, sondern mit einer Abfrage möglicher Anmeldemethoden, gefolgt von einer Auswahl. Sie können Ihren Benutzerpool so konfigurieren, dass er die wahlbasierte Authentifizierung unterstützt und die Authentifizierung mit Benutzername/Passwort, kennwortloser Authentifizierung und Hauptschlüsselauthentifizierung entsperrt. In der API ist das der Ablauf. `USER_AUTH`

So richten Sie die wahlbasierte Authentifizierung in der Amazon Cognito Cognito-Konsole ein

1. Wählen Sie den Featureplan Essentials oder Plus aus.
2. Bearbeiten Sie im Anmeldemenü Ihres Benutzerpools die Optionen für die wahlbasierte Anmeldung. Wählen und konfigurieren Sie die Authentifizierungsmethoden, die Sie für die wahlbasierte Authentifizierung aktivieren möchten.
3. Bearbeiten Sie im Menü Authentifizierungsmethoden Ihres Benutzerpools die Konfiguration der Anmeldevorgänge.

Weitere Informationen

- Authentifizierung mit Amazon Cognito Cognito-Benutzerpools

Funktionen des Plus-Plans

Der Plus-Funktionsplan bietet erweiterte Sicherheitsfunktionen für Amazon Cognito Cognito-Benutzerpools. Diese Funktionen protokollieren und analysieren den Benutzerkontext zur Laufzeit im Hinblick auf potenzielle Sicherheitsprobleme bei Geräten, Standorten, Anforderungsdaten und Passwörtern. Anschließend minimieren sie potenzielle Risiken durch automatische Reaktionen, die Benutzerkonten blockieren oder Sicherheitsvorkehrungen hinzufügen. Sie können Ihre Sicherheitsprotokolle auch zur weiteren Analyse nach Amazon S3, Amazon Data Firehose oder Amazon CloudWatch Logs exportieren.

Wenn Sie vom Essentials- zum Plus-Plan wechseln, erhalten Sie alle Funktionen von Essentials und die folgenden zusätzlichen Funktionen. Dazu gehören die Sicherheitsoptionen zum Schutz

vor Bedrohungen, die auch als erweiterte Sicherheitsfunktionen bezeichnet werden. Um Ihre Benutzerpools so zu konfigurieren, dass sie sich automatisch an Bedrohungen in Ihrem Authentifizierungs-Frontend anpassen, wählen Sie den Plus-Plan für Ihre Benutzerpools.

Die folgenden Abschnitte bieten einen kurzen Überblick über die Funktionen, die Sie Ihrer Anwendung mit dem Plus-Plan hinzufügen können. Ausführliche Informationen finden Sie auf den folgenden Seiten.

Weitere Ressourcen

- Adaptive Authentifizierung: Arbeiten Sie mit adaptiver Authentifizierung
- Kompromittierte Anmeldeinformationen: Arbeiten mit der Erkennung kompromittierter Anmeldeinformationen
- Export von Protokollen: Exportieren von Protokollen aus Amazon Cognito Cognito-Benutzerpools

Themen

- Schutz vor Bedrohungen: adaptive Authentifizierung
- Bedrohungsschutz: Erkennung kompromittierter Anmeldeinformationen
- Bedrohungsschutz: Protokollierung von Benutzeraktivitäten

Schutz vor Bedrohungen: adaptive Authentifizierung

Der Plus-Plan beinhaltet eine adaptive Authentifizierungsfunktion. Wenn Sie diese Funktion aktivieren, führt Ihr Benutzerpool für jede Benutzerauthentifizierungssitzung eine Risikobewertung durch. Anhand der resultierenden Risikoeinstufungen können Sie die Authentifizierung blockieren oder MFA für Benutzer aktivieren, die sich mit einer Risikostufe anmelden, die über einem von Ihnen festgelegten Schwellenwert liegt. Mit der adaptiven Authentifizierung blockieren oder richten Ihr Benutzerpool und Ihre Anwendung automatisch MFA für Benutzer ein, deren Konten Sie vermuten, dass sie angegriffen werden. Sie können auch Feedback zu den Risikobewertungen aus Ihrem Benutzerpool geben, um future Bewertungen anzupassen.

So richten Sie die adaptive Authentifizierung in der Amazon Cognito Cognito-Konsole ein

1. Wählen Sie den Plus-Funktionsplan aus.
2. Bearbeiten Sie im Menü Bedrohungsschutz Ihres Benutzerpools unter Bedrohungsschutz die Optionen Standard- und benutzerdefinierte Authentifizierung.

3. Stellen Sie den Erzwingungsmodus für die Standard- oder benutzerdefinierte Authentifizierung auf Vollständig ein.
4. Konfigurieren Sie unter Adaptive Authentifizierung automatische Risikoreaktionen für verschiedene Risikostufen.

Weitere Informationen

- Arbeiten Sie mit adaptiver Authentifizierung
- Erfassung von Daten für den Schutz vor Bedrohungen in Anwendungen

Bedrohungsschutz: Erkennung kompromittierter Anmeldeinformationen

Der Plus-Plan beinhaltet eine Funktion zur Erkennung kompromittierter Anmeldeinformationen. Diese Funktion schützt vor der Verwendung unsicherer Passwörter und vor der Gefahr eines unbeabsichtigten Anwendungszugriffs, die durch diese Vorgehensweise entsteht. Wenn Sie Ihren Benutzern gestatten, sich mit einem Benutzernamen und einem Passwort anzumelden, verwenden sie möglicherweise ein Passwort, das sie an anderer Stelle verwendet haben. Dieses Passwort ist möglicherweise durchgesickert oder wird nur häufig erraten. Bei der Erkennung kompromittierter Anmeldeinformationen liest Ihr Benutzerpool die von Ihren Benutzern übermittelten Passwörter und vergleicht sie mit Kennwortdatenbanken. Wenn der Vorgang zu der Entscheidung führt, dass das Passwort wahrscheinlich kompromittiert wurde, können Sie Ihren Benutzerpool so konfigurieren, dass die Anmeldung blockiert wird, und dann ein Zurücksetzen des Kennworts für den Benutzer in Ihrer Anwendung veranlassen.

Die Erkennung kompromittierter Anmeldeinformationen kann auf unsichere Passwörter reagieren, wenn sich neue Benutzer anmelden, wenn sich bestehende Benutzer anmelden und wenn Benutzer versuchen, ihre Passwörter zurückzusetzen. Mit dieser Funktion kann Ihr Benutzerpool verhindern oder davor warnen, sich mit unsicheren Passwörtern anzumelden, egal wo Benutzer sie eingeben.

So richten Sie die Erkennung kompromittierter Anmeldeinformationen in der Amazon Cognito Cognito-Konsole ein

1. Wählen Sie den Plus-Funktionsplan aus.
2. Bearbeiten Sie im Menü Bedrohungsschutz Ihres Benutzerpools unter Bedrohungsschutz die Optionen Standard- und benutzerdefinierte Authentifizierung.
3. Stellen Sie den Erzwingungsmodus für die Standard- oder benutzerdefinierte Authentifizierung auf Vollständig ein.

4. Konfigurieren Sie unter Kompromittierte Anmeldeinformationen die Arten von Authentifizierungsvorgängen, die Sie überprüfen möchten, und die automatische Antwort, die Sie von Ihrem Benutzerpool erwarten.

Weitere Informationen

- Arbeiten mit der Erkennung kompromittierter Anmeldeinformationen

Bedrohungsschutz: Protokollierung von Benutzeraktivitäten

Der Plus-Plan bietet eine Protokollierungsfunktion, die Sicherheitsanalysen und Einzelheiten zu Benutzerauthentifizierungsversuchen enthält. Sie können Risikobewertungen, Benutzer-IP-Adressen, Benutzeragenten und andere Informationen über das Gerät einsehen, das eine Verbindung zu Ihrer Anwendung hergestellt hat. Sie können mit den integrierten Funktionen zum Schutz vor Bedrohungen auf diese Informationen reagieren oder Ihre Protokolle in Ihren eigenen Systemen analysieren und geeignete Maßnahmen ergreifen. Sie können die Protokolle von Threat Protection nach Amazon S3, CloudWatch Logs oder Amazon DynamoDB exportieren.

So richten Sie die Protokollierung von Benutzeraktivitäten in der Amazon Cognito Cognito-Konsole ein

1. Wählen Sie den Plus-Funktionsplan aus.
2. Bearbeiten Sie im Menü Bedrohungsschutz Ihres Benutzerpools unter Bedrohungsschutz die Optionen Standard- und benutzerdefinierte Authentifizierung.
3. Stellen Sie den Erzwingungsmodus für die Standard- oder benutzerdefinierte Authentifizierung auf Nur Audit ein. Dies ist die Mindesteinstellung für Protokolle. Sie können es auch im Vollfunktionsmodus aktivieren und andere Funktionen zum Schutz vor Bedrohungen konfigurieren.
4. Um Ihre Protokolle AWS-Service zur Analyse durch Dritte in ein anderes zu exportieren, rufen Sie das Log-Streaming-Menü Ihres Benutzerpools auf und richten Sie ein Exportziel ein.

Weitere Informationen

- Benutzerauthentifizierungsereignisse exportieren
- Exportieren von Protokollen aus Amazon Cognito Cognito-Benutzerpools

Funktionen ausschalten, um Funktionspläne zu ändern

Funktionspläne erweitern Ihren Benutzerpool um Konfigurationsoptionen. Sie können diese Funktionen nur konfigurieren und verwenden, wenn der zugehörige Funktionsplan aktiv ist. Sie können beispielsweise die Anpassung von Zugriffstoken in den Plus- und Essentials-Plänen konfigurieren, nicht jedoch im Lite-Plan. Um diese Funktionen zu deaktivieren, müssen Sie jede aktive Komponente deaktivieren. Die Option Wechseln zu im Menü Einstellungen in der Amazon Cognito Cognito-Konsole informiert Sie über die Funktionen, die Sie deaktivieren müssen, bevor Sie Ihren Funktionsplan ändern können. In diesem Kapitel erfahren Sie, welche Änderungen die Deaktivierung an Ihrer Benutzerpool-Konfiguration vornimmt und wie Sie diese Funktionen einzeln ausschalten können.

Anpassen von Zugriffs-Token

Um zu einem Plan zu wechseln, der keine Anpassung von Zugriffstoken beinhaltet, müssen Sie den Lambda-Trigger vor der Token-Generierung aus Ihrem Benutzerpool entfernen. Um einen neuen Trigger vor der Token-Generierung ohne Anpassung des Zugriffstokens hinzuzufügen, weisen Sie dem Trigger eine neue Funktion zu und konfigurieren Sie ihn für `V1_0` Ereignisse. Diese Trigger-Ereignisse der ersten Version können nur Änderungen an ID-Token verarbeiten.

Um die Anpassung von Zugriffstoken manuell zu deaktivieren, entfernen Sie Ihren Trigger vor der Token-Generierung und fügen Sie einen neuen Trigger der Version eins hinzu.

Schutz vor Bedrohungen

Um zu einem Tarif ohne Bedrohungsschutz zu wechseln, deaktivieren Sie alle Funktionen im Bedrohungsschutz-Menü Ihres Benutzerpools.

Export von Protokollen

Um zu einem Tarif ohne Protokollexport zu wechseln, deaktivieren Sie ihn im Log-Streaming-Menü Ihres Benutzerpools. Ihr Benutzerpool generiert keine lokalen oder exportierten Benutzeraktivitätsprotokolle mehr. Sie können auch eine SetLogDeliveryConfigurationAPI-Anfrage senden, die jede Konfiguration mit einem `EventSource` Wert von `UserActivity` entfernt.

E-Mail MFA

Um zu einem Tarif ohne E-Mail-MFA zu wechseln, rufen Sie das Anmeldemenü Ihres Benutzerpools auf. Bearbeiten Sie die Multi-Faktor-Authentifizierung und deaktivieren Sie E-Mail-Nachricht als eine der verfügbaren MFA-Methoden.

Bewährte Sicherheitsmethoden für Amazon-Cognito-Benutzerpools

Auf dieser Seite werden bewährte Sicherheitsmethoden beschrieben, die Sie implementieren können, wenn Sie sich vor häufigen Bedrohungen schützen möchten. Welche Konfiguration Sie wählen, hängt vom Anwendungsfall der jeweiligen Anwendung ab. Wir empfehlen, dass Sie mindestens die geringsten Rechte auf administrative Operationen anwenden und Maßnahmen ergreifen, um Anwendungs- und Benutzergeheimnisse zu schützen. Ein weiterer fortgeschrittener, aber effektiver Schritt, den Sie ergreifen können, besteht darin, das AWS WAF Internet ACLs zu konfigurieren und auf Ihre Benutzerpools anzuwenden.

Schützen Sie Ihren Benutzerpool auf Netzwerkebene

AWS WAF Web ACLs kann die Leistung und die Kosten der Authentifizierungsmechanismen schützen, die Sie mit Amazon Cognito entwickeln. Mit Web ACLs können Sie Leitplanken vor API- und verwalteten Anmeldeanfragen implementieren. Das Web ACLs erstellt Filter auf Netzwerk- und Anwendungsebene, die den Traffic unterbrechen können oder ein CAPTCHA erfordern, das auf von Ihnen festgelegten Regeln basiert. Anfragen werden erst dann an Ihre Amazon Cognito Cognito-Ressourcen weitergeleitet, wenn sie die Voraussetzungen in Ihren Web-ACL-Regeln erfüllen. Weitere Informationen finden Sie unter AWS WAF Web ACLs.

Schützen Sie sich vor Missbrauch von SMS-Nachrichten

Wenn Sie die öffentliche Registrierung in Ihrem Benutzerpool zulassen, können Sie die Kontoverifizierung mit Codes konfigurieren, die Amazon Cognito in SMS-Textnachrichten sendet. SMS-Nachrichten können mit unerwünschten Aktivitäten in Verbindung gebracht werden und Ihre AWS Rechnung erhöhen. Konfigurieren Sie Ihre Infrastruktur so, dass sie gegen das Senden von SMS-Nachrichten unter Betrugsfällen geschützt ist. Weitere Informationen finden Sie in den folgenden AWS Blog-Beiträgen.

- Reduzieren Sie mit Amazon Cognito Cognito-Benutzerpools das Risiko von Benutzeranmeldebetrug und SMS-Pumping
- Schutz vor SMS-Pumping: Neue AWS Funktionen zur Bekämpfung von künstlich aufgeblähtem Traffic

Verstehen Sie öffentliche Authentifizierung

Amazon Cognito Cognito-Benutzerpools verfügen über CIAM-Funktionen (Customer Identity and Access Management), die Anwendungsfälle unterstützen, in denen sich Mitglieder der

Öffentlichkeit für ein Benutzerkonto registrieren und auf Ihre Anwendungen zugreifen können. Wenn ein Benutzerpool die Self-Service-Registrierung zulässt, ist er offen für Anfragen nach Benutzerkonten aus dem öffentlichen Internet. Self-Service-Anfragen kommen aus API-Vorgängen wie SignUpund und InitiateAuthaus Benutzerinteraktionen mit verwalteter Anmeldung. Sie können Benutzerpools konfigurieren, um Missbrauch durch öffentliche Anfragen zu verhindern oder öffentliche Authentifizierungsvorgänge vollständig zu deaktivieren.

Die folgenden Einstellungen sind einige der Möglichkeiten, wie Sie öffentliche und interne Authentifizierungsanfragen in Ihren Benutzerpools und App-Clients verwalten können.

Beispiele für Benutzerpooleinstellungen, die sich auf den Zugriff auf öffentliche Benutzerpools auswirken

Einstellung	Verfügbare Optionen	Konfiguriert am	Auswirkung auf die öffentliche Authentifizierung	Konsoleneinstellung	API-Betrieb und Parameter
Self-Service-Anmeldung	Erlauben Sie Benutzern, sich für ein Konto zu registrieren oder Benutzerkonten als Administrator zu erstellen.	Benutzerpool	Öffentliche Registrierung verhindern	Anmeldung — Self-Service-Anmeldung	CreateUserPool, UpdateUserPool `AdminCreateUserConfig – AllowAdminCreateUserOnly`
Bestätigung durch den Administrator	Senden Sie Bestätigungscodes an neue Benutzer oder fordern Sie Administratoren auf, sie zu bestätigen.	Benutzerpool	Verhindern Sie die Bestätigung der Anmeldung ohne Eingreifen des Administrators	Anmeldung — Cognito-gestützte Überprüfung und Bestätigung	CreateUserPool, UpdateUserPool `AccountRecoverySettings –`

Einstellung	Verfügbare Optionen	Konfiguriert am	Auswirkung auf die öffentliche Authentifizierung	Konsoleneinstellung	API-Betrieb und Parameter
					`admin_only`
Offenlegung durch Benutzer	Senden Sie bei der Anmeldung und beim Zurücksetzen des Passworts die Meldung „Benutzer nicht gefunden“ oder verhindern Sie die Offenlegung.	App-Client	Schützen Sie sich davor, den Anmeldenamen, die E-Mail-Adresse oder die Telefonnummern zu erraten	App-Clients — Vermeiden Sie Fehler bei der Existenz von Benutzern	CreateUserPoolClient, UpdateUserPoolClient `PreventUserExistenceErrors`
Clientschlüssel	Bei der Registrierung, Anmeldung und beim Zurücksetzen des Passworts ist ein geheimer Hash erforderlich oder nicht erforderlich	App-Client	Schützen Sie sich vor Authentifizierungsanfragen aus nicht autorisierten Quellen	App-Clients — Geheimer Client-Schlüssel	CreateUserPoolClient `GenerateSecret`
Web ACLs	Aktivieren oder deaktivieren Sie eine Netzwerk-Firewall für Authentifizierungsanfragen	Benutzerpool	Beschränken oder verhindern Sie den Zugriff auf der Grundlage von vom Administrator definierten Anforderungsmerkmalen und IP-Adressregeln	AWS WAF— WAF-Einstellungen	AssociateWebACL `ResourceArn`

Einstellung	Verfügbare Optionen	Konfiguriert am	Auswirkung auf die öffentliche Authentifizierung	Konsoleneinstellung	API-Betrieb und Parameter
Externer IdP	Erlauben Sie die Anmeldung von Benutzern eines Drittanbieters IdPs, des Benutzerpool-Verzeichnisses oder beidem	App-Client	Schließen Sie lokale Benutzer oder Verbundbenutzer von der Registrierung und Anmeldung aus.	App-Clients — Identitätsanbieter	CreateUserPoolClient, UpdateUserPoolClient `SupportedIdentityProviders`
Autorisierungsserver	Hosten Sie öffentliche Webseiten zur Authentifizierung oder hosten Sie nicht	Benutzerpool	Schalten Sie öffentliche Webseiten aus und lassen Sie nur SDK-basierte Authentifizierung zu	Domain	CreateUserPoolDomain Durch die Erstellung einer beliebigen Benutzerpool-Domain werden öffentliche Webseiten verfügbar.

Einstellung	Verfügbare Optionen	Konfiguriert am	Auswirkung auf die öffentliche Authentifizierung	Konsoleneinstellung	API-Betrieb und Parameter
Schutz vor Bedrohungen	Aktivieren oder deaktivieren Sie die Überwachung auf Anzeichen bösartiger Aktivitäten oder unsicherer Passwörter	Benutzerpool oder App-Client	Kann die Anmeldung automatisch blockieren oder MFA verlangen, wenn Benutzer Anzeichen einer Gefährdung anzeigen	Bedrohungsschutz — Schutzeinstellungen	SetRiskConfiguration Die Parameter von `SetRiskConfiguration` definieren Ihre Einstellungen für den Schutz vor Bedrohungen.

Schützen Sie vertrauliche Kunden mit vertraulichen Kundengeheimnissen

Das Client-Geheimnis ist eine optionale Zeichenfolge, die einem App-Client zugeordnet ist. Alle Authentifizierungsanfragen an App-Clients mit Client-Geheimnissen müssen einen geheimen Hash enthalten, der aus dem Benutzernamen, der Client-ID und dem geheimen Client-Schlüssel generiert wird. Diejenigen, die das Client-Geheimnis nicht kennen, werden von Anfang an von Ihrer Anwendung ausgeschlossen.

Kundengeheimnisse haben jedoch Einschränkungen. Wenn Sie ein Client-Geheimnis in öffentliche Client-Software einbetten, kann Ihr Client-Geheimnis eingesehen werden. Dadurch haben Sie die Möglichkeit, Benutzer zu erstellen, Anfragen zum Zurücksetzen von Kennwörtern einzureichen und andere Vorgänge in Ihrem App-Client auszuführen. Client-Geheimnisse dürfen nur implementiert werden, wenn eine Anwendung die einzige Entität ist, die Zugriff auf das Geheimnis hat. In der

Regel ist dies in serverseitigen vertraulichen Clientanwendungen möglich. Dies gilt auch für M2M-Anwendungen, bei denen ein geheimer Client-Schlüssel erforderlich ist. Speichern Sie das Client-Geheimnis in einem verschlüsselten lokalen Speicher oder AWS Secrets Manager. Lassen Sie Ihr Kundengeheimnis niemals im öffentlichen Internet sichtbar sein.

Schützen Sie andere Geheimnisse

Ihr Authentifizierungssystem mit Amazon Cognito Cognito-Benutzerpools verarbeitet möglicherweise private Daten, Passwörter und AWS Anmeldeinformationen. Im Folgenden finden Sie einige bewährte Methoden für den Umgang mit Geheimnissen, auf die Ihre Anwendung möglicherweise zugreift.

Passwörter

Benutzer geben möglicherweise Kennwörter ein, wenn sie sich bei Ihrer Anwendung anmelden. Amazon Cognito verfügt über Aktualisierungstoken, die Ihre Anwendung verwenden kann, um abgelaufene Benutzersitzungen ohne Aufforderung zur Eingabe eines neuen Kennworts fortzusetzen. Speichern Sie keine Passwörter oder Passwort-Hashes im lokalen Speicher. Entwerfen Sie Ihre Anwendung so, dass Passwörter undurchsichtig behandelt werden und sie nur an Ihren Benutzerpool weitergegeben werden.

Implementieren Sie als bewährte Methode die kennwortlose Authentifizierung mit Hauptschlüsseln. WebAuthn Wenn Sie Passwörter implementieren müssen, verwenden Sie den Secure Remote Password (SRP) -Authentifizierungsablauf und die Multi-Faktor-Authentifizierung (MFA).

AWS Anmeldeinformationen

Für die Administratorauthentifizierung und für Verwaltungsvorgänge im Benutzerpool ist eine Authentifizierung mit AWS Anmeldeinformationen erforderlich. Um diese Operationen in einer Anwendung zu implementieren, gewähren Sie sicheren Zugriff auf temporäre AWS Anmeldeinformationen. Gewähren Sie den Zugriff auf Anmeldeinformationen nur für Anwendungen, die auf einer Serverkomponente ausgeführt werden, die Sie kontrollieren. Stellen Sie Anwendungen, die AWS Anmeldeinformationen enthalten, nicht auf öffentlichen Versionskontrollsystemen wie. GitHub Verschlüsseln Sie keine AWS Anmeldeinformationen in öffentlichen clientseitigen Anwendungen.

PKCE-Code-Verifizierer

Proof Key for Code Exchange, oder PKCE, ist für OpenID Connect (OIDC) -Autorisierungscode-Erteilungen mit Ihrem Benutzerpool-Autorisierungsserver vorgesehen. Anwendungen teilen die Geheimnisse der Codeverifizierung mit Ihrem Benutzerpool, wenn sie Autorisierungscodes

anfordern. Um Autorisierungscodes gegen Token auszutauschen, müssen Kunden bestätigen, dass sie den Code-Verifier kennen. Diese Praxis schützt vor der Ausgabe von Token mit abgefangenen Autorisierungscodes.

Kunden müssen bei jeder Autorisierungsanfrage einen neuen Zufallscode-Verifier generieren. Die Verwendung eines statischen oder vorhersehbaren Codeverifizierers bedeutet, dass ein Angreifer nur dann den hartcodierten Verifier und den Autorisierungscode abfangen muss. Entwerfen Sie Ihre Anwendung so, dass sie Benutzern keine Werte der Codeverifizierung zugänglich macht.

Verwaltung des Benutzerpools, geringste Rechte

IAM-Richtlinien können die Zugriffsebene definieren, die Principals auf die Verwaltung des Amazon Cognito Cognito-Benutzerpools und die administrativen Authentifizierungsvorgänge haben. Zum Beispiel:

- Erteilen Sie einem Webserver Berechtigungen für die Authentifizierung mit administrativen API-Vorgängen.
- Erteilen Sie einem AWS IAM Identity Center Benutzer, der einen Benutzerpool in Ihrem Unternehmen verwaltet AWS-Konto, Berechtigungen für die Verwaltung und Berichterstattung des Benutzerpools.

Der Grad der Ressourcengranularität in Amazon Cognito ist für IAM-Richtlinienzwecke auf zwei Ressourcentypen beschränkt: Benutzerpool und Identitätspool. Beachten Sie, dass Sie keine Berechtigungen zur Verwaltung einzelner App-Clients anwenden können. Konfigurieren Sie Benutzerpools mit dem Wissen, dass die von Ihnen erteilten Berechtigungen für alle App-Clients gelten. Wenn Ihr Unternehmen über mehrere Anwendungsmandanten verfügt und Ihr Sicherheitsmodell eine Trennung der administrativen Zuständigkeiten zwischen den Mandanten erfordert, implementieren Sie eine Mehrmandantenfähigkeit mit einem Mandanten pro Benutzerpool.

Sie können zwar IAM-Richtlinien mit Berechtigungen für Benutzerauthentifizierungsvorgänge wie erstellen`InitiateAuth`, diese Berechtigungen haben jedoch keine Auswirkung. Öffentliche und mit Token autorisierte API-Operationen unterliegen keinen IAM-Berechtigungen. Von den verfügbaren Benutzerpool-Authentifizierungsvorgängen können Sie nur Berechtigungen für administrative serverseitige Operationen wie gewähren. `AdminInitiateAuth`

Sie können die Verwaltungsebenen des Benutzerpools mit Listen mit den geringsten Rechten einschränken. `Action` Die folgende Beispielrichtlinie richtet sich an einen Administrator, der zwar

Ressourcenserver IdPs, App-Clients und die Benutzerpooldomäne verwalten kann, jedoch nicht Benutzer oder den Benutzerpool.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UserPoolClientAdministrator",
      "Action": [
        "cognito-idp:CreateIdentityProvider",
        "cognito-idp:CreateManagedLoginBranding",
        "cognito-idp:CreateResourceServer",
        "cognito-idp:CreateUserPoolDomain",
        "cognito-idp:DeleteIdentityProvider",
        "cognito-idp:DeleteResourceServer",
        "cognito-idp:DeleteUserPoolDomain",
        "cognito-idp:DescribeIdentityProvider",
        "cognito-idp:DescribeManagedLoginBranding",
        "cognito-idp:DescribeManagedLoginBrandingByClient",
        "cognito-idp:DescribeResourceServer",
        "cognito-idp:DescribeUserPool",
        "cognito-idp:DescribeUserPoolClient",
        "cognito-idp:DescribeUserPoolDomain",
        "cognito-idp:GetIdentityProviderByIdentifier",
        "cognito-idp:GetUICustomization",
        "cognito-idp:ListIdentityProviders",
        "cognito-idp:ListResourceServers",
        "cognito-idp:ListUserPoolClients",
        "cognito-idp:ListUserPools",
        "cognito-idp:SetUICustomization",
        "cognito-idp:UpdateIdentityProvider",
        "cognito-idp:UpdateManagedLoginBranding",
        "cognito-idp:UpdateResourceServer",
        "cognito-idp:UpdateUserPoolClient",
        "cognito-idp:UpdateUserPoolDomain"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_EXAMPLE"
    }
  ]
```

```
}
```

Die folgende Beispielrichtlinie gewährt Benutzer- und Gruppenverwaltung und Authentifizierung für eine serverseitige Anwendung.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UserAdminAuthN",
      "Action": [
        "cognito-idp:AdminAddUserToGroup",
        "cognito-idp:AdminConfirmSignUp",
        "cognito-idp:AdminCreateUser",
        "cognito-idp:AdminDeleteUser",
        "cognito-idp:AdminDeleteUserAttributes",
        "cognito-idp:AdminDisableProviderForUser",
        "cognito-idp:AdminDisableUser",
        "cognito-idp:AdminEnableUser",
        "cognito-idp:AdminForgetDevice",
        "cognito-idp:AdminGetDevice",
        "cognito-idp:AdminGetUser",
        "cognito-idp:AdminInitiateAuth",
        "cognito-idp:AdminLinkProviderForUser",
        "cognito-idp:AdminListDevices",
        "cognito-idp:AdminListGroupsForUser",
        "cognito-idp:AdminListUserAuthEvents",
        "cognito-idp:AdminRemoveUserFromGroup",
        "cognito-idp:AdminResetUserPassword",
        "cognito-idp:AdminRespondToAuthChallenge",
        "cognito-idp:AdminSetUserMFAPreference",
        "cognito-idp:AdminSetUserPassword",
        "cognito-idp:AdminSetUserSettings",
        "cognito-idp:AdminUpdateAuthEventFeedback",
        "cognito-idp:AdminUpdateDeviceStatus",
        "cognito-idp:AdminUpdateUserAttributes",
        "cognito-idp:AdminUserGlobalSignOut",
        "cognito-idp:AssociateSoftwareToken",
        "cognito-idp:ListGroups",
```

```
        "cognito-idp:ListUsers",
        "cognito-idp:ListUsersInGroup",
        "cognito-idp:RevokeToken",
        "cognito-idp:UpdateGroup",
        "cognito-idp:VerifySoftwareToken"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_EXAMPLE"
    }
  ]
}
```

Token sichern und verifizieren

Token können interne Verweise auf Gruppenzugehörigkeit und Benutzerattribute enthalten, die Sie möglicherweise nicht an den Endbenutzer weitergeben möchten. Speichern Sie ID- und Zugriffstoken nicht im lokalen Speicher. Aktualisierungstoken werden mit einem Schlüssel verschlüsselt, auf den nur Ihr Benutzerpool zugreifen kann, und sind für Benutzer und Anwendungen undurchsichtig. Widerrufen Sie Aktualisierungstoken, wenn sich Benutzer abmelden oder wenn Sie feststellen, dass das Fortbestehen einer Benutzersitzung aus Sicherheitsgründen unerwünscht ist.

Verwenden Sie Zugriffstoken, um den Zugriff nur auf Systeme zu autorisieren, die unabhängig überprüfen, ob das Token gültig und nicht abgelaufen ist. Ressourcen zur Überprüfung finden Sie unter Überprüfen eines JSON-Web-Tokens.

Ermitteln Sie die Identitätsanbieter, denen Sie vertrauen möchten

Wenn Sie Ihren Benutzerpool mit SAML - oder OIDC-Identitätsanbietern (IdPs) konfigurieren, IdPs können Sie neue Benutzer erstellen, Benutzerattribute festlegen und auf Ihre Anwendungsressourcen zugreifen. SAML- und OIDC-Anbieter werden in der Regel in business-to-business (B2B-) oder Unternehmensszenarien eingesetzt, in denen Sie oder Ihr direkter Kunde die Mitgliedschaft und Konfiguration des Anbieters kontrollieren.

Soziale Anbieter bieten Benutzerkonten für alle Benutzer im Internet an und unterliegen weniger Ihrer Kontrolle als Unternehmensanbieter. Aktivieren Sie Social nur dann IdPs in Ihrem App-Client, wenn Sie bereit sind, öffentlichen Kunden die Anmeldung und den Zugriff auf Ressourcen in Ihrer Anwendung zu ermöglichen.

Machen Sie sich mit den Auswirkungen von Geltungsbereichen auf den Zugriff auf Benutzerprofile vertraut

In Ihren Authentifizierungsanfragen an den Autorisierungsserver für den Benutzerpool können Sie Bereiche für die Zugriffskontrolle anfordern. Diese Bereiche können Ihren Benutzern Zugriff auf externe Ressourcen gewähren, und sie können Benutzern Zugriff auf ihre eigenen Benutzerprofile gewähren. Konfigurieren Sie Ihre App-Clients so, dass sie die Mindestbereiche unterstützen, die für den Betrieb Ihrer Anwendung erforderlich sind.

Der `aws.cognito.signin.user.admin` Geltungsbereich ist in allen Zugriffstoken enthalten, die durch die SDK-Authentifizierung mit Vorgängen wie InitiateAuthausgegeben werden. Er ist für Self-Service-Operationen mit Benutzerprofilen in Ihrer Anwendung vorgesehen. Sie können diesen Bereich auch von Ihrem Autorisierungsserver anfordern. Dieser Bereich ist für tokenautorisierte Operationen wie und erforderlich. UpdateUserAttributesGetUser Die Wirkung dieser Operationen wird durch die Lese- und Schreibberechtigungen Ihres App-Clients begrenzt.

Die `phone` Bereiche `openidprofile`,`email`, und autorisieren Anfragen an den Autorisierungsserver in UserInfo-Endpunkt Ihrem Benutzerpool. Sie definieren die Attribute, die der Endpunkt zurückgeben kann. Wenn der `openid` Bereich ohne andere Bereiche angefordert wird, gibt er alle verfügbaren Attribute zurück. Wenn Sie jedoch mehr Bereiche in der Anforderung anfordern, wird die Antwort auf die Attribute eingegrenzt, die durch die zusätzlichen Bereiche repräsentiert werden. Der `openid` Bereich gibt auch eine Anfrage für ein ID-Token an. Wenn Sie diesen Bereich in Ihrer Anfrage weglassenAutorisieren des Endpunkts, gibt Amazon Cognito nur ein Zugriffstoken und, falls zutreffend, ein Aktualisierungstoken aus. Weitere Informationen finden Sie unter OpenID Connect-Bereiche unter. App-Client-Bedingungen

Bereinigen Sie die Eingaben für Benutzerattribute

Für Benutzerattribute, die beispielsweise als Übermittlungsmethoden und Benutzernamen verwendet werden können, gelten `email` Formatbeschränkungen. Andere Attribute können den Datentyp „Zeichenfolge“, „Boolean“ oder „Zahl“ haben. Werte von Zeichenkettenattributen unterstützen eine Vielzahl von Eingaben. Konfigurieren Sie Ihre Anwendung so, dass sie vor Versuchen schützt, unerwünschte Daten in Ihr Benutzerverzeichnis oder die Nachrichten, die Amazon Cognito an Benutzer sendet, zu schreiben. Führen Sie eine clientseitige Validierung der vom Benutzer übermittelten Zeichenkettenattributwerte in Ihrer Anwendung durch, bevor Sie sie an Amazon Cognito senden.

Benutzerpools ordnen Ihrem Benutzerpool Attribute auf der Grundlage einer von IdPs Ihnen angegebenen Attributzuordnung zu. Ordnen Sie nur sichere und vorhersehbare IdP-Attribute den Zeichenkettenattributen des Benutzerpools zu.

Authentifizierung mit Amazon Cognito Cognito-Benutzerpools

Amazon Cognito umfasst mehrere Methoden zur Authentifizierung Ihrer Benutzer. Benutzer können sich mit WebAuthn Passwörtern und Hauptschlüsseln anmelden. Amazon Cognito kann ihnen ein Einmalpasswort in einer E-Mail- oder SMS-Nachricht senden. Sie können Lambda-Funktionen implementieren, die Ihre eigene Abfolge von Herausforderungen und Antworten orchestrieren. Dies sind Authentifizierungsabläufe. Bei Authentifizierungsabläufen geben Benutzer ein Geheimnis an und Amazon Cognito überprüft das Geheimnis und gibt dann JSON-Webtoken (JWTs) für Anwendungen zur Verarbeitung mit OIDC-Bibliotheken aus. In diesem Kapitel werden wir darüber sprechen, wie Sie Ihre Benutzerpools und App-Clients für verschiedene Authentifizierungsabläufe in verschiedenen Anwendungsumgebungen konfigurieren. Sie erfahren mehr über die Optionen für die Verwendung der gehosteten Anmeldeseiten von Managed Login und für die Erstellung Ihrer eigenen Logik und Ihres eigenen Frontends in einem AWS SDK.

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.

Um die Identität von Benutzern zu überprüfen, unterstützt Amazon Cognito Authentifizierungsabläufe, die zusätzlich zu Kennwörtern wie Einmalpasswörtern und Hauptschlüsseln für E-Mail- und SMS-Nachrichten auch Challenge-Typen beinhalten.

Themen

- Implementieren Sie Authentifizierungsabläufe
- Wissenswertes zur Authentifizierung mit Benutzerpools
- Ein Beispiel für eine Authentifizierungssitzung
- Konfigurieren Sie Authentifizierungsmethoden für die verwaltete Anmeldung
- Verwalten Sie Authentifizierungsmethoden in AWS SDKs
- Authentifizierungsabläufe
- Autorisierungsmodelle für die API- und SDK-Authentifizierung

Implementieren Sie Authentifizierungsabläufe

Unabhängig davon, ob Sie eine verwaltete Anmeldung oder ein benutzerdefiniertes Anwendungs-Frontend mit einem AWS SDK für die Authentifizierung implementieren, müssen Sie Ihren App-Client für die Authentifizierungstypen konfigurieren, die Sie implementieren möchten. Die folgenden Informationen beschreiben die Einrichtung von Authentifizierungsabläufen in Ihren App-Clients und Ihrer Anwendung.

App client supported flows

Sie können unterstützte Flows für Ihre App-Clients in der Amazon Cognito Cognito-Konsole oder mit der API in einem AWS SDK konfigurieren. Nachdem Sie Ihren App-Client so konfiguriert haben, dass er diese Flows unterstützt, können Sie sie in Ihrer Anwendung bereitstellen.

Das folgende Verfahren konfiguriert die verfügbaren Authentifizierungsabläufe für einen App-Client mit der Amazon Cognito Cognito-Konsole.

So konfigurieren Sie einen App-Client für Authentifizierungsabläufe (Konsole)

1. Melden Sie sich bei der Amazon Cognito-Benutzerpools-Konsole an AWS und navigieren Sie zur Amazon Cognito Cognito-Benutzerpools-Konsole. Wählen Sie einen Benutzerpool oder erstellen Sie einen neuen.
2. Wählen Sie in Ihrer Benutzerpool-Konfiguration das Menü App-Clients aus. Wählen Sie einen App-Client oder erstellen Sie einen neuen.
3. Wählen Sie unter App-Client-Informationen die Option Bearbeiten aus.
4. Wählen Sie unter App-Client-Flows die Authentifizierungsabläufe aus, die Sie unterstützen möchten.

So konfigurieren Sie einen App-Client für Authentifizierungsabläufe (API/SDK)

Um verfügbare Authentifizierungsabläufe für einen App-Client mit der Amazon Cognito Cognito-API zu konfigurieren, legen Sie den Wert von `ExplicitAuthFlows` in einer CreateUserPoolClientUpdateUserPoolClientOR-Anfrage fest. Im Folgenden finden Sie ein Beispiel, das einem Client ein sicheres Remote-Passwort (SRP) und eine wahlbasierte Authentifizierung bereitstellt.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_AUTH",
   "ALLOW_USER_SRP_AUTH
```

```
]
```

Wenn Sie vom App-Client unterstützte Flows konfigurieren, können Sie die folgenden Optionen und API-Werte angeben.

Unterstützung für App-Client-Flows

Authentifizierungsfluss	Kompatibilität	Konsole	API
Wahlbasierte Authentifizierung	Serverseitig, clientseitig	Wählen Sie bei der Anmeldung einen Authentifizierungstyp	`ALLOW_USER_AUTH`
Melden Sie sich mit dauerhaften Passwörtern an	Clientseitig	Melden Sie sich mit Benutzername und Passwort an	`ALLOW_USER_PASSWORD_AUTH`
Melden Sie sich mit dauerhaften Passwörtern und sicherer Payload an	Serverseitig, clientseitig	Melden Sie sich mit einem sicheren Remote-Passwort (SRP) an	`ALLOW_USER_SRP_AUTH`
Tokens aktualisieren	Serverseitig, clientseitig	Rufen Sie neue Benutzertoken aus vorhandenen authentifizierten Sitzungen ab	`ALLOW_REFRESH_TOKEN_AUTH`
Serverseitige Authentifizierung	Serverseitig	Melden Sie sich mit serverseitigen Administratoranmeldedaten an	`ALLOW_ADMIN_USER_PASSWORD_AUTH`
Benutzerdefinierte Authentifizierung	Serverseitige und clientseitige maßgeschneiderte Anwendungen. Nicht kompatibe	Melden Sie sich mit benutzerdefinierten Authentifizierungsabläufen	`ALLOW_CUSTOM_AUTH`

Authentifizierungs fluss	Kompatibilität	Konsole	API
	I mit verwalteter Anmeldung.	von Lambda-Triggern an	

Implement flows in your application

Die verwaltete Anmeldung macht Ihre konfigurierten Authentifizierungsoptionen automatisch auf Ihren Anmeldeseiten verfügbar. Starten Sie in maßgeschneiderten Anwendungen die Authentifizierung mit einer Deklaration des ursprünglichen Ablaufs.

- Um aus einer Liste von Flow-Optionen für einen Benutzer auszuwählen, deklarieren Sie die wahlbasierte Authentifizierung mit dem Flow. `USER_AUTH` Für diesen Flow stehen Authentifizierungsmethoden zur Verfügung, die in clientbasierten Authentifizierungsabläufen nicht verfügbar sind, z. B. Authentifizierung mit Passkey und passwortloser Authentifizierung.
- Um Ihren Authentifizierungsablauf im Voraus auszuwählen, deklarieren Sie die clientbasierte Authentifizierung mit einem beliebigen anderen Ablauf, der in Ihrem App-Client verfügbar ist.

Wenn Sie Benutzer anmelden, muss der Text Ihrer InitiateAuthAdminInitiateAuthPR-Anforderung einen `AuthFlow` Parameter enthalten.

Wahlbasierte Authentifizierung:

```
"AuthFlow": "USER_AUTH"
```

Client-basierte Authentifizierung mit SRP:

```
"AuthFlow": "USER_SRP_AUTH"
```

Wissenswertes zur Authentifizierung mit Benutzerpools

Berücksichtigen Sie bei der Gestaltung Ihres Authentifizierungsmodells mit Amazon Cognito Cognito-Benutzerpools die folgenden Informationen.

Die Authentifizierung erfolgt über die verwaltete Anmeldung und die gehostete Benutzeroberfläche

Die verwaltete Anmeldung bietet mehr Authentifizierungsoptionen als die klassische gehostete Benutzeroberfläche. Beispielsweise können Benutzer die kennwortlose Authentifizierung und die Authentifizierung mit dem Hauptschlüssel nur bei verwalteter Anmeldung durchführen.

Benutzerdefinierte Authentifizierungsabläufe sind nur bei der SDK-Authentifizierung verfügbar AWS

Sie können keine benutzerdefinierten Authentifizierungsabläufe oder eine benutzerdefinierte Authentifizierung mit Lambda-Triggern, mit verwalteter Anmeldung oder der klassischen gehosteten Benutzeroberfläche durchführen. Die benutzerdefinierte Authentifizierung ist in der Authentifizierung mit AWS SDKs verfügbar.

Verwaltete Anmeldung für die Anmeldung mit einem externen Identitätsanbieter (IdP)

Sie können Benutzer nicht über die IdPsDrittanbieter-Authentifizierung mit anmelden. AWS SDKs Sie müssen die verwaltete Anmeldung oder die klassische gehostete Benutzeroberfläche implementieren IdPs, zu der Sie weiterleiten und das resultierende Authentifizierungsobjekt dann mit OIDC-Bibliotheken in Ihrer Anwendung verarbeiten. Weitere Informationen zur verwalteten Anmeldung finden Sie unter. Vom Benutzerpool verwaltete Anmeldung

Auswirkung der kennwortlosen Authentifizierung auf andere Benutzerfunktionen

Die Aktivierung der kennwortlosen Anmeldung mit Einmalkennwörtern oder Hauptschlüsseln in Ihrem Benutzerpool und App-Client wirkt sich auf die Benutzererstellung und Migration aus. Wenn die passwortlose Anmeldung aktiv ist:

1. Administratoren können Benutzer ohne Passwörter erstellen. Die Standardvorlage für Einladungsnachrichten wurde dahingehend geändert, dass sie den `{###}` Passwort-Platzhalter nicht mehr enthält. Weitere Informationen finden Sie unter Erstellen von Benutzerkonten als Administrator.
2. Bei SDK-basierten SignUpVorgängen müssen Benutzer bei der Registrierung kein Passwort angeben. Für die verwaltete Anmeldung und die gehostete Benutzeroberfläche ist auf der Anmeldeseite ein Passwort erforderlich, auch wenn die kennwortlose Authentifizierung zulässig ist. Weitere Informationen finden Sie unter Registrieren und Bestätigen von Benutzerkonten.
3. Aus einer CSV-Datei importierte Benutzer können sich sofort mit passwortlosen Optionen anmelden, ohne dass das Passwort zurückgesetzt werden muss, wenn ihre Attribute eine E-Mail-Adresse oder Telefonnummer für eine verfügbare passwortlose Anmeldeoption enthalten. Weitere Informationen finden Sie unter Importieren von Benutzern aus einer CSV-Datei in Benutzerpools.

4. Bei der kennwortlosen Authentifizierung wird der Lambda-Trigger für die Benutzermigration nicht aufgerufen.
5. Benutzer, die sich mit einem kennwortlosen First Factor anmelden, können ihrer Sitzung keinen MFA-Faktor (Multi-Factor Authentication) hinzufügen. Nur kennwortbasierte Authentifizierungsabläufe unterstützen MFA.

Die Partei, auf die der Passkey vertraut, URLs darf nicht auf der Liste der öffentlichen Suffixe stehen

Sie können Domainnamen, die Sie besitzen`www.example.com`, z. B. als RP-ID (Relying Party) in Ihrer Hauptschlüsselkonfiguration verwenden. Diese Konfiguration soll maßgeschneiderte Anwendungen unterstützen, die auf Domänen ausgeführt werden, deren Eigentümer Sie sind. Die öffentliche Suffixliste (PSL) enthält geschützte Domänen auf hoher Ebene. Amazon Cognito gibt einen Fehler zurück, wenn Sie versuchen, Ihre RP-URL auf eine Domain auf der PSL festzulegen.

Themen

- Dauer der Authentifizierungssitzung
- Sperrverhalten bei fehlgeschlagenen Anmeldeversuchen

Dauer der Authentifizierungssitzung

Abhängig von den Funktionen Ihres Benutzerpools können Sie am Ende auf mehrere Herausforderungen reagieren`InitiateAuth`, `RespondToAuthChallenge` 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. Bei der verwalteten Anmeldung wird die Sitzungsdauer für die Multi-Faktor-Authentifizierung auf 3 Minuten und für Codes zum Zurücksetzen von Passwörtern auf 8 Minuten festgelegt.

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 für Sitzungsdauer des Authentifizierungsflusses auf die gewünschte Gültigkeitsdauer in Minuten für SMS- und E-Mail-MFA-Codes. 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.

User pools 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 Anwendungsspezifische Einstellungen mit App-Clients.

Sperrverhalten bei fehlgeschlagenen Anmeldeversuchen

Nach fünf fehlgeschlagenen Anmeldeversuchen mit einem Benutzerkennwort, unabhängig davon, ob diese mit nicht authentifizierten oder IAM-autorisierten API-Vorgängen angefordert wurden, sperrt Amazon Cognito Ihren Benutzer für eine Sekunde. 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.

Ein Beispiel für eine Authentifizierungssitzung

Das folgende Diagramm und die folgende step-by-step Anleitung veranschaulichen ein typisches Szenario, in dem sich ein Benutzer bei einer Anwendung anmeldet. In der Beispielanwendung stehen einem Benutzer mehrere Anmeldeoptionen zur Verfügung. Sie wählen eine aus, indem sie ihre Anmeldeinformationen eingeben, einen zusätzlichen Authentifizierungsfaktor angeben und sich anmelden.

Stellen Sie sich eine Anwendung mit einer Anmeldeseite vor, auf der sich Benutzer mit einem Benutzernamen und einem Passwort anmelden, in einer E-Mail-Nachricht einen Einmalcode anfordern oder eine Fingerabdruckoption wählen können.

1. Anmeldeaufforderung: Ihre Anwendung zeigt einen Startbildschirm mit einer Anmeldeschaltfläche.
2. Anmeldung anfordern: Der Benutzer wählt Anmelden aus. Aus einem Cookie oder einem Cache ruft Ihre Anwendung ihren Benutzernamen ab oder fordert sie auf, ihn einzugeben.
3. Anforderungsoptionen: Ihre Anwendung fordert die Anmeldeoptionen des Benutzers mit einer `InitiateAuth` API-Anfrage im `USER_AUTH` Flow an, in der die verfügbaren Anmeldemethoden für den Benutzer angefordert werden.
4. Anmeldeoptionen senden: Amazon Cognito antwortet mit `PASSWORD``EMAIL_OTP`, und. `WEB_AUTHN` Die Antwort enthält eine Sitzungs-ID, die Sie in der nächsten Antwort erneut abspielen können.
5. Anzeigeoptionen: Ihre Anwendung zeigt Benutzeroberflächenelemente an, mit denen der Benutzer seinen Benutzernamen und sein Passwort eingeben, einen Einmalcode abrufen oder seinen Fingerabdruck scannen kann.
6. Wählen Sie option/Enter Anmeldeinformationen: Der Benutzer gibt seinen Benutzernamen und sein Passwort ein.

7. Authentifizierung einleiten: Ihre Anwendung stellt die Anmeldeinformationen des Benutzers mit einer `RespondToAuthChallenge` API-Anfrage zur Verfügung, die die Anmeldung mit Benutzername/Passwort bestätigt und den Benutzernamen und das Passwort bereitstellt.
8. Anmeldeinformationen validieren: Amazon Cognito bestätigt die Anmeldeinformationen des Benutzers.
9. Zusätzliche Herausforderung: Der Benutzer verfügt über eine Multi-Faktor-Authentifizierung, die mit einer Authenticator-App konfiguriert ist. Amazon Cognito gibt eine `SOFTWARE_TOKEN_MFA` Herausforderung zurück.
10. Aufforderung zur Aufforderung: In Ihrer Anwendung wird ein Formular angezeigt, in dem ein zeitbasiertes Einmalkennwort (TOTP) von der Authenticator-App des Benutzers angefordert wird.
11. Herausforderung beantworten: Der Benutzer reicht das TOTP ein.
12. Auf die Herausforderung antworten: In einer anderen `RespondToAuthChallenge` Anfrage stellt Ihre Anwendung das TOTP des Benutzers bereit.
13. Challenge-Antwort validieren: Amazon Cognito bestätigt den Code des Benutzers und stellt fest, dass Ihr Benutzerpool so konfiguriert ist, dass er keine zusätzlichen Herausforderungen an den aktuellen Benutzer ausgibt.
14. Token ausgeben: Amazon Cognito gibt ID-, Zugriffs- und Aktualisierungs-JSON-Webtoken (JWTs) zurück. Die Erstauthentifizierung des Benutzers ist abgeschlossen.
15. Token speichern: Ihre Anwendung speichert die Token des Benutzers im Cache, sodass sie auf Benutzerdaten verweisen, den Zugriff auf Ressourcen autorisieren und Token aktualisieren kann, wenn sie ablaufen.
16. Autorisierte Inhalte rendern: Ihre Anwendung bestimmt anhand ihrer Identität und Rollen den Zugriff des Benutzers auf Ressourcen und stellt Anwendungsinhalte bereit.
17. Zugriff auf Inhalte: Der Benutzer ist angemeldet und beginnt, die Anwendung zu verwenden.
18. Inhalt mit abgelaufenem Token anfordern: Später fordert der Benutzer eine Ressource an, für die eine Autorisierung erforderlich ist. Das zwischengespeicherte Token des Benutzers ist abgelaufen.
19. Aktualisierungstoken: Ihre Anwendung stellt eine `InitiateAuth` Anfrage mit dem gespeicherten Aktualisierungstoken des Benutzers.
20. Token ausgeben: Amazon Cognito gibt eine neue ID und einen neuen Zugriff JWTs zurück. Die Sitzung des Benutzers wird ohne zusätzliche Aufforderung zur Eingabe von Anmeldeinformationen sicher aktualisiert.

Sie können AWS Lambda Trigger verwenden, um die Art und Weise, wie sich Benutzer authentifizieren, 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. Sie können den Authentifizierungsablauf für die Benutzermigration verwenden, um die Benutzermigration zu ermöglichen, ohne dass Ihre Benutzer ihre Passwörter zurücksetzen müssen.

Konfigurieren Sie Authentifizierungsmethoden für die verwaltete Anmeldung

Sie können verwaltete Anmeldeseiten, ein Web-Frontend für die Benutzerpoolauthentifizierung, aufrufen, wenn Sie möchten, dass sich Benutzer anmelden, abmelden oder ihr Passwort zurücksetzen. In diesem Modell importiert Ihre Anwendung OIDC-Bibliotheken, um browserbasierte Authentifizierungsversuche mit vom Benutzerpool verwalteten Anmeldeseiten zu verarbeiten. Die Authentifizierungsformen, die Ihren Benutzern zur Verfügung stehen, hängen von der Konfiguration Ihres Benutzerpools und Ihres App-Clients ab. Implementieren Sie den `ALLOW_USER_AUTH` Ablauf in Ihrem App-Client, und Amazon Cognito fordert Benutzer auf, eine Anmeldemethode aus den verfügbaren Optionen auszuwählen. Implementieren `ALLOW_USER_PASSWORD_AUTH` und weisen Sie einen SAML-Anbieter zu, und Ihre Anmeldeseiten bieten Benutzern die Möglichkeit, ihren Benutzernamen und ihr Passwort einzugeben oder sich mit ihrem IdP zu verbinden.

Mit der Amazon Cognito Cognito-Benutzerpools-Konsole können Sie mit der Einrichtung der verwalteten Anmeldeauthentifizierung für Ihre Anwendung beginnen. Wenn Sie einen neuen Benutzerpool erstellen, geben Sie die Plattform an, für die Sie entwickeln. In der Konsole finden Sie Beispiele für die Implementierung von OIDC und OAuth Bibliotheken mit Startcode zur Implementierung von Anmelde- und Abmeldeabläufen. Sie können eine verwaltete Anmeldung mit vielen OIDC-Implementierungen von Relying-Partys erstellen. Wir empfehlen Ihnen, wenn möglich mit zertifizierten OIDC-Rely-Party-Bibliotheken zu arbeiten. Weitere Informationen finden Sie unter Erste Schritte mit Benutzerpools.

In der Regel überprüfen OIDC-Relying Party Libraries in regelmäßigen Abständen den Endpunkt Ihres Benutzerpools, um den Aussteller URLs zu ermitteln, z. B. den `.well-known/openid-configuration` Token-Endpunkt und den Autorisierungs-Endpunkt. Es hat sich bewährt, dieses automatische Erkennungsverhalten dort zu implementieren, wo Sie die Wahl haben. Die manuelle Konfiguration von Emittenten-Endpunkten kann zu Fehlern führen. Sie könnten beispielsweise Ihre Benutzerpool-Domäne ändern. Der Pfad zu `openid-configuration` ist nicht mit Ihrer Benutzerpool-Domain verknüpft, sodass Anwendungen, die Dienstendpunkte automatisch erkennen, Ihre Domainänderung automatisch übernehmen.

Benutzerpool-Einstellungen für die verwaltete Anmeldung

Möglicherweise möchten Sie die Anmeldung bei mehreren Anbietern für Ihre Anwendung zulassen, oder Sie möchten Amazon Cognito als unabhängiges Benutzerverzeichnis verwenden. Möglicherweise möchten Sie auch Benutzerattribute sammeln, MFA einrichten und danach fragen oder E-Mail-Adressen als Benutzernamen benötigen. Sie können die Felder in der verwalteten Anmeldung und in der gehosteten Benutzeroberfläche nicht direkt bearbeiten. Stattdessen legt die Konfiguration Ihres Benutzerpools automatisch fest, wie die verwalteten Login-Authentifizierungsabläufe gehandhabt werden.

Die folgenden Konfigurationselemente für den Benutzerpool bestimmen die Authentifizierungsmethoden, die Amazon Cognito Benutzern in der verwalteten Anmeldung und in der gehosteten Benutzeroberfläche anbietet.

User pool options (Sign-in menu)

Die folgenden Optionen befinden sich im Anmeldemenü eines Benutzerpools in der Amazon Cognito Cognito-Konsole.

Anmeldeoptionen für den Cognito-Benutzerpool

Hat Optionen für Benutzernamen. Ihre verwalteten Anmelde- und gehosteten UI-Seiten akzeptieren nur Benutzernamen in den von Ihnen ausgewählten Formaten. Wenn Sie beispielsweise einen Benutzerpool mit E-Mail als einziger Anmeldeoption einrichten, akzeptieren Ihre verwalteten Anmeldeseiten nur Benutzernamen in einem E-Mail-Format.

Erforderliche Attribute

Wenn Sie in Ihrem Benutzerpool ein Attribut nach Bedarf festlegen, fordert die verwaltete Anmeldung die Benutzer bei der Registrierung zur Eingabe eines Werts für dieses Attribut auf.

Optionen für die wahlbasierte Anmeldung

Enthält Einstellungen für Authentifizierungsmethoden in. Wahlbasierte Authentifizierung Hier können Sie Authentifizierungsmethoden wie Passkey und passwortlos ein- oder ausschalten. Diese Methoden sind nur für Benutzerpools mit verwalteten Anmeldedomänen und Funktionsplänen ab der Lite-Stufe verfügbar.

Multifaktor-Authentifizierung

Die verwaltete Anmeldung und die gehostete Benutzeroberfläche übernehmen die Registrierungs- und Authentifizierungsvorgänge für MFA. Wenn MFA in Ihrem Benutzerpool erforderlich

ist, werden Benutzer auf Ihren Anmeldeseiten automatisch aufgefordert, ihren zusätzlichen Faktor einzurichten. Außerdem fordern sie Benutzer mit einer MFA-Konfiguration auf, die Authentifizierung mit einem MFA-Code abzuschließen. Wenn MFA in Ihrem Benutzerpool deaktiviert oder optional ist, werden Sie auf Ihren Anmeldeseiten nicht aufgefordert, MFA einzurichten.

Wiederherstellung von Benutzerkonten

Die Self-Service-Kontowiederherstellungseinstellung Ihres Benutzerpools bestimmt, ob auf Ihren Anmeldeseiten ein Link angezeigt wird, über den Benutzer ihr Passwort zurücksetzen können.

User pool options (Domain menu)

Die folgenden Optionen befinden sich im Domain-Menü eines Benutzerpools in der Amazon Cognito Cognito-Konsole.

Domain

Ihre Wahl einer Benutzerpool-Domain legt den Pfad für den Link fest, den Benutzer öffnen, wenn Sie ihre Browser zur Authentifizierung aufrufen.

Version mit Markenkennzeichnung

Ihre Wahl einer Branding-Version bestimmt, ob in Ihrer Benutzerpool-Domain die verwaltete Anmeldung oder die gehostete Benutzeroberfläche angezeigt wird.

User pool options (Social and external providers menu)

Die folgende Option befindet sich im Menü Soziale Netzwerke und externe Anbieter eines Benutzerpools in der Amazon Cognito Cognito-Konsole.

Anbieter

Die Identitätsanbieter (IdPs), die Sie Ihrem Benutzerpool hinzufügen, können für jeden App-Client im Benutzerpool aktiv oder inaktiv bleiben.

App client options

Die folgenden Optionen befinden sich im App-Clients-Menü eines Benutzerpools in der Amazon Cognito Cognito-Konsole. Um diese Optionen zu überprüfen, wählen Sie einen App-Client aus der Liste aus.

Kurzanleitung zur Einrichtung

Die Kurzanleitung zur Einrichtung enthält Codebeispiele für eine Vielzahl von Entwicklerumgebungen. Sie enthalten die Bibliotheken, die für die Integration der verwalteten Anmeldeauthentifizierung in Ihre Anwendung erforderlich sind.

Informationen zum App-Client

Bearbeiten Sie diese Konfiguration, um sie der Anwendung IdPs zuzuweisen, die durch den aktuellen App-Client repräsentiert wird. Auf den verwalteten Anmeldeseiten zeigt Amazon Cognito Auswahlmöglichkeiten für Benutzer an. Diese Auswahlmöglichkeiten werden anhand der zugewiesenen Methoden und des IdP bestimmt. Wenn Sie beispielsweise einen SAML 2.0-IdP mit einem Namen `MySAML` und einem lokalen Benutzerpool-Login zuweisen, werden auf Ihren verwalteten Anmeldeseiten Aufforderungen zur Authentifizierungsmethode und eine Schaltfläche für angezeigt. `MySAML`

Einstellungen für die Authentifizierung

Bearbeiten Sie diese Konfiguration, um Authentifizierungsmethoden für Ihre Anwendung festzulegen. Auf den verwalteten Anmeldeseiten zeigt Amazon Cognito Auswahlmöglichkeiten für Benutzer an. Diese Auswahlmöglichkeiten hängen von der Verfügbarkeit des Benutzerpools als IdP und von den Methoden ab, die Sie zuweisen. Wenn Sie beispielsweise eine wahlbasierte `ALLOW_USER_AUTH` Authentifizierung zuweisen, werden auf Ihren verwalteten Anmeldeseiten verfügbare Optionen wie die Eingabe einer E-Mail-Adresse und die Anmeldung mit einem Hauptschlüssel angezeigt. Auf verwalteten Anmeldeseiten werden auch Schaltflächen für die zugewiesenen Benutzer angezeigt. IdPs

Anmeldeseiten

Legen Sie mit den auf dieser Registerkarte verfügbaren Optionen den visuellen Effekt Ihrer verwalteten Anmeldeseiten oder der gehosteten benutzerinteraktiven Benutzeroberflächen fest. Weitere Informationen finden Sie unter Branding auf verwaltete Anmeldeseiten anwenden.

Verwalten Sie Authentifizierungsmethoden in AWS SDKs

Benutzer in Amazon Cognito Cognito-Benutzerpools können sich mit einer Vielzahl von Optionen oder Faktoren für die erste Anmeldung anmelden. Bei einigen Faktoren können Benutzer anschließend eine Multi-Faktor-Authentifizierung (MFA) durchführen. Zu diesen ersten Faktoren gehören Benutzername und Passwort, Einmalpasswort, Hauptschlüssel und benutzerdefinierte Authentifizierung. Weitere Informationen finden Sie unter Authentifizierungsabläufe. Wenn Ihre

Anwendung über integrierte Benutzeroberflächenkomponenten verfügt und ein AWS SDK-Modul importiert, müssen Sie eine Anwendungslogik für die Authentifizierung erstellen. Sie müssen eine von zwei Hauptmethoden und aus dieser Methode die Authentifizierungsmechanismen auswählen, die Sie implementieren möchten.

Sie können eine clientbasierte Authentifizierung implementieren, bei der Ihre Anwendung oder Ihr Client den Authentifizierungstyp im Voraus deklariert. Ihre andere Option ist die wahlbasierte Authentifizierung, bei der Ihre App einen Benutzernamen erfasst und die verfügbaren Authentifizierungstypen für Benutzer anfordert. Sie können diese Modelle je nach Ihren Anforderungen zusammen in derselben Anwendung oder aufgeteilt auf mehrere App-Clients implementieren. Jede Methode verfügt über einzigartige Funktionen, z. B. benutzerdefinierte Authentifizierung bei clientbasierter Authentifizierung und kennwortlose Authentifizierung bei wahlbasierter Authentifizierung.

In maßgeschneiderten Anwendungen, die die Authentifizierung mithilfe der AWS SDK-Implementierung der Benutzerpool-API durchführen, müssen Sie Ihre API-Anfragen so strukturieren, dass sie der Benutzerpoolkonfiguration, der App-Client-Konfiguration und den clientseitigen Einstellungen entsprechen. Eine `InitiateAuth` Sitzung, die mit einem of beginnt, beginnt mit `AuthFlow` der wahlbasierten Authentifizierung`USER_AUTH`. Amazon Cognito antwortet auf Ihre API mit der Aufforderung, entweder eine bevorzugte Authentifizierungsmethode oder eine Liste von Auswahlmöglichkeiten auszuwählen. Eine Sitzung, die mit oder beginnt, `CUSTOM_AUTH` geht direkt `AuthFlow` zur benutzerdefinierten Authentifizierung mit Lambda-Triggern über.

Einige Authentifizierungsmethoden sind auf einen der beiden Flow-Typen festgelegt, und einige Methoden sind in beiden verfügbar.

Themen

- Wahlbasierte Authentifizierung
- Client-basierte Authentifizierung

Wahlbasierte Authentifizierung

Ihre Anwendung kann bei der wahlbasierten Authentifizierung die folgenden Authentifizierungsmethoden anfordern. Deklarieren Sie diese Optionen im `PREFERRED_CHALLENGE` Parameter von InitiateAuthoder AdminInitiateAuthoder im `ChallengeName` Parameter von RespondToAuthChallengeoder. AdminRespondToAuthChallenge

1. `EMAIL_OTP` und `SMS_OTP`

Passwortlose Anmeldung mit Einmalpasswörtern

2. `WEB_AUTHN`

   Passwortlose Anmeldung mit Passkeys WebAuthn

3. `PASSWORD`

   Melden Sie sich mit dauerhaften Passwörtern an

   Melden Sie sich mit dauerhaften Passwörtern und sicherer Payload an

   MFA nach der Anmeldung

Informationen zu diesen Optionen in ihrem API-Kontext finden Sie `ChallengeName` unter RespondToAuthChallenge.

Bei der wahlbasierten Anmeldung wird als Antwort auf Ihre erste Anfrage eine Aufforderung ausgegeben. Bei dieser Aufforderung wird entweder überprüft, ob eine angeforderte Option verfügbar ist, oder es wird eine Liste der verfügbaren Optionen bereitgestellt. Ihre Anwendung kann diese Optionen Benutzern anzeigen, die dann Anmeldeinformationen für ihre bevorzugte Anmeldemethode eingeben und mit der Authentifizierung bei Challenge-Antworten fortfahren.

In Ihrem Authentifizierungsablauf stehen Ihnen die folgenden auswahlbasierten Optionen zur Verfügung. Alle Anfragen dieses Typs erfordern, dass Ihre App zuerst einen Benutzernamen erfasst oder ihn aus einem Cache abruft.

1. `USERNAME`Nur Optionen mit `AuthParameters` oder anfordern. Amazon Cognito gibt eine `SELECT_CHALLENGE` Herausforderung zurück. Von dort aus kann Ihre Anwendung den Benutzer auffordern, eine Herausforderung auszuwählen und diese Antwort an Ihren Benutzerpool zurückzugeben.
2. Fordere eine bevorzugte Herausforderung mit `AuthParameters` von `PREFERRED_CHALLENGE` und den Parametern deiner bevorzugten Herausforderung an, falls vorhanden. Wenn Sie beispielsweise ein `PREFERRED_CHALLENGE` of anfordern`PASSWORD_SRP`, müssen Sie auch angeben`SRP_A`. Wenn Ihr Benutzer, Ihr Benutzerpool und Ihr App-Client alle für die bevorzugte Herausforderung konfiguriert sind, reagiert Amazon Cognito mit dem nächsten Schritt in dieser Herausforderung, z. B. `PASSWORD_VERIFIER` im `PASSWORD_SRP` Flow oder CodeDeliveryDetailsin den `EMAIL_OTP` `SMS_OTP` and-Flows. Wenn die bevorzugte

Herausforderung nicht verfügbar ist, antwortet Amazon Cognito mit `SELECT_CHALLENGE` einer Liste verfügbarer Herausforderungen.

3. Melden Sie die Benutzer zuerst an und fordern Sie dann ihre bevorzugten Authentifizierungsoptionen an. Eine GetUserAuthFactorsAnfrage mit dem Zugriffstoken eines angemeldeten Benutzers gibt seine verfügbaren auswahlbasierten Authentifizierungsfaktoren und seine MFA-Einstellungen zurück. Mit dieser Option kann sich ein Benutzer zuerst mit Benutzername und Passwort anmelden und dann eine andere Form der Authentifizierung aktivieren. Sie können diesen Vorgang auch verwenden, um zusätzliche Optionen für einen Benutzer zu überprüfen, der sich mit einer bevorzugten Herausforderung angemeldet hat.

Um Ihren App-Client für die wahlbasierte Authentifizierung zu konfigurieren, fügen Sie weitere Informationen `ALLOW_USER_AUTH` zu den zulässigen Authentifizierungsabläufen hinzu. Sie müssen auch die auswahlbasierten Faktoren auswählen, die Sie in Ihrer Benutzerpool-Konfiguration zulassen möchten. Der folgende Prozess veranschaulicht, wie Sie wahlbasierte Authentifizierungsfaktoren auswählen.

Amazon Cognito console

So konfigurieren Sie wahlbasierte Authentifizierungsoptionen in einem Benutzerpool

1. Melden Sie sich bei der Amazon Cognito-Benutzerpools-Konsole an AWS und navigieren Sie zur Amazon Cognito Cognito-Benutzerpools-Konsole. Wählen Sie einen Benutzerpool oder erstellen Sie einen neuen.
2. Wählen Sie in Ihrer Benutzerpool-Konfiguration das Anmelde-Menü aus. Suchen Sie nach Optionen für die wahlbasierte Anmeldung und wählen Sie Bearbeiten.
3. Die Option Passwort ist immer verfügbar. Dies beinhaltet die `PASSWORD_SRP` Flüsse `PASSWORD` und. Wählen Sie die zusätzlichen Optionen aus, die Sie zu den Optionen Ihrer Benutzer hinzufügen möchten. Sie können Hauptschlüssel für**`WEB_AUTHN`**, Einmalpasswort für **`EMAIL_OTP`** E-Mail-Nachrichten und Einmalkennwort für **`SMS_OTP`** SMS-Nachrichten hinzufügen.
4. Wählen Sie Änderungen speichern aus.

API/SDK

Im folgenden Teil CreateUserPool- oder UpdateUserPoolAnforderungstext werden alle verfügbaren Optionen für die wahlbasierte Authentifizierung konfiguriert.

```
"Policies": {
    "SignInPolicy": {
        "AllowedFirstAuthFactors": [
            "PASSWORD",
            "WEB_AUTHN",
            "EMAIL_OTP",
            "SMS_OTP"
        ]
    }
},
```

Client-basierte Authentifizierung

Die clientbasierte Authentifizierung unterstützt die folgenden Authentifizierungsabläufe. Deklarieren Sie diese Optionen im `AuthFlow` Parameter von InitiateAuthoder AdminInitiateAuth.

1. `USER_PASSWORD_AUTH` und `ADMIN_USER_PASSWORD_AUTH`

   Melden Sie sich mit dauerhaften Passwörtern an

   MFA nach der Anmeldung

   Dieser Authentifizierungsablauf entspricht `PASSWORD` der wahlbasierten Authentifizierung.

2. `USER_SRP_AUTH`

   Melden Sie sich mit dauerhaften Passwörtern und sicherer Payload an

   MFA nach der Anmeldung

   Dieser Authentifizierungsablauf entspricht der wahlbasierten `PASSWORD_SRP` Authentifizierung.

3. `REFRESH_TOKEN_AUTH`

   Tokens aktualisieren

   Dieser Authentifizierungsablauf ist nur bei der clientbasierten Authentifizierung verfügbar.

4. `CUSTOM_AUTH`

   Benutzerdefinierte Authentifizierung

   Dieser Authentifizierungsablauf ist nur bei der clientbasierten Authentifizierung verfügbar.

Bei der clientbasierten Authentifizierung geht Amazon Cognito davon aus, dass Sie festgelegt haben, wie sich Ihr Benutzer authentifizieren möchte, bevor er mit den Authentifizierungsabläufen beginnt. Die Logik zur Bestimmung des Anmeldefaktors, den ein Benutzer angeben möchte, muss anhand von Standardeinstellungen oder benutzerdefinierten Eingabeaufforderungen festgelegt und dann in der ersten Anfrage an Ihren Benutzerpool deklariert werden. Die `InitiateAuth` Anfrage deklariert beispielsweise eine Anmeldung`AuthFlow`, die direkt einer der aufgelisteten Optionen entspricht. `USER_SRP_AUTH` Mit dieser Deklaration enthält die Anforderung auch die Parameter für den Beginn der Authentifizierung, z. B. `USERNAMESECRET_HASH`, und. `SRP_A` Amazon Cognito könnte auf diese Anfrage mit zusätzlichen Herausforderungen reagieren, z. B. `PASSWORD_VERIFIER` für SRP oder `SOFTWARE_TOKEN_MFA` für die Kennwortanmeldung mit TOTP MFA.

Um Ihren App-Client für die clientbasierte Authentifizierung zu konfigurieren, fügen Sie alle anderen Authentifizierungsabläufe als `ALLOW_USER_AUTH` die zulässigen Authentifizierungsabläufe hinzu. Beispiele sind `ALLOW_USER_PASSWORD_AUTHALLOW_CUSTOM_AUTH`,`ALLOW_REFRESH_TOKEN_AUTH`. Um clientbasierte Authentifizierungsabläufe zuzulassen, ist keine zusätzliche Benutzerpoolkonfiguration erforderlich.

Authentifizierungsabläufe

Der Prozess der Authentifizierung mit Amazon Cognito Cognito-Benutzerpools lässt sich am besten als ein Ablauf beschreiben, bei dem Benutzer eine erste Wahl treffen, Anmeldeinformationen einreichen und auf zusätzliche Herausforderungen reagieren. Wenn Sie die verwaltete Anmeldeauthentifizierung in Ihrer Anwendung implementieren, verwaltet Amazon Cognito den Ablauf dieser Eingabeaufforderungen und Herausforderungen. Wenn Sie Flows mit einem AWS SDK in Ihrem Anwendungs-Backend implementieren, müssen Sie die Logik der Anfragen erstellen, Benutzer zur Eingabe auffordern und auf Herausforderungen reagieren.

Als Anwendungsadministrator bestimmen Ihre Benutzereigenschaften, Sicherheitsanforderungen und Ihr Autorisierungsmodell, wie Sie Benutzern die Anmeldung gestatten möchten. Stellen Sie sich die folgenden Fragen.

- Möchte ich Benutzern erlauben, sich mit Anmeldeinformationen von anderen Identitätsanbietern anzumelden (IdPs)?
- Reichen ein Benutzername und ein Passwort als Identitätsnachweis aus?
- Könnten meine Authentifizierungsanfragen zur Authentifizierung mit Benutzername/Passwort abgefangen werden? Möchte ich, dass meine Anwendung Passwörter überträgt oder die Authentifizierung mithilfe von Hashes und Salts aushandelt?

- Möchte ich Benutzern erlauben, die Passworteingabe zu überspringen und ein Einmalpasswort zu erhalten, mit dem sie sich anmelden?
- Möchte ich Benutzern erlauben, sich mit einem Fingerabdruck, einem Gesicht oder einem Hardware-Sicherheitsschlüssel anzumelden?
- Wann möchte ich eine Multi-Faktor-Authentifizierung (MFA) verlangen, wenn überhaupt?
- Möchte ich Benutzersitzungen beibehalten, ohne erneut zur Eingabe von Anmeldeinformationen aufzufordern?
- Möchte ich mein Autorisierungsmodell über die integrierten Funktionen von Amazon Cognito hinaus erweitern?

Wenn Sie die Antworten auf diese Fragen haben, können Sie lernen, wie Sie die entsprechenden Funktionen aktivieren und sie in die Authentifizierungsanfragen Ihrer Anwendung implementieren.

Nachdem Sie Anmeldeabläufe für einen Benutzer eingerichtet haben, können Sie dessen aktuellen Status auf MFA und auswahlbasierte Authentifizierungsfaktoren anhand von Anfragen an den API-Vorgang überprüfen. GetUserAuthFactors Für diesen Vorgang ist eine Autorisierung mit dem Zugriffstoken eines angemeldeten Benutzers erforderlich. Es gibt Benutzerauthentifizierungsfaktoren und MFA-Einstellungen zurück.

Themen

- Melden Sie sich bei einem Drittanbieter an IdPs
- Melden Sie sich mit dauerhaften Passwörtern an
- Melden Sie sich mit dauerhaften Passwörtern und sicherer Payload an
- Passwortlose Anmeldung mit Einmalpasswörtern
- Passwortlose Anmeldung mit Passkeys WebAuthn
- MFA nach der Anmeldung
- Tokens aktualisieren
- Benutzerdefinierte Authentifizierung
- Ablauf der Authentifizierung für die Benutzermigration

Melden Sie sich bei einem Drittanbieter an IdPs

Amazon Cognito Cognito-Benutzerpools dienen als Zwischenvermittler für Authentifizierungssitzungen zwischen Diensten IdPs wie Sign in with Apple, Login with

Amazon und OpenID Connect (OIDC). Dieser Vorgang wird auch als Verbundanmeldung oder Verbundauthentifizierung bezeichnet. Bei der Verbundauthentifizierung werden keine der Authentifizierungsabläufe verwendet, die Sie in Ihren App-Client integrieren können. Stattdessen weisen Sie Ihrem App-Client einen konfigurierten Benutzerpool IdPs zu. Die Verbundanmeldung erfolgt, wenn Benutzer ihren IdP in der verwalteten Anmeldung auswählen oder Ihre Anwendung eine Sitzung mit einer Weiterleitung zu ihrer IdP-Anmeldeseite aufruft.

Mit der föderierten Anmeldung delegieren Sie primäre und MFA-Authentifizierungsfaktoren an den IdP des Benutzers. Amazon Cognito fügt die anderen erweiterten Flows in diesem Abschnitt einem Verbundbenutzer nicht hinzu, es sei denn, Sie verknüpfen sie mit einem lokalen Benutzer. Nicht verknüpfte Verbundbenutzer haben zwar Benutzernamen, aber es handelt sich dabei um einen Speicher zugeordneter Attributdaten, die unabhängig vom browserbasierten Flow normalerweise nicht für die Anmeldung verwendet werden.

Ressourcen für die Implementierung

- Benutzerpool-Anmeldung mit externen Identitätsanbietern

Melden Sie sich mit dauerhaften Passwörtern an

In Amazon Cognito Cognito-Benutzerpools hat jeder Benutzer einen Benutzernamen. Dies kann eine Telefonnummer, eine E-Mail-Adresse oder eine gewählte oder vom Administrator bereitgestellte Kennung sein. Benutzer dieses Typs können sich mit ihrem Benutzernamen und ihrem Passwort anmelden und optional MFA bereitstellen. Benutzerpools können die Anmeldung mit Benutzername/Passwort mit öffentlichen oder IAM-autorisierten API-Operationen und SDK-Methoden durchführen. Ihre Anwendung kann das Passwort zur Authentifizierung direkt an Ihren Benutzerpool senden. Ihr Benutzerpool reagiert mit zusätzlichen Herausforderungen oder den JSON-Webtoken (JWTs), die das Ergebnis einer erfolgreichen Authentifizierung sind.

Activate password sign-in

Um die clientbasierte Authentifizierung mit Benutzername und Passwort zu aktivieren, konfigurieren Sie Ihren App-Client so, dass dies zulässig ist. Navigieren Sie in der Amazon Cognito Cognito-Konsole in Ihrer Benutzerpool-Konfiguration unter Anwendungen zum Menü App-Clients. Um die Anmeldung mit einem einfachen Passwort für eine clientseitige mobile oder native App zu ermöglichen, bearbeiten Sie einen App-Client und wählen Sie unter Authentifizierungsabläufe die Option Mit Benutzername und Passwort anmelden: ALLOW_USER_PASSWORD_AUTH aus. Um die Anmeldung mit einem

einfachen Passwort für eine serverseitige App zu ermöglichen, bearbeiten Sie einen App-Client und wählen Sie Mit serverseitigen Administratoranmeldedaten anmelden: ALLOW_ADMIN_USER_PASSWORD_AUTH.

Um die wahlbasierte Authentifizierung mit Benutzername und Passwort zu aktivieren, konfigurieren Sie Ihren App-Client so, dass dies zulässig ist. Bearbeiten Sie Ihren App-Client und wählen Sie Wahlbasierte Anmeldung: ALLOW_USER_AUTH.

Um zu überprüfen, ob die Kennwortauthentifizierung in wahlbasierten Authentifizierungsabläufen verfügbar ist, navigieren Sie zum Anmeldemenü und lesen Sie den Abschnitt unter Optionen für die wahlbasierte Anmeldung. Sie können sich mit einfacher Passwortauthentifizierung anmelden, wenn das Passwort unter Verfügbare Optionen sichtbar ist. Die Option Passwort umfasst die Varianten für die einfache Authentifizierung und die SRP-Authentifizierung mit Benutzername/Passwort.

Konfigurieren Sie `ExplicitAuthFlows` mit Ihren bevorzugten username-and-password Authentifizierungsoptionen in einer CreateUserPoolClientUpdateUserPoolClientOder-Anfrage.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_PASSWORD_AUTH",
   "ALLOW_ADMIN_USER_PASSWORD_AUTH",
   "ALLOW_USER_AUTH"
]
```

Konfigurieren Sie in einer CreateUserPoolUpdateUserPoolOr-Anforderung `Policies` mit den wahlbasierten Authentifizierungsabläufen, die Sie unterstützen möchten. Der `PASSWORD` Wert in `AllowedFirstAuthFactors` umfasst sowohl die Ablaufoptionen für die Authentifizierung mit einfachem Passwort als auch die SRP-Authentifizierung.

```
"Policies": {
   "SignInPolicy": {
      "AllowedFirstAuthFactors": [
         "PASSWORD",
         "EMAIL_OTP",
         "WEB_AUTHN"
      ]
   }
}
```

Choice-based sign-in with a password

Um einen Benutzer mit Benutzername/Passwort-Authentifizierung bei einer Anwendung anzumelden, konfigurieren Sie den Text Ihrer OR-Anforderung wie folgt. AdminInitiateAuthInitiateAuth Diese Anmeldeanforderung ist erfolgreich oder wird mit der nächsten Herausforderung fortgesetzt, wenn der aktuelle Benutzer für die Authentifizierung mit Benutzername/Passwort berechtigt ist. Andernfalls wird eine Liste der verfügbaren Herausforderungen bei der Authentifizierung mit dem Primärfaktor angezeigt. Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PREFERRED_CHALLENGE" : "PASSWORD",
      "PASSWORD" : "[User's password]"
   },
   "ClientId": "1example23456789"
}
```

Sie können den `PREFERRED_CHALLENGE` Wert auch weglassen und eine Antwort erhalten, die eine Liste der für den Benutzer in Frage kommenden Anmeldefaktoren enthält.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser"
   },
   "ClientId": "1example23456789"
}
```

Wenn Sie keine bevorzugte Herausforderung eingereicht haben oder der eingereichte Benutzer nicht für seine bevorzugte Herausforderung berechtigt ist, gibt Amazon Cognito eine Liste mit Optionen in `AvailableChallenges` zurück. Wenn ein `ChallengeName` von `AvailableChallenges` enthalten ist`PASSWORD`, können Sie die Authentifizierung mit einer Antwort RespondToAuthChallengeoder einer AdminRespondToAuthChallengeChallenge-Antwort im folgenden Format fortsetzen. Sie müssen einen `Session` Parameter übergeben, der die Challenge-Antwort mit der API-Antwort auf Ihre erste Anmeldeanforderung verknüpft. Dieser

Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

```
{
   "ChallengeName": "PASSWORD",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "PASSWORD" : "[User's Password]"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response"
}
```

Amazon Cognito beantwortet qualifizierte und erfolgreiche Preferred-Challenge-Anfragen und `PASSWORD` Challenge-Antworten mit Tokens oder einer zusätzlichen erforderlichen Herausforderung wie Multi-Faktor-Authentifizierung (MFA).

Client-based sign-in with a password

Um einen Benutzer mit Benutzername-Passwort-Authentifizierung bei einer clientseitigen App anzumelden, konfigurieren Sie den Text Ihrer Anfrage wie folgt. InitiateAuth Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

```
{
   "AuthFlow": "USER_PASSWORD_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PASSWORD" : "[User's password]"
   },
   "ClientId": "1example23456789"
}
```

Um einen Benutzer mit Benutzername-Passwort-Authentifizierung bei einer serverseitigen App anzumelden, konfigurieren Sie den Text Ihrer AdminInitiateAuthAnfrage wie folgt. Ihre Anwendung muss diese Anfrage mit Anmeldeinformationen signieren. AWS Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

```
{
   "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
```

```
        "PASSWORD" : "[User's password]"
    },
    "ClientId": "1example23456789"
}
```

Amazon Cognito reagiert auf erfolgreiche Anfragen mit Tokens oder einer zusätzlichen erforderlichen Herausforderung wie der Multi-Faktor-Authentifizierung (MFA).

Melden Sie sich mit dauerhaften Passwörtern und sicherer Payload an

Eine weitere Form der Anmeldemethoden für Benutzername/Passwort in Benutzerpools ist das Secure Remote Password (SRP) -Protokoll. Bei dieser Option wird der Kenntnisnachweis über ein Passwort — ein Kennwort-Hash und ein Salt — gesendet, den Ihr Benutzerpool überprüfen kann. Da die Anfrage an Amazon Cognito keine lesbaren geheimen Informationen enthält, ist Ihre Anwendung die einzige Entität, die die von Benutzern eingegebenen Passwörter verarbeitet. Die SRP-Authentifizierung beinhaltet mathematische Berechnungen, die am besten mit einer vorhandenen Komponente durchgeführt werden, die Sie in Ihr SDK importieren können. SRP wird in der Regel in clientseitigen Anwendungen wie mobilen Apps implementiert. Weitere Informationen zum Protokoll finden Sie auf der Stanford SRP-Homepage. Wikipedia hat auch Ressourcen und Beispiele. Eine Vielzahl von öffentlichen Bibliotheken steht zur Verfügung, um die SRP-Berechnungen für Ihre Authentifizierungsabläufe durchzuführen.

Die initiate-challenge-respond Reihenfolge der Amazon Cognito Cognito-Authentifizierung validiert Benutzer und ihre Passwörter mit SRP. Sie müssen Ihren Benutzerpool und Ihren App-Client so konfigurieren, dass sie die SRP-Authentifizierung unterstützen, und dann die Logik der Anmeldeanfragen und Challenge-Antworten in Ihrer Anwendung implementieren. Ihre SRP-Bibliotheken können Zufallszahlen und berechnete Werte generieren, die Ihrem Benutzerpool zeigen, dass Sie im Besitz eines Benutzerkennworts sind. Ihre Anwendung füllt diese berechneten Werte in die Felder im JSON-Format `AuthParameters` und in die `ChallengeParameters` Felder der Amazon Cognito Cognito-Benutzerpools, API-Operationen und SDK-Methoden für die Authentifizierung ein.

Activate SRP sign-in

Um die clientbasierte Authentifizierung mit Benutzername und SRP zu aktivieren, konfigurieren Sie Ihren App-Client so, dass dies zulässig ist. Navigieren Sie in der Amazon Cognito Cognito-Konsole in Ihrer Benutzerpool-Konfiguration unter Anwendungen zum Menü App-Clients. Um die SRP-Anmeldung für eine clientseitige mobile oder native App zuzulassen, bearbeiten Sie

einen App-Client und wählen Sie unter Authentifizierungsabläufe die Option Mit sicherem Remote-Passwort (SRP) anmelden: ALLOW_USER_SRP_AUTH aus.

Um die wahlbasierte Authentifizierung mit Benutzername und SRP zu aktivieren, bearbeiten Sie Ihren App-Client und wählen Sie Wahlbasierte Anmeldung: ALLOW_USER_AUTH.

Um zu überprüfen, ob die SRP-Authentifizierung in Ihren wahlbasierten Authentifizierungsabläufen verfügbar ist, navigieren Sie zum Anmeldemenü und lesen Sie den Abschnitt unter Optionen für die wahlbasierte Anmeldung. Sie können sich mit der SRP-Authentifizierung anmelden, wenn das Passwort unter Verfügbare Optionen sichtbar ist. Die Option Passwort umfasst die Authentifizierungsvarianten Klartext und SRP Benutzername/Passwort.

Konfigurieren Sie `ExplicitAuthFlows` mit Ihren bevorzugten username-and-password Authentifizierungsoptionen in einer CreateUserPoolClientUpdateUserPoolClientOder-Anfrage.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_SRP_AUTH",
   "ALLOW_USER_AUTH"
]
```

Konfigurieren Sie in einer CreateUserPoolUpdateUserPoolOr-Anforderung `Policies` mit den wahlbasierten Authentifizierungsabläufen, die Sie unterstützen möchten. Der `PASSWORD` Wert in `AllowedFirstAuthFactors` umfasst sowohl die Optionen für den Klartext-Passwort als auch für den SRP-Authentifizierungsfluss.

```
"Policies": {
   "SignInPolicy": {
      "AllowedFirstAuthFactors": [
         "PASSWORD",
         "EMAIL_OTP",
         "WEB_AUTHN"
      ]
   }
}
```

Choice-based sign-in with SRP

Um einen Benutzer mit Benutzername-Passwort-Authentifizierung mit SRP bei einer Anwendung anzumelden, konfigurieren Sie den Text Ihrer OR-Anforderung wie folgt.

AdminInitiateAuthInitiateAuth Diese Anmeldeanforderung ist erfolgreich oder wird mit der nächsten Herausforderung fortgesetzt, wenn der aktuelle Benutzer für die Authentifizierung mit Benutzername/Passwort berechtigt ist. Andernfalls wird eine Liste der verfügbaren Herausforderungen bei der Authentifizierung mit dem Primärfaktor angezeigt. Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PREFERRED_CHALLENGE" : "PASSWORD_SRP",
      "SRP_A" : "[g^a % N]"
   },
   "ClientId": "1example23456789"
}
```

Sie können den `PREFERRED_CHALLENGE` Wert auch weglassen und eine Antwort erhalten, die eine Liste der für den Benutzer in Frage kommenden Anmeldefaktoren enthält.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser"
   },
   "ClientId": "1example23456789"
}
```

Wenn Sie keine bevorzugte Herausforderung eingereicht haben oder der eingereichte Benutzer nicht für seine bevorzugte Herausforderung berechtigt ist, gibt Amazon Cognito eine Liste mit Optionen in `AvailableChallenges` zurück. Wenn ein `ChallengeName` von `AvailableChallenges` enthalten ist`PASSWORD_SRP`, können Sie die Authentifizierung mit einer Antwort RespondToAuthChallengeoder einer AdminRespondToAuthChallengeChallenge-Antwort im folgenden Format fortsetzen. Sie müssen einen `Session` Parameter übergeben, der die Challenge-Antwort mit der API-Antwort auf Ihre erste Anmeldeanforderung verknüpft. Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

```
{
```

```
   "ChallengeName": "PASSWORD_SRP",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "SRP_A" : "[g^a % N]"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response"
}
```

Amazon Cognito beantwortet qualifizierte Anfragen mit einer bevorzugten `PASSWORD_SRP` Herausforderung und Antworten auf Challenge-Antworten mit einer Aufforderung. `PASSWORD_VERIFIER` Ihr Kunde muss die SRP-Berechnungen durchführen und auf die Herausforderung in einer OR-Anfrage antworten. RespondToAuthChallengeAdminRespondToAuthChallenge

```
{
   "ChallengeName": "PASSWORD_VERIFIER",
   "ChallengeResponses": {
      "PASSWORD_CLAIM_SIGNATURE" : "string",
      "PASSWORD_CLAIM_SECRET_BLOCK" : "string",
      "TIMESTAMP" : "string"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

Bei einer erfolgreichen Antwort auf eine `PASSWORD_VERIFIER` Herausforderung gibt Amazon Cognito Token oder eine andere erforderliche Herausforderung wie die Multi-Faktor-Authentifizierung (MFA) aus.

Client-based sign-in with SRP

Die SRP-Authentifizierung ist bei der clientseitigen Authentifizierung üblicher als bei der serverseitigen. Sie können die SRP-Authentifizierung jedoch mit und verwenden. InitiateAuthAdminInitiateAuth Um einen Benutzer bei einer Anwendung anzumelden, konfigurieren Sie den Hauptteil Ihrer `InitiateAuth` `AdminInitiateAuth` OR-Anforderung wie folgt. Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

Der Client generiert *SRP_A* aus einem Generator Modulo N g, der auf die Potenz einer geheimen zufälligen Ganzzahl a erhöht wird.

```
{
   "AuthFlow": "USER_SRP_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "SRP_A" : "[g^a % N]"
   },
   "ClientId": "1example23456789"
}
```

Amazon Cognito reagiert mit einer `PASSWORD_VERIFIER`-Aufgabe. Ihr Kunde muss die SRP-Berechnungen durchführen und auf die Herausforderung in einer RespondToAuthChallengeOder-Anfrage antworten. AdminRespondToAuthChallenge

```
{
   "ChallengeName": "PASSWORD_VERIFIER",
   "ChallengeResponses": {
      "PASSWORD_CLAIM_SIGNATURE" : "string",
      "PASSWORD_CLAIM_SECRET_BLOCK" : "string",
      "TIMESTAMP" : "string"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

Bei einer erfolgreichen Antwort auf eine `PASSWORD_VERIFIER` Herausforderung gibt Amazon Cognito Token oder eine andere erforderliche Herausforderung wie die Multi-Faktor-Authentifizierung (MFA) aus.

Passwortlose Anmeldung mit Einmalpasswörtern

Passwörter können verloren gehen oder gestohlen werden. Möglicherweise möchten Sie nur überprüfen, ob Ihre Benutzer Zugriff auf eine verifizierte E-Mail-Adresse, Telefonnummer oder Authentifizierungs-App haben. Die Lösung hierfür ist die passwortlose Anmeldung. Ihre Anwendung kann Benutzer zur Eingabe ihres Benutzernamens, ihrer E-Mail-Adresse oder Telefonnummer auffordern. Amazon Cognito generiert dann ein Einmalkennwort (OTP), einen Code, den sie bestätigen müssen. Ein erfolgreicher Code schließt die Authentifizierung ab.

Passwortlose Authentifizierungsabläufe sind nicht mit der erforderlichen Multi-Faktor-Authentifizierung (MFA) in Ihrem Benutzerpool kompatibel. Wenn MFA in Ihrem Benutzerpool optional ist, können

sich Benutzer, die MFA aktiviert haben, nicht mit einem kennwortlosen ersten Faktor anmelden. Benutzer, die keine MFA-Präferenz in einem MFA-optionalen Benutzerpool haben, können sich mit kennwortlosen Faktoren anmelden. Weitere Informationen finden Sie unter Wissenswertes über den Benutzerpool MFA.

Wenn ein Benutzer im Rahmen der kennwortlosen Authentifizierung einen Code, den er in einer SMS oder E-Mail-Nachricht erhalten hat, korrekt eingibt, markiert Ihr Benutzerpool zusätzlich zur Authentifizierung des Benutzers auch die nicht verifizierte E-Mail-Adresse oder das Telefonnummernattribut des Benutzers als verifiziert. Der Benutzerstatus wurde ebenfalls von `UNCONFIRMED` zu geändert`CONFIRMED`, unabhängig davon, ob Sie Ihren Benutzerpool so konfiguriert haben, dass E-Mail-Adressen oder Telefonnummern automatisch verifiziert werden.

Neue Optionen mit passwortloser Anmeldung

Wenn Sie die passwortlose Authentifizierung in Ihrem Benutzerpool aktivieren, ändert sich dadurch die Funktionsweise einiger Benutzerabläufe.

1. Benutzer können sich ohne Passwort registrieren und bei der Anmeldung einen passwortlosen Faktor wählen. Sie können als Administrator auch Benutzer ohne Passwörter erstellen.
2. Benutzer, die Sie mit einer CSV-Datei importieren, können sich sofort mit einem passwortlosen Faktor anmelden. Sie müssen vor der Anmeldung kein Passwort festlegen.
3. Benutzer, die kein Passwort haben, können ChangePasswordAPI-Anfragen ohne den `PreviousPassword` Parameter einreichen.

Automatische Anmeldung mit OTPs

Benutzer, die sich registrieren und ihre Benutzerkonten per E-Mail oder SMS-Nachricht bestätigen, OTPs können sich automatisch mit dem passwortlosen Faktor anmelden, der ihrer Bestätigungsnachricht entspricht. Auf der verwalteten Anmelde-Benutzeroberfläche können Benutzer, die ihre Konten bestätigen und für die OTP-Anmeldung mit der Methode zur Übermittlung des Bestätigungscodes berechtigt sind, automatisch zu ihrer ersten Anmeldung übergehen, nachdem sie den Bestätigungscode eingegeben haben. Übergeben Sie in Ihrer maßgeschneiderten Anwendung mit einem AWS SDK die folgenden Parameter an eine Oder-Operation. InitiateAuthAdminInitiateAuth

- Der `Session` Parameter aus der ConfirmSignUpAPI-Antwort als `Session` Anforderungsparameter.
- Ein AuthFlowvon`USER_AUTH`.

Sie können eine PREFERRED_CHALLENGE von `EMAIL_OTP` oder bestehen`SMS_OTP`, dies ist jedoch nicht erforderlich. Der `Session` Parameter liefert einen Authentifizierungsnachweis und Amazon Cognito ignoriert den, `AuthParameters` wenn Sie einen gültigen Sitzungscode übergeben.

Der Anmeldevorgang gibt die Antwort zurück, die auf eine erfolgreiche Authentifizierung hinweist AuthenticationResult, ohne zusätzliche Herausforderungen, wenn die folgenden Bedingungen zutreffen.

- Der `Session` Code ist gültig und nicht abgelaufen.
- Der Benutzer hat Anspruch auf die OTP-Authentifizierungsmethode.

Activate passwordless sign-in

Konsole

Um die passwortlose Anmeldung zu aktivieren, konfigurieren Sie Ihren Benutzerpool so, dass die primäre Anmeldung mit einem oder mehreren kennwortlosen Typen zulässig ist, und konfigurieren Sie dann Ihren App-Client so, dass der Datenfluss zulässig ist. `USER_AUTH` Navigieren Sie in der Amazon Cognito Cognito-Konsole in Ihrer Benutzerpool-Konfiguration unter Authentifizierung zum Anmelde-Menü. Bearbeiten Sie die Optionen für die wahlbasierte Anmeldung und wählen Sie Einmalpasswort für E-Mail-Nachricht oder Einmalpasswort für SMS-Nachricht. Sie können beide Optionen aktivieren. Speichern Sie Ihre Änderungen.

Navigieren Sie zum Menü App-Clients und wählen Sie einen App-Client aus oder erstellen Sie einen neuen. Wählen Sie Bearbeiten und dann Wählen Sie bei der Anmeldung einen Authentifizierungstyp aus: ALLOW_USER_AUTH.

API/SDK

Konfigurieren Sie in der Benutzerpools-API `SignInPolicy` mit den entsprechenden kennwortlosen Optionen in einer Oder-Anfrage. CreateUserPoolUpdateUserPool

```
"SignInPolicy": {
    "AllowedFirstAuthFactors": [
        "EMAIL_OTP",
        "SMS_OTP"
    ]
}
```

Konfigurieren Sie Ihren App-Client `ExplicitAuthFlows` mit der erforderlichen Option in einer CreateUserPoolClientUpdateUserPoolClientOder-Anfrage.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_AUTH"
]
```

Sign in with passwordless

Bei der passwortlosen Anmeldung gibt es keine clientbasierte Anmeldung`AuthFlow`, die Sie in und angeben können. InitiateAuthAdminInitiateAuth Die OTP-Authentifizierung ist nur in der wahlbasierten Variante verfügbar. Dabei können Sie eine bevorzugte Anmeldeoption anfordern oder aus `AuthFlow` der `USER_AUTH` Option eines Benutzers die passwortlose Option auswählen. AvailableChallenges Um einen Benutzer bei einer Anwendung anzumelden, konfigurieren Sie den Text Ihrer OR-Anforderung wie folgt. `InitiateAuth` `AdminInitiateAuth` Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

In diesem Beispiel wissen wir nicht, auf welche Weise sich der Benutzer anmelden möchte. Wenn wir einen `PREFERRED_CHALLENGE` Parameter hinzufügen und die bevorzugte Herausforderung für den Benutzer verfügbar ist, antwortet Amazon Cognito mit dieser Herausforderung.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser"
   },
   "ClientId": "1example23456789"
}
```

`AuthParameters`In diesem Beispiel können Sie stattdessen `"PREFERRED_CHALLENGE": "EMAIL_OTP"` oder `"PREFERRED_CHALLENGE": "SMS_OTP"` hinzufügen. Wenn der Benutzer für diese bevorzugte Methode in Frage kommt, sendet Ihr Benutzerpool sofort einen Code an die E-Mail-Adresse oder Telefonnummer des Benutzers und gibt `"ChallengeName": "EMAIL_OTP"` oder zurück`"ChallengeName": "SMS_OTP"`.

Wenn Sie keine bevorzugte Herausforderung angeben, antwortet Amazon Cognito mit einem `AvailableChallenges` Parameter.

```
{
```

```
   "AvailableChallenges": [
      "EMAIL_OTP",
      "SMS_OTP",
      "PASSWORD"
    ],
   "Session": "[Session ID]"
}
```

Dieser Benutzer ist berechtigt, sich passwortlos mit E-Mail-Nachrichten-OTP, SMS-Nachrichten-OTP und Benutzername-Passwort anzumelden. Ihre Anwendung kann den Benutzer zur Eingabe seiner Auswahl auffordern oder eine Auswahl auf der Grundlage interner Logik treffen. Anschließend wird mit einer AdminRespondToAuthChallengeOder-Anfrage RespondToAuthChallengefortgefahren, mit der die Herausforderung ausgewählt wird. Angenommen, der Benutzer möchte die kennwortlose Authentifizierung mit einem Einmalpasswort für eine E-Mail-Nachricht abschließen.

```
{
   "ChallengeName": "SELECT_CHALLENGE",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "ANSWER" : "EMAIL_OTP"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

Amazon Cognito antwortet mit einer `EMAIL_OTP` Aufforderung und sendet einen Code an die verifizierte E-Mail-Adresse Ihres Benutzers. Ihre Anwendung muss dann erneut auf diese Aufforderung antworten.

Dies wäre auch die Antwort auf die nächste Herausforderung, wenn Sie `EMAIL_OTP` als Antwort darauf anfragen würden`PREFERRED_CHALLENGE`.

```
{
   "ChallengeName": "EMAIL_OTP",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "EMAIL_OTP_CODE" : "123456"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
```

```
}
```

Passwortlose Anmeldung mit Passkeys WebAuthn

Hauptschlüssel sind sicher und bedeuten für die Benutzer einen relativ geringen Aufwand. Die Passkey-Anmeldung verwendet Authentifikatoren, externe Geräte, mit denen sich Benutzer authentifizieren können. Reguläre Passwörter setzen Benutzer Sicherheitslücken wie Phishing, Erraten von Passwörtern und Diebstahl von Anmeldeinformationen aus. Mit Passkeys kann Ihre Anwendung von fortschrittlichen Sicherheitsmaßnahmen auf Mobiltelefonen und anderen Geräten profitieren, die an Informationssysteme angeschlossen oder in diese integriert sind. Ein gängiger Anmeldeprozess mit dem Hauptschlüssel beginnt mit einem Anruf auf Ihrem Gerät, der Ihren Passwort- oder Anmeldeinformationsmanager aufruft, z. B. den iOS-Schlüsselbund oder den Google Chrome-Passwort-Manager. Der Manager für Anmeldeinformationen auf dem Gerät fordert sie auf, einen Hauptschlüssel auszuwählen und ihn mit einem vorhandenen Mechanismus zum Entsperren von Anmeldeinformationen oder Geräten zu autorisieren. Moderne Telefone verfügen über Gesichtsscanner, Fingerabdruckscanner, Entsperrmuster und andere Mechanismen, von denen einige gleichzeitig das Prinzip der starken Authentifizierung erfüllen, das etwas, das Sie kennen, und etwas, das Sie haben, erfüllen. Im Fall der biometrischen Authentifizierung mit Passkeys stehen Kennwörter für etwas, das Sie sind.

Möglicherweise möchten Sie Passwörter durch die Authentifizierung per Fingerabdruck, Gesicht oder Sicherheitsschlüssel ersetzen. Das ist Hauptschlüssel oder Authentifizierung. WebAuthn Es ist üblich, dass Anwendungsentwickler Benutzern gestatten, ein biometrisches Gerät zu registrieren, nachdem sie sich zum ersten Mal mit einem Passwort angemeldet haben. Mit Amazon Cognito Cognito-Benutzerpools kann Ihre Anwendung diese Anmeldeoption für Benutzer konfigurieren. Die Passkey-Authentifizierung ist nicht für die Multi-Faktor-Authentifizierung (MFA) geeignet.

Passwortlose Authentifizierungsabläufe sind nicht mit der erforderlichen Multi-Faktor-Authentifizierung (MFA) in Ihrem Benutzerpool kompatibel. Wenn MFA in Ihrem Benutzerpool optional ist, können sich Benutzer, die MFA aktiviert haben, nicht mit einem kennwortlosen ersten Faktor anmelden. Benutzer, die keine MFA-Präferenz in einem MFA-optionalen Benutzerpool haben, können sich mit kennwortlosen Faktoren anmelden. Weitere Informationen finden Sie unter Wissenswertes über den Benutzerpool MFA.

Was sind Hauptschlüssel?

Passkeys vereinfachen die Benutzererfahrung, da sie sich keine komplexen Passwörter merken oder eingeben müssen. OTPs Passkeys basieren auf WebAuthn CTAP2 Standards, die vom World Wide

Web Consortium (W3C) und der FIDO (Fast Identity Online) Alliance entworfen wurden. Browser und Plattformen implementieren diese Standards und ermöglichen es Web- oder Mobilanwendungen, einen Prozess APIs zur Registrierung oder Authentifizierung von Passkeys zu starten, sowie eine Benutzeroberfläche, über die der Benutzer einen Passkey-Authentifikator auswählen und mit diesem interagieren kann.

Wenn ein Benutzer einen Authentifikator bei einer Website oder einer App registriert, erstellt der Authenticator ein öffentlich-privates key pair. WebAuthn Browser und Plattformen senden den öffentlichen Schlüssel an das Anwendungs-Backend der Website oder App. Der Authentifikator speichert den privaten Schlüssel, den Schlüssel IDs und die Metadaten über den Benutzer und die Anwendung. Wenn sich der Benutzer in der registrierten Anwendung mit seinem registrierten Authentifikator authentifizieren möchte, generiert die Anwendung eine zufällige Aufforderung. Die Antwort auf diese Herausforderung ist die digitale Signatur der Aufforderung, die mit dem privaten Schlüssel des Authentifikators für diese Anwendung und diesen Benutzer sowie den entsprechenden Metadaten generiert wird. Der Browser oder die Anwendungsplattform empfängt die digitale Signatur und leitet sie an das Anwendungs-Backend weiter. Die Anwendung validiert dann die Signatur mit dem gespeicherten öffentlichen Schlüssel.

> **Note**
>
> Ihre Anwendung erhält weder Authentifizierungsgeheimnisse, die Benutzer ihrem Authentifikator zur Verfügung stellen, noch Informationen über den privaten Schlüssel.

Im Folgenden finden Sie einige Beispiele und Funktionen von Authentifikatoren, die derzeit auf dem Markt erhältlich sind. Ein Authentifikator kann eine oder alle dieser Kategorien erfüllen.

- Einige Authentifikatoren führen eine Benutzerverifizierung anhand von Faktoren wie einer PIN, einer biometrischen Eingabe mit einem Gesicht oder einem Fingerabdruck oder einem Passcode durch, bevor sie den Zugriff gewähren. Dadurch wird sichergestellt, dass nur der legitime Benutzer Aktionen autorisieren kann. Andere Authentifikatoren verfügen nicht über Funktionen zur Benutzerverifizierung, und einige können die Benutzerverifizierung überspringen, wenn eine Anwendung sie nicht benötigt.
- Einige Authentifikatoren, zum Beispiel YubiKey Hardware-Token, sind portabel. Sie kommunizieren mit Geräten über USB-, Bluetooth- oder NFC-Verbindungen. Einige Authentifikatoren sind lokal und an eine Plattform gebunden, z. B. Windows Hello auf einem PC oder Face ID auf einem iPhone. Ein gerätegebundener Authentifikator kann vom Benutzer mitgeführt werden, wenn er klein genug ist, wie ein Mobilgerät. Manchmal können Benutzer ihren Hardware-Authentifikator mit vielen

verschiedenen Plattformen über drahtlose Kommunikation verbinden. Beispielsweise können Benutzer in Desktop-Browsern ihr Smartphone als Hauptschlüsselauthentifikator verwenden, wenn sie einen QR-Code scannen.

- Einige plattformgebundene Hauptschlüssel werden mit der Cloud synchronisiert, sodass sie von mehreren Standorten aus verwendet werden können. Beispielsweise synchronisieren Face ID-Passkeys auf iPhones die Passkey-Metadaten mit den Apple-Konten der Nutzer in deren iCloud-Schlüsselbund. Diese Hauptschlüssel ermöglichen eine nahtlose Authentifizierung auf allen Apple-Geräten, anstatt zu verlangen, dass Benutzer jedes Gerät einzeln registrieren. Softwarebasierte Authentifikator-Apps wie 1Password, Dashlane und Bitwarden synchronisieren Passkeys auf allen Plattformen, auf denen der Benutzer die App installiert hat.

In der WebAuthn Terminologie sind Websites und Apps vertrauende Parteien. Jedem Hauptschlüssel ist eine spezifische ID der vertrauenden Partei zugeordnet. Dabei handelt es sich um eine einheitliche Kennung, die die Websites oder Apps darstellt, die die Passkey-Authentifizierung akzeptieren. Entwickler müssen ihre ID der vertrauenden Partei sorgfältig auswählen, um den richtigen Authentifizierungsumfang zu gewährleisten. Eine typische ID der vertrauenden Partei ist der Stammdomänenname eines Webservers. Ein Hauptschlüssel mit dieser Spezifikation für die ID der vertrauenden Partei kann für diese Domain und Subdomänen authentifiziert werden. Browser und Plattformen verweigern die Passkey-Authentifizierung, wenn die URL der Website, auf die ein Benutzer zugreifen möchte, nicht mit der ID der vertrauenden Partei übereinstimmt. Ähnlich kann für mobile Apps ein Hauptschlüssel nur verwendet werden, wenn der App-Pfad in den `.well-known` Zuordnungsdateien enthalten ist, die die Anwendung unter dem durch die ID der vertrauenden Partei angegebenen Pfad zur Verfügung stellt.

Hauptschlüssel sind auffindbar. Sie können von einem Browser oder einer Plattform automatisch erkannt und verwendet werden, ohne dass der Benutzer einen Benutzernamen eingeben muss. Wenn ein Benutzer eine Website oder App besucht, die die Hauptschlüsselauthentifizierung unterstützt, kann er aus einer Liste von Hauptschlüsseln auswählen, die dem Browser oder der Plattform bereits bekannt sind, oder er kann einen QR-Code scannen.

Wie implementiert Amazon Cognito die Passkey-Authentifizierung?

Hauptschlüssel sind eine optionale Funktion, die in allen Funktionsplänen außer Lite verfügbar ist. Sie ist nur im wahlbasierten Authentifizierungsablauf verfügbar. Bei der verwalteten Anmeldung übernimmt Amazon Cognito die Logik der Passkey-Authentifizierung. Sie können auch die Amazon Cognito-Benutzerpools-API verwenden, AWS SDKs um die Passkey-Authentifizierung in Ihrem Anwendungs-Backend durchzuführen.

Amazon Cognito erkennt Hauptschlüssel, die mit einem der beiden asymmetrischen kryptografischen Algorithmen ES256 (-7) und (-257) erstellt wurden. RS256 Die meisten Authentifikatoren unterstützen beide Algorithmen. Standardmäßig können Benutzer jede Art von Authentifikatoren einrichten, z. B. Hardware-Token, mobile Smartphones und Software-Authenticator-Apps. Amazon Cognito unterstützt derzeit nicht die Durchsetzung von Bescheinigungen.

In Ihrem Benutzerpool können Sie die Benutzerverifizierung so konfigurieren, dass sie bevorzugt oder erforderlich ist. Diese Einstellung ist in API-Anfragen, die keinen Wert angeben, standardmäßig auf „bevorzugt“ eingestellt, und in der Amazon Cognito Cognito-Konsole ist standardmäßig „bevorzugt“ ausgewählt. Wenn Sie die Benutzerverifizierung auf bevorzugt setzen, können Benutzer Authentifikatoren einrichten, die nicht über die Funktion zur Benutzerverifizierung verfügen, und Registrierungs- und Authentifizierungsvorgänge können auch ohne Benutzerverifizierung erfolgreich sein. Um die Benutzerverifizierung bei der Registrierung und Authentifizierung mit dem Hauptschlüssel vorzuschreiben, ändern Sie diese Einstellung in „Erforderlich“.

Die ID der vertrauenden Partei (Relying Party, RP), die Sie in Ihrer Hauptschlüsselkonfiguration festgelegt haben, ist eine wichtige Entscheidung. Wenn Sie nichts anderes angeben und Ihre Domain-Branding-Version eine verwaltete Anmeldung ist, erwartet Ihr Benutzerpool standardmäßig den Namen Ihrer benutzerdefinierten Domain als RP-ID. Wenn Sie keine benutzerdefinierte Domain haben und nichts anderes angeben, verwendet Ihr Benutzerpool standardmäßig eine RP-ID Ihrer Präfix-Domain. Sie können Ihre RP-ID auch so konfigurieren, dass sie ein beliebiger Domainname ist, der nicht in der öffentlichen Suffixliste (PSL) enthalten ist. Ihr RP-ID-Eintrag gilt für die Registrierung und Authentifizierung mit dem Hauptschlüssel bei der verwalteten Anmeldung und bei der SDK-Authentifizierung. Passkey funktioniert nur in mobilen Anwendungen, bei denen Amazon Cognito eine `.well-known` Zuordnungsdatei mit Ihrer RP-ID als Domain finden kann. Es hat sich bewährt, den Wert der ID Ihrer vertrauenden Partei zu ermitteln und festzulegen, bevor Ihre Website oder App öffentlich verfügbar ist. Wenn Sie Ihre RP-ID ändern, müssen sich Ihre Benutzer erneut mit der neuen RP-ID registrieren.

Jeder Benutzer kann bis zu 20 Hauptschlüssel registrieren. Sie können einen Hauptschlüssel erst registrieren, nachdem sie sich mindestens einmal bei Ihrem Benutzerpool angemeldet haben. Durch die verwaltete Anmeldung entfällt ein erheblicher Aufwand bei der Registrierung des Hauptschlüssels. Wenn Sie die Hauptschlüsselauthentifizierung für einen Benutzerpool und einen App-Client aktivieren, erinnert Ihr Benutzerpool mit einer verwalteten Anmeldedomäne Endbenutzer daran, einen Hauptschlüssel zu registrieren, nachdem sie sich für ein neues Benutzerkonto registriert haben. Sie können die Browser der Benutzer auch jederzeit aufrufen, um sie zur Registrierung des Hauptschlüssels auf eine verwaltete Anmeldeseite weiterzuleiten. Benutzer müssen einen Benutzernamen angeben, bevor Amazon Cognito die Hauptschlüsselauthentifizierung initiieren

kann. Die verwaltete Anmeldung erledigt dies automatisch. Auf der Anmeldeseite werden Sie zur Eingabe eines Benutzernamens aufgefordert, es wird überprüft, ob der Benutzer mindestens ein Hauptschlüssel registriert hat, und Sie werden dann zur Anmeldung mit dem Hauptschlüssel aufgefordert. In ähnlicher Weise müssen SDK-basierte Anwendungen zur Eingabe eines Benutzernamens auffordern und diesen in der Authentifizierungsanfrage angeben.

Wenn Sie die Benutzerpoolauthentifizierung mit Hauptschlüsseln einrichten und über eine benutzerdefinierte Domäne und eine Präfixdomäne verfügen, verwendet die RP-ID standardmäßig den vollqualifizierten Domänennamen (FQDN) Ihrer benutzerdefinierten Domäne. Um eine Präfix-Domain als RP-ID in der Amazon Cognito Cognito-Konsole festzulegen, löschen Sie Ihre benutzerdefinierte Domain oder geben Sie den FQDN der Präfix-Domain als Drittanbieter-Domain ein.

Activate passkey sign-in

Konsole

Um die Anmeldung mit Hauptschlüsseln zu aktivieren, konfigurieren Sie Ihren Benutzerpool so, dass die primäre Anmeldung mit einem oder mehreren kennwortlosen Typen zulässig ist, und konfigurieren Sie dann Ihren App-Client so, dass er den Datenfluss zulässt. `USER_AUTH` Navigieren Sie in der Amazon Cognito Cognito-Konsole in Ihrer Benutzerpool-Konfiguration unter Authentifizierung zum Anmelde-Menü. Bearbeiten Sie die Optionen für die wahlbasierte Anmeldung und fügen Sie Passkey zur Liste der verfügbaren Optionen hinzu.

Navigieren Sie zum Menü Authentifizierungsmethoden und bearbeiten Sie Passkey.

- Die Benutzerüberprüfung ist die Einstellung, ob Ihr Benutzerpool Hauptschlüsselgeräte benötigt, die zusätzlich überprüfen, ob der aktuelle Benutzer für einen Hauptschlüssel autorisiert ist. Um Benutzer zu ermutigen, ein Gerät mit Benutzerverifizierung zu konfigurieren, diese aber nicht zu verlangen, wählen Sie Bevorzugt. Um nur Geräte mit Benutzerverifizierung zu unterstützen, wählen Sie Erforderlich aus. Weitere Informationen finden Sie unter Benutzerverifizierung auf w3.org.
- Die ID der Domain for Relying Party ist die Kennung, die Ihre Anwendung bei der Registrierung mit dem Hauptschlüssel der Benutzer weitergibt. Sie legt das Ziel der Vertrauensbeziehung mit dem Herausgeber der Hauptschlüssel der Benutzer fest. Ihre ID der vertrauenden Partei kann sein: die Domain Ihres Benutzerpools, wenn

  Cognito-Domäne

  Die Amazon Cognito-Präfixdomäne Ihres Benutzerpools.

Benutzerdefinierte Domain

Die benutzerdefinierte Domain Ihres Benutzerpools.

Domain eines Drittanbieters

Die Domäne für Anwendungen, die den Benutzerpool nicht verwenden, verwaltete Anmeldeseiten. Diese Einstellung wird normalerweise Benutzerpools zugeordnet, die keine Domäne haben und die Authentifizierung mit einem AWS SDK und der Benutzerpools-API im Backend durchführen.

Navigieren Sie zum Menü App-Clients und wählen Sie einen App-Client aus oder erstellen Sie einen neuen. Wählen Sie Bearbeiten aus und wählen Sie unter Authentifizierungsabläufe die Option Wählen Sie bei der Anmeldung einen Authentifizierungstyp aus: ALLOW_USER_AUTH.

API/SDK

Konfigurieren Sie in der Benutzerpools-API `SignInPolicy` mit den entsprechenden Passkey-Optionen in einer CreateUserPoolOder-Anfrage. UpdateUserPool Die `WEB_AUTHN` Option für die Hauptschlüsselauthentifizierung muss von mindestens einer weiteren Option begleitet werden. Für die Registrierung mit dem Passkey ist eine bestehende Authentifizierungssitzung erforderlich.

```
"SignInPolicy": {
    "AllowedFirstAuthFactors": [
        "PASSWORD",
        "WEB_AUTHN"
    ]
}
```

Konfigurieren Sie Ihre Präferenz für die Benutzerverifizierung und Ihre RP-ID im `WebAuthnConfiguration` Parameter einer SetUserPoolMfaConfigAnfrage. Das`RelyingPartyId`, das beabsichtigte Ziel der Ergebnisse der Passkey-Authentifizierung, kann Ihr Benutzerpool-Präfix oder Ihre benutzerdefinierte Domain oder eine Domain Ihrer Wahl sein.

```
"WebAuthnConfiguration": {
   "RelyingPartyId": "example.auth.us-east-1.amazoncognito.com",
   "UserVerification": "preferred"
}
```

Konfigurieren Sie Ihren App-Client `ExplicitAuthFlows` mit der erforderlichen Option in einer CreateUserPoolClientUpdateUserPoolClientOder-Anfrage.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_AUTH"
]
```

Register a passkey (managed login)

Die verwaltete Anmeldung kümmert sich um die Benutzerregistrierung von Hauptschlüsseln. Wenn die Hauptschlüsselauthentifizierung in Ihrem Benutzerpool aktiv ist, fordert Amazon Cognito Benutzer auf, einen Hauptschlüssel einzurichten, wenn sie sich für ein neues Benutzerkonto registrieren.

Amazon Cognito fordert Benutzer nicht auf, einen Hauptschlüssel einzurichten, wenn sie sich bereits registriert haben und keinen Hauptschlüssel eingerichtet haben oder wenn Sie ihr Konto als Administrator erstellt haben. Benutzer in diesem Status müssen sich mit einem anderen Faktor wie einem Passwort oder einem kennwortlosen OTP anmelden, bevor sie einen Hauptschlüssel registrieren können.

Um einen Hauptschlüssel zu registrieren

1. Leiten Sie den Benutzer zu Ihrer Anmeldeseite weiter.

   ```
   https://auth.example.com/oauth2/authorize/?
   client_id=1example23456789&response_type=code&scope=email+openid
   +phone&redirect_uri=https%3A%2F%2Fwww.example.com
   ```

2. Verarbeiten Sie das Authentifizierungsergebnis des Benutzers. In diesem Beispiel leitet Amazon Cognito sie `www.example.com` mit einem Autorisierungscode weiter, den Ihre Anwendung gegen Token eintauscht.
3. Leiten Sie den Benutzer zu Ihrer Registrierungs-Passkey-Seite weiter. Der Benutzer erhält ein Browser-Cookie, das seine angemeldete Sitzung beibehält. Die Hauptschlüssel-URL verwendet und Parameter. `client_id` `redirect_uri` Amazon Cognito erlaubt nur authentifizierten Benutzern den Zugriff auf diese Seite. Melden Sie Ihren Benutzer mit einem Passwort, E-Mail-OTP oder SMS-OTP an und rufen Sie dann eine URL auf, die dem folgenden Muster entspricht.

   Sie können dieser Anfrage auch andere Autorisieren des Endpunkts Parameter wie und hinzufügen. `response_type` `scope`

```
https://auth.example.com/passkeys/add?
client_id=1example23456789&redirect_uri=https%3A%2F%2Fwww.example.com
```

Register a passkey (SDK)

Sie registrieren Hauptschlüsselanmeldedaten mit Metadaten in einem PublicKeyCreationOptionsObjekt. Sie können dieses Objekt mit den Anmeldeinformationen eines angemeldeten Benutzers generieren und diese in einer API-Anfrage an dessen Hauptschlüsselaussteller weitergeben. Der Aussteller gibt ein RegistrationResponseJSON-Objekt zurück, das die Registrierung des Hauptschlüssels bestätigt.

Um den Prozess der Registrierung des Hauptschlüssels zu starten, melden Sie einen Benutzer mit einer vorhandenen Anmeldeoption an. Autorisieren Sie die durch das Token autorisierte StartWebAuthnRegistrationAPI-Anfrage mit dem Zugriffstoken des aktuellen Benutzers. Im Folgenden finden Sie den Hauptteil einer Beispielanforderung. `GetWebAuthnRegistrationOptions`

```
{
   "AccessToken": "eyJra456defEXAMPLE"
}
```

Die Antwort aus Ihrem Benutzerpool enthält das `PublicKeyCreationOptions` Objekt. Präsentieren Sie dieses Objekt in einer API-Anfrage an den Aussteller des Benutzers. Es enthält Informationen wie den öffentlichen Schlüssel und die ID der vertrauenden Partei. Der Emittent wird mit einem `RegistrationResponseJSON` Objekt antworten.

Präsentieren Sie die Antwort auf die Registrierung in einer CompleteWebAuthnRegistrationAPI-Anfrage, die wiederum mit dem Zugriffstoken des Benutzers autorisiert wurde. Wenn Ihr Benutzerpool mit einer HTTP 200-Antwort mit einem leeren Text antwortet, ist der Hauptschlüssel Ihres Benutzers registriert.

Sign in with a passkey

Bei der passwortlosen Anmeldung gibt es keine Option`AuthFlow`, die Sie in und angeben können. InitiateAuthAdminInitiateAuth Stattdessen müssen Sie ein `AuthFlow` of deklarieren `USER_AUTH` und eine Anmeldeoption anfordern oder Ihre passwortlose Option aus der Antwort Ihres Benutzerpools auswählen. Um einen Benutzer bei einer Anwendung anzumelden, konfigurieren Sie den Hauptteil Ihrer `AdminInitiateAuth` Oder-Anfrage wie `InitiateAuth`

folgt. Dieser Parametersatz ist das Minimum, das für die Anmeldung erforderlich ist. Zusätzliche Parameter sind verfügbar.

In diesem Beispiel wissen wir, dass sich der Benutzer mit einem Hauptschlüssel anmelden möchte, und fügen einen `PREFERRED_CHALLENGE` Parameter hinzu.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PREFERRED_CHALLENGE" : "WEB_AUTHN"
   },
   "ClientId": "1example23456789"
}
```

Amazon Cognito reagiert mit einer `WEB_AUTHN`-Aufgabe. Ihre Bewerbung muss auf diese Herausforderung reagieren. Initiieren Sie eine Anmeldeanforderung beim Passkey-Anbieter des Benutzers. Es wird ein AuthenticationResponseJSON-Objekt zurückgegeben.

```
{
   "ChallengeName": "WEB_AUTHN",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "CREDENTIAL" : "{AuthenticationResponseJSON}"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

MFA nach der Anmeldung

Sie können festlegen, dass Benutzer, die die Anmeldung mit einem Benutzername-Passwort-Schema abschließen, in einer E-Mail-Nachricht, SMS-Nachricht oder einer Anwendung zur Codegenerierung zu einer zusätzlichen Bestätigung mit einem Einmalkennwort aufgefordert werden. MFA unterscheidet sich von der passwortlosen Anmeldung, einem ersten Authentifizierungsfaktor mit WebAuthn Einmalpasswörtern oder Kennwörtern, der MFA nicht beinhaltet. MFA in Benutzerpools ist ein Challenge-Response-Modell, bei dem ein Benutzer zunächst nachweist, dass er das Passwort kennt, und dann nachweist, dass er Zugriff auf sein registriertes Second-Factor-Gerät hat.

Ressourcen für die Implementierung

- Hinzufügen der MFA zu einem Benutzerpool

Tokens aktualisieren

Wenn Sie möchten, dass Benutzer angemeldet bleiben, ohne ihre Anmeldeinformationen erneut eingeben zu müssen, sind Aktualisierungstoken das Tool, das Ihre Anwendung benötigt, um die Sitzung eines Benutzers beizubehalten. Anwendungen können Ihrem Benutzerpool Aktualisierungstoken präsentieren und diese gegen neue ID- und Zugriffstoken eintauschen. Mit der Tokenaktualisierung können Sie sicherstellen, dass ein angemeldeter Benutzer weiterhin aktiv ist, aktualisierte Attributinformationen abrufen und Zugriffsberechtigungen ohne Benutzereingriff aktualisieren.

Ressourcen für die Implementierung

- Tokens aktualisieren

Benutzerdefinierte Authentifizierung

Möglicherweise möchten Sie eine Authentifizierungsmethode für Ihre Benutzer konfigurieren, die hier nicht aufgeführt ist. Sie können dies mit einer benutzerdefinierten Authentifizierung mit Lambda-Triggern tun. In einer Abfolge von Lambda-Funktionen gibt Amazon Cognito eine Aufforderung aus, stellt eine Frage, die Benutzer beantworten müssen, überprüft die Antwort auf ihre Richtigkeit und bestimmt dann, ob eine weitere Aufforderung ausgegeben werden soll. Die Fragen und Antworten können Sicherheitsfragen, Anfragen an einen CAPTCHA-Dienst, Anfragen an eine externe MFA-Dienst-API oder alle diese nacheinander beinhalten.

Ressourcen für die Implementierung

- Lambda-Auslöser für benutzerdefinierte Authentifizierungsaufforderungen

Benutzerdefinierter Authentifizierungsablauf

Amazon-Cognito-Benutzerpools ermöglichen außerdem die Verwendung benutzerdefinierter Authentifizierungsabläufe, mit denen Sie ein auf Aufforderungen und Antworten basiertes Authentifizierungsmodell mithilfe von AWS Lambda -Auslösern erstellen können.

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`.

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 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 mit einem Lambda-Auslöser für die Benutzermigration.

Autorisierungsmodelle für die API- und SDK-Authentifizierung

Wenn Sie mit der Entwicklung Ihrer Anwendung mit Benutzerpool-Authentifizierung beginnen, müssen Sie sich für das API-Autorisierungsmodell entscheiden, das zu Ihrem Anwendungstyp passt. Ein Autorisierungsmodell ist ein System zur Autorisierung für Anfragen mit den Authentifizierungskomponenten in den API- und SDK-Integrationen der Amazon Cognito Cognito-Benutzerpools. Amazon Cognito hat drei Autorisierungsmodelle: IAM-autorisiert, öffentlich und tokenautorisiert.

Bei IAM-autorisierten Anfragen erfolgt die Autorisierung anhand einer Signatur anhand einer Reihe von AWS IAM-Anmeldeinformationen im Header einer Anfrage. `Authorization` Bei serverseitigen Anwendungen schützt diese Vorgehensweise Authentifizierungsvorgänge mit IAM-Autorisierung. Bei öffentlichen (nicht authentifizierten) Authentifizierungsanfragen ist keine Autorisierung erforderlich. Dies eignet sich für clientseitige Anwendungen, die an Benutzer verteilt

werden. Bei tokenautorisierten Vorgängen, die in der Regel in Kombination mit öffentlichen Vorgängen implementiert werden, erfolgt die Autorisierung über ein Sitzungstoken oder ein Zugriffstoken, das im Header der `Authorization` Anfrage enthalten ist. Die Amazon Cognito Cognito-Authentifizierung erfordert in der Regel, dass Sie zwei oder mehr API-Operationen nacheinander implementieren. Die von Ihnen verwendeten API-Operationen hängen von den Eigenschaften Ihrer Anwendung ab. Öffentliche Clients, bei denen die Anwendung an Benutzer verteilt wird, verwenden öffentliche Operationen, bei denen Anmeldeanfragen keine Autorisierung erfordern. Mit Tokens autorisierte Operationen setzen die Benutzersitzungen in öffentlichen Anwendungen fort. Serverseitige Clients, bei denen die Anwendungslogik auf einem Remotesystem gehostet wird, schützen Authentifizierungsvorgänge mit IAM-Autorisierung für Anmeldeanfragen. Die folgenden API-Operationspaare und ihre entsprechenden SDK-Methoden sind den verfügbaren Autorisierungsmodellen zugeordnet.

Jeder öffentliche Authentifizierungsvorgang hat irgendein serverseitiges Äquivalent, zum Beispiel UpdateUserAttributesund AdminUpdateUserAttributes. Während clientseitige Operationen vom Benutzer initiiert werden und eine Bestätigung erfordern, gehen serverseitige Operationen davon aus, dass die Änderung von einem Benutzerpooladministrator festgeschrieben wurde und die Änderungen sofort wirksam werden. In diesem Beispiel sendet Amazon Cognito eine Nachricht mit einem Bestätigungscode an den Benutzer, und das Zugriffstoken des Benutzers autorisiert eine VerifyUserAttributeAnfrage, die den Code sendet. Die serverseitige Anwendung kann sofort den Wert eines beliebigen Attributs festlegen. Allerdings gelten bei der Änderung des Werts von E-Mail-Adressen und Telefonnummern, wenn diese für die Anmeldung verwendet werden, besondere Überlegungen.

Einen Vergleich der API-Authentifizierung und eine vollständige Liste der API-Operationen und ihrer Autorisierungsmodelle finden Sie unter. Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten

Client-side (public) authentication

Im Folgenden finden Sie eine typische Abfolge von Anfragen in einer clientseitigen Anwendung

1. Bei der öffentlichen InitiateAuthOperation werden primäre Anmeldeinformationen wie ein Benutzername und ein Passwort übermittelt.
2. Der tokenautorisierte RespondToAuthChallengeVorgang sendet ein Sitzungstoken aus der `InitiateAuth` Antwort und der Antwort auf eine Herausforderung, beispielsweise MFA. Die Autorisierung mit Sitzungstoken weist auf Anfragen hin, die Teil von Authentifizierungszyklen sind. not-yet-complete

3. Der durch das Token autorisierte ConfirmDeviceVorgang sendet ein Zugriffstoken und führt den Schreibvorgang durch, bei dem ein Gerät, an das man sich erinnert, zum Benutzerprofil hinzugefügt wird. Die Autorisierung mit Zugriffstoken weist auf Anfragen hin, die sich auf Self-Service-Operationen von Benutzern beziehen, nachdem diese die Authentifizierung abgeschlossen haben.

Weitere Informationen erhalten Sie unter Optionen für die clientseitige Authentifizierung und Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten.

Server-side authentication

Im Folgenden finden Sie eine typische Abfolge von Anfragen aus einem serverseitigen Vorgang. Jede Anfrage hat einen AWS Signature Version 4-Autorisierungsheader, der mit IAM-Computeranmeldeinformationen signiert ist, die an den Anwendungsserver ausgegeben wurden.

1. Der AdminInitiateAuthVorgang übermittelt primäre Anmeldeinformationen wie einen Benutzernamen und ein Passwort.
2. AdminRespondToAuthChallengeOperation reicht die Antwort auf eine Herausforderung ein, zum Beispiel MFA.
3. Bei diesem AdminUpdateDeviceStatusVorgang wird der Geräteschlüssel aus der `AdminInitiateAuth` Antwort auf den gespeicherten Wert gesetzt.

Weitere Informationen erhalten Sie unter Serverseitige Authentifizierungsoptionen und Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten.

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.

Themen

- Serverseitige Authentifizierungsoptionen
- Optionen für die clientseitige Authentifizierung
- Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten
- Liste der API-Operationen, gruppiert nach Autorisierungsmodell

Serverseitige Authentifizierungsoptionen

Webanwendungen und andere serverseitige Anwendungen implementieren die Authentifizierung auf einem Remoteserver, den ein Client in eine Remote-Display-Anwendung wie einen Browser oder eine SSH-Sitzung lädt. Serverseitige Anwendungen weisen in der Regel die folgenden Merkmale auf.

- Sie sind in einer Anwendung integriert, die auf einem Server in Sprachen wie Java, Ruby oder Node.js installiert ist.
- Sie stellen eine Verbindung zu Benutzerpool-App-Clients her, die möglicherweise über ein geheimes Client-Geheimnis verfügen, die als vertrauliche Clients bezeichnet werden.
- Sie haben Zugriff auf AWS Anmeldeinformationen.
- Sie rufen die verwaltete Anmeldung zur Authentifizierung auf oder verwenden IAM-autorisierte Operationen in der Benutzerpools-API mit einem SDK. AWS
- Sie bedienen interne Kunden und können öffentliche Kunden bedienen.

Bei serverseitigen Vorgängen mit der Benutzerpools-API können Passwörter, Einmalkennwörter oder Hauptschlüssel als primärer Anmeldefaktor verwendet werden. Für serverseitige Apps ist die Benutzerpoolauthentifizierung mit der Authentifizierung für clientseitige Apps vergleichbar, mit Ausnahme von Folgendem:

- Die serverseitige App stellt eine API-Anfrage. AdminInitiateAuth Für diesen Vorgang sind AWS Anmeldeinformationen mit Berechtigungen wie `cognito-idp:AdminInitiateAuth` und `cognito-idp:AdminRespondToAuthChallenge` erforderlich. Der Vorgang gibt die erforderliche Abfrage oder das erforderliche Authentifizierungsergebnis zurück.
- Wenn die Anwendung eine Anfrage erhält, stellt sie eine AdminRespondToAuthChallengeAPI-Anfrage. Für den `AdminRespondToAuthChallenge` API-Vorgang sind auch AWS Anmeldeinformationen erforderlich.

Weitere Informationen zum Signieren von Amazon Cognito Cognito-API-Anfragen mit AWS Anmeldeinformationen finden Sie unter Signaturprozess für Signature Version 4 in der AWS Allgemeinen Referenz.

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 Backend-Admin-Implementierungen den Admin-Authentifizierungsablauf verwenden, unterstützt der Flow keine gespeicherten Geräte. Wenn Sie die Geräteverfolgung aktiviert haben, ist die Administratorauthentifizierung erfolgreich, aber jeder Aufruf zum Aktualisieren des Zugriffstokens schlägt fehl.

Optionen für die clientseitige Authentifizierung

Mobile Apps und andere clientseitige Anwendungstypen werden auf den Geräten der Benutzer installiert und führen die Logik der Authentifizierung und der Benutzeroberfläche lokal aus. Sie weisen in der Regel die folgenden Eigenschaften auf.

- Sie sind in Sprachen wie React Native, Flutter und Swift erstellt und auf Benutzergeräten bereitgestellt.
- Sie stellen eine Verbindung zu App-Clients für Benutzerpools her, die kein geheimes Client-Geheimnis haben, die als öffentliche Clients bezeichnet werden.
- Sie haben keinen Zugriff auf AWS Anmeldeinformationen, die IAM-autorisierte API-Anfragen autorisieren würden.
- Sie rufen die verwaltete Anmeldung zur Authentifizierung auf oder verwenden öffentliche und tokenautorisierte Operationen in der Benutzerpools-API mit einem SDK. AWS
- Sie bedienen öffentliche Kunden und ermöglichen es jedem, sich zu registrieren und anzumelden.

Bei clientseitigen Vorgängen mit der Benutzerpools-API können Passwörter, Einmalpasswörter oder Hauptschlüssel als primärer Anmeldefaktor verwendet werden. Der folgende Vorgang funktioniert für clientseitige Benutzer-Apps, die Sie mit oder dem erstellen. AWS AmplifyAWS SDKs

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 integrierten Amazon Cognito SRP-Funktionen. AWS SDKs

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).

Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten

Amazon Cognito Cognito-Benutzerpools sind eine Kombination aus mehreren Authentifizierungstechnologien. Sie verlassen sich auf externe Identitätsanbieter (IdPs). Sie sind IdPs für Anwendungen vorgesehen, die die Authentifizierung mit OpenID Connect (OIDC) implementieren. SDKs Sie ermöglichen die Authentifizierung als Aussteller von JSON-Web-Tokens (JWTs), ähnlich der OIDC-Authentifizierung, jedoch in API-Methoden, die Teil von sind. AWS SDKs Sie können auch sichere Zugangspunkte für Ihre Anwendungen sein.

Für die Registrierung, Anmeldung und Verwaltung von Benutzern in Ihrem Benutzerpool haben Sie zwei Möglichkeiten.

1. Ihre verwalteten Anmeldeseiten und die klassische gehostete Benutzeroberfläche umfassen die benutzerinteraktiven Endpunkte für die verwaltete Anmeldung und die Verbundendpunkte, die IdP- und Rely-Party-Rollen verwalten. Sie bilden ein Paket öffentlicher Webseiten, das Amazon Cognito aktiviert, wenn Sie eine Domain für Ihren Benutzerpool auswählen. Für einen schnellen Einstieg in die Authentifizierungs- und Autorisierungsfunktionen der Amazon Cognito Cognito-Benutzerpools, einschließlich Seiten für Registrierung, Anmeldung, Passwortverwaltung und Multi-Faktor-Authentifizierung (MFA), verwenden Sie die integrierte Benutzeroberfläche von Managed Login.

Die anderen Endpunkte des Benutzerpools erleichtern die Authentifizierung bei externen Identitätsanbietern (). IdPs Die von ihnen erbrachten Dienstleistungen beinhalten Folgendes.

a. Callback-Endpunkte von Dienstanbietern für authentifizierte Ansprüche von Ihnen, wie und. IdPs `saml2/idpresponse` `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 `/.well-known/jwks.json`. Ihre App verwendet diese Endpunkte, wenn sie Token verifiziert oder Benutzerprofildaten mit OIDC- oder 2.0-Entwicklerbibliotheken abruft. OAuth

2. Die Amazon Cognito Cognito-Benutzerpools-API ist eine Reihe von Tools für Ihre Web- oder Mobil-App, mit der Sie Benutzer authentifizieren können, nachdem sie Anmeldeinformationen in Ihrem eigenen benutzerdefinierten Frontend gesammelt hat. 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 Rechnung um zusätzliche Kosten. AWS

   c. Ein Aktualisierungstoken, das Anfragen nach neuen IDs und Zugriffstoken autorisiert und die Eigenschaften der Benutzeridentität und der Zugriffskontrolle aktualisiert.

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 ihre föderierte Identität in einer AdminLinkProviderForUserAPI-Anfrage mit einem lokalen Benutzer verknüpfen, kann sich dieser mit der Benutzerpools-API anmelden.

Weitere Informationen finden Sie unter Verknüpfen von Verbundbenutzern mit einem vorhandenen Benutzerprofil.

Die Amazon-Cognito-Benutzerpool-API hat zwei Verwendungszwecke.

1. Sie erstellt und konfiguriert die Ressourcen für Ihre Amazon-Cognito-Benutzerpools. Sie können beispielsweise Benutzerpools erstellen, AWS Lambda Auslöser hinzufügen und die Benutzerpool-Domain konfigurieren, die Ihre verwalteten Anmeldeseiten hostet.
2. Es führt Registrierungs-, Anmelde- und andere Benutzervorgänge 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 SignUpAPI-Anfrage 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 ConfirmSignUpAPI-Anfrage 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 InitiateAuthAPI-Anfrage 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.

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 eine verwaltete Anmeldung beinhalten, finden Sie unterBenutzerpool-Endpunkte und Referenz für verwaltete Logins.

Ihre Verbundbenutzer können mit der verwalteten Anmeldung beginnen und ihren IdP auswählen, oder Sie können die verwaltete Anmeldung überspringen und Ihre Benutzer zur Anmeldung direkt an Ihren IdP senden. 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 verwalteten Anmeldeseiten

1. Ihr Benutzer wählt die Schaltfläche zum Erstellen eines Kontos aus, die Sie in Ihrer App erstellt haben.
2. Bei der verwalteten Anmeldung wird Ihrem Benutzer eine Liste der Anbieter für soziale Identitäten angezeigt, für die Sie Entwickleranmeldedaten 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. Der Browser Ihres Benutzers öffnet die Apple-Authentifizierungsseite. Ihr Benutzer meldet sich an und autorisiert Amazon Cognito, seine Profilinformationen zu lesen.
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. Die OIDC-Bibliothek in Ihrer Anwendung tauscht den Autorisierungscode mit dem aus Token-Endpunkt und speichert ein vom Benutzerpool ausgestelltes ID-Token, Zugriffstoken und Aktualisierungstoken. Ihre App verwendet OIDC-Bibliotheken, um die Token Ihrer Benutzer zu verwalten und eine dauerhafte Sitzung für diesen Benutzer aufrechtzuerhalten.

Die Benutzerpools-API und die verwalteten Anmeldeseiten 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.

Liste der API-Operationen, gruppiert nach Autorisierungsmodell

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 Von IAM autorisierte Verwaltungsvorgänge und Von IAM autorisierte Benutzeroperationen zuweisen.

Von IAM autorisierte Verwaltungsvorgänge

Von IAM autorisierte Verwaltungsvorgänge ändern und zeigen Ihren Benutzerpool und die App-Client-Konfiguration an, wie Sie es in der tun würden. AWS Management Console

Um beispielsweise Ihren Benutzerpool in einer UpdateUserPoolAPI-Anfrage zu ändern, müssen Sie AWS Anmeldeinformationen und IAM-Berechtigungen angeben, um die Ressource zu aktualisieren.

Um diese Anfragen im AWS Command Line Interface (AWS CLI) oder einem AWS SDK zu autorisieren, konfigurieren Sie Ihre Umgebung mit Umgebungsvariablen oder einer Client-Konfiguration, die Ihrer Anfrage IAM-Anmeldeinformationen hinzufügt. Weitere Informationen finden Sie unter Zugreifen 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 Anfragen mit AWS Anmeldeinformationen autorisieren oder signieren, die Sie in die Kopfzeile Ihrer Anfrage einbetten. Weitere Informationen finden Sie unter AWS API-Anfragen signieren.

Von IAM autorisierte Verwaltungsvorgänge

AddCustomAttributes

CreateGroup

CreateIdentityProvider

CreateResourceServer

CreateUserImportJob

CreateUserPool

CreateUserPoolClient

CreateUserPoolDomain

DeleteGroup

DeleteIdentityProvider

Von IAM autorisierte Verwaltungsvorgänge

DeleteResourceServer

DeleteUserPool

DeleteUserPoolClient

DeleteUserPoolDomain

DescribeIdentityProvider

DescribeResourceServer

DescribeRiskConfiguration

DescribeUserImportJob

DescribeUserPool

DescribeUserPoolClient

DescribeUserPoolDomain

Holen CSVHeader

GetGroup

GetIdentityProviderByIdentifier

GetSigningCertificate

Bekommen UICustomization

GetUserPoolMfaConfig

ListGroups

ListIdentityProviders

ListResourceServers

ListTagsForResource

Von IAM autorisierte Verwaltungsvorgänge

ListUserImportJobs

ListUserPoolClients

ListUserPools

ListUsers

ListUsersInGroup

SetRiskConfiguration

Einstellen UICustomization

SetUserPoolMfaConfig

StartUserImportJob

StopUserImportJob

TagResource

UntagResource

UpdateGroup

UpdateIdentityProvider

UpdateResourceServer

UpdateUserPool

UpdateUserPoolClient

UpdateUserPoolDomain

Von IAM autorisierte Benutzeroperationen

IAM-autorisierte Benutzeroperationen: Registrierung, Anmeldung, Verwaltung der Anmeldeinformationen, Änderung und Anzeige Ihrer Benutzer.

Sie können beispielsweise eine serverseitige Anwendungsebene haben, die ein Web-Frontend unterstützt. Ihre serverseitige App ist ein OAuth vertraulicher Client, dem Sie privilegierten Zugriff auf Ihre Amazon Cognito Cognito-Ressourcen anvertrauen. Um einen Benutzer in der App zu registrieren, kann Ihr Server AWS Anmeldeinformationen in eine AdminCreateUserAPI-Anfrage aufnehmen. Weitere Informationen zu OAuth Clienttypen finden Sie unter Clienttypen in The OAuth 2.0 Authorization Framework.

Um diese Anfragen im AWS CLI oder einem AWS SDK zu autorisieren, konfigurieren Sie Ihre serverseitige App-Umgebung mit Umgebungsvariablen oder einer Client-Konfiguration, die Ihrer Anfrage IAM-Anmeldeinformationen hinzufügt. Weitere Informationen finden Sie unter Zugreifen 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 Anfragen mit AWS Anmeldeinformationen autorisieren oder signieren, die Sie in die Kopfzeile Ihrer Anfrage einbetten. Weitere Informationen finden Sie unter AWS API-Anfragen signieren.

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-autorisierte Benutzeroperationen

AdminAddUserToGroup

AdminConfirmSignUp

AdminCreateUser

AdminDeleteUser

AdminDeleteUserAttributes

AdminDisableProviderForUser

AdminDisableUser

AdminEnableUser

AdminForgetDevice

IAM-autorisierte Benutzeroperationen

AdminGetDevice

AdminGetUser

AdminInitiateAuth

AdminLinkProviderForUser

AdminListDevices

AdminListGroupsForUser

AdminListUserAuthEvents

AdminRemoveUserFromGroup

AdminResetUserPassword

AdminRespondToAuthChallenge

AdminSetUserMFAPreference

AdminSetUserPassword

AdminSetUserSettings

AdminUpdateAuthEventFeedback

AdminUpdateDeviceStatus

AdminUpdateUserAttributes

AdminUserGlobalSignOut

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 OAuth öffentlichen Client bereitstellen, der keinen privilegierten Zugriff auf geheime Daten gewährt. Sie können diesen Benutzer mit der API-Operation ohne Authentifizierung registrieren. SignUp

Um diese Anfragen 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. Um dieses Update durchzuführen, kann Ihr Client das Zugriffstoken des Benutzers in eine UpdateUserAttributesAPI-Anfrage aufnehmen.

Um diese Anfragen 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	✓	
StartWebAuthnRegistration	✓	
CompleteWebAuthnRegistration	✓	
DeleteWebAuthnCredential	✓	
ListWebAuthnCredentials	✓	
UpdateUserAttributes	✓	

Über Token autorisierte Benutzeroperationen	AccessToken	Sitzung
DeleteUserAttributes	✓	
DeleteUser	✓	
ConfirmDevice	✓	
ForgetDevice	✓	
GetDevice	✓	
ListDevices	✓	
UpdateDeviceStatus	✓	
GetUserAttributeVerificationCode	✓	
VerifyUserAttribute	✓	
SetUserSettings	✓	
SetUserMFAPreference	✓	
GlobalSignOut	✓	
UpdateAuthEventFeedback		✓
AssociateSoftwareToken	✓	✓
VerifySoftwareToken	✓	✓
RevokeToken[1]		

Über Token autorisierte Benutzeroperationen	AccessToken	Sitzung
GetTokensFromRefreshToken[1]		

[1] `RevokeToken` und verwenden `GetTokensFromRefreshToken` Sie Aktualisierungstoken als Autorisierungsparameter. Das Aktualisierungstoken dient als Autorisierungstoken und als Zielressource.

Benutzerpool-Anmeldung mit externen Identitätsanbietern

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 JSON-Webtoken (JWTs) für den Benutzerpool verstehen.

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 Seiten für die verwaltete Anmeldung eingerichtet. Weitere Informationen finden Sie unter Verwenden Sie Ihre eigene Domain für die verwaltete Anmeldung.
- Sie können Verbundbenutzer nicht mit API-Operationen wie InitiateAuthund AdminInitiateAuthanmelden. Verbundbenutzer können sich nur mit dem Login-Endpunkt oder dem Autorisieren des Endpunkts anmelden.
- Autorisieren des Endpunkts ist ein Umleitungsendpunkt. Wenn Sie in Ihrer Anfrage einen `identity_provider` Parameter `idp_identifier` oder angeben, leitet er unbemerkt an Ihren IdP weiter und umgeht so die verwaltete Anmeldung. Andernfalls wird es zur verwalteten Anmeldung weitergeleitet. Login-Endpunkt
- Wenn die verwaltete Anmeldung eine Sitzung an einen föderierten IdP weiterleitet, nimmt Amazon Cognito den `user-agent` Header `Amazon/Cognito` in die Anfrage auf.
- 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 erstellt eine Benutzergruppe für jeden OIDC und jeden sozialen IdP SAMl, den Sie Ihrem Benutzerpool hinzufügen. Der Name der Gruppe hat das Format`[user pool ID]_[IdP name]`, zum Beispiel oder. `us-east-1_EXAMPLE_MYSSO` `us-east-1_EXAMPLE_Google` Jedes eindeutige, automatisch generierte IdP-Benutzerprofil wird automatisch zu dieser Gruppe hinzugefügt. Verknüpfte Benutzer werden dieser Gruppe nicht automatisch hinzugefügt, aber Sie können ihre Profile der Gruppe in einem separaten Prozess hinzufügen.
- 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 UpdateIdentityProviderAPI-Anfrage aktualisieren, kann es bis zu einer Minute dauern, bis Ihre Änderungen in der verwalteten Anmeldung angezeigt werden.
- Amazon Cognito unterstützt bis zu 20 HTTP-Umleitungen zwischen Amazon Cognito und Ihrem IDP.
- Wenn sich Ihr Benutzer mit der verwalteten Anmeldung anmeldet, speichert sein Browser ein verschlüsseltes Anmeldesitzungscookie, das den Client und den Anbieter aufzeichnet, mit dem er sich angemeldet hat. Wenn sie erneut versuchen, sich mit denselben Parametern anzumelden, verwendet die verwaltete Anmeldung jede noch nicht abgelaufene bestehende Sitzung wieder, 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.
- Verwenden von Anbietern für soziale Identitäten mit einem Benutzerpool
- Verwendung von SAML-Identitätsanbietern mit einem Benutzerpool
- Verwendung von OIDC-Identitätsanbietern mit einem Benutzerpool
- Zuordnung von IdP-Attributen zu Profilen und Tokens
- Verknüpfen von Verbundbenutzern mit einem vorhandenen Benutzerprofil

Konfigurieren von Identitätsanbietern für Ihren Benutzerpool.

Mit Benutzerpools können Sie die Anmeldung über eine Vielzahl von externen Identitätsanbietern () IdPs implementieren. Dieser Abschnitt des Handbuchs enthält Anweisungen zur Einrichtung dieser Identitätsanbieter mit Ihrem Benutzerpool in der Amazon Cognito Cognito-Konsole. Alternativ können Sie die Benutzerpools-API und ein AWS SDK verwenden, um Benutzerpool-Identitätsanbieter programmgesteuert hinzuzufügen. Weitere Informationen finden Sie unter CreateIdentityProvider.

Zu den unterstützten Identitätsanbieter-Optionen gehören soziale Anbieter wie Facebook, Google und Amazon sowie OpenID Connect (OIDC) und SAML 2.0-Anbieter. Bevor Sie beginnen, richten Sie sich Administratoranmeldedaten für Ihren IdP ein. Für jeden Anbietertyp müssen Sie Ihre Anwendung registrieren, die erforderlichen Anmeldeinformationen abrufen und dann die Anbieterdetails in Ihrem Benutzerpool konfigurieren. Ihre Benutzer können sich dann mit ihren bestehenden Konten bei den verbundenen Identitätsanbietern bei Ihrer Anwendung registrieren und anmelden.

Das Menü Soziale und externe Anbieter unter Authentifizierung fügt den Benutzerpool hinzu und aktualisiert ihn IdPs. Weitere Informationen finden Sie unter Benutzerpool-Anmeldung mit externen Identitätsanbietern.

Themen

- Einrichten der Benutzeranmeldung mit einem sozialen IdP
- Einrichten der Benutzeranmeldung mit einem OIDC IdP
- Einrichten der Benutzeranmeldung mit einem SAML IdP

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. 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 das Menü Soziale Netzwerke und externe Anbieter.
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 Änderungen speichern aus.
8. Wählen Sie im Menü App-Clients einen App-Client aus der Liste aus und wählen Sie dann Bearbeiten aus. Fügen Sie unter Identity providers (Identitätsanbieter) den neuen Social-Identity-Anbieter zum App-Client hinzu.
9. Wählen Sie Änderungen speichern aus.

Weitere Informationen zu sozialen IdPs Netzwerken finden Sie unterVerwenden 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. 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 das Menü Soziale Netzwerke und externe Anbieter und wählen Sie 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 gemäß der OAuth 2.0-Spezifikation durch Leerzeichen getrennt werden.

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 das automatische Ausfüllen der Aussteller-URL, wenn Ihr Anbieter über einen öffentlichen `.well-known/openid-configuration` Endpunkt verfügt, an dem Amazon Cognito die URLs Endpunkte`authorization`, `tokenuserInfo`, und `jwks_uri` abrufen kann.
12. Geben Sie die Aussteller-URL oder`authorization`, `tokenuserInfo`, und den `jwks_uri` Endpunkt URLs Ihres IdP ein.

> **Note**
>
> Sie können nur die Portnummern 443 und 80 mit Erkennung verwenden, die automatisch ausgefüllt und manuell eingegeben wurden URLs. 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_endpointuserinfo_endpoint`, und. `jwks_uri` Ebenso können Sie, wenn Sie Manuelle Eingabe wählen, 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 Erstellen aus.
16. Wählen Sie im Menü App-Clients einen App-Client aus der Liste aus. Um den neuen SAML-Identitätsanbieter zum App-Client hinzuzufügen, navigieren Sie zur Registerkarte Anmeldeseiten und wählen Sie in der Konfiguration der verwalteten Anmeldeseiten die Option Bearbeiten aus.
17. Wählen Sie Änderungen speichern aus.

Weitere Informationen zu OIDC IdPs finden Sie unter. 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 IdPs von Drittanbietern finden Sie unter. 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. 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 das Menü Soziale Netzwerke und externer Anbieter und wählen Sie 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 Ihren SAML 2.0-Identitätsanbieter so, dass er Abmeldeantworten an den

`https://mydomain.auth.us-east-1.amazoncognito.com/saml2/logout` Endpunkt sendet, den Amazon Cognito erstellt, wenn Sie die verwaltete Anmeldung konfigurieren. 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 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 eine solche Ausnahme wäre „Fehler beim Abrufen von Metadaten von“. *`<metadata endpoint>`*

So richten Sie den SAML-IdP zum Hinzufügen eines Signaturzertifikats ein

- Gehen Sie wie folgt vor, um das Zertifikat zu erhalten, das den öffentlichen Schlüssel enthält, den der IdP zur Überprüfung der signierten Abmeldeanforderung verwendet:
  1. Gehen Sie in Ihrem Benutzerpool zum Menü Soziale Dienste und externe Anbieter.
  2. Wählen Sie Ihren SAML-Anbieter aus,
  3. Wählen Sie Signaturzertifikat anzeigen.

Weitere Informationen zu SAML IdPs finden Sie unterVerwendung 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. Sie müssen die verwaltete Anmeldung aktivieren, um die Integration mit unterstützten Anbietern von sozialen Identitäten zu ermöglichen. Wenn Amazon Cognito Ihre verwalteten Anmeldeseiten erstellt, erstellt es OAuth 2.0-Endpunkte, die Amazon Cognito und Ihr OIDC und Ihre sozialen Netzwerke IdPs zum Informationsaustausch verwenden. 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

- Richten Sie ein Social IdP-Entwicklerkonto und eine Anwendung ein
- Konfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP
- Testen der Konfiguration Ihres Social-Identity-Anbieters

Richten Sie ein Social IdP-Entwicklerkonto und eine Anwendung ein

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.

Facebook

Aktuelle Informationen zur Konfiguration von Meta-Entwicklerkonten und zur Authentifizierung finden Sie unter Meta-App-Entwicklung.

Wie registriere ich eine Anwendung bei Facebook/Meta

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.auth.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.auth.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 `/oauth2/idpresponse` Endpunkt für Ihre Benutzerpool-Domain in OAuthValid Redirect ein. URIs

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
```

15. Wählen Sie Änderungen speichern aus.

Login with Amazon

Aktuelle Informationen zur Konfiguration der Login with Amazon Amazon-Entwicklerkonten und zur Authentifizierung finden Sie in der Dokumentation Login with Amazon.

So registrieren Sie eine Anwendung mit Login with Amazon

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.auth.us-east-1.amazoncognito.com
```

10. Geben Sie Ihre Benutzerpool-Domain mit dem `/oauth2/idpresponse` Endpunkt in Allowed Return ein URLs.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
```

11. Wählen Sie Speichern.

Google

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

Wie registriere ich eine Anwendung bei Google

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 Dienste APIs und dann den Bildschirm mit der Zustimmung von OAuth 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 Bereiche die Option Bereiche hinzufügen oder entfernen und wählen Sie mindestens die folgenden Bereiche aus. OAuth

   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 Dienste APIs und dann Anmeldeinformationen aus.
11. Wählen Sie CREATE CREDENTIALS und 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://mydomain.auth.us-east-1.amazoncognito.com
    ```

14. Wählen Sie unter Autorisierte Weiterleitung URIs die Option URI HINZUFÜGEN aus. Geben Sie den Pfad zum Endpunkt `/oauth2/idpresponse` Ihrer Benutzerpool-Domäne ein.

    ```
    https://mydomain.auth.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.

Sign in with Apple

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.

Wie registriere ich eine Anwendung mit Sign in with Apple (SIWA)

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 „Neuen Identifier 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 Konfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP ausgewählt haben.
    3. Wählen Sie unter Funktionen die Option Mit Apple anmelden und dann Bearbeiten aus.
    4. Wählen Sie auf der Seite „Mit Apple anmelden: App-ID-Konfiguration“ aus, ob Sie die App entweder als primäre App oder als Gruppe mit einer anderen App einrichten möchten IDs, und wählen Sie dann Speichern aus.
    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 ID registrieren“ die Option Dienste 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 Konfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP.
    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 das Pluszeichen neben Website aus URLs.
    4. Geben Sie unter Domains and subdomains (Domänen und Subdomänen) Ihre Benutzerpool-Domäne ohne das Präfix `https://` ein.

       ```
       mydomain.auth.us-east-1.amazoncognito.com
       ```

    5. Geben Sie URLs unter Zurück den Pfad zum `/oauth2/idpresponse` Endpunkt Ihrer Benutzerpool-Domain ein.

       ```
       https://mydomain.auth.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 Konfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP ausgewählt haben.

Konfigurieren Sie Ihren Benutzerpool mit einem sozialen IdP

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 Benutzerpool.
4. Wählen Sie das Menü Soziale Netzwerke und externe Anbieter und wählen Sie dann 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 Dienst-ID-ID wird in allen Benutzerpools und and/or anderen Authentifizierungsdiensten 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 Erstellen aus.
10. Wählen Sie im Menü App-Clients einen App-Client aus der Liste aus. Um den neuen Social Identity Provider zum App-Client hinzuzufügen, navigieren Sie zur Registerkarte Anmeldeseiten und wählen Sie in der Konfiguration Verwaltete Anmeldeseiten bearbeiten aus.
11. Wählen Sie Änderungen speichern aus.

Testen der Konfiguration Ihres Social-Identity-Anbieters

In Ihrer Anwendung müssen Sie im Client des Benutzers einen Browser aufrufen, damit er sich bei seinem sozialen Anbieter anmelden kann. Testen Sie die Anmeldung bei Ihrem sozialen Anbieter, nachdem Sie die Einrichtungsverfahren in den vorherigen Abschnitten abgeschlossen haben. Die folgende Beispiel-URL lädt die Anmeldeseite für Ihren Benutzerpool mit einer Präfix-Domain.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Dieser Link ist die Seite, zu der Sie Amazon Cognito weiterleitet, wenn Sie zum Menü App-Clients gehen, einen App-Client auswählen, zum Tab Anmeldeseiten navigieren und Anmeldeseite anzeigen auswählen. Weitere Informationen zu Benutzerpool-Domains finden Sie unterKonfigurieren einer Benutzerpool-Domäne. Weitere Informationen zu App-Clients, einschließlich Client IDs und Callback URLs, finden Sie unterAnwendungsspezifische Einstellungen mit App-Clients.

Mit dem folgenden Beispiel-Link wird eine automatische Weiterleitung von Autorisieren des Endpunkts mit einem `identity_provider` Abfrageparameter zu einem sozialen Anbieter eingerichtet. Diese URL umgeht die interaktive Benutzerpool-Anmeldung mit verwalteter Anmeldung und leitet direkt zur IdP-Anmeldeseite weiter.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/
authorize?identity_provider=Facebook|Google|LoginWithAmazon|
SignInWithApple&response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com
```

Verwendung 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 verwalteten Anmeldung authentifiziert Amazon Cognito lokale IdP-Benutzer und Drittanbieter-IdP-Benutzer und gibt JSON-Webtoken aus (). JWTs 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.

Kurzübersicht 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 Anwendungsspezifische Einstellungen mit App-Clients.

Themen

- Wissenswertes über SAML IdPs in Amazon Cognito Cognito-Benutzerpools
- Groß- und Kleinschreibung von SAML-Benutzernamen
- Konfiguration Ihres externen SAML-Identitätsanbieters
- Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool
- Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools
- SAML-Benutzer mit Single Sign-Out abmelden
- SAML-Signatur und Verschlüsselung
- Namen und Kennungen von SAML-Identitätsanbietern

Wissenswertes über SAML IdPs in Amazon Cognito Cognito-Benutzerpools

Die Implementierung eines SAML 2.0-IdP ist mit einigen Anforderungen und Einschränkungen verbunden. Lesen Sie diesen Abschnitt, wenn Sie Ihren IdP implementieren. Sie finden dort auch Informationen, die für die Behebung von Fehlern beim SAML-Verbund mit einem Benutzerpool nützlich sind.

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.auth.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 Zuordnung von IdP-Attributen zu Profilen und Tokens.

Für das Assertion-Format gelten spezifische Anforderungen

Ihr SAML-IdP muss die folgenden Ansprüche in die SAML-Assertion aufnehmen.

- `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. Um SAML-Benutzer einem konsistenten Benutzerprofil in Ihrem Benutzerpool zuzuweisen, weisen Sie Ihren `NameID` Anspruch anhand eines Attributs mit einem Wert zu, der sich nicht ändert.

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:persistent">
  carlos
</saml2:NameID>
```

Ein Wert `Format` in Ihrem `NameID` IdP-Anspruch von `urn:oasis:names:tc:SAML:1.1:nameid-format:persistent` bedeutet, dass Ihr IdP einen unveränderlichen Wert übergibt. Amazon Cognito benötigt diese Formatdeklaration nicht und weist ein Format von zu, `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified` falls Ihr IdP kein Format für den Anspruch angibt. `NameID` Dieses Verhalten entspricht Abschnitt 2.2.2 Complex Type Name der SAML IDType 2.0-Spezifikation.

- 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>
```

- Bei SP-initiiertem Single Sign-On ein `Response` Element mit einem `InResponseTo` Wert der ursprünglichen SAML-Anforderungs-ID.

```
<saml2p:Response Destination="https://mydomain.auth.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`

- 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.auth.us-
east-1.amazoncognito.com/saml2/idpresponse"/>
```

Von 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 im Menü Soziale Netzwerke und externe Anbieter 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 weitergibt. 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

Wenn Sie `InvalidParameterException` beim Erstellen eines SAML-IdP mit einer HTTPS-Metadatenendpunkt-URL beispielsweise „Fehler beim Abrufen von Metadaten von“ sehen*`<metadata endpoint>`*, stellen Sie sicher, dass für den Metadaten-Endpunkt SSL korrekt eingerichtet ist und dass ihm ein gültiges SSL-Zertifikat zugeordnet ist. Weitere Informationen zur Validierung von Zertifikaten finden Sie unter Was ist ein Zertifikat? SSL/TLS .

Der Metadaten-Endpunkt muss sich auf dem Standard-TCP-Port für HTTP oder HTTPS befinden

Amazon Cognito akzeptiert nur Metadaten URLs für SAML-Anbieter auf den Standard-TCP-Ports 80 für HTTP und 443 für HTTPS. Aus Sicherheitsgründen empfiehlt es sich, SAML-Metadaten unter einer TLS-verschlüsselten URL mit dem Präfix zu hosten. `https://` Geben Sie Metadaten URLs im Format oder ein. *`http://www.example.com/saml2/metadata.xml`* *`https://www.example.com/saml2/metadata.xml`* Die Amazon Cognito Cognito-Konsole akzeptiert Metadaten URLs nur mit dem `https://` Präfix. Sie können IdP-Metadaten auch mit CreateIdentityProviderund UpdateIdentityProviderkonfigurieren.

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.

Rotation von Zertifikaten für die Signierung

Amazon Cognito speichert SAML-Metadaten für bis zu sechs Stunden im Cache, wenn Sie Metadaten mit einer URL angeben. Wenn Sie eine Zertifikatsrotation für die Signatur von Metadaten durchführen, konfigurieren Sie Ihre Metadatenquelle so, dass sowohl das ursprüngliche als auch das neue Zertifikat für mindestens sechs Stunden veröffentlicht werden. Wenn Amazon Cognito den Cache über die Metadaten-URL aktualisiert, behandelt es jedes Zertifikat als gültig und Ihr SAML-IdP kann beginnen, SAML-Assertionen mit dem neuen Zertifikat zu signieren. Nach Ablauf dieses Zeitraums können Sie das Originalzertifikat aus den veröffentlichten Metadaten entfernen.

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

- Konfiguration Ihres externen SAML-Identitätsanbieters
- Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool
- Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools
- SAML-Benutzer mit Single Sign-Out abmelden
- SAML-Signatur und Verschlüsselung
- Namen und Kennungen von SAML-Identitätsanbietern

Konfiguration Ihres externen SAML-Identitätsanbieters

Wenn Sie Ihrem Benutzerpool einen SAML-Identitätsanbieter (IdP) hinzufügen möchten, müssen Sie einige Konfigurationsaktualisierungen in der Verwaltungsoberfläche Ihres IdP vornehmen. In diesem Abschnitt wird beschrieben, wie Sie die Werte formatieren, die Sie Ihrem IdP zur Verfügung stellen müssen. Sie können auch erfahren, wie Sie das statische oder aktive URL-Metadatendokument abrufen, das den IdP und seine SAML-Ansprüche an Ihren Benutzerpool identifiziert.

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.auth.us-east-1.amazoncognito.com`*`/saml2/idpresponse` Wenn Ihr Benutzerpool über eine Amazon Cognito Cognito-Domain verfügt, finden Sie Ihren Benutzerpool-Domänenpfad im Domain-Menü Ihres Benutzerpools in der Amazon Cognito Cognito-Konsole.

Bei einigen IdPs SAMLs müssen Sie die auch als `urn` Zielgruppen-URI oder SP-Entitäts-ID bezeichnete ID 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 auch benutzerdefinierte Konfigurationsinformationen für den SAML 2.0-Verbund mit IAM oder. AWS IAM Identity Center Um zu erfahren, wie Sie die Amazon Cognito Cognito-Integration einrichten, suchen Sie nach allgemeinen Anweisungen zum Abrufen des Metadatendokuments und verwalten Sie den Rest der Konfiguration in Ihrem Benutzerpool.

Lösung	Weitere Informationen
Microsoft Entra ID	Föderations-Metadaten
Okta	So laden Sie die IdP-Metadaten und SAML-Signaturzertifikate für eine SAML-App-Integration herunter
Auth0	Konfigurieren Sie Auth0 als SAML-Identitätsanbieter

Lösung	Weitere Informationen
Ping-Identität () PingFederate	Exportieren von SAML-Metadaten von PingFederate
JumpCloud	Hinweise zur SAML-Konfiguration
SecureAuth	Integration von SAML-Anwendungen

Hinzufügen und Verwalten von SAML-Identitätsanbietern in einem Benutzerpool

Nachdem Sie Ihren Identitätsanbieter für die Zusammenarbeit mit Amazon Cognito konfiguriert haben, können Sie ihn zu Ihren Benutzerpools und App-Clients hinzufügen. 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 das Menü Soziale Netzwerke und externe Anbieter und wählen Sie dann 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.auth.us-east-1.amazoncognito.com/saml2/logout` Endpunkt sendet, der bei der Konfiguration der verwalteten Anmeldung erstellt wird. 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 aus.

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 das `<SAML metadata XML>` 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 das `<SAML metadata XML>` 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-to-SP Antwortnachricht verwendet.

  ```
  https://mydomain.auth.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, mit dem Ihr IdP SAML-Abmeldeanfragen validieren kann, wählen Sie das Menü Soziale Netzwerke und externe Anbieter in Ihrem Benutzerpool, 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 das Menü Soziale Netzwerke und externe Anbieter.
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.

Themen

- Implementieren Sie die SP-initiierte SAML-Anmeldung
- Implementieren Sie die IDP-initiierte SAML-Anmeldung

Implementieren Sie die SP-initiierte 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.

Der folgende Prozess zeigt, wie Benutzer die SP-initiierte Anmeldung bei Ihrem Benutzerpool über einen SAML-Anbieter abschließen.

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 Anwendung erfassen oder die verwaltete Anmeldung in der Webansicht aufrufen.

   Sie können Ihre verwalteten Anmeldeseiten so konfigurieren, dass sie eine Liste von E-Mail-Adressen anzeigen IdPs oder zur Eingabe einer E-Mail-Adresse auffordern und diese mit der ID Ihres SAML-IdP abgleichen. Um nach einer E-Mail-Adresse zu fragen, bearbeiten Sie Ihren Branding-Stil für verwaltete Logins und suchen Sie in Foundation nach Authentifizierungsverhalten und legen Sie unter Anbieteranzeige die Option Anzeigestil auf Domain-Sucheingabe fest.
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 optional signierten SAML-Anfrage 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 verwalteten Anmeldung 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 das 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 einem Gültigkeitsbereich von 2.0. 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. Bei der verwalteten Anmeldung werden Benutzersitzungen in einem Browser-Cookie verwaltet, das 1 Stunde lang gültig ist.

Implementieren Sie die IDP-initiierte 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.

1. Ein Benutzer fordert die SAML-Anmeldung mit Ihrer Anwendung an.
2. Ihre Anwendung ruft einen Browser auf oder leitet den Benutzer zur Anmeldeseite seines SAML-Anbieters weiter.
3. Der IdP authentifiziert den Benutzer interaktiv oder mit einer gespeicherten Sitzung in einem Browser-Cookie.
4. Der IdP leitet Ihren Benutzer mit der SAML-Assertion oder -Antwort in seinem POST-Text zu Ihrer Anwendung weiter.
5. Ihre Anwendung fügt die SAML-Assertion dem POST-Text einer Anfrage an Ihren Benutzerpool-Endpunkt hinzu. `saml2/idpresponse`
6. Amazon Cognito gibt Ihrem Benutzer einen Autorisierungscode aus.

7. Ihr Benutzer präsentiert seinen Autorisierungscode in Ihrer App, die den Code gegen JSON-Webtoken (JWTs) eintauscht.
8. Ihre Anwendung akzeptiert und verarbeitet das ID-Token Ihres Benutzers als Authentifizierung, generiert autorisierte Anfragen an Ressourcen mit seinem Zugriffstoken und speichert das Aktualisierungstoken.

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 verwaltete Anmeldeautorisierungsanfragen 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 ein Beispiel für eine Anfrage und Antwort für einen vom 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 zum Menü Soziale Dienste und externe Anbieter.
4. Bearbeiten oder fügen Sie einen SAML-Anbieter 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 vom IdP initiiertes SAML unterstützt.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

SAML-Benutzer mit Single Sign-Out abmelden

Amazon Cognito unterstützt SAML 2.0 Single Logout (SLO). Mit SLO kann Ihre Anwendung Benutzer von ihren SAML-Identitätsanbietern (IdPs) abmelden, wenn sie sich aus Ihrem Benutzerpool abmelden. Auf diese Weise müssen sich Benutzer, wenn sie sich erneut bei Ihrer Anwendung anmelden möchten, mit ihrem SAML-IdP authentifizieren. Andernfalls verfügen sie möglicherweise über IdP- oder Benutzerpool-Browser-Cookies, die sie an Ihre Anwendung weiterleiten, ohne dass sie Anmeldeinformationen angeben müssen.

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 im Menü Soziale Netzwerke und externe Anbieter Ihres Benutzerpools Ihren IdP aus und suchen Sie das Signaturzertifikat.
4. Wählen Sie Als .crt herunterladen.
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

Bei der SAML 2.0-Anmeldung steht der Benutzer einer Anwendung als Träger von Anfragen und Antworten in seinem Authentifizierungsablauf im Mittelpunkt. Möglicherweise möchten Sie sicherstellen, dass Benutzer diese SAML-Dokumente während der Übertragung nicht lesen oder ändern. Um dies zu erreichen, fügen Sie den SAML-Identitätsanbietern (IdPs) in Ihrem Benutzerpool SAML-Signatur und Verschlüsselung hinzu. Mit der SAML-Signatur fügen Ihre Benutzerpools SAML-Anmelde- und Abmeldeanfragen eine Signatur hinzu. Mit dem öffentlichen Schlüssel Ihres Benutzerpools kann Ihr IdP überprüfen, ob er unveränderte SAML-Anfragen empfängt. Wenn Ihr IdP dann antwortet und SAML-Assertionen an die Browsersitzungen der Benutzer weiterleitet, kann der IdP diese Antwort verschlüsseln, sodass der Benutzer seine eigenen Attribute und Berechtigungen nicht überprüfen kann.

Bei der SAML-Signatur und -Verschlüsselung müssen alle kryptografischen Operationen während SAML-Vorgängen im Benutzerpool 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 Dauer gültig und der öffentliche Schlüssel ändert sich nicht zwischen den Zertifikaten. 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 im Menü Soziale Netzwerke und externe Anbieter Ihres Benutzerpools 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"
}
```

Um das Verschlüsselungszertifikat aus Ihrem Benutzerpool abzurufen, stellen Sie eine DescribeIdentityProviderAPI-Anfrage und rufen Sie den Wert von `ActiveEncryptionCertificate` im Antwortparameter `ProviderDetails` ab. Speichern Sie dieses Zertifikat und stellen Sie es Ihrem IdP als Verschlüsselungszertifikat für Anmeldeanfragen aus Ihrem Benutzerpool zur Verfügung.

SAML-Anfragen signieren

Die Fähigkeit, 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 im Menü Soziale Netzwerke und externe Anbieter Ihres Benutzerpools 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 An- 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 verwaltete Anmeldung die E-Mail-Adresse jedes Benutzers an, bewertet die Domain in seiner E-Mail-Adresse und leitet sie 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 verwaltete Anmeldung 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.auth.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.

Bei IdP-Identifikatoren in einem Domänenformat fordert die verwaltete Anmeldung 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 verwaltete Anmeldung 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 Seite`bob@exampleA.co.uk`, mit der er interagiert, die IdP-Anmeldeseite unter. `https://auth.exampleA.co.uk/sso/saml`

Sie können dieselbe Logik auch unabhängig 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 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 bei der verwalteten Anmeldung zu analysieren, weisen Sie jedem SAML-IdP, den Sie Ihrem App-Client zugewiesen haben, mindestens eine Kennung zu. Standardmäßig zeigt der Anmeldebildschirm für die verwaltete Anmeldung eine Schaltfläche für jedes Element an IdPs, das Sie Ihrem App-Client zugewiesen haben. Wenn Sie jedoch erfolgreich Kennungen zugewiesen haben, sieht Ihre klassische Hosted UI-Anmeldeseite wie in der folgenden Abbildung aus.

> **Note**
>
> In der klassischen gehosteten Benutzeroberfläche werden Sie auf der Anmeldeseite für Ihren App-Client automatisch zur Eingabe einer E-Mail-Adresse aufgefordert, wenn Sie Ihrer E-Mail-Adresse Kennungen zuweisen. IdPs Bei der verwalteten Anmeldung müssen Sie dieses Verhalten im Branding-Editor aktivieren. Wählen Sie in der Kategorie Einstellungen für das Authentifizierungsverhalten unter der Überschrift Anbieteranzeige die Option Domain-Sucheingabe aus.

Die Domainanalyse bei verwalteter Anmeldung erfordert, dass Sie Domains als Ihre IdP-Identifikatoren verwenden. Wenn Sie jeder SAML IdPs für einen App-Client eine Kennung beliebigen Typs zuweisen, zeigt die verwaltete Anmeldung 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 Ihren verwalteten Anmeldeseiten eine Liste von 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 verwalteten Anmeldeseite eine Schaltfläche angezeigt, mit der Sie sich mit diesem IdP anmelden können.
- Wenn Sie jedem SAML-IdP, den Sie für Ihren App-Client aktivieren, eine Kennung zuweisen, wird auf der verwalteten Anmeldeseite eine Benutzereingabeaufforderung für eine E-Mail-Adresse angezeigt.
- Wenn Sie mehrere haben IdPs und nicht allen eine Kennung zuweisen, wird auf der Anmeldeseite für verwaltete Anmeldung eine Schaltfläche angezeigt, mit der Sie sich bei jedem zugewiesenen IdP anmelden können.
- Wenn Sie Ihren Identifikatoren zugewiesen haben IdPs und Sie möchten, dass auf Ihren verwalteten Anmeldeseiten eine Auswahl von IdP-Schaltflächen angezeigt wird, 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..

Verwendung von OIDC-Identitätsanbietern mit einem Benutzerpool

Benutzer können sich mit ihren vorhandenen Konten von OpenID Connect (OIDC) - Identitätsanbietern () bei Ihrer Anwendung anmelden. IdPs Mit OIDC-Anbietern können Benutzer unabhängiger Single Sign-On-Systeme bestehende Anmeldeinformationen angeben, während Ihre Anwendung OIDC-Token im gemeinsamen Format von Benutzerpools erhält. Um einen OIDC-IdP zu konfigurieren, richten Sie Ihren IdP so ein, dass er Ihren Benutzerpool als RP verwaltet, und konfigurieren Sie Ihre Anwendung so, dass Ihr Benutzerpool als IdP verwaltet wird. Amazon Cognito dient als Zwischenschritt zwischen mehreren OIDC IdPs und Ihren Anwendungen. Ihr Benutzerpool wendet Regeln zur Attributzuweisung auf die Ansprüche in den ID- und Zugriffstoken an, die Ihr Anbieter direkt an Ihren Benutzerpool weitergibt. Amazon Cognito gibt dann neue Token aus, die auf den zugewiesenen Benutzerattributen und allen zusätzlichen Anpassungen basieren, die Sie am Authentifizierungsablauf mit Lambda-Triggern vorgenommen haben.

Benutzer, die sich mit einem OIDC-IdP anmelden, müssen keine neuen Anmeldeinformationen oder Informationen angeben, um auf Ihre Benutzerpool-Anwendung zuzugreifen. Ihre Anwendung kann sie zur Anmeldung im Hintergrund an ihren IdP weiterleiten, wobei ein Benutzerpool als Tool im Hintergrund das Tokenformat für Ihre Anwendung standardisiert. Weitere Informationen zur IdP-Umleitung finden Sie unter. Autorisieren des Endpunkts

Wie bei anderen Identitätsanbietern von Drittanbietern müssen Sie Ihre Anwendung beim OIDC-Anbieter registrieren und Informationen über die IdP-Anwendung abrufen, die Sie mit Ihrem Benutzerpool verbinden möchten. Ein Benutzerpool-OIDC-IdP benötigt eine Client-ID, einen geheimen Client-Schlüssel, Bereiche, die Sie anfordern möchten, und Informationen zu Providerdienst-Endpunkten. Ihr Benutzerpool kann die Provider-OIDC-Endpunkte von einem Discovery-Endpunkt aus ermitteln oder Sie können sie manuell eingeben. Sie müssen auch Provider-ID-Token untersuchen und Attributzuordnungen zwischen dem IdP und den Attributen in Ihrem Benutzerpool erstellen.

Weitere Informationen Authentifizierungsablauf für den OIDC-Benutzerpool-IdP zu diesem Authentifizierungsablauf finden Sie unter.

> **Note**
>
> Die Anmeldung über einen Drittanbieter (Verbund) in Amazon-Cognito-Benutzerpools wird unterstützt. Diese Funktion ist unabhängig vom OIDC-Verbund mit Amazon Cognito Cognito-Identitätspools.

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
- Registrieren Sie eine Anwendung bei einem OIDC-IdP
- Hinzufügen eines OIDC-IdP zu Ihrem Benutzerpool
- 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-IDkid-Anspruch am `jwks_uri` und enthält einen `kid`-Anspruch in seinen Token.

- Präsentiert einen nicht abgelaufenen öffentlichen Schlüssel mit einer gültigen Root-CA-Vertrauenskette.

Registrieren Sie eine Anwendung bei einem OIDC-IdP

Bevor Sie Ihrer Benutzerpool-Konfiguration einen OIDC-IdP hinzufügen und ihn App-Clients zuweisen, richten Sie in Ihrem IdP eine OIDC-Client-Anwendung ein. Ihr Benutzerpool ist die Rely-Party-Anwendung, die die Authentifizierung mit Ihrem IdP verwaltet.

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	Salesforce als OpenID Connect-Identitätsanbieter	`https://MyDomainName.my.salesforce.com/.well-known/openid-configuration`
OneLogin	Eine OIDC-fähige App Connect	`https://your-domain.onelogin.com/oidc/2/.well-known/openid-configuration`
JumpCloud	SSO mit OIDC	`https://oauth.id.jumpcloud.com/.well-known/openid-configuration`
Okta	Installieren eines Okta-Identitätsanbieters	`https://Your Okta subdomain.okta.com/.well-known/openid-configuration`

OIDC-IdP	Installieren	URL für die OIDC-Erkennung
Microsoft Entra ID	OpenID Connect auf der Microsoft-Identitätsplattform	`https://login.microsoftonline.com/ {tenant}/v2.0` Die Werte von `tenant` können eine Mandanten-ID, `commonorganizations` , oder `consumers` enthalten.

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.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
```

3. Wählen Sie die Bereiche aus, die Ihr Benutzerverzeichnis mit Ihrem Benutzerpool gemeinsam nutzen soll. Der Bereich openid ist erforderlich, damit OIDC Benutzerinformationen IdPs anbieten kann. Der `email` Umfang ist erforderlich, um Zugriff auf die Ansprüche und Angaben zu gewähren. `emailemail_verified` Zusätzliche Bereiche in der OIDC-Spezifikation gelten `profile` für alle Benutzerattribute und für undphone. `phone_number` `phone_number_verified`
4. Der OIDC-Identitätsanbieter stellt Ihnen eine Client-ID und einen geheimen Client-Schlüssel bereit. Notieren Sie sich diese Werte und fügen Sie sie der Konfiguration des OIDC-IdP hinzu, den Sie später zu Ihrem Benutzerpool hinzufügen.

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.auth.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 die OAuth Einstellungen und geben Sie die URL des **`/oauth2/idpresponse`** Endpunkts für Ihre Benutzerpool-Domain in das Feld Callback-URL ein. Dies ist die URL, unter der Salesforce den Autorisierungscode ausgibt, den Amazon Cognito gegen ein OAuth Token eintauscht.

   ```
   https://mydomain.auth.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 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.

Hinzufügen eines OIDC-IdP zu Ihrem Benutzerpool

Nachdem Sie Ihren IdP eingerichtet haben, können Sie Ihren Benutzerpool so konfigurieren, dass Authentifizierungsanfragen mit einem OIDC-IdP bearbeitet werden.

Amazon Cognito console

Einen OIDC-IdP in der Konsole hinzufügen

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 das Menü Soziale Netzwerke und externe Anbieter und wählen Sie dann Identitätsanbieter hinzufügen aus.
5. Wählen Sie einen OpenID Connect-IDP aus.
6. Geben Sie einen eindeutigen Anbieternamen ein.
7. Geben Sie die IdP-Client-ID ein. Dies ist die ID des Anwendungsclients, den Sie in Ihrem OIDC-IdP erstellen. Bei der von Ihnen angegebenen Client-ID muss es sich um einen OIDC-Anbieter handeln, den Sie mit der Callback-URL konfiguriert haben. `https://[your user pool domain]/oauth2/idpresponse`
8. Geben Sie den geheimen IdP-Client-Schlüssel ein. Dies muss der geheime Client-Schlüssel für denselben Anwendungsclient aus dem vorherigen Schritt sein.
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 gemäß der OAuth2.0-Spezifikation durch Leerzeichen getrennt werden.

   Ihr IdP fordert Benutzer möglicherweise auf, der Bereitstellung dieser Attribute für Ihre Anwendung zuzustimmen, wenn sie sich anmelden.

10. Wählen Sie eine Methode für die Anforderung von Attributen. IdPserfordert möglicherweise, dass Anfragen an ihre `userInfo` Endpunkte entweder als oder `GET` formatiert werden. `POST` Der Amazon Cognito `userInfo` Cognito-Endpunkt erfordert beispielsweise `HTTP GET` Anfragen.
11. Wählen Sie eine Setup-Methode für die Art und Weise, wie Ihr Benutzerpool den Pfad zu den wichtigsten OIDC-Federation-Endpunkten bei Ihrem IdP bestimmen soll. IdPs Hosten Sie einen `/well-known/openid-configuration` Endpunkt in der Regel unter einer Basis-URL eines Emittenten. Wenn dies bei Ihrem Anbieter der Fall ist, fordert Sie die Option Automatisch durch Aussteller-URL ausfüllen auf, diese Basis-URL einzugeben, versucht, von dort aus auf den `/well-known/openid-configuration` Pfad zuzugreifen, und liest die dort aufgelisteten Endpunkte. Möglicherweise haben Sie untypische Endpunktpfade oder möchten Anfragen über einen alternativen Proxy an einen oder mehrere Endpunkte weiterleiten. Wählen Sie in diesem Fall Manuelle Eingabe aus und geben Sie Pfade für die `authorization` Endpunkte, `tokenuserInfo`, und `jwks_uri` an.

    > **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.

12. Konfigurieren Sie Ihre Regeln für die Attributzuweisung unter Attribute zwischen Ihrem OpenID Connect-Anbieter und Ihrem Benutzerpool zuordnen. Das Benutzerpool-Attribut ist das Zielattribut im Amazon Cognito-Benutzerprofil und das OpenID Connect-Attribut ist das Quellattribut, das Amazon Cognito in einem ID-Token-Antrag oder einer Antwort finden soll. `userInfo` Amazon Cognito ordnet den `username` OIDC-Claim Sub automatisch dem Zielbenutzerprofil zu.

    Weitere Informationen finden Sie unter Zuordnung von IdP-Attributen zu Profilen und Tokens.
13. Wählen Sie Identitätsanbieter hinzufügen.
14. Wählen Sie im Menü App-Clients einen App-Client aus der Liste aus. Navigieren Sie zur Registerkarte Anmeldeseiten und wählen Sie unter Konfiguration der verwalteten

Anmeldeseiten die Option Bearbeiten aus. Suchen Sie nach Identitätsanbietern und fügen Sie Ihren neuen OIDC-IdP hinzu.

15. Wählen Sie Änderungen speichern aus.

API/CLI

Die OIDC-Konfiguration in Beispiel zwei finden Sie unter. CreateIdentityProvider
Sie können diese Syntax ändern und sie als Anforderungstext von`CreateIdentityProvider`,`UpdateIdentityProvider`, oder als `--cli-input-json` Eingabedatei für verwenden. create-identity-provider

Testen Ihrer OIDC-IdP-Konfiguration

In Ihrer Anwendung müssen Sie im Client des Benutzers einen Browser aufrufen, damit sich dieser bei seinem OIDC-Anbieter anmelden kann. Testen Sie die Anmeldung bei Ihrem Anbieter, nachdem Sie die Einrichtungsverfahren in den vorherigen Abschnitten abgeschlossen haben. Die folgende Beispiel-URL lädt die Anmeldeseite für Ihren Benutzerpool mit einer Präfix-Domain.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Dieser Link ist die Seite, zu der Sie Amazon Cognito weiterleitet, wenn Sie zum Menü App-Clients gehen, einen App-Client auswählen, zum Tab Anmeldeseiten navigieren und Anmeldeseite anzeigen auswählen. Weitere Informationen zu Benutzerpool-Domains finden Sie unterKonfigurieren einer Benutzerpool-Domäne. Weitere Informationen zu App-Clients, einschließlich Client IDs und Callback URLs, finden Sie unterAnwendungsspezifische Einstellungen mit App-Clients.

Mit dem folgenden Beispiel-Link wird eine automatische Umleitung zum `MyOIDCIdP` Anbieter Autorisieren des Endpunkts mit einem `identity_provider` Abfrageparameter eingerichtet. Diese URL umgeht die interaktive Benutzerpool-Anmeldung mit verwalteter Anmeldung und leitet direkt zur IdP-Anmeldeseite weiter.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
identity_provider=MyOIDCIdP&response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com
```

Authentifizierungsablauf für den OIDC-Benutzerpool-IdP

Mit der OpenID Connect (OIDC) -Anmeldung automatisiert Ihr Benutzerpool den Anmeldevorgang mit dem Autorisierungscode bei Ihrem Identitätsanbieter (IdP). Nachdem Ihr Benutzer die Anmeldung mit seinem IdP abgeschlossen hat, sammelt Amazon Cognito seinen Code am `oauth2/idpresponse` Endpunkt des externen Anbieters. Mit dem resultierenden Zugriffstoken fragt Ihr Benutzerpool den `userInfo` IdP-Endpunkt ab, um Benutzerattribute abzurufen. Ihr Benutzerpool vergleicht dann die empfangenen Attribute mit den von Ihnen eingerichteten Regeln für die Attributzuweisung und füllt das Benutzerprofil und das ID-Token entsprechend aus.

Die OAuth 2.0-Bereiche, die Sie in Ihrer OIDC-Anbieterkonfiguration anfordern, definieren die Benutzerattribute, die der IdP Amazon Cognito zur Verfügung stellt. Aus Sicherheitsgründen sollten Sie nur die Bereiche anfordern, die den Attributen entsprechen, die Sie Ihrem Benutzerpool zuordnen möchten. Wenn Ihr Benutzerpool beispielsweise Anfragen stelltopenid `profile`, erhalten Sie alle möglichen Attribute, aber wenn Sie eine Anfrage stellen, erhalten `openid email phone_number` Sie nur die E-Mail-Adresse und Telefonnummer des Benutzers. Sie können die Bereiche, die Sie von OIDC anfordern, so konfigurieren, dass sie sich von denen unterscheiden IdPs, die Sie im App-Client und in der Benutzerpool-Authentifizierungsanfrage autorisieren und anfordern.

Wenn sich Ihr Benutzer mit einem OIDC-IdP bei Ihrer Anwendung anmeldet, führt Ihr Benutzerpool den folgenden Authentifizierungsablauf durch.

1. Ein Benutzer greift auf Ihre verwaltete Anmeldeseite zu und entscheidet sich dafür, sich mit seinem OIDC-IdP anzumelden.
2. Ihre Anwendung leitet den Browser des Benutzers zum Autorisierungsendpunkt Ihres Benutzerpools weiter.
3. Ihr Benutzerpool leitet die Anfrage an den Autorisierungsendpunkt des OIDC-IdP weiter.
4. Ihr IdP zeigt eine Anmeldeaufforderung an.
5. In Ihrer Anwendung zeigt die Sitzung Ihres Benutzers eine Anmeldeaufforderung für den OIDC-IdP an.
6. Der Benutzer gibt seine Anmeldeinformationen für den IdP ein oder präsentiert ein Cookie für eine bereits authentifizierte Sitzung.
7. Nachdem sich Ihr Benutzer authentifiziert hat, leitet der OIDC-IdP mit einem Autorisierungscode zu Amazon Cognito weiter.
8. Ihr Benutzerpool tauscht den Autorisierungscode gegen ID- und Zugriffstoken aus. Amazon Cognito erhält Zugriffstoken, wenn Sie Ihren IdP mit den Bereichen konfigurieren. `openid` Die

Ansprüche im ID-Token und in der `userInfo` Antwort werden durch zusätzliche Bereiche aus Ihrer IdP-Konfiguration bestimmt, z. B. `profile` und. `email`

9. Ihr IdP stellt die angeforderten Token aus.
10. Ihr Benutzerpool bestimmt den Pfad zum `jwks_uri` IdP-Endpunkt vom Aussteller URLs in Ihrer IdP-Konfiguration und fordert die Tokensignaturschlüssel vom JSON-Webschlüsselsatz (JWKS) - Endpunkt an.
11. Der IdP gibt Signaturschlüssel vom JWKS-Endpunkt zurück.
12. Ihr Benutzerpool validiert die IdP-Token anhand der Signatur- und Ablaufdaten in den Token.
13. Ihr Benutzerpool autorisiert eine Anfrage an den `userInfo` IdP-Endpunkt mit dem Zugriffstoken. Der IdP antwortet mit Benutzerdaten, die auf den Gültigkeitsbereichen des Zugriffstokens basieren.
14. Ihr Benutzerpool vergleicht das ID-Token und die `userInfo` Antwort des IdP mit den Attributzuordnungsregeln in Ihrem Benutzerpool. Es schreibt zugeordnete IdP-Attribute in Benutzerpool-Profilattribute.
15. Amazon Cognito stellt die Bearer-Token für Ihre Anwendung aus, wobei es sich unter anderem um Identitäts-, Zugriffs- und Aktualisierungs-Token handeln kann.
16. Ihre Anwendung verarbeitet die Benutzerpool-Token und meldet den Benutzer an.

> **Note**
>
> Amazon Cognito storniert Authentifizierungsanfragen, die nicht innerhalb von 5 Minuten abgeschlossen werden, und leitet den Benutzer zur verwalteten Anmeldung weiter. Für die Seite wird eine `Something went wrong`-Fehlermeldung angezeigt.

OIDC ist eine Identitätsebene auf OAuth 2.0, die Identitätstoken im JSON-Format (JWT) spezifiziert, die von an OIDC-Client-Apps (vertrauende Parteien) 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 Zugriffstoken ist ein Standard-2.0-Token. OAuth 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 Managed Login 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. Amazon Cognito aktualisiert den Signaturschlüssel vom JWKS-Endpunkt in Ihrer IdP-Konfiguration für jedes IdP-ID-Token, das es verarbeitet.
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.

Zuordnung von IdP-Attributen zu Profilen und Tokens

Identitätsanbieterdienste (IdP), einschließlich Amazon Cognito, können in der Regel mehr Informationen über einen Benutzer aufzeichnen. Möglicherweise möchten Sie wissen, für welches Unternehmen sie arbeiten, wie Sie sie kontaktieren können und welche anderen identifizierenden Informationen Sie benötigen. Das Format dieser Attribute ist jedoch von Anbieter zu Anbieter unterschiedlich. Richten Sie beispielsweise drei IdPs von drei verschiedenen Anbietern mit Ihrem Benutzerpool ein und untersuchen Sie jeweils ein Beispiel für eine SAML-Assertion, ein ID-Token oder eine `userInfo` Payload. Einer stellt die E-Mail-Adresse des Benutzers als`email`, ein anderer als und der dritte als `emailaddress` dar. `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

Ein großer Vorteil, der sich aus der Konsolidierung IdPs mit einem Benutzerpool ergibt, ist die Möglichkeit, die Vielzahl von Attributnamen in einem einzigen OIDC-Token-Schema mit konsistenten, vorhersehbaren, gemeinsamen Attributnamen abzubilden. Auf diese Weise müssen Ihre Entwickler nicht die Logik für die Verarbeitung einer komplexen Vielzahl von Single Sign-On-Ereignissen beibehalten. Bei dieser Formatkonsolidierung handelt es sich um eine Attributzuordnung. Bei der Zuordnung von Benutzerpoolattributen werden IdP-Attributnamen den entsprechenden Benutzerpool-Attributnamen zugewiesen. Sie können Ihren Benutzerpool beispielsweise so konfigurieren, dass der Wert eines `emailaddress` Claims in das Standard-Benutzerpool-Attribut geschrieben wird. `email`

Jeder Benutzerpool-IdP hat ein separates Attributzuordnungsschema. Um Attributzuordnungen für Ihren IdP anzugeben, konfigurieren Sie einen Benutzerpool-Identitätsanbieter in der Amazon Cognito Cognito-Konsole, ein AWS SDK oder die REST-API für Benutzerpools.

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`

- Wenn in einem Benutzerpool nicht zwischen Groß- und Kleinschreibung unterschieden wird, wandelt Amazon Cognito das Quellattribut des Benutzernamens in automatisch generierten Benutzernamen von Verbundbenutzern in Kleinbuchstaben um. Im Folgenden finden Sie ein Beispiel für einen Benutzernamen für einen Benutzerpool, bei dem Groß- und Kleinschreibung

beachtet wird:. `MySAML_TestUser@example.com` Das Folgende ist derselbe Benutzername für einen Benutzerpool, bei dem Groß- und Kleinschreibung nicht berücksichtigt wird:. `MySAML_testuser@example.com`

In Benutzerpools ohne Berücksichtigung der Groß- und Kleinschreibung müssen Ihre Lambda-Trigger, die den Benutzernamen verarbeiten, diese Änderung aller Ansprüche in gemischter Groß- und Kleinschreibung für Quellattribute von Benutzernamen berücksichtigen. Um Ihren IdP mit einem Benutzerpool zu verknüpfen, der eine andere Einstellung zur Berücksichtigung von Groß- und Kleinschreibung hat als Ihr aktueller Benutzerpool, erstellen Sie einen neuen Benutzerpool.

- 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. Um ein veränderbares benutzerdefiniertes Attribut in der Amazon Cognito Cognito-Konsole zu erstellen, aktivieren Sie das Kontrollkästchen Veränderbar für das Attribut, das Sie hinzufügen, wenn Sie im Anmeldemenü Benutzerdefinierte Attribute hinzufügen auswählen. 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, die in den eckigen Klammern und eingeschlossen ist. `[ ]` Das Amazon Cognito Cognito-URL-Formular kodiert die Werte, die nicht-alphanumerische Zeichen

enthalten, mit Ausnahme von,, und. `.` `-` `*` `_` 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 das Menü Soziale Netzwerke und externe Anbieter.
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 ***<provider>***Attributspalte 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 das Menü Soziale Netzwerke und externe Anbieter.
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 Verwendung 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 Änderungen speichern aus.

Angeben von Zuordnungen von Identitätsanbieter-Attributen für Ihren Benutzerpool (und Ihre API)AWS CLIAWS

Im folgenden Anfragetext CreateIdentityProviderwerden die UpdateIdentityProvider„MyIdP“ -Attribute des SAML-Anbietersemailaddress,birthdate, und phone den Benutzerpool-Attributenemail, und birthdatephone_number, in dieser Reihenfolge zugeordnet. Dies ist ein vollständiger Anfragetext für einen SAML 2.0-Anbieter. Ihr Anfragetext variiert je nach IdP-Typ und spezifischen Details. Die Attributzuordnung ist im Parameter enthalten. AttributeMapping

```
{
   "AttributeMapping": {
      "email" : "emailaddress",
      "birthdate" : "birthdate",
      "phone_number" : "phone"
   },
   "IdpIdentifiers": [
      "IdP1",
      "pdxsaml"
   ],
   "ProviderDetails": {
      "IDPInit": "true",
      "IDPSignout": "true",
      "EncryptedResponses" : "true",
      "MetadataURL": "https://auth.example.com/sso/saml/metadata",
      "RequestSigningAlgorithm": "rsa-sha256"
   },
   "ProviderName": "MyIdP",
   "ProviderType": "SAML",
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

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**
  >
  > Wenn der irgendwelche Anführungszeichen (") *<SAML metadata XML>* enthält, müssen sie maskiert 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
  ```

- API/SDK: 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`

- API/SDK: 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>`
- API/SDK: 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, die mehrere IdP-Benutzer hat, ein einheitliches Erlebnis in Ihrer App haben. 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. Beispiel, 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, und. IdPs ADFS1 ADFS2 ADFS3

> **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 Benutzer mit jedem IdP aus bis zu fünf IdP-Attributansprüchen verknüpfen, wie durch den `ProviderAttributeName` Parameter von `SourceUser` in einer `AdminLinkProviderForUser` API-Anfrage definiert. Wenn Sie beispielsweise mindestens einen Benutzer mit den Quellattributen`email`,,`phone`, und `location` verknüpft haben `departmentgiven_name`, können Sie weitere Benutzer nur mit einem dieser fünf Attribute 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 Identitätsansprüche im ID-Token von einem OIDC-IdP und überprüft auch den `userInfo` Endpunkt von OAuth 2.0- und 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 der AWS SDKs.

Vom Benutzerpool verwaltete Anmeldung

Sie können eine Webdomain auswählen, um Dienste für Ihren Benutzerpool zu hosten. Ein Amazon Cognito Cognito-Benutzerpool erhält die folgenden Funktionen, wenn Sie eine Domain hinzufügen, die zusammen als verwaltete Anmeldung bezeichnet wird.

- Ein Autorisierungsserver, der als Identitätsanbieter (IdP) für Anwendungen fungiert, die mit OAuth 2.0 und OpenID Connect (OIDC) funktionieren. Der Autorisierungsserver leitet Anfragen weiter, gibt aus und verwaltet JSON-Webtoken (JWTs) und stellt Benutzerattributinformationen bereit.
- Eine ready-to-use Benutzeroberfläche (UI) für Authentifizierungsvorgänge wie Anmeldung, Abmeldung und Passwortverwaltung. Die verwalteten Anmeldeseiten dienen als Web-Frontend für Authentifizierungsdienste.
- Ein Service Provider (SP) oder Relying Party (RP) für SAML 2.0 IdPs, OIDC IdPs, Facebook, Login with Amazon, Sign in with Apple und Google.

Eine zusätzliche Option, die einige Funktionen mit der verwalteten Anmeldung teilt, ist die klassische gehostete Benutzeroberfläche. Die klassische gehostete Benutzeroberfläche ist eine Version der Managed Login Services der ersten Generation. Gehostete UI-IdP- und RP-Dienste haben im Allgemeinen dieselben Eigenschaften wie verwaltete Anmeldeseiten, aber die Anmeldeseiten haben ein einfacheres Design und weniger Funktionen. Beispielsweise ist die Anmeldung mit dem Hauptschlüssel in der klassischen gehosteten Benutzeroberfläche nicht verfügbar. Im Lite-Featureplan ist die klassische gehostete Benutzeroberfläche Ihre einzige Option für Benutzerpool-Domänendienste.

Bei den verwalteten Anmeldeseiten handelt es sich um eine Sammlung von Weboberflächen für grundlegende Aktivitäten wie Registrierung, Anmeldung, mehrstufige Authentifizierung und Kennwortzurücksetzung in Ihrem Benutzerpool. Sie verbinden Benutzer auch mit einem oder mehreren externen Identitätsanbietern (IdPs), wenn Sie Benutzern die Wahl zwischen verschiedenen Anmeldeoptionen geben möchten. Ihre App kann Ihre verwalteten Anmeldeseiten in den Browsern der Benutzer aufrufen, wenn Sie Benutzer authentifizieren und autorisieren möchten.

Sie können die Benutzererfahrung mit verwalteten Logins mit benutzerdefinierten Logos, Hintergründen und Stilen an Ihre Marke anpassen. Sie haben zwei Optionen für das Branding, das Sie auf Ihre verwaltete Anmelde-Benutzeroberfläche anwenden können: den Branding-Editor für die verwaltete Anmeldung und das (klassische) Branding der gehosteten Benutzeroberfläche für die gehostete Benutzeroberfläche.

Branding-Editor

Eine aktualisierte Benutzererfahrung mit den meisten up-to-date Authentifizierungsoptionen und einem visuellen Editor in der Amazon Cognito Cognito-Konsole.

Branding der gehosteten Benutzeroberfläche

Eine vertraute Benutzererfahrung für frühere Anwender von Amazon Cognito Cognito-Benutzerpools. Das Branding für die gehostete Benutzeroberfläche ist ein dateibasiertes System. Um das Branding auf gehostete UI-Seiten anzuwenden, laden Sie eine Logo-Bilddatei und eine Datei hoch, in der Werte für mehrere vordefinierte CSS-Stiloptionen festgelegt sind.

Der Branding-Editor ist nicht in allen Funktionsplänen für Benutzerpools verfügbar. Weitere Informationen finden Sie unter Funktionspläne für Benutzerpools.

Weitere Informationen zur Erstellung von Anfragen für verwaltete Anmelde- und gehostete Benutzeroberflächendienste finden Sie unterBenutzerpool-Endpunkte und Referenz für verwaltete Logins.

> **Note**
>
> Die von Amazon Cognito verwaltete Anmeldung unterstützt keine benutzerdefinierte Authentifizierung mit Lambda-Triggern für benutzerdefinierte Authentifizierungsherausforderungen.

Themen

- Lokalisierung verwalteter Logins
- Verwaltete Anmeldung einrichten mit AWS Amplify
- Verwaltete Anmeldung mit der Amazon Cognito Cognito-Konsole einrichten
- Anzeigen Ihrer Anmeldeseite
- Anpassen Ihrer Authentifizierungsseiten
- Wissenswertes zur verwalteten Anmeldung und zur gehosteten Benutzeroberfläche
- Konfigurieren einer Benutzerpool-Domäne
- Branding auf verwaltete Anmeldeseiten anwenden

Lokalisierung verwalteter Logins

Bei der verwalteten Anmeldung wird auf benutzerinteraktiven Seiten standardmäßig die englische Sprache verwendet. Sie können Ihre verwalteten Anmeldeseiten in lokalisierter Sprache für die Sprache Ihrer Wahl anzeigen. Die verfügbaren Sprachen sind diejenigen, die in der verfügbar sind AWS Management Console. Fügen Sie dem Link, den Sie an Benutzer verteilen, einen `lang` Abfrageparameter hinzu, wie im folgenden Beispiel gezeigt.

```
https://<your domain>/oauth2/authorize?lang=es&response_type=code&client_id=<your app
 client id>&redirect_uri=<your relying-party url>
```

Amazon Cognito setzt nach der ersten Anfrage mit einem `lang` Parameter ein Cookie im Browser der Benutzer mit ihrer Sprachpräferenz. Nachdem das Cookie gesetzt wurde, bleibt die Sprachauswahl bestehen, ohne dass Sie den Parameter anzeigen oder in Anfragen angeben müssen. Wenn ein Benutzer beispielsweise eine Anmeldeanfrage mit einem `lang=de` Parameter gestellt hat, werden seine verwalteten Anmeldeseiten auf Deutsch dargestellt, bis er seine Cookies löscht oder eine neue Anfrage mit einem neuen Lokalisierungsparameter wie stellt. `lang=en`

Die Lokalisierung ist nur für verwaltete Logins verfügbar. Sie müssen den Feature-Plan Essentials oder Plus nutzen und Ihrer Domain das Branding für verwaltete Logins zugewiesen haben.

Die Auswahl, die Ihr Benutzer bei der verwalteten Anmeldung vornimmt, ist für benutzerdefinierte E-Mail- oder SMS-Absender-Trigger nicht verfügbar. Wenn Sie diese Auslöser implementieren, müssen Sie andere Mechanismen verwenden, um die bevorzugte Sprache eines Benutzers zu bestimmen. In Anmeldeabläufen kann das `locale` Attribut je nach Standort die bevorzugte Sprache des Benutzers angeben. In Anmeldeabläufen kann die Region oder die App-Client-ID Ihres Benutzerpools auf eine bevorzugte Sprache hinweisen.

Die folgenden Sprachen sind verfügbar.

Verwaltete Anmeldesprachen

Sprache	Code
Deutsch	de
Englisch	en
Spanisch	es

Sprache	Code
Französisch	fr
Bahasa Indonesia	id
Italienisch	it
Japanisch	ja
Koreanisch	ko
Portugiesisch (Brasilien)	pt-BR
Chinesisch (vereinfacht)	zh-CN
Chinesisch (traditionell)	zh-TW

Verwaltete Anmeldung einrichten mit AWS Amplify

Wenn Sie AWS Amplify Ihrer Web- oder mobilen App Authentifizierung hinzufügen, können Sie Ihre verwalteten Anmeldeseiten in der Amplify-Befehlszeilenschnittstelle (CLI) und Bibliotheken im Amplify-Framework einrichten. Um Ihrer App eine Authentifizierung hinzuzufügen, fügen Sie die `Auth` Kategorie zu Ihrem Projekt hinzu. Authentifizieren Sie dann in Ihrer Anwendung Benutzerpool-Benutzer mit Amplify-Clientbibliotheken.

Sie können verwaltete Anmeldeseiten zur Authentifizierung aufrufen oder Benutzer über einen Autorisierungsendpunkt zusammenführen, der zu einem IdP weiterleitet. Nachdem sich ein Benutzer erfolgreich beim Anbieter authentifiziert hat, erstellt Amplify einen neuen Benutzer in Ihrem Benutzerpool und übergibt die Token des Benutzers an Ihre App.

Die folgenden Beispiele zeigen, wie Sie AWS Amplify die verwaltete Anmeldung mit sozialen Anbietern in Ihrer App einrichten können.

- React
- Swift
- Flattern
- Android

Verwaltete Anmeldung mit der Amazon Cognito Cognito-Konsole einrichten

Die erste Voraussetzung für die verwaltete Anmeldung und die gehostete Benutzeroberfläche ist eine Benutzerpool-Domain. Navigieren Sie in der Benutzerpools-Konsole zur Registerkarte Domain Ihres Benutzerpools und fügen Sie eine Cognito-Domain oder eine benutzerdefinierte Domain hinzu. Sie können auch während der Erstellung eines neuen Benutzerpools eine Domain auswählen. Weitere Informationen finden Sie unter Konfigurieren einer Benutzerpool-Domäne. Wenn eine Domain in Ihrem Benutzerpool aktiv ist, stellen alle App-Clients öffentliche Authentifizierungsseiten in dieser Domain bereit.

Wenn Sie eine Benutzerpool-Domain erstellen oder ändern, legen Sie die Branding-Version für Ihre Domain fest. Bei dieser Branding-Version haben Sie die Wahl zwischen verwalteter Anmeldung und gehosteter Benutzeroberfläche (klassisch). Die von Ihnen gewählte Branding-Version gilt für alle App-Clients, die die Anmeldedienste in Ihrer Domain verwenden.

Der nächste Schritt besteht darin, auf der Registerkarte App-Clients in Ihrem Benutzerpool einen App-Client zu erstellen. Bei der Erstellung eines App-Clients fragt Amazon Cognito Sie nach Informationen zu Ihrer Anwendung und fordert Sie dann auf, eine Rückgabe-URL auszuwählen. Die Rückgabe-URL wird auch als Relying Party (RP) -URL, Umleitungs-URI und Callback-URL bezeichnet. Dies ist die URL, von der aus Ihre Anwendung ausgeführt wird, zum Beispiel `https://www.example.com` oder `myapp://example`.

Nachdem Sie in Ihrem Benutzerpool eine Domain und einen App-Client mit einem Branding-Stil konfiguriert haben, sind Ihre verwalteten Anmeldeseiten im Internet verfügbar.

Anzeigen Ihrer Anmeldeseite

Wählen Sie in der Amazon Cognito Cognito-Konsole im Tab Anmeldeseiten für Ihren App-Client im Menü App-Clients die Schaltfläche Anmeldeseiten anzeigen. Über diese Schaltfläche gelangen Sie zu einer Anmeldeseite in Ihrer Benutzerpool-Domain 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 Anmeldeseite anzeigen ist nützlich, wenn Sie die Grundfunktionen Ihrer verwalteten Anmeldeseiten testen möchten. Ihre Anmeldeseiten entsprechen der Branding-Version, die Sie

Ihrer Benutzerpool-Domain zugewiesen haben. Sie können Ihre Anmelde-URL mit zusätzlichen und geänderten Parametern anpassen. In den meisten Fällen entsprechen die automatisch generierten Parameter des Links Anmeldeseite 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 Benutzerpool-Endpunkte und Referenz für verwaltete Logins.

Die Anmelde-Webseite 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 relying-party url>
```

Sie können die Domainzeichenfolge Ihres Benutzerpools im Domain-Menü in Ihrem Benutzerpool nachschlagen. Im Menü App-Clients können Sie den App-Client IDs, seinen Callback URLs, seine zulässigen Bereiche und andere Konfigurationen identifizieren.

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.

Beispiel für eine Anfrage für einen impliziten Zuschuss

Sie können Ihre Anmeldeseite mit der folgenden URL für die implizite Codegewährung aufrufen, wo. `response_type=token` Nach einer erfolgreichen Anmeldung gibt Amazon Cognito Benutzerpool-Token in die Adresszeile Ihres Webbrowsers aus.

```
            https://mydomain.auth.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://auth.example.com/
#id_token=eyJraaBcDeF1234567890&access_token=eyJraGhIjKlM1112131415&expires_in=3600&token_type=
```

Anpassen Ihrer Authentifizierungsseiten

In der Vergangenheit hostete Amazon Cognito nur Anmeldeseiten mit der klassischen gehosteten Benutzeroberfläche, einem einfachen Design, das Authentifizierungswebseiten ein universelles Aussehen verleiht. Sie könnten Amazon Cognito Cognito-Benutzerpools mit einem Logobild anpassen und einige Stile mit einer Datei optimieren, die einige CSS-Stilwerte spezifiziert. Später führte Amazon Cognito Managed Login ein, einen aktualisierten gehosteten Authentifizierungsdienst. Die verwaltete Anmeldung wurde look-and-feel mit dem Branding-Editor aktualisiert. Der Branding-Editor ist ein visueller Editor ohne Code und bietet mehr Optionen als die gehostete Benutzerschnittstelle zur Anpassung der Benutzeroberfläche. Bei der verwalteten Anmeldung wurden auch benutzerdefinierte Hintergrundbilder und ein Dunkelmodus-Thema eingeführt.

Sie können in Benutzerpools zwischen verwalteter Anmeldung und gehosteten UI-Branding-Erlebnissen wechseln. Weitere Informationen zum Anpassen Ihrer verwalteten Anmeldeseiten finden Sie unterBranding auf verwaltete Anmeldeseiten anwenden.

Wissenswertes zur verwalteten Anmeldung und zur gehosteten Benutzeroberfläche

Das einstündige Cookie für verwaltete Anmeldung und gehostete UI-Sitzung

Wenn sich ein Benutzer mit Ihren Anmeldeseiten oder einem Drittanbieter anmeldet, setzt Amazon Cognito ein Cookie in seinem Browser. Mit diesem Cookie können sich Benutzer eine Stunde lang erneut mit derselben Authentifizierungsmethode anmelden. Wenn sie sich mit ihrem Browser-Cookie anmelden, erhalten sie neue Token, die für die in Ihrer App-Client-Konfiguration angegebene Dauer gültig sind. Änderungen an Benutzerattributen oder Authentifizierungsfaktoren haben keinen Einfluss auf ihre Fähigkeit, sich erneut mit ihrem Browser-Cookie anzumelden.

Durch die Authentifizierung mit dem Sitzungscookie wird die Cookie-Dauer nicht auf eine weitere Stunde zurückgesetzt. Benutzer müssen sich erneut anmelden, wenn sie mehr als eine Stunde nach ihrer letzten erfolgreichen interaktiven Authentifizierung versuchen, auf Ihre Anmeldeseiten zuzugreifen.

Bestätigung von Benutzerkonten und Überprüfung der Benutzerattribute

Für lokale Benutzer im Benutzerpool funktionieren die verwaltete Anmeldung und die gehostete Benutzeroberfläche am besten, wenn Sie Ihren Benutzerpool so konfigurieren, dass Cognito automatisch Nachrichten zur Überprüfung und Bestätigung sendet. 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, wird auf Ihren Anmeldeseiten nach der Registrierung eine Fehlermeldung angezeigt. 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.

Ihre Änderungen an der Konfiguration werden angezeigt

Wenn Sie Stiländerungen an Ihren Seiten vornehmen und diese nicht sofort angezeigt werden, warten Sie ein paar Minuten und aktualisieren Sie dann die Seite.

Benutzerpool-Tokens decodieren

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

TLS-Version

Für verwaltete Anmeldeseiten und gehostete Benutzeroberflächen ist eine Verschlüsselung bei der Übertragung erforderlich. Benutzerpool-Domänen, die von Amazon Cognito bereitgestellt werden, erfordern, dass die Browser der Benutzer eine TLS-Mindestversion von 1.2 aushandeln. Benutzerdefinierte Domains unterstützen Browserverbindungen mit TLS Version 1.2. Die gehostete Benutzeroberfläche (klassisch) erfordert kein TLS 1.2 für benutzerdefinierte Domänen, aber die neuere verwaltete Anmeldung erfordert TLS Version 1.2 sowohl für benutzerdefinierte Domänen als auch für Präfixdomänen. Da Amazon Cognito die Konfiguration Ihrer Domain-Services verwaltet, können Sie die TLS-Anforderungen Ihrer Benutzerpool-Domain nicht ändern.

CORS-Richtlinien

Weder die verwaltete Anmeldung noch die gehostete Benutzeroberfläche unterstützen benutzerdefinierte Ursprungsrichtlinien für Cross-Origin Resource Sharing (CORS). Eine CORS-Richtlinie würde Benutzer daran hindern, Authentifizierungsparameter in ihren Anfragen zu

übergeben. Implementieren Sie stattdessen eine CORS-Richtlinie in Ihrem Anwendungs-Frontend. Amazon Cognito gibt einen `Access-Control-Allow-Origin: *` Antwort-Header auf Anfragen an die folgenden Endpunkte zurück.

1. Token-Endpunkt
2. Widerrufen des Endpunkts
3. UserInfo-Endpunkt

Cookies

Die verwaltete Anmeldung und die gehostete Benutzeroberfläche 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 `csrf-state-legacy` Cookie für Sitzungskonsistenz, das von Amazon Cognito als Fallback gelesen wird, wenn Ihr Browser das Attribut nicht unterstützt. `SameSite`
- Ein `cognito` Sitzungscookie, das erfolgreiche Anmeldeversuche eine Stunde lang speichert.
- Ein `lang` Cookie, das die vom Benutzer gewählte Sprachlokalisierung bei der verwalteten Anmeldung beibehält.
- Ein `page-data` Cookie, das die erforderlichen Daten speichert, wenn ein Benutzer zwischen verwalteten Anmeldeseiten navigiert.

In iOS können Sie alle Cookies blockieren. Diese Einstellung ist nicht mit der verwalteten Anmeldung oder der gehosteten Benutzeroberfläche kompatibel. Um mit Benutzern zu arbeiten, die diese Einstellung möglicherweise aktivieren, bauen Sie die Benutzerpoolauthentifizierung in eine native iOS-App mit einem AWS SDK ein. In diesem Szenario können Sie Ihren eigenen Sitzungsspeicher erstellen, der nicht auf Cookies basiert.

Auswirkungen der Änderung der verwalteten Anmeldeversion

Beachten Sie die folgenden Auswirkungen des Hinzufügens von Domänen und der Festlegung der Version für die verwaltete Anmeldung.

- Wenn Sie eine Präfix-Domain hinzufügen, entweder mit verwaltetem Login oder mit gehosteter Benutzeroberfläche (klassisch), kann es bis zu 60 Sekunden dauern, bis Ihre Anmeldeseiten verfügbar sind.
- Wenn Sie eine benutzerdefinierte Domain hinzufügen, entweder mit verwaltetem Login oder mit gehosteter Benutzeroberfläche (klassisch), kann es bis zu fünf Minuten dauern, bis Ihre Anmeldeseiten verfügbar sind.
- Wenn Sie die Branding-Version für Ihre Domain ändern, kann es bis zu vier Minuten dauern, bis Ihre Anmeldeseiten in der neuen Branding-Version verfügbar sind.
- Wenn Sie zwischen verwalteter Anmeldung und gehostetem UI-Branding (klassisch) wechseln, verwaltet Amazon Cognito keine Benutzersitzungen. Sie müssen sich erneut mit der neuen Oberfläche anmelden.

Standardstil

Wenn Sie in der einen App-Client erstellen AWS Management Console, weist Amazon Cognito Ihrem App-Client automatisch einen Branding-Stil zu. Wenn Sie mit dem CreateUserPoolClientVorgang programmgesteuert einen App-Client erstellen, wird kein Branding-Stil erstellt. Die verwaltete Anmeldung ist für einen App-Client, der mit einem AWS SDK erstellt wurde, erst verfügbar, wenn Sie einen mit einer CreateManagedLoginBrandingAnfrage erstellen.

Bei der Aufforderung zur verwalteten Anmeldeauthentifizierung läuft das Zeitlimit

Amazon Cognito storniert Authentifizierungsanfragen, die nicht innerhalb von fünf Minuten abgeschlossen werden, und leitet den Benutzer zur verwalteten Anmeldung weiter. Für die Seite wird eine `Something went wrong`-Fehlermeldung angezeigt.

Konfigurieren einer Benutzerpool-Domäne

Die Konfiguration einer Domain ist ein optionaler Teil der Einrichtung eines Benutzerpools. Eine Benutzerpool-Domain hostet Funktionen für die Benutzerauthentifizierung, den Verbund mit Drittanbietern und OpenID Connect (OIDC) -Flows. Es verfügt über eine verwaltete Anmeldung, eine vorgefertigte Oberfläche für wichtige Operationen wie Registrierung, Anmeldung und Kennwortwiederherstellung. Es hostet auch die standardmäßigen OpenID Connect (OIDC) -Endpunkte wie authorize, UserInfo und token für machine-to-machine (M2M) -Autorisierung und andere OIDC- und 2.0-Authentifizierungs- und Autorisierungsabläufe. OAuth

Benutzer authentifizieren sich mit verwalteten Anmeldeseiten in der Domain, die Ihrem Benutzerpool zugeordnet ist. Sie haben zwei Möglichkeiten, diese Domain zu konfigurieren: Sie können

entweder die von Amazon Cognito gehostete Standarddomain verwenden oder Sie können eine benutzerdefinierte Domain konfigurieren, die Ihnen gehört.

Die benutzerdefinierte Domain-Option bietet mehr Optionen für Flexibilität, Sicherheit und Kontrolle. Beispielsweise kann eine vertraute, unternehmenseigene Domain das Vertrauen der Benutzer stärken und den Anmeldevorgang intuitiver gestalten. Der Ansatz für benutzerdefinierte Domänen erfordert jedoch zusätzlichen Aufwand, z. B. die Verwaltung des SSL-Zertifikats und der DNS-Konfiguration.

Die OIDC-Discovery-Endpunkte `/.well-known/openid-configuration` für Endgeräte URLs und `/.well-known/jwks.json` Tokensignaturschlüssel werden nicht auf Ihrer Domain gehostet. Weitere Informationen finden Sie unter Identitätsanbieter und Endpunkte der vertrauenden Partei.

Zu verstehen, wie Sie die Domain für Ihren Benutzerpool konfigurieren und verwalten, ist ein wichtiger Schritt bei der Integration der Authentifizierung in Ihre Anwendung. Die Anmeldung mit der Benutzerpools-API und einem AWS SDK kann eine Alternative zur Konfiguration einer Domain sein. Das API-basierte Modell stellt Token direkt in einer API-Antwort bereit. Für Implementierungen, die die erweiterten Funktionen von Benutzerpools als OIDC-IdP nutzen, müssen Sie jedoch eine Domain konfigurieren. Weitere Informationen zu den Authentifizierungsmodellen, die in Benutzerpools verfügbar sind, finden Sie unter. Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten

Themen

- Wissenswertes über Benutzerpool-Domänen
- Verwenden der Amazon Cognito-Präfixdomäne für die verwaltete Anmeldung
- Verwenden Sie Ihre eigene Domain für die verwaltete Anmeldung

Wissenswertes über Benutzerpool-Domänen

Benutzerpool-Domänen sind eine Anlaufstelle für OIDC-vertrauende Parteien in Ihren Anwendungen und für Benutzeroberflächenelemente. Beachten Sie die folgenden Details, wenn Sie die Implementierung einer Domäne für Ihren Benutzerpool planen.

Vorbehaltene Bedingungen

Sie können den Text `aws` oder nicht `cognito` im Namen einer Amazon Cognito-Präfix-Domain verwenden. `amazon`

Discovery-Endpunkte befinden sich auf einer anderen Domain

Die Discovery-Endpunkte `.well-known/openid-configuration` des Benutzerpools befinden sich `.well-known/jwks.json` nicht in Ihrer benutzerdefinierten Benutzerpool-Domäne oder Präfixdomäne. Der Pfad zu diesen Endpunkten lautet wie folgt.

- `https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/.well-known/openid-configuration`
- `https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/.well-known/jwks.json`

Zeitpunkt des Inkrafttretens von Domainänderungen

Es kann bis zu einer Minute dauern, bis Amazon Cognito die Branding-Version einer Präfix-Domain startet oder aktualisiert. Es kann bis zu fünf Minuten dauern, bis Änderungen an einer benutzerdefinierten Domain übernommen werden. Die Verbreitung neuer benutzerdefinierter Domains kann bis zu einer Stunde dauern.

Benutzerdefinierte Domains und Präfix-Domains gleichzeitig

Sie können einen Benutzerpool mit einer benutzerdefinierten Domäne und einer Präfixdomäne einrichten, deren Eigentümer ist AWS. Da die Endpunkte für die Benutzerpool-Erkennung in einer anderen Domäne gehostet werden, bedienen sie nur die benutzerdefinierte Domäne. Sie `openid-configuration` geben beispielsweise einen einzigen Wert für `"authorization_endpoint"` of an`"https://auth.example.com/oauth2/authorize"`.

Wenn Sie sowohl benutzerdefinierte Domänen als auch Präfixdomänen in einem Benutzerpool haben, können Sie die benutzerdefinierte Domäne mit allen Funktionen eines OIDC-Anbieters verwenden. Die Präfixdomäne in einem Benutzerpool mit dieser Konfiguration hat weder Erkennungs- noch token-signing-key Endpunkte und sollte entsprechend verwendet werden.

Benutzerdefinierte Domänen werden als ID der vertrauenden Partei für den Hauptschlüssel bevorzugt

Wenn Sie die Benutzerpoolauthentifizierung mit Hauptschlüsseln einrichten, müssen Sie eine Relying Party (RP) -ID festlegen. Wenn Sie eine benutzerdefinierte Domain und eine Präfix-Domain haben, können Sie die RP-ID nur als Ihre benutzerdefinierte Domain festlegen. Um eine Präfix-Domain als RP-ID in der Amazon Cognito Cognito-Konsole festzulegen, löschen Sie Ihre benutzerdefinierte Domain oder geben Sie den vollqualifizierten Domainnamen (FQDN) der Präfix-Domain als Drittanbieter-Domain ein.

Verwenden Sie keine benutzerdefinierten Domains auf verschiedenen Ebenen Ihrer Domain-Hierarchie

Sie können separate Benutzerpools so konfigurieren, dass sie benutzerdefinierte Domains in derselben Top-Level-Domain (TLD) haben, zum Beispiel auth.example.com und auth2.example.com. Das Cookie für die verwaltete Anmeldesitzung ist für eine benutzerdefinierte Domain und alle Subdomains gültig, z. B. *.auth.example.com. Aus diesem Grund sollte kein Benutzer Ihrer Anwendungen auf die verwaltete Anmeldung für eine übergeordnete Domain und Subdomain zugreifen. Wenn benutzerdefinierte Domains dieselbe TLD verwenden, behalten Sie sie auf derselben Subdomain-Ebene bei.

Angenommen, Sie haben einen Benutzerpool mit der benutzerdefinierten Domain auth.example.com. Dann erstellen Sie einen weiteren Benutzerpool und weisen die benutzerdefinierte Domain uk.auth.example.com zu. . Ein Benutzer meldet sich mit auth.example.com an. und ruft ein Cookie ab, das sein Browser auf jeder Website im Platzhalterpfad *.auth.example.com präsentiert. Sie versuchen dann, sich bei uk.auth.example.com anzumelden. . Sie übergeben ein ungültiges Cookie an Ihre Benutzerpool-Domain und erhalten statt einer Anmeldeaufforderung eine Fehlermeldung. Im Gegensatz dazu hat ein Benutzer mit einem Cookie für *.auth.example.com keine Probleme, eine Anmeldesitzung auf auth2.example.com zu starten.

Branding-Version

Wenn Sie eine Domain erstellen, legen Sie eine Branding-Version fest. Ihre Optionen sind das neuere verwaltete Anmeldeerlebnis und das klassische gehostete Benutzeroberflächenerlebnis. Diese Auswahl gilt für alle App-Clients, die Dienste in Ihrer Domain hosten.

Verwenden der Amazon Cognito-Präfixdomäne für die verwaltete Anmeldung

Das Standarderlebnis für die verwaltete Anmeldung wird auf einer Domain gehostet, die Eigentümer ist. AWS Dieser Ansatz hat eine niedrige Eintrittsbarriere — wählen Sie einen Präfixnamen und er ist aktiv —, verfügt aber nicht über die vertrauenserweckenden Funktionen einer benutzerdefinierten Domain. Es gibt keinen Kostenunterschied zwischen der Amazon Cognito Cognito-Domain-Option und der benutzerdefinierten Domain-Option. Der einzige Unterschied besteht in der Domain in der Webadresse, zu der Sie Ihre Benutzer weiterleiten. Bei IdP-Weiterleitungen von Drittanbietern und beim Fluss von Kundenanmeldedaten hat die gehostete Domain kaum sichtbare Auswirkungen. Eine benutzerdefinierte Domain eignet sich besser für Fälle, in denen sich Ihre Benutzer mit verwaltetem Login anmelden und mit einer Authentifizierungsdomäne interagieren würden, die nicht mit der Anwendungsdomäne übereinstimmt.

Die gehostete Amazon Cognito Cognito-Domain hat ein Präfix Ihrer Wahl, wird jedoch in der Stammdomain `amazoncognito.com` gehostet. Im Folgenden wird ein Beispiel gezeigt:

```
https://cognitoexample.auth.ap-south-1.amazoncognito.com
```

Alle Präfix-Domains folgen diesem Format:*`prefix`*. `auth`. *`AWS-Region code`*. `amazoncognito.com`. Benutzerdefinierte Domain-Benutzerpools können die verwalteten Anmelde- oder gehosteten UI-Seiten auf jeder Domain hosten, die Sie besitzen.

> **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 Domänen des Benutzerpools haben die folgenden Formate.
>
> ```
> auth.Region.amazoncognito.com
> auth-fips.Region.amazoncognito.com
> ```

Informationen zum Hinzufügen eines App-Clients und einer Benutzerpool-Domäne mit dem AWS Management Console finden Sie unterErstellen eines App-Clients.

Themen

- Voraussetzungen
- Konfigurieren Sie ein Amazon Cognito Cognito-Domainpräfix
- Überprüfen Sie Ihre Anmeldeseite

Voraussetzungen

Bevor Sie anfangen, benötigen Sie:

- Einen Benutzerpool mit einem App-Client. Weitere Informationen finden Sie unter Erste Schritte mit Benutzerpools.

Konfigurieren Sie ein Amazon Cognito Cognito-Domainpräfix

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 unter Branding zum Domain-Menü.
2. Wählen Sie neben Domain die Option Actions und anschließend Create Cognito domain aus. Wenn Sie bereits eine Benutzerpool-Präfixdomäne konfiguriert haben, wählen Sie Cognito-Domäne löschen, bevor Sie Ihre neue benutzerdefinierte Domäne erstellen.
3. Geben Sie ein verfügbares Domain-Präfix zur Verwendung mit einer Amazon-Cognito-Domäne ein. Informationen zum Einrichten einer benutzerdefinierten Domain finden Sie unterVerwenden Sie Ihre eigene Domain für die verwaltete Anmeldung.
4. Wählen Sie eine Branding-Version. Ihre Branding-Version gilt für alle benutzerinteraktiven Seiten in dieser Domain. Ihr Benutzerpool kann entweder verwaltetes Login oder gehostetes UI-Branding für alle App-Clients hosten.

   > **Note**
   >
   > Sie können eine benutzerdefinierte Domain und eine Präfix-Domain haben, aber Amazon Cognito bedient nur den `/.well-known/openid-configuration` Endpunkt für die benutzerdefinierte Domain.

5. 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> --managed-login-version 2`

- API-Betrieb für Benutzerpools: 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>`
- API-Betrieb für Benutzerpools: 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>`
- API-Betrieb für Benutzerpools: DeleteUserPoolDomain

Überprüfen Sie Ihre 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 Sie Ihre eigene Domain für die verwaltete Anmeldung

Nachdem Sie einen App-Client eingerichtet haben, können Sie Ihren Benutzerpool mit einer benutzerdefinierten Domäne für die Domänendienste von Managed Login konfigurieren. Mit einer benutzerdefinierten Domain können sich Benutzer mit Ihrer eigenen Webadresse statt mit der `amazoncognito.com` Standardpräfixdomäne bei Ihrer Anwendung anmelden. Benutzerdefinierte Domänen verbessern das Vertrauen der Benutzer in Ihre Anwendung mit einem vertrauten Domainnamen, insbesondere wenn die Stammdomäne mit der Domain übereinstimmt, die

Ihre Anwendung hostet. Benutzerdefinierte Domänen können die Einhaltung organisatorischer Sicherheitsanforderungen verbessern.

Für eine benutzerdefinierte Domain sind einige Voraussetzungen erforderlich, darunter ein Benutzerpool, ein App-Client und eine Webdomäne, die Sie besitzen. Für benutzerdefinierte Domains ist außerdem ein SSL-Zertifikat für die benutzerdefinierte Domain erforderlich, die mit AWS Certificate Manager (ACM) in USA Ost (Nord-Virginia) verwaltet wird. Amazon Cognito erstellt eine CloudFront Amazon-Distribution, die während des Transports mit Ihrem ACM-Zertifikat gesichert ist. Da Sie Eigentümer der Domain sind, müssen Sie einen DNS-Eintrag erstellen, der den Datenverkehr an die CloudFront Distribution für Ihre benutzerdefinierte Domain weiterleitet.

Sobald diese Elemente fertig sind, können Sie die benutzerdefinierte Domain über die Amazon Cognito Cognito-Konsole oder API zu Ihrem Benutzerpool hinzufügen. Dazu müssen Sie den Domainnamen und das SSL-Zertifikat angeben und anschließend Ihre DNS-Konfiguration mit dem angegebenen Alias-Ziel aktualisieren. Nachdem Sie diese Änderungen vorgenommen haben, können Sie überprüfen, ob die Anmeldeseite in Ihrer benutzerdefinierten Domain zugänglich ist.

Der einfachste Weg, eine benutzerdefinierte Domain zu erstellen, ist eine öffentlich gehostete Zone in Amazon Route 53. Die Amazon Cognito Cognito-Konsole kann in wenigen Schritten die richtigen DNS-Einträge erstellen. Bevor Sie beginnen, sollten Sie erwägen, eine Route 53-Hosting-Zone für eine Domain oder Subdomain zu erstellen, die Sie besitzen.

Themen

- Hinzufügen einer benutzerdefinierten Domäne zu einem Benutzerpool
- Voraussetzungen
- Schritt 1: Eingeben des benutzerdefinierten Domänennamens
- Schritt 2: Hinzufügen eines Alias-Ziels und einer Subdomäne
- Schritt 3: Überprüfen Ihrer Anmeldeseite
- 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. Um versehentliche Auswirkungen auf die Kundeninfrastruktur zu verhindern, unterstützt Amazon Cognito die Verwendung von Top-Level-Domains (TLDs) für benutzerdefinierte Domains nicht. Weitere Informationen finden Sie unter Domänennamen.
- Die Möglichkeit zum Erstellen einer Unterdomäne für die benutzerdefinierte Domäne. Wir empfehlen Auth für Ihren Subdomainnamen. 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 öffentliches SSL/TLS Zertifikat, das von ACM in USA East (Nord-Virginia) verwaltet wird. Das Zertifikat muss in us-east-1 sein, da das Zertifikat mit einer Verteilung in CloudFront einem globalen Dienst verknüpft wird.
- Browserclients, die Server Name Indication (SNI) unterstützen. Die CloudFront Verteilung, die Amazon Cognito benutzerdefinierten Domains zuweist, erfordert SNI. Sie können diese Einstellung nicht ändern. Weitere Informationen zu SNI in CloudFront Distributionen finden Sie unter Verwenden von SNI zur Bearbeitung von HTTPS-Anfragen im Amazon CloudFront Developer Guide.
- Eine Anwendung, die es Ihrem Benutzerpool-Autorisierungsserver ermöglicht, Cookies zu Benutzersitzungen hinzuzufügen. Amazon Cognito setzt mehrere erforderliche Cookies für verwaltete Anmeldeseiten. 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 verwaltete Anmelde-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 bei der Größenbeschränkung zu vermeiden, empfehlen wir, dass Ihre Anwendung keine Cookies auf der Subdomain setzt, die die Domänendienste Ihres Benutzerpools hostet.

- Erlaubnis zur Aktualisierung von CloudFront Amazon-Distributionen. Fügen Sie hierzu die folgende IAM-Richtlinienanweisung einem Benutzer in Ihrem AWS-Konto an:

  JSON

```
{
    "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. Navigieren Sie unter Branding zum Domain-Menü.
2. Navigieren Sie dann zu Domäne und wählen Sie Aktionen, Benutzerdefinierte Domäne erstellen oder Amazon-Cognito-Domäne erstellen aus. Wenn Sie bereits eine benutzerdefinierte Domäne für den Benutzerpool konfiguriert haben, wählen Sie Benutzerdefinierte Domäne löschen aus, bevor Sie Ihre neue benutzerdefinierte Domäne erstellen.
3. Wählen Sie neben Domain die Option Aktionen und anschließend Benutzerdefinierte Domain erstellen aus. Wenn Sie bereits eine benutzerdefinierte Domain konfiguriert haben, wählen Sie Benutzerdefinierte Domain löschen, um die bestehende Domain zu löschen, bevor Sie Ihre neue benutzerdefinierte Domain erstellen.
4. 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.
5. 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.
6. Wählen Sie eine Branding-Version. Ihre Branding-Version gilt für alle benutzerinteraktiven Seiten in dieser Domain. Ihr Benutzerpool kann entweder verwaltetes Login oder gehostetes UI-Branding für alle App-Clients hosten.

   > **Note**
   >
   > Sie können eine benutzerdefinierte Domain und eine Präfix-Domain haben, aber Amazon Cognito bedient nur den `/.well-known/openid-configuration` Endpunkt für die benutzerdefinierte Domain.
7. Wählen Sie Erstellen aus.

8. Amazon Cognito bringt Sie zum Domain-Menü 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

Der folgende CreateUserPoolDomainAnfragetext erstellt eine benutzerdefinierte Domain.

```
{
   "Domain": "auth.example.com",
   "UserPoolId": "us-east-1_EXAMPLE",
   "ManagedLoginVersion": 2,
   "CustomDomainConfig": {
    "CertificateArn": "arn:aws:acm:us-east-1:111122223333:certificate/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
   }
}
```

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. Wenn Sie dazu aufgefordert werden, geben Sie Ihre AWS Anmeldeinformationen ein.

2. Wenn Sie in Route 53 keine öffentlich gehostete Zone haben, erstellen Sie eine Zone mit einem Stamm, der Ihrer benutzerdefinierten Domäne übergeordnet ist. Weitere Informationen finden Sie unter Creating a public hosted zone im Amazon Route 53 Developer Guide.

   a. Wählen Sie Create Hosted Zone.

   b. Geben Sie beispielsweise `auth.example.com` die übergeordnete Domain Ihrer benutzerdefinierten Domain aus der Domainnamenliste ein. `myapp.auth.example.com`

   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 Hosting-Zone für Ihre benutzerdefinierte Domain mit einer Delegierung in der übergeordneten Hosting-Zone erstellen, die Anfragen an die Host-Zone der Subdomain 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. Erstellen Sie einen DNS-Eintrag für die übergeordnete Domain mit den folgenden Eigenschaften:

   - Name des Eintrags: Leer lassen.
   - Datensatztyp:A.
   - Alias: Nicht aktivieren.
   - Wert: Geben Sie ein Ziel Ihrer Wahl ein. Dieser Datensatz muss etwas ergeben, aber der Wert des Datensatzes spielt für Amazon Cognito keine Rolle.
   - TTL: Stellen Sie Ihre bevorzugte TTL ein oder belassen Sie sie als Standard.

- Routing-Richtlinie: Wählen Sie Simple Routing.

5. Wählen Sie Create records (Datensätze erstellen). Im Folgenden finden Sie einen Beispieldatensatz für die Domain*example.com*:

```
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.

6. Fügen Sie einen weiteren DNS-Eintrag für Ihre benutzerdefinierte Domain mit den folgenden Eigenschaften hinzu:

   - Datensatzname: Ihr benutzerdefiniertes Domainpräfix, `auth` um beispielsweise einen Eintrag für zu erstellen`auth.example.com`.
   - Datensatztyp:`A`.
   - Alias: Aktivieren.
   - Traffic weiterleiten an: Wählen Sie Alias to Cloudfront Distribution. Geben Sie beispielsweise `123example.cloudfront.net` das Alias-Ziel ein, das Sie zuvor aufgezeichnet haben.
   - Routing-Richtlinie: Wählen Sie Simple Routing.

7. Wählen Sie Create records (Datensätze erstellen).

> **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 einer benutzerdefinierten Domain *example.com* mit der *auth* Subdomain:

  ```
  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 das Domain-Menü.
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“.

Branding auf verwaltete Anmeldeseiten anwenden

Möglicherweise möchten Sie für ein einheitliches Benutzererlebnis zwischen Ihrem Authentifizierungsdienst und Ihrer Anwendung sorgen. Sie können dieses Ziel entweder mit benutzerdefinierten Formularen und Back-End-API-Vorgängen in einem AWS SDK oder mit verwalteter Anmeldung erreichen. Die verwaltete Anmeldung und die klassische gehostete Benutzeroberfläche sind Web-Frontends für die Komponente Ihrer Anwendung, die die Authentifizierung mit Benutzerpools ermöglicht. Um Ihre verwalteten Authentifizierungsdienste mit der UX Ihrer Anwendung zu synchronisieren, haben Sie zwei Anpassungsoptionen: den Branding-Editor und das gehostete UI-Branding. Sie können Ihr bevorzugtes Erlebnis in der Amazon Cognito Cognito-Konsole und bei API-Vorgängen für den Benutzerpool wählen.

Der Branding-Editor

Der Branding-Editor ist die neueste Anpassungsoption für die Benutzeroberfläche der neuesten Benutzerpools, verwaltete Anmeldung. Der Branding-Editor ist ein visueller Editor ohne Code für verwaltete Anmelde-Assets und -Stile sowie eine Reihe von API-Operationen für die

programmatische Konfiguration einer Vielzahl von Konfigurationsoptionen. Benutzerpools, die Sie mit einer Domain und einer verwalteten Anmeldung konfigurieren, rendern automatisch die Branding-Designer-Version Ihrer Anmeldeseiten.

Branding für gehostete Benutzeroberflächen (klassisch)

Das Branding-Erlebnis für gehostete Benutzeroberflächen (klassisch) bietet zwei Optionen: das Ändern einer CSS-Datei (Cascading Stylesheets) mit einem festen Satz von Stiloptionen und das Hinzufügen eines benutzerdefinierten Logobilds. Sie können diese Optionen in der Amazon Cognito Cognito-Konsole oder mit dem Vorgang „UICustomizationAPI einrichten“ festlegen. Zum Zeitpunkt der Einführung des Dienstes hatte Amazon Cognito nur diese Option. Benutzerpools, die Sie mit einer Domain und der gehosteten UI-Branding-Version konfigurieren, rendern automatisch die klassische Version Ihrer Anmeldeseiten. Ihr Featureplan unterstützt möglicherweise auch nur die gehostete Benutzeroberfläche.

> **Note**
>
> Der Branding-Editor und das klassische Branding-Erlebnis ändern die visuellen Eigenschaften Ihres gehosteten Authentifizierungsdienstes. Derzeit können Sie den Text, der auf Ihren verwalteten Anmeldeseiten angezeigt wird, nicht ändern, es sei denn, Sie können die Lokalisierung in eine von mehreren Sprachen anwenden. Weitere Informationen zur Lokalisierung finden Sie unterLokalisierung verwalteter Logins.

Wählen Sie ein Branding-Erlebnis und weisen Sie Stile zu

In der Amazon Cognito Cognito-Konsole wird für neue Benutzerpools standardmäßig das Branding-Erlebnis Managed Login verwendet. Benutzerpools, die Sie eingerichtet haben, bevor die verwaltete Anmeldung verfügbar war, erhalten das Branding Hosted UI (klassisch). Sie können zwischen verwalteter Anmeldung und gehostetem UI-Branding wechseln. Wenn Sie Ihre Branding-Version ändern, wendet Amazon Cognito die Änderung sofort auf die benutzerinteraktiven Seiten Ihrer Benutzerpool-Domain an. Mit der verwalteten Anmeldung und der gehosteten Benutzeroberfläche kann Ihr Benutzerpool für jeden App-Client einen eigenen Stil haben.

Jeder App-Client kann einen eigenen Branding-Stil haben, aber eine Benutzerpool-Domain dient entweder der verwalteten Anmeldung oder der gehosteten Benutzeroberfläche. Ein Stil ist ein Satz von Anpassungseinstellungen, die auf einen App-Client angewendet werden. Sie können eine benutzerdefinierte Domäne und eine Präfixdomäne pro Benutzerpool einrichten. Sie können Ihren

benutzerdefinierten Domains und Präfix-Domains unterschiedliche Branding-Versionen zuweisen. Eine Präfixdomäne ist jedoch nicht voll funktionsfähig, wenn Sie auch über eine benutzerdefinierte Domain verfügen — die `.well-known` OIDC-Discovery-Endpunkte bieten nur benutzerdefinierte Domänenpfade. Sie können die Präfixdomäne nur für Operationen verwenden, für die keine Endpoint Discovery (`openid-configuration`) in einem Benutzerpool mit dieser Konfiguration erforderlich ist. Aufgrund dieser Eigenschaften von Benutzerpools können Sie effektiv eine Branding-Version pro Benutzerpool auswählen.

Sie können den App-Clients in einem Benutzerpool, in dem für eine Domain die Branding-Version mit verwalteter Anmeldung festgelegt ist, Stile zuweisen. Stile sind eine Reihe von visuellen Einstellungen, die sich aus Bilddateien, Anzeigeoptionen und CSS-Werten zusammensetzen. Wenn Sie einem App-Client einen Stil zuweisen, überträgt Amazon Cognito Ihre Aktualisierungen sofort auf Ihre benutzerinteraktiven Anmeldeseiten. Amazon Cognito rendert Ihre benutzerinteraktiven Seiten mit der von Ihnen ausgewählten Branding-Version und den Anpassungen, die Sie darauf vorgenommen haben.

Stile aktualisieren und löschen

Wenn Sie einen Stil erstellen, verknüpfen Sie ihn mit einem App-Client. Um eine Stilzuweisung für einen App-Client zu ändern, müssen Sie zuerst den ursprünglichen Stil löschen. Derzeit können Sie keine Einstellungen zwischen Stilen kopieren. Sie müssen dies programmgesteuert tun. Um Einstellungen zwischen Stilen und App-Clients zu replizieren, rufen Sie die Einstellungen für einen Stil mit der DescribeManagedLoginBrandingAPI-Operation ab und wenden Sie sie mit oder an. CreateManagedLoginBrandingUpdateManagedLoginBranding Sie können die einem App-Client zugewiesenen Stile nicht ändern — Sie können nur das Original löschen und einen neuen festlegen. Weitere Informationen zur Verwaltung von Stilen mit API- und SDK-Vorgängen finden Sie unter. API- und SDK-Operationen für verwaltetes Login-Branding

> **Note**
>
> Programmatische Anfragen, die einen Branding-Stil erstellen oder aktualisieren, dürfen eine Anforderungsgröße von nicht mehr als 2 MB haben. Wenn Ihre Anfrage diesen Grenzwert überschreitet, teilen Sie Ihre Anfrage in mehrere `UpdateManagedLoginBranding` Anfragen für Parametergruppen auf, die die maximale Anforderungsgröße nicht überschreiten. Diese Anfragen führen nicht dazu, dass nicht spezifizierte Parameter auf die Standardwerte gesetzt werden, sodass Sie Teilanfragen senden können, ohne dass dies Auswirkungen auf die vorhandenen Einstellungen hat.

Sie löschen einen Stil in der Amazon Cognito Cognito-Konsole aus dem Menü Verwaltete Anmeldung. Wählen Sie unter Stile den Stil aus, den Sie löschen möchten, und wählen Sie Stil löschen aus.

Auf oberster Ebene besteht der Prozess der Zuordnung des Brandings zu einer Domain aus den folgenden Schritten.

1. Erstellen Sie eine Domain und legen Sie die Branding-Version fest.
2. Erstellen Sie einen Branding-Stil und weisen Sie ihn einem App-Client zu.

Um einem App-Client einen Stil zuzuweisen

1. Erstellen Sie im Domain-Menü Ihres Benutzerpools eine Domain und setzen Sie die Branding-Version auf Verwaltete Anmeldung.
2. Navigieren Sie zum Menü Verwaltete Anmeldung. Wählen Sie unter Styles die Option Create a style aus.
3. Wählen Sie den App-Client aus, dem Sie Ihren Stil zuweisen möchten, oder erstellen Sie einen neuen App-Client.
4. Um mit der Konfiguration Ihrer Branding-Einstellungen zu beginnen, wählen Sie Branding-Editor starten.

Themen

- Der Branding-Editor und die Anpassung der verwalteten Anmeldung
- Anpassen des Brandings für gehostete Benutzeroberflächen (klassisch)

Der Branding-Editor und die Anpassung der verwalteten Anmeldung

Der Branding-Editor ist ein visuelles Design- und Bearbeitungstool für Ihre verwalteten Login-Webseiten. Es ist in die Amazon Cognito Cognito-Konsole integriert. Im Branding-Editor beginnen Sie mit einer Vorschau Ihrer Anmeldeseiten und können dann zu einer Schnelleinrichtungsoption oder einer Detailansicht mit erweiterten Optionen übergehen. Sie können Stilparameter ändern und eine Vorschau anzeigen oder ein benutzerdefiniertes Hintergrundbild und Logo hinzufügen. Sie können den Hellmodus und den Dunkelmodus konfigurieren.

Erstellen Sie zunächst einen Stil, den Sie auf Ihren Benutzerpool oder einen App-Client anwenden können.

Um mit dem Branding-Editor zu beginnen

1. Erstellen Sie über den Domain-Tab eine Domain oder aktualisieren Sie Ihre bestehende Domain. Stellen Sie unter Branding-Version Ihre Domain so ein, dass sie die verwaltete Anmeldung verwendet.
2. Löschen Sie den vorhandenen App-Client-Stil, falls vorhanden.
   a. Wählen Sie im Menü App-Clients Ihren App-Client aus.
   b. Wählen Sie unter Verwalteter Anmeldestil den Stil aus, der Ihrem App-Client zugewiesen ist.
   c. Wählen Sie Stil löschen. Bestätigen Sie Ihre Auswahl.
3. Navigieren Sie in Ihrem Benutzerpool zum Menü Verwaltete Anmeldung. Falls Sie dies noch nicht getan haben, folgen Sie der Aufforderung, um einen Funktionsplan auszuwählen, der eine verwaltete Anmeldung beinhaltet. Sie können auch „Diese Funktion in der Vorschau anzeigen“ auswählen, wenn Sie den Branding-Editor auschecken möchten, ohne Änderungen vorzunehmen.
4. Wählen Sie unter „Stile“ die Option „Stil erstellen“ aus.
5. Wählen Sie den App-Client aus, dem Sie Ihren Stil zuweisen möchten, und wählen Sie Erstellen aus. Sie können auch einen neuen App-Client erstellen.
6. Die Amazon Cognito Cognito-Konsole startet den Branding-Editor.
7. Wählen Sie eine Registerkarte aus, auf der Sie mit der Bearbeitung beginnen möchten, oder wählen Sie Editor starten und rufen Sie die Schnellkonfiguration auf. Die folgenden Tabs sind verfügbar:

   Vorversion

   Sehen Sie sich auf Ihren verwalteten Anmeldeseiten an, wie Ihre aktuellen Auswahlen aussehen.

   Grundlage

   Lege ein allgemeines Design fest, konfiguriere Links zu externen Identitätsanbietern und gestalte Formularfelder.

   Komponenten

   Konfigurieren Sie Stile für Kopf- und Fußzeilen und einzelne Benutzeroberflächenelemente.

8. Rufen Sie Quick Setup auf, um Entscheidungen über die Grundeinstellungen zu treffen. Wählen Sie die Kategorie Einstellungen ändern und anschließend Schnelleinrichtung aus. Wenn Sie Weiter auswählen, wird der Branding-Editor mit einer Reihe grundlegender Optionen gestartet, die Sie konfigurieren müssen.

Text und Lokalisierung

Sie können Text im Branding-Editor nicht ändern oder lokalisieren. Fügen Sie stattdessen der URL, die Sie an Benutzer verteilen, einen `lang` Abfrageparameter hinzu. Dieser Parameter bewirkt, dass Ihre verwalteten Anmeldeseiten in eine von mehreren verfügbaren Sprachen lokalisiert werden. Weitere Informationen finden Sie unter Lokalisierung verwalteter Logins.

Quick Setup

Mit der Schaltfläche Branding-Editor starten wird ein visueller Editor für Ihre verwaltete Anmeldekonfiguration geladen, in dem Sie aus einer Vielzahl von primären Anpassungsoptionen auswählen können. Während Sie eine Auswahl treffen, rendert Amazon Cognito Ihre verwalteten Anmeldeänderungen in einem Vorschaufenster. Um zum Menü mit den detaillierten Einstellungen zurückzukehren, wählen Sie die Schaltfläche „Einstellungskategorie ändern“.

Wie sollte das Gesamtbild aussehen und sich anfühlen?

Konfigurieren Sie die grundlegenden Theme-Einstellungen für die verwaltete Anmeldung.

Anzeigemodus

Wählen Sie für Ihre verwaltete Anmeldung einen Lichtmodus, einen Dunkelmodus oder ein adaptives Erlebnis. Die adaptiven Einstellungen richten sich nach den Browsereinstellungen des Benutzers, wenn Amazon Cognito die verwaltete Anmeldung rendert. Wenn Sie einen browseradaptiven Modus wählen, können Sie verschiedene Farben und Logobilder für den hellen und dunklen Modus wählen.

Spacing

Legen Sie den Standardabstand zwischen Elementen auf der Seite fest.

Radius des Randes

Legt die Rundungstiefe des äußeren Randes von Elementen fest.

Wie sollte der Seitenhintergrund aussehen?

Typ des Hintergrunds

Das Kontrollkästchen Bild anzeigen gibt an, ob Sie ein Hintergrundbild oder eine einfarbige Hintergrundfarbe verwenden möchten.

1. Um ein Bild zu verwenden, wählen Sie „Bild anzeigen“ und wählen Sie ein Hintergrundbild für den hellen und dunklen Modus. Sie können auch eine Seitenhintergrundfarbe im Dunkel- und Hellmodus für Bereiche des Hintergrunds festlegen, die nicht vom Bild verdeckt werden.
2. Wenn Sie nur eine Farbe für den Hintergrund verwenden möchten, deaktivieren Sie die Option „Bild anzeigen“ und wählen Sie eine Seitenhintergrundfarbe im Hell- und Dunkelmodus.

Wie sollten Formulare aussehen?

Konfigurieren Sie die Einstellungen für die Formularelemente der verwalteten Anmeldung. Beispiele für Formularelemente sind Anmelde- und Codeaufforderungen.

Horizontale Ausrichtung

Legen Sie die horizontale Ausrichtung der Formularfelder fest.

Das Logo des Formulars

Legen Sie die Positionierung Ihres Logo-Bildes fest.

Logo-Bild

Wählen Sie eine Logo-Bilddatei aus, die in das Formularelement für den Hell- und Dunkelmodus aufgenommen werden soll. Um ein Bild hochzuladen, wählen Sie das Drop-down-Menü Logobild aus, wählen Sie Neues Asset hinzufügen und fügen Sie eine Logodatei hinzu.

Primäre Markenfarbe

Lege eine Designfarbe für den hellen und dunklen Modus fest. Diese Farbe wird als Hintergrundfarbe auf alle Elemente angewendet, die als primär klassifiziert sind.

Wie sollten Überschriften aussehen?

Wählen Sie aus, ob Sie eine Kopfzeile in Ihre verwalteten Anmeldeseiten aufnehmen möchten. Die Kopfzeile kann ein Logobild enthalten.

Header-Logo

Legen Sie die Position des Logo-Bildes in Ihrer Kopfzeile fest.

Logo-Bild

Wählen Sie eine Logo-Position und eine Logo-Bilddatei aus, die in die Kopfzeile aufgenommen werden sollen. Um ein Bild hochzuladen, wählen Sie das Drop-down-Menü Logobild aus, wählen Sie Neues Asset hinzufügen und fügen Sie eine Logodatei hinzu.

Hintergrundfarbe der Überschrift

Stellen Sie die Farben im Hell- und Dunkelmodus für den Hintergrund der Kopfzeile ein.

Detaillierte Einstellungen

In der Ansicht mit den detaillierten Einstellungen können Sie einzelne Komponenten in den Bereichen Foundation und Components ändern. Auf der Registerkarte Vorschau wird eine Vorschau der verwalteten Anmeldung im aktuellen Kontext mit Ihren Anpassungen angezeigt.

Um den visuellen Editor für eine Komponente aufzurufen, wählen Sie das Bearbeitungssymbol in der Kachel für die Komponente. Im Theme Studio-Editor können Sie mit der Schaltfläche Einstellungskategorie ändern zwischen den Komponenten wechseln.

Grundlage

App-Stil

Konfigurieren Sie die Grundlagen Ihrer verwalteten Login-Konfiguration. Diese Kategorie enthält Einstellungen für das allgemeine Thema, den Textabstand sowie die Kopf- und Fußzeile der Seite.

Anzeigemodus

Wählen Sie für Ihre verwalteten Anmeldeseiten einen hellen Modus, einen dunklen Modus oder ein adaptives Erlebnis. Wenn Sie einen browseradaptiven Modus wählen, können Sie verschiedene Farben und Logobilder für den Hell- und Dunkelmodus wählen.

Spacing

Legen Sie den Standardabstand zwischen Elementen auf der Seite fest.

Verhalten bei der Authentifizierung

Konfigurieren Sie Stile für die Schaltflächen, die Ihre Benutzer mit externen Identitätsanbietern verbinden (IdPs). Dieser Abschnitt enthält die Option Domain-Sucheingabe, mit der Benutzer die Anmeldeaufforderung nach einer E-Mail-Adresse verwalten und diese mit ihrer SAML-Identity Provider-ID abgleichen können.

Verhalten von Formularen

Konfigurieren Sie Stile für Eingabeformulare: die Positionierung von Eingaben, Farben und Ausrichtung von Elementen.

Komponenten

Schaltflächen

Stile für Schaltflächen, die Amazon Cognito auf verwalteten Anmeldeseiten rendert.

Teiler

Stile für Trennlinien und Grenzen zwischen verwalteten Anmeldeelementen wie dem Eingabeformular und der Anmeldeselektion für externe Anbieter.

Dropdown

Stile für Dropdownmenüs.

Favicon

Stile für das Bild, das Amazon Cognito für das Tab- und Lesezeichensymbol bereitstellt.

Fokusringe

Stile für die Markierungen, die auf eine aktuell ausgewählte Eingabe hinweisen.

Formularcontainer

Stile für die Elemente, die ein Formular verbinden.

Globale Fußzeile

Stile für die Fußzeile, die Amazon Cognito unten auf verwalteten Anmeldeseiten anzeigt.

Globaler Header

Stile für die Kopfzeile, die Amazon Cognito oben auf verwalteten Anmeldeseiten anzeigt.

Hinweise

Stile für Fehler- und Erfolgsmeldungen.

Steuerelemente für Optionen

Stile für Kontrollkästchen, Mehrfachauswahlen und andere Eingabeaufforderungen.

Hintergrund der Seite

Stile für den Gesamthintergrund der verwalteten Anmeldung.

Eingaben

Stile für Eingabeaufforderungen in Formularfeldern.

Link

Stile für Hyperlinks auf verwalteten Anmeldeseiten.

Text für Seite

Stile für Text innerhalb der Seite.

Text für das Feld

Stile für den Text rund um Formulareingaben.

API- und SDK-Operationen für verwaltetes Login-Branding

Mit den API-Operationen CreateManagedLoginBrandingund können Sie auch ein Branding auf einen verwalteten Anmeldestil anwenden UpdateManagedLoginBranding. Diese Operationen sind ideal, um identische oder leicht modifizierte Versionen eines Branding-Stils für einen anderen App-Client oder Benutzerpool zu erstellen. Fragen Sie das verwaltete Login-Branding eines vorhandenen Stils mit der API-Operation ab DescribeManagedLoginBranding, ändern Sie dann die Ausgabe nach Bedarf und wenden Sie sie auf eine andere Ressource an.

Durch den `UpdateManagedLoginBranding` Vorgang wird der App-Client, auf den Ihr Stil angewendet wird, nicht geändert. Es aktualisiert nur den vorhandenen Stil, der einem App-Client zugewiesen ist. Um den Stil für einen App-Client vollständig zu ersetzen, löschen Sie den vorhandenen Stil mit DeleteManagedLoginBrandingund weisen Sie ihm einen neuen Stil mit zu`CreateManagedLoginBranding`. In der Amazon Cognito Cognito-Konsole gilt dasselbe: Sie müssen den vorhandenen Stil löschen und einen neuen erstellen.

Um das Branding für verwaltete Logins in einer API- oder SDK-Anfrage einzurichten, müssen Ihre Einstellungen in eine JSON-Datei eingebettet werden, die in einen `Document` Datentyp konvertiert

wird. Im Folgenden finden Sie Anleitungen für Bilder, die Sie hinzufügen können, und für das Generieren programmatischer Anfragen zur Konfiguration eines Branding-Stils.

Bild-Assets

CreateManagedLoginBrandingund UpdateManagedLoginBrandingfügen Sie einen `Assets` Parameter hinzu. Dieser Parameter ist ein Array von Bilddateien im Base64-codierten Binärformat.

> **Note**
>
> Programmatische Anfragen, die den Branding-Stil erstellen oder aktualisieren, dürfen eine Anforderungsgröße von nicht mehr als 2 MB haben. Aufgrund der in Ihrer Anfrage enthaltenen Elemente wird diese Obergrenze möglicherweise überschritten. Wenn dies der Fall ist, teilen Sie Ihre Anfrage in mehrere `UpdateManagedLoginBranding` Anfragen für Parametergruppen auf, die die maximale Anforderungsgröße nicht überschreiten. Diese Anfragen führen nicht dazu, dass nicht spezifizierte Parameter auf die Standardwerte gesetzt werden, sodass Sie Teilanfragen senden können, ohne dass dies Auswirkungen auf die vorhandenen Einstellungen hat.

Für einige Ressourcen gelten Einschränkungen in Bezug auf die Dateitypen, die Sie einreichen können.

Komponente	Zulässige Dateierweiterungen
FAVICON_ICO	ico
FAVICON_SVG	svg
E-MAIL_GRAFIK	png, svg, jpeg
SMS_GRAFIK	png, svg, jpeg
AUTH_APP_GRAPHIC	png, svg, jpeg
PASSWORT_GRAFIK	png, svg, jpeg
PASSKEY_GRAPHIC	png, svg, jpeg
SEITENÜBERSCHRIFT_LOGO	png, svg, jpeg

Komponente	Zulässige Dateierweiterungen
SEITENKOPFHINTERGRUND	png, svg, jpeg
SEITE_FOOTER_LOGO	png, svg, jpeg
SEITENFUSSZEILE_HINTERGRUND	png, svg, jpeg
SEITENHINTERGRUND	png, svg, jpeg
FORM_HINTERGRUND	png, svg, jpeg
FORM_LOGO	png, svg, jpeg
IDP_BUTTON_ICON	ico, svg

Dateien des Typs SVG unterstützen die folgenden Attribute und Elemente.

Attributes

```
accent-height, accumulate, additivive, alignment-baseline, ascent, attributename,
 attributetype, azimuth, basefrequency, baseline-shift, begin, bias, by, class,
 clip, clip-path, clip-rule, color, color-interpolation, color-interpolation-
filters, color-profile, color-rendering, cx, cy, d, dx, dy, diffuseconstant,
 direction, display, divisor, dur, edgemode, elevation, end, fill, fill-opacity,
 fill-rule, filter, filterunits, flood-color, flood-opacity, font-family, font-
size, font-size-adjust, font-stretch, font-style, font-variant, font-weight, fx,
 fy, g1, g2, glyph-name, glyphref, gradientunits, gradienttransform, height, href,
 id, image-rendering, in, in2, k, k1, k2, k3, k4, kerning, keypoints, keysplines,
 keytimes, lang, lengthadjust, letter-spacing, kernelmatrix, kernelunitlength,
 lighting-color, local, marker-end, marker-mid, marker-start, markerheight,
 markerunits, markerwidth, maskcontentunits, maskunits, max, mask, media,
 method, mode, min, name, numoctaves, offset, operator, opacity, order, orient,
 orientation, origin, overflow, paint-order, path, pathlength, patterncontentunits,
 patterntransform, patternunits, points, preservealpha, preserveaspectratio, r,
 rx, ry, radius, refx, refy, repeatcount, repeatdur, restart, result, rotate,
 scale, seed, shape-rendering, specularconstant, specularexponent, spreadmethod,
 stddeviation, stitchtiles, stop-color, stop-opacity, stroke-dasharray, stroke-
dashoffset, stroke-linecap, stroke-linejoin, stroke-miterlimit, stroke-opacity,
 stroke, stroke-width, style, surfacescale, tabindex, targetx, targety, transform,
 text-anchor, text-decoration, text-rendering, textlength, type, u1, u2, unicode,
 values, viewbox, visibility, vert-adv-y, vert-origin-x, vert-origin-y, width, word-
```

```
spacing, wrap, writing-mode, xchannelselector, ychannelselector, x, x1, x2, xmlns,
 y, y1, y2, z, zoomandpan
```

Elements

```
svg, a, altglyph, altglyphdef, altglyphitem, animatecolor, animatemotion,
 animatetransform, audio, canvas, circle, clippath, defs, desc, ellipse, filter,
 font, g, glyph, glyphref, hkern, image, line, lineargradient, marker, mask,
 metadata, mpath, path, pattern, polygon, polyline, radialgradient, rect, stop,
 style, switch, symbol, text, textpath, title, tref, tspan, video, view, vkern,
 feBlend, feColorMatrix, feComponentTransfer, feComposite, feConvolveMatrix,
 feDiffuseLighting, feDisplacementMap, feDistantLight, feFlood, feFuncA, feFuncB,
 feFuncG, feFuncR, feGaussianBlur, feMerge, feMergeNode, feMorphology, feOffset,
 fePointLight, feSpecularLighting, feSpotLight, feTile, feTurbulence
```

Tools für verwaltete Login-Branding-Operationen

Amazon Cognito verwaltet eine Datei im JSON-Schemaformat für das Objekt mit den Branding-Einstellungen für verwaltete Logins. Im Folgenden erfahren Sie, wie Sie Ihren Branding-Stil programmgesteuert aktualisieren können.

Um das Branding in der Benutzerpools-API zu aktualisieren

1. Erstellen Sie in der Amazon Cognito Cognito-Konsole im Menü Verwaltete Anmeldung Ihres Benutzerpools einen standardmäßigen Branding-Stil für verwaltete Logins. Weisen Sie ihn einem App-Client zu.
2. Notieren Sie sich beispielsweise die ID des App-Clients, für den Sie den Stil erstellt haben`1example23456789`.
3. Rufen Sie die Einstellungen für den Branding-Stil mit einer DescribeManagedLoginBrandingByClientAPI-Anfrage mit der `ReturnMergedResources` Einstellung auf ab`true`. Im Folgenden finden Sie ein Beispiel für einen Anfragetext.

   ```
   {
      "ClientId": "1example23456789",
      "ReturnMergedResources": true,
      "UserPoolId": "us-east-1_EXAMPLE"
   }
   ```

4. Ändern Sie die Ausgabe von `DescribeManagedLoginBrandingByClient` mit Ihren Anpassungen.

a. Der Antworttext ist in ein `ManagedLoginBranding` Element eingeschlossen, das nicht Teil der Syntax für Erstellungs- und Aktualisierungsvorgänge ist. Entfernen Sie diese oberste Ebene des JSON-Objekts.

b. Um Bilder zu ersetzen, ersetzen Sie den `Bytes` Wert durch die Base64-codierten Binärdaten jeder Bilddatei.

c. Um die Einstellungen zu aktualisieren, ändern Sie die Ausgabe des `Settings` Objekts und nehmen Sie es in Ihre nächste Anfrage auf. Amazon Cognito ignoriert alle Werte in Ihrem `Settings` Objekt, die nicht in dem Schema enthalten sind, das Sie in Ihrer API-Antwort erhalten.

5. Verwenden Sie den aktualisierten Antworttext in einer CreateManagedLoginBrandingUpdateManagedLoginBrandingOder-Anfrage. Wenn diese Anfrage mehr als 2 MB groß ist, teilen Sie sie in mehrere Anfragen auf. Diese Operationen funktionieren in einem `PATCH` Modell, in dem die ursprünglichen Einstellungen unverändert bleiben, sofern Sie nichts anderes angeben.

Anpassen des Brandings für gehostete Benutzeroberflächen (klassisch)

Sie können die AWS Management Console API oder die AWS CLI oder verwenden, um klassische Anpassungseinstellungen für die gehostete Benutzeroberfläche anzugeben. Sie können ein benutzerdefiniertes Logo-Image hochladen, das Sie in der App anzeigen möchten. Sie können auch einige Cascading Style Sheets (CSS) -Optionen auf das Erscheinungsbild der Benutzeroberfläche anwenden.

Sie können die Standardeinstellungen der Benutzeroberfläche anpassen und einzelne App-Clients mit bestimmten Einstellungen überschreiben. Amazon Cognito wendet die Standardkonfiguration auf jeden App-Client an, der keine Einstellungen auf Client-Ebene hat.

In der Amazon Cognito Cognito-Konsole und in API-Anfragen darf die Anfrage, die Ihre UI-Anpassung festlegt, 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 Anfrage zu groß ist, gibt die AWS Management Console oder Ihre `SetUICustomization` API-Anfrage einen `request parameters too large` Fehler 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.

Spezifizierung eines benutzerdefinierten Logos im klassischen Branding

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 `ImageFile` ein SetUICustomizationkonvertieren Sie in der API Ihre Datei in eine Base64-kodierte Textzeichenfolge oder geben Sie in der AWS CLI einen Dateipfad an und lassen Sie ihn von Amazon Cognito für Sie codieren.

Spezifizierung von CSS-Anpassungen im klassischen Branding

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.
- color ist die Textfarbe. Muss ein zulässiger CSS-Farbwert sein.

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.
- color ist die Textfarbe. Muss ein zulässiger CSS-Farbwert sein.

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.
- Farbe ist die Textfarbe für IdP-Abschnittsüberschriften, zum Beispiel Melden Sie sich mit Ihrer Unternehmens-ID an. Muss ein zulässiger CSS-Farbwert sein.

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.
- background-color ist die Hintergrundfarbe für Protokolle mit transparenten Abschnitten. Muss ein zulässiger CSS-Farbwert sein.

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.

Anpassen der gehosteten Benutzeroberfläche mit klassischem Branding in AWS Management Console

Sie können das verwenden AWS Management Console , um die Einstellungen für die Anpassung der Benutzeroberfläche für Ihre App festzulegen.

> **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. Erstellen Sie über den Domain-Tab eine Domain oder aktualisieren Sie Ihre bestehende Domain. Stellen Sie unter Branding-Version Ihre Domain so ein, dass sie Hosted UI (klassisch) verwendet.
4. Wählen Sie das Menü Verwaltete Anmeldung.
5. Um die Benutzeroberflächeneinstellungen für alle App-Clients anzupassen, suchen Sie unter Einstellungen für gehostete Benutzeroberflächen nach Style und wählen Sie Bearbeiten aus.
6. Um die UI-Einstellungen für einen App-Client anzupassen, rufen Sie das Menü App-Clients auf und wählen Sie den App-Client aus, den Sie ändern möchten. Suchen Sie dann nach dem Stil der gehosteten Benutzeroberfläche (klassisch) und wählen Sie Überschreiben aus. Wählen Sie Bearbeiten aus.
7. Um Ihre eigene Logo-Bilddatei hochzuladen, wählen Sie Choose file (Datei auswählen) oder Replace current file (Aktuelle Datei ersetzen) aus.
8. 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.

Anpassen der gehosteten Benutzeroberfläche mit klassischem Branding in der Benutzerpools-API und mit AWS CLI

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 aus der 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 einem 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

Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern

Amazon Cognito arbeitet mit 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 Authentifizierungsablaufs 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 der Anfrage- und Antwortereignisse können Sie Ihre eigenen Authentifizierungsherausforderungen einführen, 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 Laufzeitoperationen für Ihre AWS Umgebung APIs, externe 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. Ein Ereignis, das unverändert zurückgegeben wird, benachrichtigt Ihren Benutzerpool, mit der Standardaktion für das Ereignis fortzufahren. Beispielsweise kann Ihr Trigger vor der Registrierung automatisch Benutzer für die `PreSignUp_SignUp` Triggerquelle bestätigen, das Ereignis jedoch für externe und vom Administrator erstellte Benutzer unverändert zurückgeben.

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

Benutzerpool-Ablauf	Operation	Beschreibung
	Antwort auf Authentifizierungs aufforderung überprüfen	Bestimmt, ob eine Antwort in einem benutzerdefinierten Authentifizierungsablauf richtig ist
Authentifizierungsereignisse	the section called “Vorauthentifizierung”	Benutzerdefinierte Validierung zur Akzeptierung oder Ablehnung einer Anmeldeanforderung
	the section called “Nachauthentifizierung”	Protokolliert Ereignisse für benutzerdefinierte Analysen
	the section called “Generierung von Pre-Token”	Erhöht oder unterdrückt Token-Ansprüche
Registrieren	the section called “Voranmeldung”	Führt eine benutzerdefinierte Validierung durch, die die Anmeldeanforderung akzeptiert oder ablehnt
	the section called “Nachbestätigung”	Fügt benutzerdefinierte Begrüßungsnachrichten oder Ereignisprotokollierung für benutzerdefinierte Analysen hinzu
	the section called “Benutzer migrieren”	Migriert Benutzer aus einem vorhandenen Benutzerverzeichnis in Benutzerpools
Nachrichten	the section called “Benutzer definierte Nachricht”	Führt eine erweiterte Anpassung und Lokalisierung von Nachrichten durch

Benutzerpool-Ablauf	Operation	Beschreibung
Erstellung von Tokens	the section called “Generierung von Pre-Token”	Fügt Attribute in ID- und Zugriffstoken hinzu oder entfernt sie
Drittanbieter von E-Mails und SMS	the section called “Benutzerdefinierte Absender”	Sendet SMS- und E-Mail-Nachrichten mit einem Drittanbieter

Themen

- Wissenswertes über Lambda-Trigger
- Fügen Sie einen Lambda-Trigger für einen Benutzerpool hinzu
- Lambda-Auslöserereignis für einen Benutzerpool
- Allgemeine Parameter von Lambda-Auslösern für Benutzerpools
- Client-Metadaten
- 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

Wissenswertes über Lambda-Trigger

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. Um Zugriffstoken in a anzupassenLambda-Auslöser für die Vorab-Generierung von Token, müssen Sie Ihren Benutzerpool mit einem anderen Funktionsplan als Lite konfigurieren und Ihre Lambda-Trigger-Konfiguration aktualisieren, um Event Version 2 zu verwenden.
- 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 AWS Lambda Developer Guide.

  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 in einer CreateUserPoolUpdateUserPoolAPI-Anfrage `LambdaArn` 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.

Managed Login gibt Fehler zurück, die von Lambda-Triggern generiert werden, als Fehlertext über der Anmeldeaufforderung. 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`print()`, um beispielsweise sensible Informationen oder Debugging-Informationen in Logs zu protokollieren. CloudWatch Ein Beispiel finden Sie unter Beispiel für die Vorab-Registrierung: Anmeldung ablehnen, wenn der Benutzername weniger als fünf Zeichen enthält.

- Sie können einer anderen eine Lambda-Funktion AWS-Konto als Auslöser für Ihren Benutzerpool hinzufügen. Sie müssen kontoübergreifende Trigger mit den UpdateUserPoolAPI-Operationen CreateUserPoolund oder deren Entsprechungen in und dem hinzufügen. AWS CloudFormation AWS CLI Sie können keine kontoübergreifenden Funktionen in der hinzufügen. AWS Management Console
- 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 über die Lambda-Konsole hinzufügen oder den AddPermissionLambda-API-Vorgang 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.

Fügen Sie einen Lambda-Trigger für einen Benutzerpool hinzu

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 das Menü Erweiterungen und suchen Sie nach Lambda-Triggern.
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 Lambda-Funktion zuweisen und wählen Sie eine Funktion aus, die sich AWS-Region in Ihrem Benutzerpool befindet.

   > **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 “Wissenswertes”.

8. Wählen Sie Änderungen speichern aus.
9. Sie können CloudWatch in der Lambda-Konsole Ihre Lambda-Funktion protokollieren. Weitere Informationen finden Sie unter Zugreifen auf CloudWatch Protokolle 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. Wenn Ihre Funktion das Eingabeereignis ohne Änderung zurückgibt, fährt Amazon Cognito mit dem

Standardverhalten fort. Im Folgenden werden die Parameter aufgeführt, die allen Lambda-Trigger-Eingabeereignissen gemeinsam sind. Informationen zur triggerspezifischen Ereignissyntax finden Sie im Ereignisschema im Abschnitt dieses Handbuchs für jeden Trigger.

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)

Der AWS-Region als Instanz`AWSRegion`.

userPoolId

Die ID des Benutzerpools.

userName

Der Benutzername des aktuellen Benutzers.

callerContext

Metadaten über die Anfrage und die Codeumgebung. Es enthält die Felder awsSdkVersionund die 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.

Client-Metadaten

Sie können benutzerdefinierte Parameter an Ihre Lambda-Triggerfunktionen in API-Vorgängen und Token-Endpunkt Anfragen senden. Mithilfe von Client-Metadaten kann Ihre Anwendung zusätzliche

Informationen über die Umgebung sammeln, aus der Anfragen stammen. Wenn Sie Client-Metadaten an Ihre Lambda-Funktionen übergeben, können diese die zusätzlichen Daten verarbeiten und sie für die Protokollierung oder Anpassung von Authentifizierungsabläufen verwenden. Client-Metadaten sind Zeichenkettenpaare Ihrer Wahl und Ihres Designs in einem JSON-Schlüsselwertformat.

Anwendungsfälle für Client-Metadaten

- Übergeben Sie Geolokalisierungsdaten bei der Anmeldung an den Trigger vor der Registrierung und verhindern Sie so die Anmeldung von unerwünschten Standorten aus.
- Übergeben Sie Mandanten-ID-Daten an benutzerdefinierte Challenge-Trigger und stellen Sie Kunden aus verschiedenen Geschäftsbereichen unterschiedliche Herausforderungen.
- Übergeben Sie das Token eines Benutzers an den Trigger vor der Token-Generierung und generieren Sie ein Protokoll des Prinzipals, für den eine M2M-Anfrage gestellt wurde. Eine Beispielanforderung finden Sie unter Kundenanmeldedaten mit Basisautorisierung.

Hier ist ein Beispiel für die Übergabe von Client-Metadaten an den Trigger vor der Anmeldung.

SignUp request

Im Folgenden finden Sie eine SignUpBeispielanforderung mit Client-Metadaten, die Amazon Cognito an einen Trigger vor der Anmeldung weiterleitet.

```
POST HTTP/1.1
Host: cognito-idp.us-east-1.amazonaws.com
X-Amz-Date: 20230613T200059Z
Accept-Encoding: gzip, deflate, br
X-Amz-Target: AWSCognitoIdentityProviderService.SignUp
User-Agent: <UserAgentString>
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
 Signature=<Signature>
Content-Length: <PayloadSizeBytes>

{
    "ClientId": "1example23456789",
    "Username": "mary_major",
    "Password": "<Password>",
    "SecretHash": "<Secret hash>",
    "ClientMetadata": {
        "IpAddress" : "192.0.2.252",
        "GeoLocation" : "Netherlands (Kingdom of the) [NL]"
```

```
    }
    "UserAttributes": [
        {
            "Name": "name",
            "Value": "Mary"
        },
        {
            "Name": "email",
            "Value": "mary_major@example.com"
        },
        {
            "Name": "phone_number",
            "Value": "+12065551212"
        }
    ],
}
```

Lambda trigger input event

Die Anfrage führt zum folgenden Anfragetext für Ihre Funktion vor der Anmeldung.

```
{
    "callerContext": {
        "awsSdkVersion": "aws-sdk-unknown-unknown",
        "clientId": "1example23456789"
    },
    "region": "us-west-2",
    "request": {
        "clientMetadata": {
            "GeoLocation": "Netherlands (Kingdom of the) [NL]",
            "IpAddress": "192.0.2.252"
        },
        "userAttributes": {
            "email": "mary_major@example.com",
            "name": "Mary",
            "phone_number": "+12065551212"
        },
        "validationData": null
    },
    "response": {
        "autoConfirmUser": false,
        "autoVerifyEmail": false,
        "autoVerifyPhone": false
    },
```

```
    "triggerSource": "PreSignUp_SignUp",
    "userName": "mary_major2",
    "userPoolId": "us-west-2_EXAMPLE",
    "version": "1"
}
```

Client-Metadaten für machine-to-machine (M2M) -Client-Anmeldeinformationen

Sie können Client-Metadaten in M2M-Anfragen übergeben. Client-Metadaten sind zusätzliche Informationen aus einer Benutzer- oder Anwendungsumgebung, die zu den Ergebnissen einer Lambda-Auslöser für die Vorab-Generierung von Token beitragen können. Bei Authentifizierungsvorgängen mit einem Benutzerprinzipal können Sie Client-Metadaten im Hauptteil von AdminRespondToAuthChallengeund RespondToAuthChallengeAPI-Anfragen an den Trigger vor der Token-Generierung übergeben. Da Anwendungen den Ablauf für die Generierung von Zugriffstoken für M2M mit direkten Anfragen an die durchführenToken-Endpunkt, haben sie ein anderes Modell. Übergeben Sie im POST-Hauptteil der Token-Anfragen für Client-Anmeldeinformationen einen `aws_client_metadata` Parameter mit dem Client-Metadatenobjekt URL-encoded () `x-www-form-urlencoded` an string. Eine Beispielanforderung finden Sie unter Kundenanmeldedaten mit Basisautorisierung. Im Folgenden finden Sie einen Beispielparameter, der die Schlüssel-Wert-Paare übergibt. `{"environment": "dev", "language": "en-US"}`

```
aws_client_metadata=%7B%22environment%22%3A%20%22dev%22,%20%22language%22%3A%20%22en-US
%22%7D
```

Temporäre Benutzerattribute: **`validationData`**

Einige Authentifizierungsvorgänge haben auch einen `validationData` Parameter. Wie bei Client-Metadaten bietet dies die Möglichkeit, externe Informationen, die Amazon Cognito nicht automatisch sammelt, an Lambda-Trigger weiterzugeben. Das Validierungsdatenfeld soll Ihrer Lambda-Funktion zusätzlichen Benutzerkontext bei Registrierungs- und Anmeldevorgängen bieten. SignUpund zum Trigger vor der `validationData` Anmeldung AdminCreateUserübergehen. InitiateAuthund AdminInitiateAuthübergibt `ClientMetadata` den Hauptteil der API-Anfrage wie `validationData` im Eingabeereignis an die Benutzerauslöser vor der Authentifizierung und Migration.

Informationen zur Zuordnung von API-Vorgängen zu den Funktionen, an die sie Client-Metadaten übergeben können, finden Sie in den folgenden Abschnitten zu den Trigger-Quellen.

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, verwaltete Anmeldung oder Benutzerpool-Endpunkte anmeldet, ruft Amazon Cognito Ihre Lambda-Funktionen basierend auf dem Sitzungskontext auf. Weitere Informationen zur Amazon-Cognito-Benutzerpool-API und zu Benutzerpool-Endpunkten finden Sie unter Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten. 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
- Lambda-Trigger für lokale Amazon Cognito Cognito-Benutzer bei verwalteter Anmeldung
- Lambda-Auslöser für Verbundbenutzer

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`

API-Vorgang	Lambda-Trigger	Trigger-Quelle
	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`
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`

API-Vorgang	Lambda-Trigger	Trigger-Quelle
	Benutzer migrieren	`UserMigration_Authentication`
	Benutzerdefinierte Nachricht	`CustomMessage_Authentication`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_AccountTakeOverNotification` `CustomEmailSender_Authentication`
	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`
UpdateUserAttributes AdminUpdateUserAttributes	Benutzerdefinierte Nachricht	`CustomMessage_UpdateUserAttribute`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_UpdateUserAttribute`

API-Vorgang	Lambda-Trigger	Trigger-Quelle
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_UpdateUserAttribute`
VerifyUserAttributes	Benutzerdefinierte Nachricht	`CustomMessage_VerifyUserAttribute`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_VerifyUserAttribute`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_VerifyUserAttribute`
GetTokensFromRefreshToken	Generierung von Pre-Token	`TokenGeneration_Authentication`

Lambda-Trigger für lokale Amazon Cognito Cognito-Benutzer bei verwalteter Anmeldung

In der folgenden Tabelle werden die Quellzeichenfolgen für die Lambda-Trigger beschrieben, die Amazon Cognito aufrufen kann, wenn sich ein lokaler Benutzer mit verwalteter Anmeldung bei Ihrem Benutzerpool anmeldet.

Lokale Benutzer lösen Quellen bei der verwalteten Anmeldung aus

Verwaltete Anmelde-URI	Lambda-Trigger	Trigger-Quelle
`/signup`	Voranmeldung	`PreSignUp_SignUp`
	Benutzerdefinierte Nachricht	`CustomMessage_SignUp`
	Benutzerdefinierter E-Mail-Absender	`CustomEmailSender_SignUp`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_SignUp`

Verwaltete Anmelde-URI	Lambda-Trigger	Trigger-Quelle
`/confirmuser`	Nachbestätigung	`PostConfirmation_ConfirmSignUp`
`/login`	Vorauthentifizierung	`PreAuthentication_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` `CustomEmailSender_Authentication`
	Benutzerdefinierter SMS-Absender	`CustomSMSSender_Authentication`
`/forgotpassword`	Benutzer migrieren	`UserMigration_ForgotPassword`
	Benutzerdefinierte Nachricht	`CustomMessage_ForgotPassword`

Verwaltete Anmelde-URI	Lambda-Trigger	Trigger-Quelle
	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 verwaltete Anmeldung verwenden, um sich anzumelden, oder Sie können eine Anfrage an den senden, die sie Autorisieren des Endpunkts unbemerkt auf die Anmeldeseite ihres Identitätsanbieters umleitet. 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`

Anmeldeereignisse	Lambda-Trigger	Trigger-Quelle
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	Ereignis
Voranmeldung	`PreSignUp_SignUp`	Voranmeldung.

Auslöser	triggerSource-Wert	Ereignis
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	Ereignis
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	Ereignis
Generierung von Pre-Token	`TokenGeneration_HostedAuth`	Amazon Cognito authentifiziert den Benutzer von Ihrer verwalteten Anmeldeseite aus.
Generierung von Pre-Token	`TokenGeneration_Authentication`	Die Benutzerauthentifizierung oder Token-Aktualisierung ist 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	Ereignis
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	Ereignis
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 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.

Auslöser	triggerSource-Wert	Ereignis
Benutzerdefinierte Nachricht	`CustomMessage_Authentication`	Benutzerdefinierte Nachricht, wenn sich ein Benutzer anmeldet, der SMS MFA konfiguriert hat.

Benutzerdefinierte Absender-Trigger

Auslöser	triggerSource-Wert	Ereignis

Benutzerdefinierter Absender	`CustomEmailSender_SignUp`
	
	`CustomSmsSender_SignUp`
		Wenn sich ein Benutzer in Ihrem Benutzerpool anmeldet.

Benutzerdefinierter Absender	`CustomEmailSender_AdminCreateUser`
	
	`CustomSmsSender_AdminCreateUser`
		Wenn Sie einen Benutzer als Administrator erstellen und Amazon Cognito ihm ein temporäres Passwort sendet.

Benutzerdefinierter Absender	`CustomEmailSender_ForgotPassword`
	
	`CustomSmsSender_ForgotPassword`
		Wenn Ihr Benutzer ein Zurücksetzen des Passworts anfordert.

Benutzerdefinierter Absender	`CustomEmailSender_UpdateUserAttribute`
	
	`CustomSmsSender_UpdateUserAttribute`
		Wenn ein Benutzer seine E-Mail-Adresse oder Telefonnummer ändert und Amazon Cognito einen Bestätigungscode sendet.

Benutzerdefinierter Absender	`CustomEmailSender_VerifyUserAttribute`	Wenn ein Benutzer eine E-Mail-Adresse oder Telefonnummer hinzufügt und Amazon

Auslöser	triggerSource-Wert	Ereignis
	`CustomSmsSender_VerifyUserAttribute`	Cognito einen Bestätigungscode sendet.

Benutzerdefinierter Absender	`CustomEmailSender_Authentication`
	
	`CustomSmsSender_Authentication`
		Wenn sich ein Benutzer anmeldet, der SMS oder E-Mail MFA oder OTP konfiguriert hat.

Benutzerdefinierter Absender	`CustomEmailSender_AccountTakeOverNotification`	Wenn Ihre Einstellungen für den Bedrohungsschutz eine automatische Aktion gegen den Anmeldeversuch eines Benutzers ergreifen und die Aktion für die Risikostufe eine Benachrichtigung beinhaltet.

Lambda-Auslöser für die Vorab-Registrierung

Möglicherweise möchten Sie den Anmeldevorgang in Benutzerpools anpassen, die über Self-Service-Anmeldeoptionen verfügen. Häufig wird der Trigger vor der Anmeldung verwendet, um benutzerdefinierte Analysen und Aufzeichnungen neuer Benutzer durchzuführen, Sicherheits- und Governance-Standards anzuwenden oder Benutzer von einem Drittanbieter-IdP mit einem konsolidierten Benutzerprofil zu verknüpfen. Möglicherweise haben Sie auch vertrauenswürdige Benutzer, die sich keiner Überprüfung und Bestätigung unterziehen müssen.

Unmittelbar bevor Amazon Cognito die Erstellung eines neuen lokalen oder Verbundbenutzers abschließt, aktiviert es die Lambda-Funktion vor der Anmeldung. Das `userAttributes` in der Anfrage an diese Funktion gesendete Objekt enthält Attribute, die bei der lokalen Benutzerregistrierung bereitgestellt wurden oder die erfolgreich den Anbieterattributen für einen Verbundbenutzer zugeordnet wurden. Ihr Benutzerpool ruft diesen Trigger bei der Self-Service-Registrierung SignUpoder der ersten Anmeldung bei einem vertrauenswürdigen Identitätsanbieter und bei der Benutzererstellung mit auf. AdminCreateUser Im Rahmen des Anmeldevorgangs können Sie diese Funktion verwenden, um das Anmeldeereignis mit benutzerdefinierter Logik zu analysieren und den neuen Benutzer zu ändern oder abzulehnen.

Themen

- Lambda-Auslöserparameter für die Vorab-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-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 im ValidationData Parameter Ihrer AdminCreateUseroder SignUpAPI-Anfrage an Ihre Lambda-Funktion.

Amazon Cognito legt Ihre ValidationData Daten nicht als Attribute des von Ihnen erstellten Benutzers fest. ValidationData sind temporäre Benutzerinformationen, die Sie für Ihren Lambda-Trigger 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 übergeben, indem Sie den ClientMetadata Parameter in den folgenden API-Aktionen verwenden: AdminCreateUser, AdminRespondToAuthChallengeForgotPassword, 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.

Beispiel für Vorab-Registrierung: Automatische Bestätigung von Benutzern aus einer registrierten Domäne

Dies ist ein Beispiel für einen Lambda-Triggercode. Der Trigger vor der Anmeldung wird unmittelbar vor der Verarbeitung der Registrierungsanfrage durch Amazon Cognito ausgelöst. 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

```
export const handler = async (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 mithilfe der ListUsers Benutzerpools-API überprüfen, ob es einen bestehenden Benutzer gibt, der die Telefonnummer oder E-Mail-Adresse des neuen Benutzers bereits als Alias verwendet.

Node.js

```
exports.handler = (event, context, callback) => {
  // 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 to Amazon Cognito
  callback(null, event);
};
```

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

```
export const 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. Ihr Benutzerpool ruft Ihre Funktion nach der Bestätigung bei ConfirmSignUpAdminConfirmSignUp, und ConfirmForgotPasswordauf. Dieser Trigger wird auch ausgeführt, wenn Benutzer bei der verwalteten Anmeldung die Registrierung oder das Zurücksetzen des Passworts bestätigen.

Themen

- Lambda-Auslöserparameter nach der Bestätigung
- Beispiel für „Nachbestätigung“

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 ClientMetadata Parameter in den folgenden API-Aktionen verwenden: AdminConfirmSignUp, ConfirmForgotPasswordConfirmSignUp, und SignUp.

Antwortparameter nach der Bestätigung

Erwartungsgemäß enthält die Antwort keine weiteren Informationen.

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, es sei denn, die `PreventUserExistenceErrors` Einstellung eines Benutzerpool-App-Clients ist auf `ENABLED` eingestellt. Die Erneuerung einer bestehenden Authentifizierungssitzung aktiviert diesen Trigger ebenfalls nicht.

Themen

- Überblick über den Ablauf
- Lambda-Auslöseparameter für die Vorab-Authentifizierung
- Beispiel für „Vorauthentifizierung“

Überblick über den Ablauf

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 Ein Beispiel für eine Authentifizierungssitzung.

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 ClientMetadata Parameter in den Aktionen InitiateAuthund AdminInitiateAuthAPI.

Antwortparameter für die Vorab-Authentifizierung

Amazon Cognito verarbeitet keine zusätzlichen Informationen, die Ihre Funktion in der Antwort zurückgibt. 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

Der Trigger nach der Authentifizierung ändert den Authentifizierungsablauf für einen Benutzer nicht. Amazon Cognito ruft dieses Lambda auf, nachdem die Authentifizierung abgeschlossen ist, bevor ein Benutzer Token erhalten hat. Fügen Sie einen Trigger nach der Authentifizierung hinzu, wenn Sie eine benutzerdefinierte Nachverarbeitung von Authentifizierungsereignissen hinzufügen möchten, z. B. Protokollierung oder Anpassungen des Benutzerprofils, die bei der nächsten Anmeldung berücksichtigt werden.

Ein Lambda nach der Authentifizierung, das den Anfragetext nicht an Amazon Cognito zurücksendet, kann dennoch dazu führen, dass die Authentifizierung nicht abgeschlossen werden kann. Weitere Informationen finden Sie unter Wissenswertes über Lambda-Trigger.

Themen

- Authentifizierungsprozess – Übersicht
- Lambda-Auslöseparameter nach der Authentifizierung
- Beispiel für „Nachauthentifizierung“

Authentifizierungsprozess – Übersicht

Weitere Informationen finden Sie unter Ein Beispiel für eine Authentifizierungssitzung.

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. Um diese Daten an Ihre Lambda-Funktion zu übergeben, können Sie den ClientMetadata Parameter in den Aktionen AdminRespondToAuthChallengeund RespondToAuthChallengeAPI verwenden. Amazon Cognito bezieht keine Daten aus dem ClientMetadata Parameter in AdminInitiateAuthund

InitiateAuthAPI-Operationen in die Anfrage ein, die es an die Post-Authentifizierungsfunktion weiterleitet.

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.

Beispiel für „Nachauthentifizierung“

Diese Lambda-Beispielfunktion nach der Authentifizierung sendet Daten von 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

Beim Aufbau Ihrer Authentifizierungsabläufe für Ihren Amazon Cognito Cognito-Benutzerpool stellen Sie möglicherweise fest, dass Sie Ihr Authentifizierungsmodell über die integrierten Abläufe hinaus erweitern möchten. Ein häufiger Anwendungsfall für die benutzerdefinierten Challenge-Trigger ist die Implementierung zusätzlicher Sicherheitsüberprüfungen, die über Benutzername, Passwort und Multi-Faktor-Authentifizierung (MFA) hinausgehen. Eine benutzerdefinierte Herausforderung ist jede Frage und Antwort, die Sie in einer von Lambda unterstützten Programmiersprache generieren können. Möglicherweise möchten Sie von Benutzern verlangen, dass sie ein CAPTCHA lösen oder eine Sicherheitsfrage beantworten, bevor sie sich authentifizieren dürfen. Ein weiterer potenzieller Bedarf besteht in der Integration mit speziellen Authentifizierungsfaktoren oder -geräten. Oder Sie haben möglicherweise bereits Software entwickelt, die Benutzer mit einem Hardware-Sicherheitsschlüssel oder einem biometrischen Gerät authentifiziert. Die Definition des Authentifizierungserfolgs für eine benutzerdefinierte Abfrage ist die Antwort, die Ihre Lambda-Funktion als richtig akzeptiert: zum Beispiel eine feste Zeichenfolge oder eine zufriedenstellende Antwort von einer externen API.

Sie können die Authentifizierung mit Ihrer benutzerdefinierten Abfrage beginnen und den Authentifizierungsprozess vollständig kontrollieren, oder Sie können eine Benutzername-Passwort-Authentifizierung durchführen, bevor Ihre Anwendung Ihre benutzerdefinierte Abfrage erhält.

Der Lambda-Trigger für benutzerdefinierte Authentifizierungsherausforderungen:

Definiert

Initiiert eine Challenge-Sequenz. Legt fest, ob Sie eine neue Herausforderung initiieren, die Authentifizierung als abgeschlossen markieren oder den Authentifizierungsversuch beenden möchten.

Erzeugt

Stellt die Frage an Ihre Anwendung, die der Benutzer beantworten muss. Diese Funktion kann eine Sicherheitsfrage oder einen Link zu einem CAPTCHA enthalten, das Ihre Anwendung Ihrem Benutzer anzeigen soll.

Überprüft

Kennt die erwartete Antwort und vergleicht sie mit der Antwort, die Ihre Anwendung in der Challenge-Antwort liefert. Die Funktion ruft möglicherweise die API Ihres CAPTCHA-Dienstes auf, um die erwarteten Ergebnisse des Lösungsversuchs Ihres Benutzers abzurufen.

Diese drei Lambda-Funktionen sind miteinander verknüpft, um einen Authentifizierungsmechanismus zu bieten, den Sie vollständig kontrollieren können und den Sie selbst entworfen haben. Da für die benutzerdefinierte Authentifizierung Anwendungslogik in Ihrem Client und in den Lambda-Funktionen erforderlich ist, können Sie die benutzerdefinierte Authentifizierung nicht innerhalb der verwalteten Anmeldung verarbeiten. Dieses Authentifizierungssystem erfordert zusätzlichen Entwickleraufwand. Ihre Anwendung muss den Authentifizierungsablauf mit der Benutzerpools-API durchführen und die daraus resultierende Herausforderung mit einer maßgeschneiderten Anmeldeschnittstelle bewältigen, die die Frage in den Mittelpunkt der benutzerdefinierten Authentifizierungsherausforderung stellt.

Weitere Informationen zur Implementierung der benutzerdefinierten Authentifizierung finden Sie unter Benutzerdefinierter Authentifizierungsablauf und Aufforderungen

Authentifizierung zwischen den API-Vorgängen InitiateAuthoder AdminInitiateAuth, und RespondToAuthChallengeoder AdminRespondToAuthChallenge. 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. Eine Antwort auf eine Herausforderung könnte eine neue Herausforderung sein. In diesem Fall reagiert Ihre Anwendung so oft wie nötig auf neue Herausforderungen. Eine erfolgreiche Authentifizierung erfolgt,

wenn die Funktion Define Auth Challenge die bisherigen Ergebnisse analysiert, feststellt, dass alle Challenges beantwortet wurden, und ein Ergebnis zurückgibt`IssueTokens`.

Themen

- SRP-Authentifizierung in benutzerdefinierten Challenge-Flows
- Lambda-Auslöser für die Definition einer Authentifizierungsaufforderung
- Lambda-Auslöser für die Erstellung einer Authentifizierungsaufforderung
- Lambda-Auslöser für die Verifizierung der Antwort auf eine Authentifizierungsaufforderung

SRP-Authentifizierung in benutzerdefinierten Challenge-Flows

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.

Im Folgenden finden Sie ein Beispiel für eine `InitiateAuth` Startanforderung, die der benutzerdefinierten Authentifizierung mit einem SRP-Flow vorausgeht.

```
{
    "AuthFlow": "CUSTOM_AUTH",
    "ClientId": "1example23456789",
    "AuthParameters": {
        "CHALLENGE_NAME": "SRP_A",
        "USERNAME": "testuser",
        "SRP_A": "[SRP_A]",
        "SECRET_HASH": "[secret hash]"
    }
}
```

Das Passwort wurde im SRP-Flow für die benutzerdefinierte Authentifizierung zurückgesetzt

Wenn sich Benutzer im `FORCE_CHANGE_PASSWORD` Status befinden, muss Ihr benutzerdefinierter Authentifizierungsablauf den Schritt zur Passwortänderung integrieren und gleichzeitig die Integrität Ihrer Authentifizierungsherausforderungen wahren. Amazon Cognito ruft während der Herausforderung Ihren Lambda-Trigger für die Definitionsauthentifizierungsherausforderung auf. `NEW_PASSWORD_REQUIRED` In diesem Szenario kann ein Benutzer, der sich mit einem benutzerdefinierten Challenge-Flow und SRP-Authentifizierung anmeldet, ein neues Passwort festlegen, wenn er sich in einem Zustand befindet, in dem das Passwort zurückgesetzt wurde.

Wenn sich Benutzer im `FORCE_CHANGE_PASSWORD` Status `RESET_REQUIRED` oder befinden, müssen sie auf eine Aufforderung mit einem `NEW_PASSWORD_REQUIRED`antworten. `NEW_PASSWORD` Bei der benutzerdefinierten Authentifizierung mit SRP gibt Amazon Cognito eine `NEW_PASSWORD_REQUIRED` Aufforderung zurück, nachdem Benutzer die `PASSWORD_VERIFIER` SRP-Abfrage abgeschlossen haben. Ihr Trigger zur Definition einer Authentifizierungsabfrage empfängt beide Challenge-Ergebnisse im `session` Array und kann mit weiteren benutzerdefinierten Abfragen fortfahren, nachdem der Benutzer sein Passwort erfolgreich geändert hat.

Ihr Lambda-Trigger zur Definierung der Authentifizierungsherausforderung muss die Abfragesequenz durch SRP-Authentifizierung, Kennwortrücksetzung und nachfolgende benutzerdefinierte Abfragen verwalten. Der Trigger empfängt eine Reihe abgeschlossener Herausforderungen im `session` Parameter, einschließlich sowohl `PASSWORD_VERIFIER` der Ergebnisse als auch der Ergebnisse. `NEW_PASSWORD_REQUIRED` Eine Beispielimplementierung finden Sie unterBeispiel für „Authentifizierungsaufforderung definieren“.

Schritte des Authentifizierungsablaufs

Für Benutzer, die ihr Passwort vor benutzerdefinierten Herausforderungen verifizieren müssen, folgt der Prozess diesen Schritten:

1. Ihre App initiiert die Anmeldung, indem `InitiateAuth` oder `AdminInitiateAuth` mit der `AuthParameters`-Karte aufgerufen werden. Die Parameter müssen Folgendes enthalten: `CHALLENGE_NAME: SRP_A` und Werte für `SRP_A` und`USERNAME`.
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 Passwortverifizierung erfolgreich ist, passiert eines von zwei Dingen:

   Für Benutzer mit normalem Status:

   Amazon Cognito ruft Ihre Lambda-Funktion erneut mit einer neuen Sitzung auf, die und enthält. `challengeName: PASSWORD_VERIFIER` `challengeResult: true`

   Ihre Lambda-Funktion initiiert Ihre benutzerdefinierten Aufforderungen, indem sie mit `challengeName: CUSTOM_CHALLENGE`, `issueTokens: false` und `failAuthentication: false` antwortet.

   Für Benutzer im Status oder**`RESET_REQUIRED`**: **`FORCE_CHANGE_PASSWORD`**

   Amazon Cognito ruft Ihre Lambda-Funktion mit einer Sitzung auf, die und enthält. `challengeName: PASSWORD_VERIFIER` `challengeResult: true`

   Ihre Lambda-Funktion sollte mit `challengeName: NEW_PASSWORD_REQUIRED`, `issueTokens: false` und `failAuthentication: false` antworten.

   Nach erfolgreicher Kennwortänderung ruft Amazon Cognito Ihre Lambda-Funktion mit einer Sitzung auf, die sowohl die als auch die `PASSWORD_VERIFIER` Ergebnisse enthält. `NEW_PASSWORD_REQUIRED`

   Ihre Lambda-Funktion initiiert Ihre benutzerdefinierten Aufforderungen, indem sie mit `challengeName: CUSTOM_CHALLENGE`, `issueTokens: false` und `failAuthentication: false` antwortet.
5. Die Aufforderungsschleife wird so lange ausgeführt, bis alle Aufforderungen beantwortet sind.

Wenn Sie Ihren benutzerdefinierten Authentifizierungsablauf nicht mit Passwortüberprüfung starten möchten, können Sie die Anmeldung mit der `AuthParameters`-Karte einschließlich `CHALLENGE_NAME: CUSTOM_CHALLENGE` initiieren.

Sitzungsverwaltung

Der Authentifizierungsablauf gewährleistet die Kontinuität der Sitzung während einer Reihe von Sitzungs IDs - und Challenge-Ergebnissen. Jede Challenge-Antwort generiert eine neue Sitzungs-ID, um Fehler bei der Wiederverwendung von Sitzungen zu vermeiden. Dies ist besonders wichtig bei mehrstufigen Authentifizierungsabläufen.

Challenge-Ergebnisse werden chronologisch in dem Session-Array gespeichert, das Ihre Lambda-Trigger empfangen. Für Benutzer mit `FORCE_CHANGE_PASSWORD` Status enthält das Sitzungs-Array:

1. `session[0]`- Erste `SRP_A` Herausforderung
2. `session[1]`- `PASSWORD_VERIFIER` Ergebnis
3. `session[2]`- `NEW_PASSWORD_REQUIRED` Ergebnis
4. Nachfolgende Elemente — Ergebnisse zusätzlicher benutzerdefinierter Herausforderungen

Beispiel für einen Authentifizierungsablauf

Das folgende Beispiel zeigt einen vollständigen benutzerdefinierten Authentifizierungsablauf für einen Benutzer mit `FORCE_CHANGE_PASSWORD` Status, der sowohl eine Passwortänderung als auch eine benutzerdefinierte CAPTCHA-Abfrage durchführen muss.

1. InitiateAuth Anfrage

```
{
    "AuthFlow": "CUSTOM_AUTH",
    "ClientId": "1example23456789",
    "AuthParameters": {
        "CHALLENGE_NAME": "SRP_A",
        "USERNAME": "testuser",
        "SRP_A": "[SRP_A]"
    }
}
```

2. InitiateAuth Antwort

```
{
```

```
    "ChallengeName": "PASSWORD_VERIFIER",
    "ChallengeParameters": {
        "USER_ID_FOR_SRP": "testuser"
    },
    "Session": "[session_id_1]"
}
```

3. RespondToAuthChallenge Anfrage mit **`PASSWORD_VERIFIER`**

```
{
    "ChallengeName": "PASSWORD_VERIFIER",
    "ClientId": "1example23456789",
    "ChallengeResponses": {
        "PASSWORD_CLAIM_SIGNATURE": "[claim_signature]",
        "PASSWORD_CLAIM_SECRET_BLOCK": "[secret_block]",
        "TIMESTAMP": "[timestamp]",
        "USERNAME": "testuser"
    },
    "Session": "[session_id_1]"
}
```

4. RespondToAuthChallenge Antwort mit **`NEW_PASSWORD_REQUIRED`** Herausforderung

```
{
    "ChallengeName": "NEW_PASSWORD_REQUIRED",
    "ChallengeParameters": {},
    "Session": "[session_id_2]"
}
```

5. RespondToAuthChallenge Anfrage mit **`NEW_PASSWORD_REQUIRED`**

```
{
    "ChallengeName": "NEW_PASSWORD_REQUIRED",
    "ClientId": "1example23456789",
    "ChallengeResponses": {
        "NEW_PASSWORD": "[password]",
        "USERNAME": "testuser"
    },
    "Session": "[session_id_2]"
}
```

6. RespondToAuthChallenge Antwort mit benutzerdefinierter CAPTCHA-Herausforderung

```
{
    "ChallengeName": "CUSTOM_CHALLENGE",
    "ChallengeParameters": {
        "captchaUrl": "url/123.jpg"
    },
    "Session": "[session_id_3]"
}
```

7. RespondToAuthChallenge Anfrage mit Antwort auf die benutzerdefinierte CAPTCHA-Herausforderung

```
{
    "ChallengeName": "CUSTOM_CHALLENGE",
    "ClientId": "1example23456789",
    "ChallengeResponses": {
        "ANSWER": "123",
        "USERNAME": "testuser"
    },
    "Session": "[session_id_3]"
}
```

6. Endgültige Erfolgsantwort

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "IdToken": "eyJra789ghiEXAMPLE",
        "RefreshToken": "eyJjd123abcEXAMPLE",
        "TokenType": "Bearer"
    },
    "ChallengeParameters": {}
}
```

Lambda-Auslöser für die Definition einer Authentifizierungsaufforderung

Der Define Auth Challenge Trigger ist eine Lambda-Funktion, die die Abfragesequenz in einem benutzerdefinierten Authentifizierungsablauf beibehält. Sie erklärt den Erfolg oder Misserfolg der Challenge-Sequenz und legt die nächste Challenge fest, falls die Sequenz noch nicht abgeschlossen ist.

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.

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 Challenge-Typen:`CUSTOM_CHALLENGE`,`SRP_A`,`PASSWORD_VERIFIER`,`SMS_MFA`,`EMAIL_OTP`,`SOFTWARE_TOKEN`
`DEVICE_SRP_AUTHDEVICE_PASSWORD_VERIFIER`, oder`ADMIN_NO_SRP_AUTH`.

Wenn Ihre Define Auth Challenge-Funktion eine `PASSWORD_VERIFIER` Herausforderung für einen Benutzer ausgibt, der die Multifaktor-Authentifizierung eingerichtet hat, folgt Amazon Cognito darauf mit einer `SMS_MFAEMAIL_OTP`, oder -Aufforderung. `SOFTWARE_TOKEN_MFA` Dies sind die Eingabeaufforderungen zur Eingabe eines Multi-Faktor-Authentifizierungscodes. Schließen Sie in Ihrer Funktion die Behandlung

von Eingabeereignissen aus `SMS_MFAEMAIL_OTP`, und `SOFTWARE_TOKEN_MFA` Herausforderungen ein. Sie müssen in Ihrer Define Auth Challenge-Funktion keine MFA-Challenges 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 RespondToAuthChallengeAPI-Operationen AdminRespondToAuthChallengeund verwenden. Die Anfrage, die die Funktion define auth challenge aufruft, enthält keine Daten, die in den ClientMetadata Parameter-In AdminInitiateAuth- und InitiateAuthAPI-Vorgängen 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. Wenn Benutzer die SRP-Authentifizierung mit den `SRP_A` und `PASSWORD_VERIFIER` Challenges abschließen, übergibt ihnen diese Funktion eine, `CUSTOM_CHALLENGE` die den Create Auth Challenge-Trigger auslöst. In Kombination mit unserem Beispiel für die Create Auth Challenge liefert diese Sequenz eine CAPTCHA-Herausforderung für Herausforderung drei und eine Sicherheitsfrage für Herausforderung vier.

Nachdem der Benutzer das CAPTCHA gelöst und die Sicherheitsfrage beantwortet hat, bestätigt diese Funktion, dass Ihr Benutzerpool Token ausstellen kann. Eine SRP-Authentifizierung ist nicht erforderlich. Sie können das CAPTCHA und die Sicherheitsfrage auch als Herausforderung eins und zwei festlegen. Für den Fall, dass Ihre Funktion zum Definieren einer Authentifizierungsabfrage keine SRP-Herausforderungen deklariert, hängt der Erfolg Ihrer Benutzer ausschließlich von ihren Antworten auf Ihre benutzerdefinierten Eingabeaufforderungen ab.

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

Der Create Auth Challenge-Trigger ist eine Lambda-Funktion, die die Details jeder Abfrage enthält, die durch den Define Auth Challenge-Trigger deklariert wurde. Sie verarbeitet den durch den Define Auth Challenge-Trigger deklarierten Challenge-Namen und gibt einen Wert zurück`publicChallengeParameters`, den Ihre Anwendung dem Benutzer präsentieren muss. Diese Funktion liefert Ihrem Benutzerpool dann die Antwort auf die Herausforderung`privateChallengeParameters`, die Ihr Benutzerpool an den Verify Auth Challenge-Trigger weiterleitet. Während Ihr Define Auth Challenge-Trigger die Challenge-Sequenz verwaltet, verwaltet Ihr Create Auth Challenge-Trigger den Inhalt der Herausforderung.

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: Einer von:`"CUSTOM_CHALLENGE"`,,`"PASSWORD_VERIFIER"`,`"SMS_MFA"`, `"DEVICE_SRP_AUTH"` `"DEVICE_PASSWORD_VERIFIER""NEW_PASSWORD_REQUIRED"`, oder. `"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 AdminRespondToAuthChallengeund RespondToAuthChallengeAPI-Aktionen verwenden, um diese Daten an Ihre Lambda-Funktion zu übergeben. Die Anfrage, die die Funktion create auth challenge aufruft, enthält keine Daten, die in den ClientMetadata Parameter-In AdminInitiateAuth- und InitiateAuthAPI-Vorgängen 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“

Diese Funktion hat zwei benutzerdefinierte Herausforderungen, die der Challenge-Sequenz in unserem Beispiel für eine Define Auth Challenge entsprechen. Bei den ersten beiden Herausforderungen handelt es sich um die SRP-Authentifizierung. Bei der dritten Challenge gibt diese Funktion in der Challenge-Antwort eine CAPTCHA-URL an Ihre Anwendung zurück. Ihre Anwendung rendert das CAPTCHA unter der angegebenen URL und gibt die Benutzereingabe zurück. 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.

Für die vierte Herausforderung gibt diese Funktion eine Sicherheitsfrage zurück. Ihre Anwendung rendert die Frage und fordert den Benutzer zur Eingabe seiner Antwort auf. Nachdem Benutzer beide benutzerdefinierten Herausforderungen gelöst haben, bestätigt der Trigger zur Definition der Authentifizierungsabfrage, dass Ihr Benutzerpool Token ausstellen kann.

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

Der Verify Auth Challenge-Trigger ist eine Lambda-Funktion, die die von einem Benutzer bereitgestellte Antwort mit einer bekannten Antwort vergleicht. Diese Funktion teilt Ihrem Benutzerpool mit, ob der Benutzer die Anfrage richtig beantwortet hat. Wenn der Challenge-Trigger zur Überprüfung `answerCorrect` der Authentifizierung mit einem Aus antwortet`true`, kann die Authentifizierungssequenz fortgesetzt werden.

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. 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 Verify Auth Challenge-Funktion weiterleitet.

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“

Diese Funktion zur Überprüfung der Authentifizierung überprüft, ob die Antwort des Benutzers auf eine Abfrage mit der erwarteten Antwort übereinstimmt. Die Antwort des Benutzers wird durch Eingaben aus Ihrer Anwendung definiert, und die bevorzugte Antwort wird durch die Antwort aus der Antwort `privateChallengeParameters.answer` auf den Trigger „Create Auth

Challenge“ definiert. Sowohl die richtige Antwort als auch die angegebene Antwort sind Teil des Eingabeereignisses für diese Funktion.

Wenn in diesem Beispiel die Antwort des Benutzers mit der erwarteten Antwort übereinstimmt, setzt Amazon Cognito den `answerCorrect` Parameter auf`true`.

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 dem Featureplan Essentials oder Plus können Sie Version zwei oder ein Trigger-Ereignis mit Anpassung des Zugriffstokens und das `V2_0` Trigger-Ereignis der Version drei oder `V3_0` das Trigger-Ereignis mit Anpassung des Zugriffstokens für (M2M) -Zugriffstoken für machine-to-machine (M2M) -Zugriffs-Zugangsdaten generieren.

Amazon Cognito sendet ein `V1_0`-Ereignis als Anfrage an Ihre Funktion mit Daten, die in das Identitätstoken geschrieben werden. Ein `V2_0` `V3_0` OR-Ereignis ist eine einzelne Anfrage mit den Daten, die Amazon Cognito sowohl in die Identitäts- als auch in die Zugriffstoken schreiben würde. Um beide Token anzupassen, müssen Sie Ihre Funktion so aktualisieren, dass sie die Trigger-Version zwei oder drei verwendet, und Daten für beide Token in derselben Antwort senden.

Amazon Cognito wendet Ereignisantworten der zweiten Version auf Zugriffstoken aus der Benutzerauthentifizierung an, bei der ein menschlicher Benutzer Anmeldeinformationen für Ihren Benutzerpool vorgelegt hat. Die Ereignisantworten der dritten Version gelten für Zugriffstoken aus der Benutzerauthentifizierung und der Maschinenauthentifizierung, bei denen automatisierte Systeme Zugriffstoken-Anfragen mit App-Client-Geheimnissen autorisieren. Abgesehen von den Umständen, unter denen die resultierenden Zugriffstoken entstanden sind, sind die Ereignisse der Versionen zwei und drei identisch.

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` `V3_0` OR-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 bei der Konfiguration Ihres Triggers in der Amazon Cognito Cognito-Konsole eine Trigger-Event-Version von Basisfunktionen + Zugriffstoken-Anpassung für Benutzeridentitäten aus. Um `V3_0` Ereignisse zu erzeugen, wählen Sie Basisfunktionen + Anpassung des Zugriffstokens für Benutzer- und Maschinenidentitäten. Sie können den Wert von auch `LambdaVersion` in den LambdaConfigParametern einer UpdateUserPool oder CreateUserPool API-Anfrage festlegen. Die Event-Versionen eins, zwei und drei sind in den Featureplänen Essentials und Plus verfügbar. M2M-Operationen für Ereignisse der Version drei haben eine Preisstruktur, die sich von der Formel für monatlich aktive Benutzer (MAU) unterscheidet. Weitere Informationen finden Sie unter Amazon Cognito – Preise.

> **Note**
>
> Benutzerpools, die am oder vor dem 22. November 2024 um 18:00 Uhr GMT mit der Option Erweiterte Sicherheitsfunktionen betriebsbereit waren und weiterhin auf der Lite-Funktionsebene verbleiben, haben Zugriff auf die Eventversionen eins und zwei des Triggers vor der Token-Generierung. Benutzerpools in dieser älteren Stufe ohne erweiterte Sicherheitsfunktionen haben Zugriff auf Event Version 1. Version drei ist nur in Essentials und Plus verfügbar.

Referenz zu Ansprüchen und Geltungsbereichen

Amazon Cognito schränkt die Ansprüche und Bereiche ein, die Sie in Zugriffs- und Identitäts-Token hinzufügen, ändern oder unterdrücken können. In der folgenden Tabelle werden die Ansprüche beschrieben, die Ihre Lambda-Funktion ändern kann und die nicht, sowie die Triggerereignisparameter, die sich auf das Vorhandensein oder den Wert des Anspruchs auswirken.

Antrag	Standa Tokent	Kann ich hinzufü n?	Kann ändern	Kann unterdr ken?	Ereignisp arameter — hinzufüge n oder ändern	Ereignisp arameter — unterdrüc ken	Identität styp	Version des Ereignisses
Jeder Anspruch, der nicht im Token-Sch ema des Benutzerp ools enthalten ist	Keine	Ja	Ja	N/A	`claimsTo` `ddOrOver` `ide`	`claimsTo` `uppress`	Benutzer, Maschine [1]	Alle [2]
`scope`	Zugriff	Ja	Ja	Ja	`scopesTo` `dd`	`scopesTo` `uppress`	Benutzer, Maschine [1]	`v2_0`, `v3_0`
`cognito:` `roups`	ID, Zugang	Ja	Ja	Ja	`groupsTo` `verride`	`claimsTo` `uppress`	Benutzer	Alle [2]
`cognito:` `referred_` `role`	ID	Ja	Ja	Ja	`preferre` `Role`	`claimsTo` `uppress` [3]	Benutzer	Alle
`cognito:` `oles`	ID	Ja	Ja	Ja	`iamRoles` `oOverrid`	`claimsTo` `uppress` [3]	Benutzer	Alle
`cognito:` `sername`	ID	Nein	Nein	Nein	N/A	N/A	Benutzer	N/A

Antrag	Standa Tokent	Kann ich hinzufü n?	Kann ändern	Kann unterdr ken?	Ereignisp arameter — hinzufüge n oder ändern	Ereignisp arameter — unterdrüc ken	Identität styp	Version des Ereignisses
Jeder andere Anspruch mit einem `cognito:` Präfix	Keine	Nein	Nein	Nein	N/A	–	–	N/A
`username`	Zugriff	Nein	Nein	Nein	N/A	N/A	Benutzer	`v2_0`, `v3_0`
`sub`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer	N/A
Standard-OIDC-Attribut	ID	Ja	Ja	Ja	`claimsTo` `ddOrOver` `ide`	`claimsTo` `uppress`	Benutzer	Alle
`custom:` Attribut	ID	Ja	Ja	Ja	`claimsTo` `ddOrOver` `ide`	`claimsTo` `uppress`	Benutzer	Alle
`dev:` Attribut	ID	Nein	Nein	Ja	N/A	`claimsTo` `uppress`	Benutzer	Alle
`identiti` `s`	ID	Nein	Nein	Nein	N/A	N/A	Benutzer	N/A
`aud`[4]	ID	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`client_i`	Zugriff	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A

Antrag	Standa Tokent	Kann ich hinzufü n?	Kann ändern	Kann unterdr ken?	Ereignisp arameter — hinzufüge n oder ändern	Ereignisp arameter — unterdrüc ken	Identität styp	Version des Ereignisses
`event_id`	Zugriff	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`device_ke y`	Zugriff	Nein	Nein	Nein	N/A	N/A	Benutzer	N/A
`version`	Zugriff	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`acr`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`amr`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`at_hash`	ID	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`auth_time`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`azp`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`exp`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`iat`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`iss`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A

Antrag	Standa Tokent	Kann ich hinzufü n?	Kann ändern	Kann unterdr ken?	Ereignisp arameter — hinzufüge n oder ändern	Ereignisp arameter — unterdrüc ken	Identität styp	Version des Ereignisses
`jti`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`nbf`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`nonce`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`origin_ji i`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A
`token_use`	ID, Zugang	Nein	Nein	Nein	N/A	N/A	Benutzer, Maschine	N/A

[1] Zugriffstoken für Maschinenidentitäten sind nur mit dem `v3_0` Trigger-Eingabeereignis verfügbar. Event-Version drei ist nur in den Feature-Tarifen Essentials und Plus verfügbar. Benutzerpools auf der Lite-Stufe können `v1_0` Ereignisse empfangen. Benutzerpools auf der Lite-Stufe mit erweiterten Sicherheitsfunktionen können `v2_0` Ereignisse empfangen`v1_0`.

[2] Konfigurieren Sie Ihren Trigger vor der Token-Generierung auf die Event-Version nur `v1_0` für ID-Token, `v2_0` für ID und Zugriffstoken, `v3_0` für ID und Zugriffstoken mit Funktionen für Maschinenidentitäten.

[3] Um die `cognito:roles` Ansprüche `cognito:preferred_role` und zu unterdrücken, fügen Sie `cognito:groups` zu `claimsToSuppress` hinzu.

[4] Sie können einen `aud` Anspruch auf Zugriffstoken hinzufügen, dessen Wert muss jedoch 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

Bei allen Eventversionen des Lambda-Triggers vor der Token-Generierung können Sie den Inhalt eines Identitätstokens (ID) aus Ihrem Benutzerpool 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 Grundlegendes zum Identitätstoken (ID).

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 den Ereignisversionen zwei und drei des Lambda-Triggers vor der Token-Generierung können Sie den Inhalt eines Zugriffstokens aus Ihrem Benutzerpool anpassen. Das Zugriffstoken autorisiert Benutzer, Informationen aus zugriffsgeschützten Ressourcen wie Token-autorisierten Amazon Cognito Cognito-API-Vorgängen und Drittanbietern abzurufen. APIs Für die machine-to-machine (M2M-) Autorisierung mit einer Gewährung von Kundenanmeldedaten ruft Amazon Cognito den Trigger vor der Token-Generierung nur auf, wenn Ihr Benutzerpool für ein Ereignis der Version drei () `V3_0` konfiguriert ist. Weitere Informationen zu Zugriffs-Token finden Sie unter Das Zugriffstoken verstehen.

Der Lambda-Trigger vor der Token-Generierung mit einem Zugriffs-Token kann unter anderem wie folgt verwendet werden.

- Fügen Sie dem Anspruch Bereiche hinzu oder unterdrücken Sie diese. `scope` 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. Wählen Sie das Menü Erweiterungen und suchen Sie nach Lambda-Triggern.
4. Fügen Sie einen Trigger für die Pre-Token-Generierung hinzu oder bearbeiten Sie ihn.
5. Wählen Sie unter Zuweisen einer Lambda-Funktion eine Lambda-Funktion aus.
6. Wählen Sie eine Trigger-Event-Version von Basisfunktionen + Anpassung von Zugriffstoken für Benutzeridentitäten oder Basisfunktionen + Zugriffstoken-Anpassung für Benutzer- und Maschinenidentitäten. 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 CreateUserPoolodder UpdateUserPoolAPI-Anfrage. Sie müssen einen Wert für alle Parameter angeben, die nicht auf einen Standardwert festgelegt werden sollen. Weitere Informationen finden Sie unter Aktualisierung der Benutzerpool- und App-Client-Konfiguration.

Nehmen Sie den folgenden Inhalt in den `LambdaVersion`-Parameter Ihrer Anfrage auf. Ein `LambdaVersion` Wert von `V2_0` veranlasst Ihren Benutzerpool, Parameter für Zugriffstoken

hinzuzufügen und Änderungen an diesen vorzunehmen. Der `LambdaVersion` Wert von `V3_0` erzeugt dasselbe Ereignis wie`V2_0`, veranlasst Ihren Benutzerpool jedoch, auch Änderungen an M2M-Zugriffstoken vorzunehmen. 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": "V3_0"
},
```

Client-Metadaten für machine-to-machine (M2M-) Client-Anmeldeinformationen

Sie können Client-Metadaten in M2M-Anfragen übergeben. Client-Metadaten sind zusätzliche Informationen aus einer Benutzer- oder Anwendungsumgebung, die zu den Ergebnissen einer Lambda-Auslöser für die Vorab-Generierung von Token beitragen können. Bei Authentifizierungsvorgängen mit einem Benutzerprinzipal können Sie Client-Metadaten im Hauptteil von AdminRespondToAuthChallengeund RespondToAuthChallengeAPI-Anfragen an den Trigger vor der Token-Generierung übergeben. Da Anwendungen den Ablauf für die Generierung von Zugriffstoken für M2M mit direkten Anfragen an die durchführenToken-Endpunkt, haben sie ein anderes Modell. Übergeben Sie im POST-Hauptteil der Token-Anfragen für Client-Anmeldeinformationen einen `aws_client_metadata` Parameter mit dem Client-Metadatenobjekt URL-encoded () `x-www-form-urlencoded` an string. Eine Beispielanforderung finden Sie unter Kundenanmeldedaten mit Basisautorisierung. Im Folgenden finden Sie einen Beispielparameter, der die Schlüssel-Wert-Paare übergibt. `{"environment": "dev", "language": "en-US"}`

```
aws_client_metadata=%7B%22environment%22%3A%20%22dev%22,%20%22language%22%3A%20%22en-US
%22%7D
```

Weitere -Quellen

- So passen Sie Zugriffstoken in Amazon Cognito Cognito-Benutzerpools an

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 während der Authentifizierung von der verwalteten Anmeldeseite mit Amazon Cognito aus aufgerufen.
`TokenGeneration_Authentication`	Wird aufgerufen, nachdem Benutzer-Authentifizierung-Abläufe abgeschlossen sind.
`TokenGeneration_NewPasswordChallenge`	Wird aufgerufen, nachdem der Benutzer von einem Administrator erstellt wurde. Dieser Ablauf wird aufgerufen, wenn der Benutzer ein temporäres Passwort ändern muss.
`TokenGeneration_ClientCredentials`	Wird nach der Gewährung einer M2M-Client-Anmeldedaten aufgerufen. Ihr Benutzerpool sendet dieses Ereignis nur, wenn es Ihre Event-Version ist`V3_0`.
`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 one

Mit dem Versions-Eins-Token können Gruppenmitgliedschaften, IAM-Rollen und neue Ansprüche in ID-Token festgelegt werden. Überschreibungen der Gruppenmitgliedschaft gelten auch für den `cognito:groups` Anspruch in Zugriffstoken.

```
{
    "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"
            }
        }
    }
}
```

Versions two and three

In den Versionen zwei und drei werden Anforderungsereignisse mit Feldern hinzugefügt, mit denen das Zugriffstoken individuell angepasst werden kann. Benutzerpools wenden Änderungen von Ereignissen aus Version drei auf Zugriffstoken für Maschinenidentitäten an. Diese Versionen bieten auch Unterstützung für komplexe `claimsToOverride` Datentypen im Antwortobjekt. 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

Name	Beschreibung	Minimale Trigger-Ereignisversion
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
Bereiche	Greifen Sie auf Token-Bereiche zu. 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`-Auslöser ereignis.	1
claimsAndScopeOverrideDetails	Ein Container für alle Elemente in einem Ereignis `V2_0` oder einem `V3_0` Trigger-Ereignis.	2
idTokenGeneration	Die Ansprüche, die Sie im ID-Token Ihres Benutzers überschreiben, hinzufügen oder unterdrücken möchten. Diese Werte für die Anpassung des übergeordneten ID-Tokens werden nur in Ereignisversion 2 und höher angezeigt, die untergeordneten Elemente jedoch in Ereignissen der Version 1.	2
accessTokenGeneration	Die Ansprüche, die Sie im Zugriffs-Token Ihres Benutzers überschreiben, hinzufügen oder unterdrücken möchten. Dieses übergeordnete Element für den Zugriff auf Token-Anpassungswerte erscheint nur in Eventversion 2 und höher.	2
claimsToAddOrOverride	Eine Zuordnung von einem oder mehreren Ansprüchen und deren Werte, die Sie hinzufügen oder ändern möchten. Für gruppenbezogene Ansprüche verwenden Sie stattdessen `groupOverrideDetails` . In Eventversion 2 und höher erscheint dieses Element `accessTokenGeneration` sowohl unter als auch `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.	1

Name	Beschreibung	Minimale Trigger-Ereignisversion
	In Eventversion 2 und höher erscheint dieses Element `accessTokenGeneration` sowohl unter als auchidTokenGeneration .	
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. Gruppenüberschreibungen sind die einzigen Änderungen am Zugriffstoken, die Ereignisse der Version 1 vornehmen können.	1
scopesToAdd	Eine Liste der Bereiche, die Sie dem `scope` Anspruch im Zugriffstoken Ihres Benutzers hinzufügen möchten. Sie können keine Bereichswerte hinzufügen, die ein oder mehrere Leerzeichen enthalten.	2
scopesToSuppress	Eine Liste der Bereiche, die Sie aus dem `scope` Anspruch im Zugriffstoken Ihres Benutzers entfernen möchten.	2

* Antwortobjekte auf Ereignisse der ersten Version können Zeichenketten zurückgeben. Antwortobjekte auf Ereignisse der Versionen zwei und drei 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 den `email` Anspruch 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"]
            },
            "accessTokenGeneration": {
                "claimsToAddOrOverride": claims,
                "claimsToSuppress": ["email"],
                "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::XXXXXXXXXXXX: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 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	Ereignis
`UserMigration_Authentication` [1]	Benutzermigration bei der Anmeldung.
`UserMigration_ForgotPassword`	Benutzermigration während des Ablaufs bei vergessenem Passwort.

[1] Amazon Cognito ruft diesen Trigger nicht auf, wenn sich Benutzer mit passwortloser Anmeldung authentifizieren.

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 ClientMetadata Parameter in den Aktionen InitiateAuthund AdminInitiateAuthAPI 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 ClientMetadata Parameter in den Aktionen AdminRespondToAuthChallengeund ForgotPasswordAPI 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	Wenn erforderliche Attribute für die Migration fehlen, verwendet Amazon Cognito Standardwerte.

Attribute	Anforderung
Benutzerpools als erforderlich gekennzeichnet sind	
`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 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.
> Wenn das Passwort dann nicht der Richtlinie entspricht, setzen Sie es finalUserStatus aufRESET_REQUIRED.

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 setzen `TRUE` und die Telefonnummer oder E-Mail-Adresse im UserAttributes Parameter bereits als Alias für einen anderen Benutzer existiert, 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

```
exports.handler = (event, context, callback) => {
  var user;

  if (event.triggerSource == "UserMigration_Authentication") {
    // authenticate the user with your existing user directory service
    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";
      context.succeed(event);
    } else {
      // Return error to Amazon Cognito
      callback("Bad password");
    }
  } else if (event.triggerSource == "UserMigration_ForgotPassword") {
    // Lookup the user in your existing user directory service
    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";
      context.succeed(event);
    } else {
      // Return error to Amazon Cognito
      callback("Bad password");
    }
  } else {
    // Return error to Amazon Cognito
    callback("Bad triggerSource " + event.triggerSource);
  }
};
```

Lambda-Auslöser für benutzerdefinierte Nachrichten

Wenn Sie über einen externen Standard für die E-Mail- und SMS-Nachrichten verfügen, die Sie an Ihre Benutzer senden möchten, oder wenn Sie zur Laufzeit Ihre eigene Logik auf die Formatierung von Benutzernachrichten anwenden möchten, fügen Sie Ihrem Benutzerpool einen benutzerdefinierten Nachrichtenauslöser hinzu. Die benutzerdefinierte Nachricht Lambda empfängt den Inhalt aller E-Mail- und SMS-Nachrichten, bevor Ihr Benutzerpool sie sendet. Ihre Lambda-Funktion hat dann die Möglichkeit, den Inhalt und den Betreff der Nachricht zu ändern.

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.

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**
>
> Das Eingabeereignis für eine Lambda-Funktion mit benutzerdefinierter Nachricht und der `CustomMessage_AdminCreateUser` Triggerquelle umfasst einen Benutzernamen und einen Bestätigungscode. Da ein vom Administrator erstellter Benutzer sowohl seinen Benutzernamen als auch seinen Code erhalten muss, muss die Antwort Ihrer Funktion Platzhaltervariablen für den Benutzernamen und den Code enthalten. Die

Platzhalter für Ihre Nachricht sind die Werte von und. `request.usernameParameter` `request.codeParameter` Diese Werte sind in der Regel `{username}` und`{####}`. Es hat sich bewährt, auf die Eingabewerte zu verweisen, anstatt die Variablennamen fest zu codieren.

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

triggerSource-Wert	Ereignis
	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 Anfrage, die eine benutzerdefinierte Nachrichtenfunktion aufruft, enthält keine Daten, die in den ClientMetadata Parameter-In AdminInitiateAuth- und InitiateAuthAPI-Vorgängen 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 Zeichenketteneinschränkungen, 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 `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",
 "region": "us-west-2",
 "userPoolId": "us-west-2_EXAMPLE",
 "userName": "test-user",
 "callerContext": {
  "awsSdkVersion": "aws-sdk-unknown-unknown",
  "clientId": "1example23456789"
 },
 "triggerSource": "CustomMessage_SignUp",
```

```
 "request": {
  "userAttributes": {
   "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
   "cognito:user_status": "CONFIRMED",
   "email_verified": "true",
   "phone_number_verified": "true",
   "phone_number": "+12065551212",
   "email": "test-user@example.com"
  },
  "codeParameter": "{####}",
  "linkParameter": "{##Click Here##}",
  "usernameParameter": "None"
 },
 "response": {
  "smsMessage": "None",
  "emailMessage": "None",
  "emailSubject": "None"
 }
}
```

Benutzerdefinierte Nachricht für das Beispiel, bei dem der Administrator einen Benutzer anlegt

Die Anfrage, die Amazon Cognito an diese Lambda-Funktion für eine benutzerdefinierte Beispielnachricht gesendet hat, hat den `triggerSource` Wert `CustomMessage_AdminCreateUser` und einen Benutzernamen und ein temporäres Passwort. Die Funktion wird `${event.request.codeParameter}` anhand des temporären Kennworts in der Anfrage und anhand des Benutzernamens in `${event.request.usernameParameter}` der Anfrage aufgefüllt.

Ihre benutzerdefinierten Nachrichten müssen die Werte von `codeParameter` und `usernameParameter` in `smsMessage` und `emailMessage` in das 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

Das Lambda löst E-Mail `CustomEmailSender` - und SMS-Benachrichtigungen von Drittanbietern in Benutzerpools aus und `CustomSMSSender` unterstützt diese. Sie können SMS- und E-Mail-Anbieter auswählen, um Benachrichtigungen an Benutzer aus Ihrem Lambda-Funktionscode zu senden. Wenn Amazon Cognito Einladungen, MFA-Codes, Bestätigungscodes, Bestätigungscodes und temporäre Passwörter an Benutzer sendet, 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 vom AWS KMS Kunden verwalteten Schlüssel und dem. AWS Encryption SDK Die AWS Encryption SDK ist eine clientseitige Verschlüsselungsbibliothek, mit der Sie generische Daten ver- und entschlüsseln können.

CustomEmailSender

Amazon Cognito ruft diesen Auslöser auf, um E-Mail-Benachrichtigungen an Benutzer zu senden.

Custom (Benutzerdefiniert)SMSSender

Amazon Cognito ruft diesen Auslöser auf, um SMS-Benachrichtigungen an Benutzer zu senden.

Konzepte der Verschlüsselung

Amazon Cognito sendet Benutzercodes in den Ereignissen, die es an benutzerdefinierte Absenderauslöser sendet, nicht im Klartext. Die Lambda-Funktionen müssen die Codes in den Ereignissen entschlüsseln. Die folgenden Konzepte beschreiben die Verschlüsselungsarchitektur, die Ihre Funktion verwenden muss, um Codes abzurufen, die sie Benutzern zur Verfügung stellen kann.

AWS KMS

AWS KMS ist ein verwalteter Dienst 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. Um einen symmetrischen KMS-Schlüssel zu verwenden, müssen Sie anrufen AWS KMS. Amazon Cognito verwendet symmetrische Schlüssel. Derselbe Schlüssel verschlüsselt und entschlüsselt. Weitere Informationen finden Sie unter Symmetrische KMS-Schlüssel.

Wissenswertes über Lambda-Trigger mit benutzerdefiniertem Sender

- 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.

  Die `UpdateUserPool` Operation legt die Lambda-Konfiguration fest. Anfragen für diesen Vorgang erfordern 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. Fügen Sie, wie im folgenden AWS CLI-Beispiel gezeigt, Einträge für alle Lambda-Funktionen hinzu, die Sie zu Ihrem Benutzerpool hinzufügen oder in diesem behalten möchten. Weitere Informationen finden Sie unter Aktualisierung der Benutzerpool- und App-Client-Konfiguration.

```
    #Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
```

```
    #user pool configurations. This snippet also includes a pre sign-up trigger for
 syntax reference. The pre sign-up trigger
    #doesn't have a role in custom sender triggers.

      --lambda-config "PreSignUp=lambda-arn, \
                       CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                       CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                       KMSKeyID=key-id"
```

Für Anfragen, die den JSON-Text `UpdateUserPool` des folgenden `LambdaConfig` Snippets verwenden, werden benutzerdefinierte SMS- und E-Mail-Absenderfunktionen zugewiesen.

```
"LambdaConfig": {
   "KMSKeyID": "arn:aws:kms:us-
east-1:111122223333:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
   "CustomEmailSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:111122223333:function:MyFunction",
      "LambdaVersion": "V1_0"
   },
   "CustomSMSSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:111122223333:function:MyFunction",
      "LambdaVersion": "V1_0"
   }
```

- Um einen benutzerdefinierten Lambda-Sender-Trigger mit einem `update-user-pool` AWS CLI Befehl zu entfernen, lassen Sie den `CustomEmailSender` Parameter `CustomSMSSender` oder von weg `--lambda-config` und schließen Sie alle anderen Trigger ein, 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.
- 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.

Lambda-Trigger mit benutzerdefiniertem Absender aktivieren

Um benutzerdefinierte Logik zum Senden von SMS- oder E-Mail-Nachrichten für Ihren Benutzerpool zu verwenden, richten Sie benutzerdefinierte Absender-Trigger ein. Anhand des folgenden Verfahrens weisen Sie Ihrem Benutzerpool einen benutzerdefinierten SMS-Auslöser, einen benutzerdefinierten E-Mail-Auslöser oder beides zu. Nachdem Sie Ihren benutzerdefinierten Absender-Trigger hinzugefügt haben, sendet Amazon Cognito immer Benutzerattribute, einschließlich der Telefonnummer und des Einmalcodes, an Ihre Lambda-Funktion, anstatt das Standardverhalten, das eine SMS- oder E-Mail-Nachricht sendet.

1. Erstellen Sie in () einen symmetrischen Verschlüsselungsschlüssel. AWS Key Management Service AWS KMS Amazon Cognito generiert Geheimnisse — temporäre Passwörter, Bestätigungscodes, Einmalkennwörter für die Authentifizierung und Bestätigungscodes — und verwendet dann diesen KMS-Schlüssel, um die Geheimnisse zu verschlüsseln. AWS Encryption SDK Sie können dann die Funktion AWS Encryption SDK in Ihrer Lambda-Funktion verwenden, um die Geheimnisse zu entschlüsseln und sie im Klartext an den Benutzer zu senden.
2. Der IAM-Prinzipal, der Ihren Benutzerpool erstellt oder aktualisiert, erstellt einen einmaligen Zuschuss für den KMS-Schlüssel, den Amazon Cognito zur Verschlüsselung des Codes verwendet. Erteilen Sie diesem `CreateGrant` Prinzipalberechtigungen für Ihren KMS-Schlüssel. Damit dieses Beispiel für eine KMS-Schlüsselrichtlinie wirksam wird, muss der Administrator, der den Benutzerpool aktualisiert, mit einer Sitzung mit angenommener Rolle für die IAM-Rolle angemeldet sein. `arn:aws:iam::111222333444:role/my-example-administrator-role`

   Wenden Sie die folgende, an Ihre Umgebung angepasste, ressourcenbasierte Richtlinie auf Ihren KMS-Schlüssel an.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:role/my-example-administrator-role"
        },
        "Action": "kms:CreateGrant",
        "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1example-2222-3333-4444-999example",
        "Condition": {
```

```
            "StringEquals": {
               "kms:EncryptionContext:userpool-id": "us-west-2_EXAMPLE"
            }
        }
    },
    {
        "Sid": "Allow Lambda to decrypt",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:role/my-lambda-function-role"
        },
        "Action": "kms:Decrypt",
        "Resource": "*"
    }]
}
```

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.

   a. Weisen Sie eine Lambda-Ausführungsrolle zu, die mindestens über `kms:Decrypt` Berechtigungen für Ihren KMS-Schlüssel verfügt.

   b. Verfassen Sie den Lambda-Funktionscode, um Ihre Nachrichten zu senden. Das Eingabeereignis für Ihre Funktion enthält ein Geheimnis. Entschlüsseln Sie in Ihrer Funktion das Geheimnis mit dem AWS Encryption SDK 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.

   c. Fügen Sie das 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.

      i. Exportieren Sie Ihre Lambda-Funktion als ZIP-Datei in die AWS Management Console.

      ii. Öffnen Sie Ihre Funktion und fügen Sie die AWS Encryption SDK hinzu. Weitere Informationen und Download-Links finden Sie unter AWS Encryption SDK – Programmiersprachen im Entwicklerhandbuch zum AWS Encryption SDK .

      iii. 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 .

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 Erlaubnis, Ihre Lambda-Funktion aufzurufen:

   ```
   aws lambda add-permission --function-name lambda_arn --statement-id
    "CognitoLambdaInvokeAccess" --action lambda:InvokeFunction --principal cognito-
   idp.amazonaws.com
   ```

5. Generieren Sie eine UpdateUserPoolAPI-Anfrage mit einem `LambdaConfig` Parameter, der benutzerdefinierte Lambda-Sender-Trigger hinzufügt. Sie können in der Amazon Cognito Cognito-Konsole keine Auslöser dieses Typs hinzufügen. Benutzerdefinierte Absender-Trigger erfordern die `LambdaConfig` Parameter `KMSKeyID` und `CustomSMSSender` oder `CustomEmailSender` (oder beide).

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 an Ihre Benutzer senden. Der benutzerdefinierte Code Ihrer Funktion muss alle E-Mail-Nachrichten aus Ihrem Benutzerpool verarbeiten und versenden.

Dieser Trigger eignet sich für Szenarien, in denen Sie möglicherweise mehr Kontrolle darüber haben möchten, wie Ihr Benutzerpool E-Mail-Nachrichten sendet. Ihre Lambda-Funktion kann den Aufruf von Amazon SES SES-API-Vorgängen anpassen, z. B. wenn Sie mehrere verifizierte Identitäten oder Kreuzungen verwalten möchten. AWS-Regionen Ihre Funktion kann auch Nachrichten an ein anderes Übermittlungsmedium oder einen Drittanbieter-Service weiterleiten.

Informationen zur Konfiguration eines benutzerdefinierten E-Mail-Absender-Triggers finden Sie unterLambda-Trigger mit benutzerdefiniertem Absender aktivieren.

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	Ereignis
`CustomEmailSender_SignUp`	Ein Benutzer meldet sich an und Amazon Cognito sendet eine Willkommensnachricht.
`CustomEmailSender_Authentication`	Ein Benutzer meldet sich an und Amazon Cognito sendet einen Multi-Faktor-Authentifizierungscode (MFA).
`CustomEmailSender_ForgotPassword`	Ein Benutzer fordert einen Code an, um sein Passwort zurückzusetzen.
`CustomEmailSender_ResendCode`	Ein Benutzer fordert einen Ersatzcode für die Kontobestätigung an.
`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.
`CustomEmailSender_AccountTakeOverNotification`	Amazon Cognito erkennt einen Versuch, ein Benutzerkonto zu übernehmen, und sendet dem Benutzer eine Benachrichtigung.

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

Typ

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. Um diese Daten an Ihre Lambda-Funktion zu übergeben, können Sie den ClientMetadata Parameter in den Aktionen AdminRespondToAuthChallengeund RespondToAuthChallengeAPI verwenden. Amazon Cognito bezieht keine Daten aus dem ClientMetadata Parameter in AdminInitiateAuthund InitiateAuthAPI-Operationen in die Anfrage ein, die es an die Post-Authentifizierungsfunktion weiterleitet.

> **Note**
>
> Amazon Cognito sendet bei Ereignissen mit `ClientMetadata` den folgenden Triggerquellen an benutzerdefinierte E-Mail-Auslösefunktionen:
>
> - `CustomEmailSender_ForgotPassword`

- `CustomEmailSender_SignUp`
- `CustomEmailSender_Authentication`

Amazon Cognito sendet keine Trigger-Ereignisse mit `ClientMetadata` der Quelle `CustomEmailSender_AccountTakeOverNotification` ein.

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 Lambda-Funktion muss das Ereignis interpretieren und den Code entschlüsseln und dann den Nachrichteninhalt übermitteln. Eine typische Funktion stellt eine E-Mail-Nachricht zusammen und leitet sie an ein SMTP-Relay eines Drittanbieters weiter.

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_ID`

Die ID des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln der Benutzercodes verwenden möchten.

`KEY_ARN`

Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln von Benutzercodes verwenden möchten.

Um diese Funktion bereitzustellen

1. Installieren Sie die neueste Version von NodeJS in Ihrem Developer Workspace.
2. Erstellen Sie ein neues NodeJS-Projekt in Ihrem Workspace.
3. Initialisieren Sie Ihr Projekt mit. `npm init -y`
4. Erstellen Sie das Skript für die Lambda-Funktion:`touch index.mjs`.

5. Fügen Sie den Inhalt des folgenden Beispiels in ein`index.mjs`.
6. Laden Sie die Projektabhängigkeit herunter, AWS Encryption SDK:`npm install @aws-crypto/client-node`.
7. Komprimieren Sie das Projektverzeichnis in eine Datei:`zip -r my_deployment_package.zip .`.
8. Stellen Sie die ZIP-Datei für Ihre Funktion bereit.

Diese Beispielfunktion entschlüsselt den Code und simuliert bei Anmeldeereignissen das Senden einer E-Mail-Nachricht an die E-Mail-Adresse des Benutzers.

```
import { KmsKeyringNode, buildClient, CommitmentPolicy } from '@aws-crypto/client-
node';

// Configure the encryption SDK client with the KMS key from the environment variables
const { encrypt, decrypt } = buildClient(
    CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT
);

const generatorKeyId = process.env.KEY_ID;
const keyIds = [process.env.KEY_ARN];
const keyring = new KmsKeyringNode({ generatorKeyId, keyIds });

// Example function to simulate sending email.
// This example logs message details to CloudWatch Logs from your Lambda function.
// Update this function with custom logic that sends an email message to 'emailaddress'
 with body 'message'.
const sendEmail = async (emailAddress, message) => {
    // Log the destination with the email address masked.
    console.log(`Simulating email send to ${emailAddress.replace(/[^@.]/g, '*')}`);
    // Log the message with the code masked.
    console.log(`Message content: ${message.replace(/\b\d{6,8}\b/g, '********')}`);
    // Simulate API delay
    await new Promise(resolve => setTimeout(resolve, 100));
    console.log('Email sent successfully');
    return true;
};

export const handler = async (event) => {
    try {
        // Decrypt the secret code using encryption SDK
        let plainTextCode;
```

```
        if (event.request.code) {
            const { plaintext, messageHeader } = await decrypt(keyring,
 Buffer.from(event.request.code, 'base64'));
            plainTextCode = Buffer.from(plaintext).toString('utf-8');
        }

        // Handle different trigger sources
        if (event.triggerSource == 'CustomEmailSender_SignUp') {
            const emailAddress = event.request.userAttributes.email;
            const message = `Welcome! Your verification code is: ${plainTextCode}`;
            await sendEmail(emailAddress, message);
        }
        else if (event.triggerSource == 'CustomEmailSender_ResendCode') {
            // Handle resend code
        }
        else if (event.triggerSource == 'CustomEmailSender_ForgotPassword') {
            // Handle forgot password
        }
        else if (event.triggerSource == 'CustomEmailSender_UpdateUserAttribute') {
            // Handle update attribute
        }
        else if (event.triggerSource == 'CustomEmailSender_VerifyUserAttribute') {
            // Handle verify attribute
        }
        else if (event.triggerSource == 'CustomEmailSender_AdminCreateUser') {
            // Handle admin create user
        }
        else if (event.triggerSource == 'CustomEmailSender_Authentication') {
            // Handle authentication
        }
        else if (event.triggerSource ==
 'CustomEmailSender_AccountTakeOverNotification') {
            // Handle account takeover notification
        }

        return;
    } catch (error) {
        console.error('Error in custom email sender:', error);
        throw error;
    }
};
```

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 Absender-Trigger 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.

Dieser Auslöser eignet sich für Szenarien, in denen Sie möglicherweise mehr Kontrolle darüber haben möchten, wie Ihr Benutzerpool SMS-Nachrichten sendet. Ihre Lambda-Funktion kann den Aufruf von Amazon SNS SNS-API-Vorgängen anpassen, z. B. wenn Sie mehrere Originationen oder Kreuzungen verwalten möchten. IDs AWS-Regionen Ihre Funktion kann auch Nachrichten an ein anderes Übermittlungsmedium oder einen Drittanbieter-Service weiterleiten.

Informationen zur Konfiguration eines benutzerdefinierten E-Mail-Absender-Triggers finden Sie unterLambda-Trigger mit benutzerdefiniertem Absender aktivieren.

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.

TriggerSource value	Ereignis
CustomSMSSender_UpdateUserAttribute	Ein Benutzer aktualisiert eine E-Mail-Adresse oder ein Telefonnummernattribut und Amazon Cognito sendet einen Code zur Verifizierung des Attributs.
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.

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 Aktionen AdminRespondToAuthChallengeund RespondToAuthChallengeAPI verwenden. Amazon Cognito bezieht keine Daten aus dem ClientMetadata Parameter in AdminInitiateAuthund InitiateAuthAPI-Operationen in die Anfrage ein, die es an die Post-Authentifizierungsfunktion weiterleitet.

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.

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_ID`

Die ID des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln der Benutzercodes verwenden möchten.

KEY_ARN

Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels, den Sie zum Verschlüsseln und Entschlüsseln von Benutzercodes verwenden möchten.

Um diese Funktion bereitzustellen

1. Installieren Sie die neueste Version von NodeJS in Ihrem Developer Workspace.
2. Erstellen Sie ein neues NodeJS-Projekt in Ihrem Workspace.
3. Initialisieren Sie Ihr Projekt mit. `npm init -y`
4. Erstellen Sie das Skript für die Lambda-Funktion:`touch index.mjs`.
5. Fügen Sie den Inhalt des folgenden Beispiels in ein`index.mjs`.
6. Laden Sie die Projektabhängigkeit herunter, AWS Encryption SDK:`npm install @aws-crypto/client-node`.
7. Komprimieren Sie das Projektverzeichnis in eine Datei:`zip -r my_deployment_package.zip .`.
8. Stellen Sie die ZIP-Datei für Ihre Funktion bereit.

```
import { KmsKeyringNode, buildClient, CommitmentPolicy } from '@aws-crypto/client-
node';

// Configure the encryption SDK client with the KMS key from the environment variables
const { encrypt, decrypt } = buildClient(
    CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT
);

const generatorKeyId = process.env.KEY_ID;
const keyIds = [process.env.KEY_ARN];
const keyring = new KmsKeyringNode({ generatorKeyId, keyIds });

// Example function to simulate sending SMS.
// This example logs message details to CloudWatch Logs from your Lambda function.
// Update this function with custom logic that sends an SMS message to 'phoneNumber'
 with body 'message'.
const sendSMS = async (phoneNumber, message) => {
    // Log the destination with the phone number masked.
    console.log(`Simulating SMS send to ${phoneNumber.replace(/[^+]/g, '*')}`);
    // Log the message with the code masked.
```

```
    console.log(`Message content: ${message.replace(/\b\d{6,8}\b/g, '********')}`);
    // Simulate API delay
    await new Promise(resolve => setTimeout(resolve, 100));
    console.log('SMS sent successfully');
    return true;
};

export const handler = async (event) => {
    try {
        // Decrypt the secret code using encryption SDK
        let plainTextCode;
        if (event.request.code) {
            const { plaintext, messageHeader } = await decrypt(keyring,
 Buffer.from(event.request.code, 'base64'));
            plainTextCode = Buffer.from(plaintext).toString('utf-8');
        }

        // Handle different trigger sources
        if (event.triggerSource == 'CustomSMSSender_SignUp') {
            const phoneNumber = event.request.userAttributes.phone_number;
            const message = `Welcome! Your verification code is: ${plainTextCode}`;
            await sendSMS(phoneNumber, message);
        }
        else if (event.triggerSource == 'CustomSMSSender_ResendCode') {
            // Handle resend code
        }
        else if (event.triggerSource == 'CustomSMSSender_ForgotPassword') {
            // Handle forgot password
        }
        else if (event.triggerSource == 'CustomSMSSender_UpdateUserAttribute') {
            // Handle update attribute
        }
        else if (event.triggerSource == 'CustomSMSSender_VerifyUserAttribute') {
            // Handle verify attribute
        }
        else if (event.triggerSource == 'CustomSMSSender_AdminCreateUser') {
            // Handle admin create user
        }
        return;
    } catch (error) {
        console.error('Error in custom SMS sender:', error);
        throw error;
    }
```

```
};
```

Themen

- Bewerten von SMS-Nachrichtenfunktionen mit einer benutzerdefinierten SMS-Sender-Funktion

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-Absender finden Sie unter amazon-cognito-user-pool- development-and-testing-with - sms-redirected-to-email in der aws-samples-Bibliothek unter. GitHub Das Repository enthält AWS CloudFormation Vorlagen, mit denen Sie einen neuen Benutzerpool erstellen oder mit einem Benutzerpool arbeiten können, den Sie bereits haben. 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 Zuweisung der benutzerdefinierten SMS-Absenderfunktion aus Ihrem Benutzerpool.

> **Important**
>
> Verwenden Sie die Vorlagen in amazon-cognito-user-pool- development-and-testing-with - nichtsms-redirected-to-email, um eine Produktionsumgebung aufzubauen. 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.

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.

Die Verwaltung von Benutzern in Ihrem Amazon Cognito Cognito-Benutzerpool umfasst eine Vielzahl von Konfigurationsoptionen und Verwaltungsaufgaben. Benutzerpools können auf Millionen von Benutzern skaliert werden. Ein Benutzerverzeichnis dieser Größenordnung erfordert gleichermaßen skalierbare und wiederholbare Verwaltungstools. Möglicherweise möchten Sie viele Benutzerprofile erstellen, inaktive Benutzer verwalten, Governance- und Compliance-Berichte erstellen oder Self-Service-Tools einrichten, bei denen Benutzer die meiste Arbeit erledigen. Nachdem Sie einen Benutzerpool erstellt haben, können Sie steuern, wie Benutzer sich registrieren und ihre Konten bestätigen, einschließlich der Anforderung, dass E-Mail oder Telefonnummer verifiziert werden müssen. Administratoren können Benutzerkonten auch direkt erstellen und die Willkommensnachrichten und Kennwortanforderungen anpassen.

Benutzerpools verfügen über Benutzergruppen, in denen Sie den Zugriff auf Ressourcen auf der Grundlage der Gruppenmitgliedschaft eines Benutzers verwalten können. Sie können diesen Gruppen IAM-Rollen zuweisen, um den Zugriff AWS-Services mit Identitätspools zu verwalten. Die Gruppenmitgliedschaft der Benutzer ist sowohl in ID- als auch in Zugriffstoken enthalten. Mit diesen Informationen können Sie zur Laufzeit in Ihrer Anwendung oder mit einer Richtlinien-Engine wie Amazon Verified Permissions Entscheidungen zur Zugriffskontrolle treffen.

Benutzerpools haben oft viele Benutzer. Sie werden häufig nach Benutzerkonten suchen und diese aktualisieren müssen. Die Amazon Cognito Cognito-Konsole und die API unterstützen die Abfrage von Benutzern anhand von Standardattributen wie Benutzername, E-Mail und Telefonnummer. Administratoren können auch Passwörter zurücksetzen, Konten deaktivieren und den Verlauf von Benutzerereignissen einsehen.

Für die Migration vorhandener Benutzerdaten bietet Amazon Cognito Optionen zum Importieren von Benutzern aus einer CSV-Datei und zur Verwendung eines Lambda-Triggers, um Benutzer bei der

ersten Anmeldung automatisch zu migrieren. Diese Optionen unterstützen Benutzerübergänge von anderen Benutzerverzeichnissen zu Ihrem Benutzerpool.

Sie können die Benutzerverwaltungsfunktionen in Benutzerpools verwenden, um eine genaue Kontrolle über den Benutzerlebenszyklus und die Authentifizierungserfahrung zu haben. Die Kombination aus Self-Service-Registrierung, vom Administrator erstellten Konten, Gruppen und Migrationstools macht Amazon Cognito Cognito-Benutzerpools zu einem flexiblen Benutzerverzeichnis.

Themen

- Konfigurieren von Richtlinien für die Benutzererstellung
- Registrieren und Bestätigen von Benutzerkonten
- Erstellen von Benutzerkonten als Administrator
- Hinzufügen von Gruppen zu einem Benutzerpool
- Verwalten von und Suchen nach Benutzerkonten
- Passwörter, Kontowiederherstellung und Passwortrichtlinien
- Importieren von Benutzern in einen Benutzerpool
- Mit Benutzerattributen arbeiten

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 verwaltetes Login verfügt, erstellt ein Benutzerprofil für jeden Benutzer im Internet, der 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

Das Anmeldemenü Ihres Benutzerpools enthält 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 festlegen, wenn Sie einen Benutzerpool erstellen.
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 Cognito-Konsole oder mit AdminCreateUserAPI-

Anfragen erstellen. In einem Benutzerpool, in dem die Selbstregistrierung inaktiv ist, kehren SignUpAPI-Anfragen zurück **`NotAuthorizedException`** und bei verwalteter Anmeldung wird kein Anmeldelink angezeigt.

Für Benutzerpools, in denen Sie Benutzer als Administrator erstellen möchten, können Sie die Dauer ihrer temporären Passwörter in der Einstellung im Menü Authentifizierungsmethoden 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 im Menü Authentifizierungsmethoden 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-Geheimnissen im App-Client-Menü der Amazon Cognito Cognito-Konsole erstellen.

Amazon Cognito user pools API

Sie können die Parameter für die Erstellung von Benutzern in einem Benutzerpool programmgesteuert in einer CreateUserPooloder UpdateUserPoolAPI-Anfrage festlegen.

Das AdminCreateUserConfigElement 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 CreateUserPoolUpdateUserPoolAPI-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.

Passwörter bei der Anmeldung

Amazon Cognito verlangt Passwörter von allen Benutzern, wenn sie sich registrieren, außer unter den folgenden Bedingungen. Wenn alle diese Bedingungen erfüllt sind, können Sie Passwörter bei Anmeldevorgängen weglassen.

1. Die passwortlose Anmeldung ist in Ihrem Benutzerpool und App-Client aktiv.
2. Ihre Anwendung wurde mit Authentifizierungsmodulen in einem SDK maßgeschneidert. AWS Für die verwaltete Anmeldung und die gehostete Benutzeroberfläche sind immer Passwörter erforderlich.
3. Benutzer geben Attributwerte für die kennwortlosen Anmeldemethoden — Einmalpasswörter für E-Mail oder SMS-Nachrichten () — an, die Sie zulassen. OTPs Wenn Sie beispielsweise die Anmeldung mit E-Mail und Telefon-OTP zulassen, können Benutzer entweder eine Telefonnummer oder eine E-Mail-Adresse angeben. Wenn Sie jedoch nur die Anmeldung mit E-Mail zulassen, müssen sie eine E-Mail-Adresse angeben.
4. Ihr Benutzerpool überprüft automatisch die Attribute, die Benutzer bei der kennwortlosen Anmeldung verwenden können.
5. Für jede SignUpAnfrage gibt der Benutzer keinen Wert für den Password-Parameter an.

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.

Weitere -Quellen

- Erkennung und Behebung inaktiver Benutzerkonten mit Amazon Cognito

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 einer der AWS SDKs.

Wenn ein Benutzer über eine überprüfte Kontaktmethode verfügt, sendet Amazon Cognito dem Benutzer automatisch eine Nachricht, wenn der Benutzer die Zurücksetzung des Passworts anfordert.

Andere Aktionen, die Benutzerattribute bestätigen und verifizieren

Die folgende Benutzeraktivität überprüft Benutzerattribute. Sie müssen diese Attribute nicht für die automatische Überprüfung festlegen: Die aufgelisteten Aktionen kennzeichnen sie in allen Fällen als verifiziert.

E-Mail-Adresse

1. Die passwortlose Authentifizierung mit einem E-Mail-Einmalpasswort (OTP) wurde erfolgreich abgeschlossen.
2. Erfolgreicher Abschluss der Multi-Faktor-Authentifizierung (MFA) mit einem E-Mail-OTP.

Phone number (Telefonnummer)

1. Erfolgreicher Abschluss der passwortlosen Authentifizierung mit einem SMS-OTP.
2. Erfolgreicher Abschluss von MFA mit einem SMS-OTP.

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 das Anmeldemenü und suchen Sie nach Attributverifizierung 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 Cognito-Konsole zu bestätigen, navigieren Sie zum Menü 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 Cognito-Benutzerpools APIs akzeptieren einen geheimen Client-Hashwert in einem Parameter. `SecretHash`

- ConfirmForgotPassword
- ConfirmSignUp
- ForgotPassword
- ResendConfirmationCode
- SignUp

Darüber hinaus APIs akzeptieren die folgenden Benutzer einen geheimen Client-Hashwert in einem `SECRET_HASH` Parameter, entweder in Authentifizierungsparametern oder in einer Challenge-Antwort.

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 `HMAC_SHA256` steht er für eine + Verkettung, für eine Funktion, die mithilfe von Hmac einen HMAC-Wert erzeugtSHA256, und für eine Funktion, die eine `Base64` Base-64-kodierte Version der Hashausgabe erzeugt.

```
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

In Benutzerpools, die Sie mit mehreren Anmeldenamen konfigurieren, können Benutzer bei der Anmeldung eine Telefonnummer oder eine E-Mail-Adresse als Benutzernamen eingeben. Wenn sie ihre E-Mail-Adresse oder Telefonnummer in Ihrer App aktualisieren, kann Amazon Cognito

ihnen sofort eine Nachricht mit einem Code senden, der bestätigt, dass sie Eigentümer des neuen Attributwerts sind. Informationen zum Aktivieren des automatischen Versendens dieser Bestätigungscodes finden Sie unter. Konfigurieren der Verifizierung per E-Mail und Telefon

Benutzer, die einen Bestätigungscode erhalten, müssen diesen Code in einer VerifyUserAttributeAnfrage an Amazon Cognito zurücksenden. Nachdem sie den Code eingegeben haben, wird ihr Attribut als verifiziert markiert. Wenn Benutzer ihre E-Mail-Adresse oder Telefonnummer aktualisieren, sollten Sie in der Regel überprüfen, ob sie Eigentümer des neuen Werts sind, bevor sie ihn für die Anmeldung und den Empfang von Nachrichten verwenden können. Benutzerpools verfügen über eine konfigurierbare Option, mit der festgelegt wird, ob Benutzer Aktualisierungen ihrer E-Mail-Adresse oder Telefonnummer verifizieren müssen.

Bei dieser Option handelt es sich um eine Eigenschaft des Benutzerpools`AttributesRequireVerificationBeforeUpdate`. Konfigurieren Sie es in einer CreateUserPoolUpdateUserPoolOder-Anfrage oder mit der Einstellung Ursprünglichen Attributwert aktiv lassen, wenn ein Update aussteht im Anmeldemenü der Amazon Cognito Cognito-Konsole.

Wie Ihr Benutzerpool mit Aktualisierungen von E-Mail-Adressen und Telefonnummern umgeht, hängt von der Benutzernamenkonfiguration Ihres Benutzerpools ab. Benutzerpool-Benutzernamen können sich in einer Konfiguration mit Benutzernamenattributen befinden, bei der die Anmeldenamen E-Mail-Adresse, Telefonnummer oder beides sind. Sie können sich auch in einer Alias-Attributkonfiguration befinden, bei der das `username` Attribut ein Anmeldename zusammen mit E-Mail-Adresse, Telefonnummer oder bevorzugtem Benutzernamen als alternative Anmeldenamen ist. Weitere Informationen finden Sie unter Anpassen von Anmeldeattributen.

Sie können auch einen Lambda-Trigger für benutzerdefinierte Nachrichten verwenden, um die Bestätigungsnachricht anzupassen. Weitere Informationen finden Sie unter Lambda-Auslöser für benutzerdefinierte Nachrichten. Wenn die E-Mail-Adresse oder Telefonnummer eines Benutzers nicht verifiziert ist, sollte Ihre Anwendung den Benutzer darüber informieren, dass er das Attribut überprüfen muss, und eine Schaltfläche oder einen Link bereitstellen, über den Benutzer ihren Bestätigungscode eingeben können.

In der folgenden Tabelle wird beschrieben, wie `AttributesRequireVerificationBeforeUpdate` und wie die Aliaseinstellungen das Ergebnis bestimmen, wenn Benutzer den Wert ihrer Anmeldeattribute ändern.

Konfiguration des Benutzernamens	Verhalten, wenn Benutzer neue Attribute verifizieren müssen	Verhalten, wenn Benutzer keine neuen Attribute verifizieren müssen
Benutzernamenattribute	Das ursprüngliche Attribut bleibt verifiziert, berechtigt zur Anmeldung und hat seinen ursprünglichen Wert. Wenn der Benutzer einen neuen Wert verifiziert, aktualisiert Amazon Cognito den Attributwert, markiert ihn als verifiziert und macht ihn für die Anmeldung berechtigt.	Amazon Cognito aktualisiert das Attribut auf einen neuen Wert. Ein neuer Wert ist für die Anmeldung berechtigt. Wenn der Benutzer einen neuen Wert verifiziert, markiert Amazon Cognito ihn als verifiziert.
Aliasattribute	Das ursprüngliche Attribut bleibt verifiziert, berechtigt zur Anmeldung und hat den ursprünglichen Wert. Wenn der Benutzer einen neuen Wert verifiziert, aktualisiert Amazon Cognito den Attributwert, markiert ihn als verifiziert und macht ihn für die Anmeldung berechtigt.	Amazon Cognito aktualisiert das Attribut auf einen neuen Wert. Weder der ursprüngliche noch der neue Attributwert sind für die Anmeldung berechtigt. Wenn der Benutzer einen neuen Wert verifiziert, aktualisiert Amazon Cognito den Attributwert, markiert ihn als verifiziert und macht ihn für die Anmeldung berechtigt.

Beispiel 1

Benutzer 1 meldet sich mit der E-Mail-Adresse bei Ihrer Anwendung an `user1@example.com` und hat den Benutzernamen `user1` (Alias-Attribute). Ihr Benutzerpool ist so konfiguriert, dass Aktualisierungen der Anmeldeattribute überprüft und automatisch Bestätigungsnachrichten gesendet werden. Sie bitten darum, ihre E-Mail-Adresse auf zu `user1+foo@example.com` aktualisieren. Sie erhalten eine Bestätigungs-E-Mail unter `user1+foo@example.com` und können sich nur mit der E-Mail-Adresse erneut anmelden`user1@example.com`. Später geben sie ihren Bestätigungscode ein und können sich nur mit der E-Mail-Adresse erneut anmelden`user1+foo@example.com`.

Beispiel 2

Benutzer 2 meldet sich mit der E-Mail-Adresse bei Ihrer Anwendung an `user2@example.com` und hat einen Benutzernamen (Alias-Attribute). Ihr Benutzerpool ist so konfiguriert, dass Aktualisierungen der Anmeldeattribute nicht überprüft und Bestätigungsnachrichten automatisch gesendet werden. Sie bitten darum, ihre E-Mail-Adresse auf zu `user2+bar@example.com` aktualisieren. Sie erhalten eine Bestätigungs-E-Mail unter `user2+bar@example.com` und können sich nicht erneut anmelden. Später geben sie ihren Bestätigungscode ein und können sich nur mit der E-Mail-Adresse erneut anmeldenuser2+bar@example.com.

Beispiel 3

Benutzer 3 meldet sich mit der E-Mail-Adresse bei Ihrer Anwendung an `user3@example.com` und hat keinen Benutzernamen (Benutzernamenattribute). Ihr Benutzerpool ist so konfiguriert, dass Aktualisierungen der Anmeldeattribute nicht überprüft und Bestätigungsnachrichten automatisch gesendet werden. Sie bitten darum, ihre E-Mail-Adresse auf zu `user3+baz@example.com` aktualisieren. Sie erhalten eine Bestätigungs-E-Mail unteruser3+baz@example.com, können sich aber sofort anmelden, ohne dass zusätzliche Maßnahmen mit dem Bestätigungscode ergriffen werden müssen.

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 (sieheImportieren 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

Im Menü Authentifizierungsmethoden können Sie Einstellungen für die E-Mail- oder Telefonverifizierung 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 Bestätigungscode oder Link wählen, wenn Sie Ihre Vorlage für Bestätigungsnachrichten im Menü Nachrichtenvorlagen in der Amazon Cognito 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 Ausgabenkontingente auf Konto- und Nachrichtenebene und wie funktionieren sie? im Amazon SNS. FAQs

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 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 im Anmeldemenü unter Attributverifizierung und Bestätigung des Benutzerkontos die Option Bearbeiten 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.

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 im Menü Authentifizierungsmethoden der Amazon Cognito Cognito-Konsole unter SMS die Option Bearbeiten aus, um eine Rolle festzulegen.

Konfiguration von MFA-, Authentifizierungs-, Überprüfungs- und Einladungsnachrichten

Mit Amazon Cognito können Sie die SMS- und E-Mail-Authentifizierung, Verifizierung und Benutzereinladungsnachrichten individuell anpassen, um die Sicherheit und Benutzererfahrung Ihrer Anwendung zu verbessern. Bei einigen Nachrichten können Sie zwischen codebasierter Linküberprüfung und Linküberprüfung mit nur einem Klick wählen. In diesem Thema wird erläutert, wie Sie die Authentifizierungs- und Bestätigungskommunikation in der Amazon Cognito Cognito-Konsole personalisieren können.

Im Menü Nachrichtenvorlagen können Sie Folgendes anpassen:

- Ihre E-Mail- und SMS-Nachrichtenvorlagen für Einmalpasswort (OTP) und Multi-Faktor-Authentifizierung (MFA)
- Ihrer Nachrichten zur SMS- und E-Mail-Verifizierung
- Der Verifizierungstyp für den E-Mail-Code oder -Link

> **Note**
>
> Amazon Cognito sendet Links mit Ihrer linkbasierten Vorlage in den Bestätigungsnachrichten, wenn sich Benutzer registrieren oder einen Bestätigungscode erneut senden. E-Mails aus Vorgängen zur Aktualisierung von Attributen und zum Zurücksetzen von Kennwörtern verwenden die Codevorlage.

- Ihre Nachrichten zur Einladung von Benutzern
- Absender- und Empfänger-E-Mail-Adressen für E-Mails des Benutzerpools

> **Note**
>
> Die Vorlagen für SMS- und E-Mail-Bestätigungsnachrichten werden nur angezeigt, wenn Sie ausgewählt haben, dass die Bestätigung von Telefonnummer und E-Mail erforderlich ist. Auch die Vorlage der SMS-MFA-Nachricht wird nur angezeigt, wenn die MFA-Einstellung als erforderlich oder optional festgelegt ist.

Themen

- Nachrichtenvorlagen
- E-Mail- und SMS-MFA-Nachrichten anpassen
- Anpassen von Nachrichten zur E-Mail-Verifizierung
- Anpassen von Nachrichten zur Einladung von Benutzern
- Anpassen Ihrer E-Mail-Adresse
- Autorisieren von Amazon Cognito zum Senden von Amazon-SES-E-Mails in Ihrem Auftrag (über eine benutzerdefinierte Absender-E-Mail-Adresse)

Nachrichtenvorlagen

Sie können Nachrichtenvorlagen verwenden, um Platzhalter in Ihre Nachrichten einzufügen. Amazon Cognito ersetzt die Platzhalter durch die entsprechenden Werte. Sie können in Nachrichtenvorlagen beliebigen Typs auf Platzhalter für universelle Vorlagen verweisen, obwohl diese Werte nicht in allen Nachrichtentypen vorhanden sein werden.

Universelle Platzhalter für Vorlagen

Beschreibung	Token	Nachrichtentyp
Verifizierungscode	`{####}`	Überprüfungs-, Bestätigungs- und MFA-Nachrichten
Temporäres Passwort	`{####}`	Passwort und Einladung snachrichten vergessen
Benutzername	`{username}`	Einladung und erweiterte Sicherheitsnachrichten

Eine der verfügbaren automatisierten Antworten mit Bedrohungsschutz besteht darin, den Benutzer darüber zu informieren, dass Amazon Cognito potenziell bösartige Aktivitäten entdeckt hat. 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. Amazon Cognito Threat Protection kann 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.

Threat Protection fügt die folgenden Platzhalter hinzu, die Sie in die Nachrichtenvorlagen einfügen können. Diese Platzhalter gelten für Nachrichten zur adaptiven Authentifizierung, d. h. Benachrichtigungen, die Amazon Cognito an Benutzer sendet, deren Sitzungen auf ein gewisses Risiko hin bewertet wurden. Um Nachrichtenvorlagen mit diesen Variablen zu konfigurieren, aktualisieren Sie die Konfiguration Ihres Bedrohungsschutzes mit vollem Funktionsumfang in der Amazon Cognito Cognito-Konsole oder reichen Sie Vorlagen in einer Anfrage ein SetRiskConfiguration.

Platzhalter in der Vorlage für die erweiterte Sicherheit

Beschreibung	Token
IP-Adresse	`{ip-address}`
Ort	`{city}`
Country (Land)	`{country}`
Anmeldezeit	`{login-time}`
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}`

E-Mail- und SMS-MFA-Nachrichten anpassen

Um die SMS- und E-Mail-Nachrichten für die Multi-Faktor-Authentifizierung (MFA) anzupassen, bearbeiten Sie die MFA-Nachricht im Menü Nachrichtenvorlagen in der Amazon Cognito Cognito-Benutzerpools-Konsole.

> **Important**
>
> Die benutzerdefinierte Nachricht muss den `{####}`-Platzhalter enthalten. Dieser Platzhalter wird mit dem Authentifizierungscode ersetzt, bevor die Nachricht gesendet wird.

Amazon Cognito legt eine maximale Länge für SMS-Nachrichten, einschließlich des Authentifizierungscodes, von 140 UTF-8-Zeichen fest.

Anpassen von Nachrichten zur SMS-Verifizierung

Um die SMS-Nachricht für die Überprüfung der Telefonnummer anzupassen, bearbeiten Sie die Vorlage für die Bestätigungsnachricht im Menü Nachrichtenvorlagen Ihres Benutzerpools.

> **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.

Um den E-Mail-Betreff und den Nachrichteninhalt für Bestätigungsnachrichten der E-Mail-Adresse anzupassen, bearbeiten Sie die Vorlage für Bestätigungsnachrichten im Menü Nachrichtenvorlagen 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 die Benutzereinladungsnachricht, die Amazon Cognito an neue Benutzer per SMS oder E-Mail sendet, anpassen, indem Sie die Vorlage für Einladungsnachrichten im Menü Nachrichtenvorlagen 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 das Menü Authentifizierungsmethoden. 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 von einer verifizierten Domain zu verwenden, konfigurieren Sie die E-Mail-Einstellungen in der AWS Command Line Interface oder 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.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "arn:aws:ses:us-
east-1:111122223333:identity/support@example.com",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
east-1:111122223333:userpool/us-east-1_EXAMPLE"
                }
            }
        }
    ]
}
```

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

Benutzerpools sind nicht nur ein CIAM-Benutzerverzeichnis (Customer Identity and Access Management), in dem sich jeder im Internet für ein Benutzerprofil in Ihrer Anwendung registrieren kann. Sie können die Self-Service-Registrierung deaktivieren. Möglicherweise kennen Sie Ihre Kunden bereits und möchten nur Kunden zulassen, die im Voraus autorisiert wurden. Sie können Ihre Anwendung mit einem privaten SAML 2.0- oder OIDC-Identitätsanbieter manuell authentifizieren, indem Sie Benutzer importieren, Benutzer bei der Registrierung überprüfen oder Benutzer mit administrativen API-Vorgängen erstellen. Ihr Workflow für die administrative Erstellung von Benutzern kann programmatisch sein und Benutzer nach der Registrierung in einem anderen System bereitstellen, oder er kann auf case-by-case oder Testbasis in der Amazon Cognito Cognito-Konsole erfolgen.

Wenn Sie Benutzer als Administrator erstellen, legt Amazon Cognito ein temporäres Passwort für sie fest und sendet eine Willkommens- oder Einladungsnachricht. Sie können dem Link in ihrer Einladungsnachricht folgen und sich zum ersten Mal anmelden, ein Passwort festlegen und ihr Konto bestätigen. Auf der folgenden Seite wird beschrieben, wie Sie neue Benutzer erstellen und die Willkommensnachricht konfigurieren. Weitere Informationen zur Benutzererstellung mit der Benutzerpools-API und einem AWS SDK oder CDK finden Sie unter AdminCreateUser.

Nachdem Sie Ihren Benutzerpool erstellt haben, können Sie Benutzer sowohl mit der AWS Management Console Amazon Cognito API als auch mit der AWS Command Line Interface oder der Amazon Cognito API 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.

Im Folgenden finden Sie einige Beispiele dafür, wie Administratoren Benutzer in Benutzerpools verwalten können.

- Erstellen Sie ein neues Benutzerprofil in der Amazon Cognito Cognito-Konsole oder mit der `AdminCreateUser` API-Operation.
- Stellen Sie username-and-password Ihrem Benutzerpool und App-Client Authentifizierungsabläufe ohne Passwort, Hauptschlüssel und benutzerdefinierte Authentifizierungsabläufe zur Verfügung.
- Legen Sie Benutzerattributwerte fest.
- Erstellen Sie benutzerdefinierte Attribute.
- Legen Sie den Wert unveränderlicher benutzerdefinierter Attribute in API-Anfragen fest. `AdminCreateUser` Dieses Feature ist in der Amazon-Cognito-Konsole nicht verfügbar.
- Geben Sie ein temporäres Passwort an, erstellen Sie einen Benutzer ohne Passwort oder erlauben Sie Amazon Cognito, automatisch ein Passwort zu generieren.
- Erstellen Sie neue Benutzer und bestätigen Sie automatisch ihre Konten, verifizieren Sie ihre E-Mail-Adressen oder ihre Telefonnummern.
- Geben Sie benutzerdefinierte SMS- und E-Mail-Einladungsnachrichten für neue Benutzer über die AWS Management Console oder Lambda-Trigger wie benutzerdefinierte Nachricht, benutzerdefinierter SMS-Absender und benutzerdefinierter E-Mail-Absender an.
- 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.
- Unterdrücken Sie das Senden der Einladungsnachricht, wenn der Benutzer erstellt wird.

- Geben Sie eine Ablaufzeit von bis zu 90 Tagen für neue Benutzerkonten an.
- Benutzer gestatten, sich anzumelden, oder verlangen, dass neue Benutzer nur durch den Administrator hinzugefügt werden.

Administratoren können Benutzer auch mit AWS Anmeldeinformationen in einer serverseitigen Anwendung anmelden. Weitere Informationen finden Sie unter Autorisierungsmodelle für die API- und SDK-Authentifizierung.

Abläufe bei der Benutzerauthentifizierung und Erstellung von Benutzern

Für die administrative Erstellung von Benutzern gibt es Optionen, die sich je nach Konfiguration Ihres Benutzerpools unterscheiden. Die Authentifizierungsabläufe oder Methoden, die Benutzern für die Anmeldung und MFA zur Verfügung stehen, können die Art und Weise, wie Sie Benutzer erstellen und welche Nachrichten Sie an sie senden, ändern. Im Folgenden sind einige Authentifizierungsabläufe aufgeführt, die in Benutzerpools verfügbar sind.

- Nutzername und Passwort
- Hauptschlüssel
- Melden Sie sich bei einem Drittanbieter an IdPs
- Passwortlos mit E-Mail- und SMS-Einmalpasswörtern () OTPs
- Multi-Faktor-Authentifizierung mit E-Mail, SMS und Authenticator-App OTPs
- Benutzerdefinierte Authentifizierung mit Lambda-Triggern

Weitere Informationen zur Konfiguration dieser Anmeldefaktoren finden Sie unter. Authentifizierung mit Amazon Cognito Cognito-Benutzerpools

Benutzer ohne Passwörter erstellen

Wenn Sie die kennwortlose Anmeldung für Ihren Benutzerpool aktiviert haben, können Sie Benutzer ohne Passwörter erstellen. Um einen Benutzer ohne Passwort zu erstellen, müssen Sie Attributwerte für einen verfügbaren kennwortlosen Anmeldefaktor angeben. Wenn in Ihrem Benutzerpool beispielsweise eine passwortlose E-Mail-OTP-Anmeldung verfügbar ist, können Sie einen Benutzer ohne Passwort und mit einem E-Mail-Adressattribut erstellen. Wenn die einzigen Authentifizierungsabläufe, die neuen Benutzern zur Verfügung stehen, ein Passwort erfordern, z. B. einen Hauptschlüssel oder ein Benutzername-Passwort, müssen Sie für jeden neuen Benutzer ein temporäres Passwort erstellen oder generieren.

Um einen neuen Benutzer ohne Passwort zu erstellen

- Wählen Sie in der Amazon Cognito Cognito-Konsole kein Passwort festlegen
- Lassen Sie den `TemporaryPassword` Parameter Ihrer API-Anfrage weg oder lassen Sie `AdminCreateUser` ihn leer

Benutzer ohne Passwörter werden automatisch bestätigt

Normalerweise erhalten neue Benutzer ein temporäres Passwort und gehen in einen `FORCE_CHANGE_PASSWORD` Status über, wenn Sie sie erstellen. Wenn Sie Benutzer ohne Passwörter erstellen, gehen sie sofort in einen `CONFIRMED` Status über. Sie können Bestätigungscodes nicht erneut an diese Benutzer in dem `CONFIRMED` Bundesstaat senden.

Einladungsnachrichten ändern sich für Benutzer ohne Passwort.

Standardmäßig sendet Amazon Cognito eine Einladungsnachricht an neue Benutzer mit dem Hinweis `Your username is {userName} and your password is {####}.` Wenn Sie Benutzer ohne Passwort erstellen, lautet die Meldung `Your username is {userName}.` Passen Sie Ihre Einladungsnachricht an, um zu verdeutlichen, ob Sie Passwörter für Benutzer festlegen. Lassen Sie die `{####}` Kennwortvariable in kennwortlosen Authentifizierungsmodellen weg.

Sie können Passwörter nicht automatisch generieren, wenn kennwortlose Faktoren verfügbar sind

Wenn Sie Ihren Benutzerpool so konfiguriert haben, dass er die kennwortlose Anmeldung per E-Mail oder Telefon per OTP unterstützt, können Sie nicht automatisch ein Passwort generieren. Für jeden Benutzer, der über ein Passwort verfügen wird, müssen Sie bei der Erstellung seines Profils ein temporäres Passwort festlegen.

Kennwortlose Benutzer müssen Werte für alle erforderlichen Attribute haben

Wenn Sie einen Benutzer ohne Passwort erstellen, ist Ihre Anfrage nur erfolgreich, wenn der Benutzer Werte für alle Attribute bereitstellt, die Sie in Ihrem Benutzerpool als erforderlich markiert haben. Dies gilt für alle erforderlichen Attribute, nicht nur für die Telefonnummer und die E-Mail-Attribute, die für die OTP-Zustellung erforderlich sind.

Benutzer erstellen, die später die erforderlichen Attributwerte angeben

Möglicherweise möchten Sie Attribute in Ihrem Benutzerpool vorschreiben, aber diese Attribute sammeln, nachdem Sie Benutzer administrativ erstellt haben, während der Benutzerinteraktion in Ihrer Anwendung. Administratoren können Werte für erforderliche Attribute weglassen, wenn

sie Benutzer mit temporären Kennwörtern erstellen. Sie können erforderliche Attributwerte für kennwortlose Benutzer nicht weglassen.

Benutzer mit fehlenden Werten für erforderliche Attribute und einem temporären Passwort erhalten bei der ersten Anmeldung die Aufforderung NEW_PASSWORD_REQUIRED. Anschließend können sie im Parameter einen Wert für die fehlenden erforderlichen Attribute angeben. `requiredAttributes` Sie können Benutzer nur dann mit Passwörtern und ohne erforderliche Attribute erstellen, wenn alle erforderlichen Attribute veränderbar sind. Benutzer können die Anmeldung mit `NEW_PASSWORD_REQUIRED` Herausforderungen und Werten für erforderliche Attribute nur abschließen, wenn die erforderlichen Attribute vom App-Client aus beschreibbar sind, mit dem sie sich anmelden.

Wenn Sie ein permanentes Passwort für einen vom Administrator erstellten Benutzer einrichten, ändert sich sein Status `CONFIRMED` und Ihr Benutzerpool fordert die Benutzer bei der ersten Anmeldung nicht zur Eingabe eines neuen Kennworts oder der erforderlichen Attribute auf.

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 die Mindestkomplexität von Kennwörtern konfigurieren und festlegen, ob sich Benutzer APIs in Ihrem Benutzerpool mit public registrieren 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 Benutzerpool.
3. Wählen Sie das Menü Authentifizierungsmethoden und suchen Sie nach 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 Benutzerpool.
3. Wählen Sie das Anmeldemenü und suchen Sie nach Self-Service-Anmeldung. Wählen Sie Edit (Bearbeiten) aus.
4. Wählen Sie aus, ob Sie die Selbstregistrierung aktivieren möchten. Die Selbstregistrierung wird in der Regel bei öffentlichen App-Clients verwendet, die neue Benutzer in Ihrem Benutzerpool registrieren müssen, ohne ein geheimes Client-Geheimnis 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 Benutzerpool.
3. Wählen Sie das Menü Nachrichtenvorlagen und wählen Sie Bestätigungsnachricht, Einladungsnachricht oder MFA-Nachricht aus und wählen Sie Bearbeiten aus.
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. Um Konten als Administrator zu bestätigen, können Sie auch die API AWS CLI oder verwenden oder Benutzerprofile mit einem föderierten Identitätsanbieter 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 das Menü „Benutzer“ und wählen Sie „Benutzer erstellen“.
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. Um die Einladungsnachricht zu unterdrücken, wählen Sie Keine Einladung senden.

> **Note**
>
> Bevor Sie Einladungsnachrichten versenden können, müssen Sie im Menü Authentifizierungsmethoden Ihres Benutzerpools einen Absender und einen AWS-Region mit Amazon Simple Notification Service und Amazon Simple Email Service 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. Die Option zum Generieren eines Passworts ist nicht verfügbar, wenn die passwortlose Anmeldung im Benutzerpool verfügbar ist. Jedes temporäre Passwort muss der Passwortrichtlinie des Benutzerpools entsprechen.
8. Wählen Sie Create (Erstellen) aus.
9. Wählen Sie das Menü Benutzer und wählen Sie den Eintrag 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.

Amazon Cognito erstellt eine Benutzergruppe für jeden OIDC und jeden Social Identity Provider (IdP), den Sie Ihrem Benutzerpool hinzufügen. SAMl Der Name der Gruppe hat das Format`[user pool ID]_[IdP name]`, zum Beispiel oder. `us-east-1_EXAMPLE_MYSSO` `us-east-1_EXAMPLE_Google` Jedes eindeutige, automatisch generierte IdP-Benutzerprofil wird automatisch zu dieser Gruppe hinzugefügt. Verknüpfte Benutzer werden dieser Gruppe nicht automatisch hinzugefügt, aber Sie können ihre Profile der Gruppe in einem separaten Prozess hinzufügen.

Sie können Gruppen in einem Benutzerpool über die AWS Management Console, die und die APIs 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 sie AWS Anmeldeinformationen für einen Benutzer über einen Identitätspool anfordert, indem Sie im GetCredentialsForIdentity`CustomRoleARN`Parameter einen Rollen-ARN angeben. 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 beiden Gruppen unterschiedliche Rollen haben ARNs, wird der `cognito:preferred_role` Anspruch nicht in den ID-Token der Benutzer 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 Steuern des Zugriffs auf HTTP APIs mit JWT-Autorisierern im API Gateway Developer Guide.

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 das Menü Gruppen und dann 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

Benutzerpools können Millionen von Benutzern enthalten. Die Arbeit mit einem Datensatz dieser Größe ist eine Herausforderung für Administratoren. Amazon Cognito verfügt über Tools zum Suchen und Ändern von Benutzerprofilen. Die wichtigsten Methoden zum Auffinden von Benutzern sind das Benutzermenü der Amazon Cognito Cognito-Konsole und mit ListUsers. Von den Methoden, mit denen Informationen über Benutzer abgerufen werden, sind dies die Optionen, die sich nicht auf die Kosten auswirken, im Gegensatz zu beispielsweise. AdminGetUser

Dieser Abschnitt des Handbuchs enthält Informationen zum Suchen und Aktualisieren von Benutzerprofilen in einem Benutzerpool.

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. 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 das Menü Benutzer und wählen Sie 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. 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 das Menü Benutzer und wählen Sie 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.

Benutzerkonten aktivieren, deaktivieren und löschen

Sie können unbenutzte Benutzerprofile löschen oder, wenn Sie den Zugriff vorübergehend verhindern möchten, sie deaktivieren. Benutzer können ihre eigenen Konten löschen, aber nur Benutzerpooladministratoren können Benutzerkonten aktivieren und deaktivieren.

Auswirkung des Löschens

Benutzer können sich nicht mit gelöschten Benutzerkonten anmelden. Um wieder Zugriff darauf zu erhalten, müssen sie sich registrieren oder erneut erstellt werden.

Auswirkung der Deaktivierung von Konten

Wenn Sie ein Benutzerkonto deaktivieren, macht Amazon Cognito automatisch alle authentifizierten Sitzungen ungültig, deaktiviert das Benutzerkonto für die Anmeldung und widerruft deren Zugriffs- und Aktualisierungstoken. Amazon Cognito gibt einen `invalid_request` Fehler mit der Meldung zurück`User is not enabled`, wenn ein Benutzer versucht, sich bei einem Konto anzumelden, das Sie deaktiviert haben. Dieses Verhalten ändert sich nicht mit Ihren Einstellungen zur Offenlegung der Benutzerexistenz für den App-Client. Sie können lokale Benutzerkonten und die lokalen Profile von Verbundbenutzerkonten deaktivieren. Wenn sich Benutzer mit der verwalteten Anmeldung oder der klassischen gehosteten Benutzeroberfläche anmelden, deaktivieren Sie ihr Konto und sie versuchen

dann erneut, sich mit dem Browser-Cookie anzumelden, das ihre authentifizierte Sitzung beibehält, Amazon Cognito leitet sie auf die Anmeldeseite weiter.

Auswirkung der Aktivierung von Konten

Benutzer können sich sofort bei Konten anmelden, nachdem Sie sie aktiviert haben. Benutzerkonten sind standardmäßig aktiviert. Die Attribute und Passwörter der Benutzer bleiben dieselben wie vor der Deaktivierung ihres Kontos. Token, die Ihre Anwendung gesperrt hat, unabhängig davon, ob Sie das Benutzerkonto deaktiviert oder das Aktualisierungstoken separat gesperrt haben, bleiben ungültig, nachdem Sie das Benutzerkonto aktiviert haben, dem das Token gehörte.

Delete a user account (console)

Um ein Benutzerkonto zu löschen

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 das Menü Benutzer und wählen Sie das Optionsfeld neben dem Benutzernamen eines Benutzers in der Liste aus.
5. Wählen Sie Löschen aus.
6. Wählen Sie Benutzerzugriff deaktivieren.
7. Wählen Sie Löschen aus.

Delete a user account (API)

Benutzer können ihre Konten mit dem access-token-authorized DeleteUserSelf-Service-API-Vorgang löschen. Im Folgenden finden Sie ein Beispiel für einen `DeleteUser` Anfragetext.

```
{
   "AccessToken": "eyJra456defEXAMPLE"
}
```

Administratoren können Benutzerkonten mit dem von IAM autorisierten AdminDeleteUserAPI-Vorgang löschen. Im Folgenden finden Sie ein Beispiel für einen `AdminDeleteUser` Anfragetext.

```
{
```

```
    "Username": "testuser",
    "UserPoolId": "us-west-2_EXAMPLE"
}
```

Disable a user account (console)

Um ein Benutzerkonto zu deaktivieren

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 das Menü Benutzer und wählen Sie den Benutzernamen eines Benutzers in der Liste aus.
5. Wählen Sie auf der Seite mit den Benutzerdetails die Optionen Aktionen, Benutzerzugriff deaktivieren aus.
6. Wählen Sie in dem daraufhin erstellten Dialogfeld die Option Deaktivieren aus.

Disable a user account (API)

Administratoren können Benutzerkonten mit dem von IAM autorisierten AdminDisableUserAPI-Vorgang deaktivieren. Im Folgenden finden Sie ein Beispiel für einen `AdminDisableUser` Anfragetext.

```
{
    "Username": "testuser",
    "UserPoolId": "us-west-2_EXAMPLE"
}
```

Enable a user account (console)

Um ein Benutzerkonto zu aktivieren

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 das Menü Benutzer und wählen Sie den Benutzernamen eines Benutzers in der Liste aus.
5. Wählen Sie auf der Seite mit den Benutzerdetails die Optionen Aktionen, Benutzerzugriff aktivieren aus.
6. Wählen Sie in dem daraufhin erstellten Dialogfeld die Option Aktivieren aus.

Enable a user account (API)

Administratoren können Benutzerkonten mit dem von IAM autorisierten AdminEnableUserAPI-Vorgang aktivieren. Im Folgenden finden Sie ein Beispiel für einen `AdminEnableUser` Anfragetext.

```
{
   "Username": "testuser",
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

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 Cognito-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, wird eine paginierte Liste mit null oder mehr Benutzern ListUsers zurückgegeben. 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 im Benutzerhandbuch unter Filtern AWS CLI der Ausgabe. AWS Command Line Interface

Auf der Suche nach Benutzern mit dem 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 Suchanfragen 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. 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.
4. Wählen Sie das Menü Benutzer und geben Sie den Benutzernamen 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

Verwenden Sie die Amazon Cognito ListUsers Cognito-API, um in Ihrer App nach Benutzern zu suchen. 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. 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"
}
```

Passwörter, Kontowiederherstellung und Passwortrichtlinien

Allen Benutzern, die sich bei einem Benutzerpool anmelden, auch Verbundbenutzern, sind ihren Benutzerprofilen Kennwörter zugewiesen. Lokale Benutzer und verknüpfte Benutzer müssen bei der Anmeldung ein Passwort angeben. Verbundbenutzer verwenden keine Benutzerpool-Passwörter, sondern melden sich mit ihrem Identity Provider (IdP) an. Sie können Benutzern gestatten, ihre eigenen Passwörter zurückzusetzen, Passwörter als Administrator zurückzusetzen oder zu ändern, und Richtlinien für die Komplexität und den Verlauf von Kennwörtern festlegen.

Amazon Cognito speichert Benutzerkennwörter nicht im Klartext. Stattdessen speichert es einen Hash der Passwörter jedes Benutzers mit einem benutzerspezifischen Salt. Aus diesem Grund können Sie keine vorhandenen Passwörter aus den Benutzerprofilen in Ihren Benutzerpools abrufen. Es hat sich bewährt, Benutzerkennwörter im Klartext-Format nirgends zu speichern. Führen Sie Kennwortzurücksetzungen durch, wenn Benutzer ihre Passwörter vergessen haben.

Zurücksetzen und Wiederherstellen von Passwörtern

Benutzer vergessen ihre Passwörter. Möglicherweise möchten Sie, dass sie ihr Passwort selbst zurücksetzen können, oder Sie möchten verlangen, dass ein Administrator ihr Passwort für sie zurücksetzt. Amazon Cognito Cognito-Benutzerpools bieten Optionen für beide Modelle. Dieser Teil des Handbuchs behandelt die Benutzerpool-Einstellungen und die API-Operationen für das Zurücksetzen von Passwörtern.

Der ForgotPasswordAPI-Vorgang und die verwaltete Anmeldeoption Haben Sie Ihr Passwort vergessen? Senden Sie Benutzern einen Code, mit dem sie, wenn sie bestätigen, dass

sie den richtigen Code haben, die Möglichkeit haben, ein neues Passwort festzulegen ConfirmForgotPassword. Dies ist das Self-Service-Modell zur Kennwortwiederherstellung.

Wiederherstellung nicht verifizierter Benutzer

Sie können Wiederherstellungsnachrichten an Benutzer senden, die ihre E-Mail-Adresse oder Telefonnummer bestätigt haben. Wenn sie keine bestätigte Wiederherstellungs-E-Mail oder Telefonnummer haben, kann ein Benutzerpooladministrator ihre E-Mail-Adresse oder Telefonnummer als verifiziert markieren. Bearbeiten Sie die Benutzerattribute des Benutzers in der Amazon Cognito Cognito-Konsole und aktivieren Sie das Kontrollkästchen neben Telefonnummer als verifiziert markieren oder E-Mail-Adresse als verifiziert markieren. Sie können `email_verified` oder in einer AdminUpdateUserAttributesAnfrage auch `phone_number_verified` auf true setzen. Neuen Benutzern sendet der ResendConfirmationCodeAPI-Vorgang einen neuen Code an ihre E-Mail-Adresse oder Telefonnummer und sie können die Bestätigung und Verifizierung per Self-Service durchführen.

Setzen Sie Passwörter als Administrator zurück

Bei den AdminSetUserPasswordund AdminResetUserPasswordAPI-Vorgängen handelt es sich um vom Administrator initiierte Methoden zum Zurücksetzen von Kennwörtern. `AdminSetUserPassword`legt ein temporäres oder permanentes Passwort fest und `AdminResetUserPassword` sendet Benutzern auf dieselbe Weise einen Code zum Zurücksetzen des Passworts wie. `ForgotPassword`

Konfigurieren Sie das Zurücksetzen und Wiederherstellen von Kennwörtern

Amazon Cognito wählt Ihre Kontowiederherstellungsoptionen automatisch aus den erforderlichen Attributen und Anmeldeoptionen aus, die Sie beim Erstellen eines Benutzerpools in der Konsole auswählen. Sie können diese Standardeinstellungen ändern.

Die bevorzugte MFA-Methode eines Benutzers beeinflusst die Methoden, mit denen er sein Passwort wiederherstellen kann. Benutzer, deren bevorzugte MFA per E-Mail-Nachricht erfolgt, können keinen Code zum Zurücksetzen des Passworts per E-Mail erhalten. Benutzer, deren bevorzugte MFA per SMS-Nachricht erfolgt, können keinen Code zum Zurücksetzen des Passworts per SMS erhalten.

Ihre Einstellungen für die Kennwortwiederherstellung müssen eine alternative Option bieten, wenn Benutzer nicht für Ihre bevorzugte Methode zum Zurücksetzen des Passworts in Frage kommen. Beispielsweise könnten Ihre Wiederherstellungsmechanismen E-Mail als erste Priorität haben und E-Mail-MFA könnte eine Option in Ihrem Benutzerpool sein. Fügen Sie in diesem

Fall die Kontowiederherstellung per SMS-Nachricht als zweite Option hinzu oder verwenden Sie administrative API-Operationen, um die Passwörter für diese Benutzer zurückzusetzen.

Amazon Cognito beantwortet Anfragen zum Zurücksetzen von Passwörtern von Benutzern, die keine gültige Wiederherstellungsmethode haben, mit einer Fehlerantwort. `InvalidParameterException`

> **Note**
>
> Benutzer können MFA- und Passwort-Reset-Codes nicht an dieselbe E-Mail-Adresse oder Telefonnummer erhalten. Wenn sie Einmalpasswörter (OTPs) aus E-Mail-Nachrichten für MFA verwenden, müssen sie SMS-Nachrichten für die Kontowiederherstellung verwenden. Wenn sie OTPs SMS-Nachrichten für MFA verwenden, müssen sie E-Mail-Nachrichten für die Kontowiederherstellung verwenden. In Benutzerpools mit MFA können Benutzer die Self-Service-Kennwortwiederherstellung möglicherweise nicht abschließen, wenn sie Attribute für ihre E-Mail-Adresse, aber keine Telefonnummer haben, oder ihre Telefonnummer, aber keine E-Mail-Adresse.
> Um zu verhindern, dass Benutzer ihre Passwörter in Benutzerpools mit dieser Konfiguration nicht zurücksetzen können, legen Sie die `phone_number` Attribute `email` und nach Bedarf fest. Als Alternative können Sie Prozesse einrichten, bei denen diese Attribute immer erfasst und festgelegt werden, wenn sich Benutzer registrieren oder wenn Ihre Administratoren Benutzerprofile erstellen. Wenn Benutzer über beide Attribute verfügen, sendet Amazon Cognito automatisch Codes zum Zurücksetzen des Passworts an das Ziel, das nicht dem MFA-Faktor des Benutzers entspricht.

Das folgende Verfahren konfiguriert die Self-Service-Kontowiederherstellung in einem Benutzerpool.

Configure self-service password reset (API/SDK)

Der `AccountRecoverySetting` Parameter ist der Benutzerpool-Parameter, der die Methoden festlegt, mit denen Benutzer ihr Passwort in ForgotPasswordAPI-Anfragen wiederherstellen können oder wenn sie Passwort vergessen? auswählen bei der verwalteten Anmeldung. `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 so festgelegt1, dass sie die höchste Priorität hat. Amazon Cognito sendet eine Bestätigung nur an eine der angegebenen Methoden. Im folgenden Beispiel werden E-Mail-Adressen als primäres Ziel für Kontowiederherstellungscodes `AccountRecoverySetting` festgelegt, wobei auf SMS-Nachrichten zurückgegriffen wird, wenn der Benutzer kein E-Mail-Adressattribut hat.

```
"AccountRecoverySetting": {
   "RecoveryMechanisms": [
      {
         "Name": "verified_email",
         "Priority": 1
      },
      {
         "Name": "verified_phone_number",
         "Priority": 2
      }
   ]
}
```

Mit diesem Wert `admin_only` wird die Self-Service-Kontowiederherstellung deaktiviert, sodass Benutzer sich für das Zurücksetzen des Kennworts an ihren Administrator wenden müssen. Sie können `admin_only` nicht mit einem anderen Mechanismus für die Kontowiederherstellung verwenden. Das folgende e

```
"AccountRecoverySetting": {
   "RecoveryMechanisms": [
      {
         "Name": "admin_only",
         "Priority": 1
      }
   ]
}
```

Wenn Sie nichts angeben`AccountRecoverySetting`, sendet Amazon Cognito den Wiederherstellungscode zuerst an eine verifizierte Telefonnummer und an eine verifizierte E-Mail-Adresse, wenn Benutzer kein Telefonnummernattribut haben.

Weitere Informationen zu `AccountRecoverySetting` finden Sie unter CreateUserPool und UpdateUserPool.

Configure self-service password reset (console)

Konfigurieren Sie die Optionen für die Kontowiederherstellung und das Zurücksetzen des Passworts im Anmeldemenü Ihres Benutzerpools.

So richten Sie die Wiederherstellung von Benutzerkonten ein

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 das Anmeldemenü. Suchen Sie nach Wiederherstellung des Benutzerkontos und wählen Sie Bearbeiten
5. Um Benutzern das Zurücksetzen ihrer eigenen Passwörter zu ermöglichen, wählen Sie Self-Service-Kontowiederherstellung aktivieren.
6. Konfigurieren Sie die Übermittlungsmethode für die Codes zur Kennwortwiederherstellung, die Ihr Benutzerpool an Benutzer sendet. Wählen Sie unter Zustellungsmethode für Nachrichten zur Wiederherstellung von Benutzerkonten eine verfügbare Option aus. Es hat sich bewährt, eine Option auszuwählen, die über eine sekundäre Methode zum Senden von Nachrichten verfügt, z. B. E-Mail, falls verfügbar, andernfalls SMS. Mit einer sekundären Versandmethode kann Amazon Cognito Codes an Benutzer senden, sodass sie für das Zurücksetzen des Passworts ein anderes Medium als für MFA verwenden müssen.
7. Wählen Sie Änderungen speichern aus.

Verhalten bei „Passwort vergessen“

In einer bestimmten Stunde erlauben wir einem Benutzer zwischen 5 und 20 Versuche, im Rahmen von „Passwort vergessen“ und Aktionen einen Code zum Zurücksetzen des Passworts anzufordern oder einzugeben. confirm-forgot-password Der genaue Wert hängt von den Risikoparametern ab, die den Anforderungen zugeordnet sind. Bitte beachten Sie, dass sich dieses Verhalten ändert.

Hinzufügen von Benutzerpool-Passwortanforderungen

Starke, komplexe Passwörter sind eine bewährte Sicherheitsmethode für Ihren Benutzerpool. Insbesondere in Anwendungen, die offen für das Internet sind, können schwache Passwörter dazu

führen, dass die Anmeldeinformationen Ihrer Benutzer an Systeme weitergegeben werden, die Passwörter erraten und versuchen, auf Ihre Daten zuzugreifen. Je komplexer ein Passwort ist, desto schwieriger ist es, es zu erraten. Amazon Cognito bietet zusätzliche Tools für sicherheitsbewusste Administratoren, wie Bedrohungsschutz 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. Es hat sich bewährt, Uhrzeit, Datum und Metadaten für das Zurücksetzen von Benutzerkennwörtern in einem externen System zu protokollieren. Mit einem externen Protokoll zum Passwortalter kann Ihre Anwendung oder ein Lambda-Trigger das Passwortalter eines Benutzers nachschlagen und nach einem bestimmten Zeitraum einen Reset erfordern.

Sie können Ihren Benutzerpool so konfigurieren, dass eine Mindestkomplexität des Kennworts erforderlich ist, die Ihren Sicherheitsstandards entspricht. Komplexe Passwörter haben eine Mindestlänge von mindestens acht Zeichen. Sie enthalten auch eine Mischung aus Großbuchstaben, Zahlen und Sonderzeichen.

Bei den Funktionsstufen Essentials oder Plus können Sie auch eine Richtlinie für die Wiederverwendung von Passwörtern festlegen. Sie können verhindern, dass ein Benutzer sein Passwort auf ein neues Passwort zurücksetzt, das seinem aktuellen Passwort oder einem von bis zu 23 zusätzlichen vorherigen Passwörtern entspricht, also insgesamt maximal 24.

So richten Sie eine Benutzerpool-Passwortrichtlinie ein

1. Erstellen Sie einen Benutzerpool und navigieren Sie zum Schritt Sicherheitsanforderungen konfigurieren, oder greifen Sie auf einen vorhandenen Benutzerpool zu und navigieren Sie zum Menü Authentifizierungsmethoden.
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.

Kennwörter können mindestens eines der folgenden Zeichen enthalten. Nachdem Amazon Cognito überprüft hat, dass Passwörter die mindestens erforderlichen Zeichen enthalten, können die Passwörter Ihrer Benutzer zusätzliche Zeichen beliebigen Typs 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-Benutzerpools-API zu aktualisieren, legen Sie TemporaryPasswordValidityDays in Ihrer CreateUserPoolodder UpdateUserPoolAPI-Anfrage einen Wert für fest.
7. Legen Sie einen Wert für Verwendung früherer Passwörter verhindern fest, falls verfügbar. Um diese Funktion zu verwenden, wählen Sie in Ihrem Benutzerpool die Featurestufe Essentials oder Plus aus. Der Wert dieses Parameters gibt die Anzahl der vorherigen Passwörter an, mit denen ein neues Passwort nicht übereinstimmt, wenn ein Benutzer sein Passwort zurücksetzt.

Gehen Sie wie folgt vor, um den Zugriff für ein abgelaufenes Benutzerkonto zurückzusetzen:

- Senden Sie ein neues temporäres Passwort und setzen Sie den Ablaufzeitraum mit einer AdminCreateUserAPI-Anfrage zurück, die auf `MessageAction` eingestellt ist`RESEND`.
- Löschen Sie das Benutzerprofil und erstellen Sie es neu.
- Generieren Sie einen neuen Bestätigungscode in einer AdminResetUserPasswordAPI-Anfrage.

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.

Weitere -Quellen

- Ansätze für die Migration von Benutzern zu Amazon Cognito Cognito-Benutzerpools
- AWS re:InForce 2023 — Migration zu Amazon Cognito

Themen

- Importieren von Benutzern mit einem Lambda-Auslöser für die Benutzermigration
- Importieren von Benutzern aus einer CSV-Datei in Benutzerpools

Importieren von Benutzern 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 Amazon Cognito-Benutzerpools-API oder über eine verwaltete Anmeldung an. Weitere Informationen darüber, wie Sie die Anmeldung mit Amazon Cognito erleichtern können 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 erstellt haben, muss Ihre App InitiateAuthoder AdminInitiateAuthmit dem `USER_PASSWORD_AUTH` Oder-Flow verwenden. `ADMIN_USER_PASSWORD_AUTH` 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 den Protokollen angezeigt. CloudWatch

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 Ihren Benutzer in Ihrer App zur Eingabe seines Bestätigungscodes und eines neuen Passworts auf und senden Sie diese Informationen dann in einer ConfirmForgotPasswordAPI-Anfrage an Amazon Cognito. Sie können auch die integrierten Seiten für den Ablauf „Passwort vergessen“ bei der verwalteten Anmeldung verwenden.

Weitere Ressourcen

- Ansätze für die Migration von Benutzern zu Amazon Cognito Cognito-Benutzerpools

Importieren von Benutzern aus einer CSV-Datei in Benutzerpools

Wenn Sie über einen externen Identitätsspeicher verfügen und Zeit haben, Ihren Benutzerpool für neue lokale Benutzer vorzubereiten, kann ein Massenimport von Benutzern aus einer CSV-Datei (Comma-Separated Values) eine einfache und kostengünstige Option für eine Migration zu einem Amazon Cognito Cognito-Benutzerpool sein. Ein CSV-Dateiimport ist ein Vorgang, bei dem eine Vorlagendatei heruntergeladen und gefüllt wird. Anschließend wird die Datei in einem Importjob an Ihren Benutzerpool übergeben. Sie können einen CSV-Import verwenden, um schnell Testbenutzer zu erstellen. Sie können die Datei auch programmgesteuert mit API-Leseanforderungen an Ihren externen Identitätsspeicher füllen und anschließend deren Details und Attribute analysieren, um Schreibvorgänge in die Datei durchzuführen.

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 in einem `RESET_REQUIRED` Zustand, wenn sie mit dieser Methode importiert werden.

Der einfachste Weg, Benutzer aus einer CSV-Datei zu importieren, besteht darin, die passwortlose Anmeldung in Ihrem Benutzerpool zu aktivieren. Mit den Attributen E-Mail-Adresse und Telefonnummer und der richtigen Benutzerpool-Konfiguration können sich Benutzer sofort nach Abschluss Ihres Importvorgangs mit E-Mail- oder SMS-Einmalkennwörtern (OTPs) anmelden. Weitere Informationen finden Sie unter Von importierten Benutzer verlangen, dass sie ihre Passwörter zurücksetzen.

Sie können die Passwörter Ihrer Benutzer auch mit einer AdminSetUserPasswordAPI-Anfrage festlegen, die den `Permanent` Parameter auf festlegt`true`. Der CSV-Import trägt nicht zur Abrechnung der monatlich aktiven Nutzer (MAUs) in Ihrem Benutzerpool bei. Bei Vorgängen zum Zurücksetzen von Passwörtern wird jedoch etwas generiert. MAUs Um die Kosten zu senken, wenn Sie eine große Anzahl von Benutzern mit einem Kennwort importieren, die möglicherweise nicht sofort aktiv sind, richten Sie Ihre Anwendung so ein, dass Benutzer bei der Anmeldung und der Aufforderung zur Eingabe eines neuen Kennworts aufgefordert werden. `RESET_REQUIRED`

> **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.

Schritte zum Erstellen eines Benutzerimportauftrags

1. Erstellen Sie eine Amazon CloudWatch Logs-Rolle 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. Wird verwendet CloudWatch , um das Ereignisprotokoll zu überprüfen.
7. Verlangen Sie von den importierten Benutzer, dass sie ihre Passwörter zurücksetzen.

Weitere -Quellen

- Referenzarchitektur für den Export von Cognito-Benutzerprofilen zum Exportieren von Benutzerkonten zwischen Benutzerpools

Themen

- Die IAM-Rolle „ CloudWatch Logs“ erstellen
- Erstellen der CSV-Datei für den Benutzerimport
- Erstellen und Ausführen des Amazon-Cognito-Benutzerpool-Importauftrags
- Die Importergebnisse des Benutzerpools in der CloudWatch Konsole anzeigen
- Von importierten Benutzer verlangen, dass sie ihre Passwörter zurücksetzen

Die IAM-Rolle „ CloudWatch Logs“ erstellen

Wenn Sie die Amazon Cognito CLI oder API verwenden, müssen Sie eine CloudWatch IAM-Rolle erstellen. Das folgende Verfahren beschreibt, wie Sie eine IAM-Rolle erstellen, mit der Amazon Cognito die Ergebnisse Ihres Importauftrags in Logs schreiben kann. CloudWatch

> **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 CloudWatch Logs für den Benutzerpool-Import (AWS CLI, API)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Erstellen Sie eine neue IAM-Rolle für eine. 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. *REGION*Ersetzen Sie sie beispielsweise `us-east-1` durch die AWS-Region Ihres Benutzerpools. *ACCOUNT*Ersetzen Sie es zum Beispiel durch Ihre AWS-Konto ID`111122223333`.

      JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
```

```
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333: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 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:

   JSON

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "cognito-idp.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }
```

7. Wählen Sie Richtlinie aktualisieren.
8. Beachten Sie die 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. 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.
4. Wählen Sie das Menü Benutzer.
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. Dabei *USER_POOL_ID* handelt es sich um die Benutzerpool-ID für den Benutzerpool, in den Sie Benutzer importieren möchten:

```
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 Mit Benutzerattributen arbeiten.

- 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 das Geburtsdatum, falls vorhanden, müssen das folgende Format *mm/dd/yyyy*haben. 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 mithilfe der Amazon Cognito Cognito-Konsole und der 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. 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.
4. Wählen Sie das Menü Benutzer.
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 Die IAM-Rolle „ CloudWatch Logs“ erstellen beschriebenen Berechtigungen und Vertrauensrichtlinien verfügen.
8. Wählen Sie unter CSV hochladen die Option Datei auswählen und hängen Sie die CSV-Datei an, die Sie vorbereitet haben.
9. 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.
10. Wenn Sie Ihren Auftrag erstellt, aber noch nicht gestartet haben, können Sie ihn später starten. Wählen Sie im Menü Benutzer unter Benutzer importieren Ihren Importjob aus und wählen Sie dann Start aus. Sie können auch eine StartUserImportJobAPI-Anfrage von einem AWS SDK aus einreichen.

11. Überwachen Sie den Fortschritt Ihres Benutzerimportauftrags im Menü Benutzer unter Benutzer importieren. Wenn Ihr Auftrag nicht erfolgreich ist, können Sie den Status-Wert auswählen. Wenn Sie weitere Informationen wünschen, wählen Sie CloudWatch Protokolle anzeigen aus, um weitere Informationen zu erhalten, und überprüfen Sie alle Probleme in der CloudWatch Protokollkonsole.

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 Benutzerimportauftrag, indem Sie den folgenden CLI-Befehl ausführen. Dabei *JOB_NAME* stehen der Name, den Sie für den Job wählen, *USER_POOL_ID* die Benutzerpool-ID für den Benutzerpool, dem die neuen Benutzer hinzugefügt werden, und *ROLE_ARN* der Rollen-ARN, in dem Sie erhalten habenDie IAM-Rolle „ CloudWatch Logs“ erstellen:

```
aws cognito-idp create-user-import-job --job-name "JOB_NAME" --user-pool-id
 "USER_POOL_ID" --cloud-watch-logs-role-arn "ROLE_ARN"
```

Der in der Antwort *PRE_SIGNED_URL* zurückgegebene Wert ist 15 Minuten gültig. Nach dieser Zeit läuft sie ab, und Sie müssen einen neuen Benutzer-Importauftrag erstellen, um eine neue URL zu erhalten.

Example Antwort:

```
{
    "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. Ihre Benutzerpools behalten die Informationen in Ihren Importdateien nicht bei, nachdem Sie Ihre Importjobs ausgeführt haben. Nachdem sie abgeschlossen sind oder abgelaufen sind, löscht Amazon Cognito Ihre hochgeladene CSV-Datei.

Beschreiben eines Benutzer-Importauftrags

Um eine Beschreibung Ihres Benutzerimport-Jobs zu erhalten, verwenden Sie den folgenden Befehl. Dabei *`USER_POOL_ID`* steht Ihre Benutzerpool-ID und *`JOB_ID`* die Job-ID, die bei der Erstellung des Benutzerimport-Jobs 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 *PRE_SIGNED_URL* ist dies die URL, zu der Sie die CSV-Datei hochgeladen haben. Das *ROLE_ARN* ist der ARN der CloudWatch Logs-Rolle, den Sie bei der Erstellung 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 *PAGINATION_TOKEN* Zeichenfolge nach dem zweiten Job gibt an, dass für diesen Listenbefehl zusätzliche Ergebnisse 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
    }
}
```

Die Importergebnisse des Benutzerpools in der CloudWatch Konsole anzeigen

Sie können die Ergebnisse Ihres Importauftrags in der CloudWatch Amazon-Konsole einsehen.

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 der an AWS Management Console 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 hat die Form*`JOB_ID`*/*`JOB_NAME`*. 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

Der Status von erfolgreich importierten Benutzern ist 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 übernommen werden, 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

Wenn Ihr Benutzerpool nur die kennwortbasierte Anmeldung anbietet, müssen Benutzer ihre Passwörter nach dem Import zurücksetzen. Bei der ersten Anmeldung können sie ein beliebiges Passwort eingeben. Amazon Cognito fordert sie auf, in der API-Antwort auf die Anmeldeanfrage Ihrer Anwendung ein neues Passwort einzugeben.

Wenn Ihr Benutzerpool kennwortlose Authentifizierungsfaktoren hat, verwendet Amazon Cognito standardmäßig die für importierte Benutzer. Sie werden nicht zur Eingabe eines neuen Passworts aufgefordert und können sich sofort mit einem passwortlosen E-Mail- oder SMS-OTP anmelden. Sie können Benutzer auch auffordern, ein Passwort festzulegen, damit sie andere Anmeldemethoden wie Benutzername/Passwort und Hauptschlüssel verwenden können. Die folgenden Bedingungen gelten für die kennwortlose Anmeldung nach dem Benutzerimport.

1. Sie müssen Benutzer mit einem Attribut importieren, das einem verfügbaren kennwortlosen Anmeldefaktor entspricht. Wenn sich Benutzer mit einer E-Mail-Adresse anmelden können, müssen Sie ein Attribut importieren. `email` Wenn es sich um eine Telefonnummer handelt, müssen Sie ein `phone_number` Attribut importieren. Wenn beides der Fall ist, importieren Sie einen Wert für eines der Attribute.
2. Normalerweise importieren Benutzer in einem `RESET_REQUIRED` Zustand, in dem sie ihr Passwort zurücksetzen müssen. Wenn sie mit der Möglichkeit importiert werden, sich mit einem kennwortlosen Faktor anzumelden, setzt Amazon Cognito ihren Status auf. `CONFIRMED`

Weitere Informationen zur kennwortlosen Authentifizierung, einschließlich deren Einrichtung und Aufbau des Authentifizierungsablaufs in Ihrer Anwendung, finden Sie unter. Authentifizierung mit Amazon Cognito Cognito-Benutzerpools

Das folgende Verfahren beschreibt die Benutzererfahrung bei einem benutzerdefinierten Anmeldemechanismus mit lokalen Benutzern in einer `RESET_REQUIRED` nach dem Import einer CSV-Datei. Wenn sich Ihre Benutzer mit verwaltetem Login anmelden, weisen Sie sie an, die Option Passwort vergessen? auszuwählen Option, geben Sie den Code aus ihrer E-Mail oder Textnachricht ein und legen Sie ein Passwort fest.

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.

Mit Benutzerattributen arbeiten

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 im auch benutzerdefinierte Attribute 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.

Bereinigen Sie die Eingaben für Zeichenkettenwerte von Benutzerattributen, bevor Sie sie an Ihren Benutzerpool senden. Eine Methode zur Analyse der vorgeschlagenen Benutzerattributwerte ist ein Lambda-Trigger wie bei der Anmeldung.

> **Note**
>
> In einigen Dokumentationen und Standards werden Attribute als Member bezeichnet.

Themen

- Standardattribute
- Benutzernamen und bevorzugter Benutzername
- Anpassen von Anmeldeattributen
- Custom attributes (Benutzerdefinierte Attribute)
- Attributberechtigungen und -bereiche

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:

- `name`
- `family_name`
- `given_name`
- `middle_name`
- `nickname`
- `preferred_username`
- `profile`
- `picture`
- `website`
- `gender`
- `birthdate`

- `zoneinfo`
- `locale`
- `updated_at`
- `address`
- `email`
- `phone_number`
- `sub`

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.

Standardattribute haben vordefinierte Eigenschaften, die Sie im `SchemaAttributes` Parameter einer DescribeUserPool API-Antwort anzeigen können. Sie können benutzerdefinierte Werte für diese Attributeigenschaften wie Datentyp, Veränderbarkeit und Längenbeschränkungen festlegen. Um Standardattributeigenschaften zu ändern, legen Sie ihre benutzerdefinierten Werte im CreateUserPool Schema-Parameter fest. Im Schema legen Sie auch die erforderlichen Attribute fest. Sie können die Eigenschaften von Standardattributen nicht ändern, wenn Sie Benutzerpools in der Amazon Cognito Cognito-Konsole erstellen.

> **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 ein gültiges 10-stelliges Datum im Format sein YYYY-MM-DD.

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 im Benutzermenü der Amazon Cognito Cognito-Konsole bearbeiten, 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 folgt und bis zu 2048 Zeichen lang sein 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 im Benutzermenü der Amazon Cognito Cognito-Konsole bearbeiten, 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 das Anmelde-Menü.
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)

Ein Attribut ist ein Alias, wenn Benutzer einen Benutzernamen haben, sich aber auch mit diesem Attribut anmelden können. Richten Sie Aliase ein, wenn Sie Ihren Benutzern die Möglichkeit

geben möchten, zwischen ihrem Benutzernamen und anderen Attributwerten im Feld für den Benutzernamen Ihres Anmeldeformulars zu wählen. Das `username` Attribut 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.

Phone number or email address (console)

Sie müssen E-Mail-Adresse und Telefonnummer als Alias-Attribute festlegen, wenn Sie einen Benutzerpool erstellen.

So erstellen Sie einen Benutzerpool mit Benutzernamen-Aliasnamen in der Amazon Cognito Cognito-Konsole

1. Wechseln Sie zu Amazon Cognito in der AWS Management Console. Wenn Sie von der Konsole dazu aufgefordert werden, geben Sie Ihre Anmeldeinformationen ein. AWS
2. Erstellen Sie mit den Schaltflächen Erste Schritte oder Benutzerpool erstellen einen neuen Benutzerpool.
3. Wählen Sie unter Anwendung definieren die Anwendungseinstellungen aus.
4. Aktivieren Sie unter Optionen für Anmeldekennungen konfigurieren das Kontrollkästchen neben Benutzername und mindestens eine der anderen Optionen, E-Mail und Telefonnummer.
5. Wählen Sie Ihre Alias-Attribute als Erforderliche Attribute für die Registrierung aus. Im Anmeldeformular für verwaltete Logins fordert Amazon Cognito neue Benutzer auf, Werte für die erforderlichen Attribute anzugeben.
6. Richten Sie unter Rückgabe-URL hinzufügen eine Rückruf-URL für die Anwendung ein, die nach der verwalteten Anmeldung weitergeleitet werden soll.
7. Wählen Sie Erstellen aus.

Phone number or email address (API/SDK)

Erstellen Sie mit dem CreateUserPoolAPI-Vorgang einen neuen Benutzerpool. Konfigurieren Sie den `AliasAttributes` Parameter wie gezeigt. Sie können den `email` Eintrag entfernen, wenn Sie nur Telefonnummern-Aliase verwenden möchten, oder den `phone_number` Eintrag entfernen, wenn Sie nur E-Mail-Adress-Aliase verwenden möchten.

```
"AliasAttributes": [
   "email",
   "phone_number"
],
```

Preferred username (API/SDK)

Die Amazon Cognito Cognito-Konsole erstellt Benutzerpools ohne `preferred_username` Alias. Um Benutzerpools mit einem `preferred_username` Alias zu erstellen, richten Sie Benutzerpools mit CreateUserPoolAPI-Anfragen in einem AWS SDK ein. Um die Erstellung von bevorzugten Benutzernamenattributen bei der Registrierung zu unterstützen, legen Sie diese `preferred_username` als erforderliches Attribut fest. Im Anmeldeformular für verwaltete Logins fordert Amazon Cognito neue Benutzer auf, Werte für die erforderlichen Attribute anzugeben. Sie können es in der Amazon Cognito Cognito-Konsole `preferred_username` als erforderliches Attribut festlegen, aber dadurch ist es nicht als Alias verfügbar.

Als Alias konfigurieren

`preferred_username`Als Alias im `AliasAttributes` Parameter einer `CreateUserPool` Anfrage konfigurieren, wie in der Abbildung gezeigt. Entfernen Sie alle Werte, die Sie nicht als Alias-Attribute verwenden möchten, aus der Liste.

```
"AliasAttributes": [
   "email",
   "phone_number",
   "preferred_username"
],
```

Konfigurieren Sie nach Bedarf

Im Anmeldeformular für verwaltete Logins fordert Amazon Cognito neue Benutzer auf, Werte für die erforderlichen Attribute anzugeben. Konfigurieren Sie `preferred_username` die Konfiguration nach Bedarf im `SchemaAttributes` Parameter einer Anfrage. CreateUserPool

Um den bevorzugten Benutzernamen als erforderliches Attribut festzulegen, konfigurieren Sie ihn wie in der Abbildung gezeigt. Im folgenden Beispiel wird das Standardschema von so geändert`preferred_username`, dass es wie erforderlich festgelegt wird. Andere Schemaparameter wie `AttributeDataType` (standardmäßig`string`) und `StringAttributeConstraints` (standardmäßig 1—99 Zeichen lang) gehen von Standardwerten aus.

```
"Schema": [
   {
      "Name": "preferred_username",
```

```
        "Required": true
    }
]
```

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.

Um Benutzernamenattribute auszuwählen, wählen Sie beim Erstellen Ihres Benutzerpools nicht Benutzername als Anmeldeoption aus.

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.

Wenn Sie sowohl E-Mail-Adresse als auch Telefonnummer als Benutzernamenattribute auswählen, können sich Benutzer mit dem einen oder anderen anmelden, auch wenn sie Werte für beide Attribute angeben. Der Benutzername für die Anmeldung basiert auf dem Wert, den Sie im `Username` Parameter von SignUpübergeben.

> **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.

Username attributes (console)

Mit dem folgenden Verfahren wird ein Benutzerpool mit den Attributen E-Mail-Adresse oder Telefonnummer erstellt. Der Unterschied beim Verfahren für Benutzernamenattribute in der Amazon Cognito Cognito-Konsole besteht darin, dass Sie den Benutzernamen nicht auch als Anmeldeattribut festlegen.

So erstellen Sie einen Benutzerpool mit Benutzernamenattributen in der Amazon Cognito Cognito-Konsole

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. Erstellen Sie mit den Schaltflächen Erste Schritte oder Benutzerpool erstellen einen neuen Benutzerpool.
3. Wählen Sie unter Anwendung definieren die Anwendungseinstellungen aus.
4. Wählen Sie unter Optionen für Anmeldekennungen konfigurieren unter Optionen für Anmeldekennungen Ihre Benutzernamenattribute aus: E-Mail, Telefonnummer oder beides. Lassen Sie das Häkchen für den Benutzernamen deaktiviert.
5. Es hat sich bewährt, dass Sie Ihre Benutzernamenattribute als Erforderliche Attribute für die Registrierung auswählen. Im Anmeldeformular für verwaltete Logins fordert Amazon Cognito neue Benutzer auf, Werte für die erforderlichen Attribute anzugeben. Wenn Sie Ihre Benutzernamenattribute nicht wie erforderlich festlegen, fordert Amazon Cognito neue Benutzer nicht auf, Werte für sie anzugeben. In diesem Szenario müssen Sie Ihre Anwendung so konfigurieren, dass E-Mail-Adressen oder Telefonnummern für jeden Benutzer erfasst und übermittelt werden, bevor dieser sich anmelden kann.
6. Richten Sie unter Rückgabe-URL hinzufügen eine Rückruf-URL für die Anwendung ein, die nach der verwalteten Anmeldung weitergeleitet wird.
7. Wählen Sie Erstellen aus.

Username attributes (API/SDK)

Konfigurieren Sie den `UsernameAttributes` Parameter in einer CreateUserPoolAnfrage wie gezeigt. Um die Anmeldung nur mit Benutzernamen mit E-Mail-Adresse zu ermöglichen, geben Sie in dieser Liste die Option „`email`Allein“ an. Um die Anmeldung nur mit Nutzernamen mit Telefonnummer zu ermöglichen, geben Sie „Nur“ an. `phone_number` Dieser Parameter hat Vorrang vor dem Benutzernamen als Anmeldeoption.

```
"UsernameAttributes": [
   "email",
   "phone_number"
],
```

Wenn Sie Benutzernamenattribute konfigurieren, können Sie SignUpAPI-Anfragen stellen, die eine E-Mail-Adresse oder Telefonnummer im `username` Parameter übergeben. Im Folgenden wird das Verhalten der `SignUp` Code-API-Operation mit Benutzernamenattributen beschrieben.

- Wenn die `username` Zeichenfolge beispielsweise ein gültiges E-Mail-Adressformat hat`user@example.com`, füllt der Benutzerpool das `email` Attribut des Benutzers automatisch mit dem `username` Wert aus.
- Wenn die `username` Zeichenfolge beispielsweise ein gültiges Telefonnummernformat hat`+12065551212`, füllt der Benutzerpool das `phone_number` Attribut des Benutzers automatisch mit dem `username` Wert aus.
- Wenn die `username`-Zeichenfolge kein gültiges E-Mail-Adress- oder Telefonnummernformat besitzt, gibt die `SignUp`-API eine Ausnahme zurück.
- Wenn die Zeichenfolge `username` eine E-Mail-Adresse oder Telefonnummer enthält, die bereits verwendet wird, gibt die `SignUp`-API eine Ausnahme zurück.
- Die `SignUp` API füllt das `username` Attribut mit einer UUID für Ihren Benutzer. Diese UUID hat den gleichen Wert wie der `sub`-Anspruch im Token für die Benutzeridentität.

Sie können eine E-Mail-Adresse oder Telefonnummer anstelle des Benutzernamens in allen APIs Fällen verwenden, mit Ausnahme des Vorgangs. ListUsers In `ListUsers` API-Anfragen können Sie einen Wert `Filter` von `email` oder angebenphone_number. Wenn Sie nach filternusername, müssen Sie den UUID-Benutzernamen angeben, nicht die E-Mail-Adresse oder Telefonnummer.

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. Der Name eines benutzerdefinierten Attributs muss dem Muster für reguläre Ausdrücke entsprechen, das im `Name` Parameter von beschrieben ist. SchemaAttributeType

Jedes benutzerdefinierte Attribut hat die folgenden Eigenschaften:

- Sie können es als eine Zeichenfolge, eine Zahl, einen booleschen Wert oder `DateTime` ein Objekt definieren. Amazon Cognito schreibt benutzerdefinierte Attributwerte nur als Strings in das ID-Token.

> **Note**
>
> In der Amazon Cognito Cognito-Konsole können Sie nur benutzerdefinierte Attribute der Datentypen Zeichenfolge und Zahl hinzufügen. Zusätzliche Optionen wie boolesche Datentypen und `DateTime` Attributdatentypen sind nur in den `SchemaAttributes` Property of CreateUserPool- und UpdateUserPoolAPI-Anfragen verfügbar.

- 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 das Anmeldemenü und wählen Sie im Abschnitt Benutzerdefinierte Attribute die Option 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 Änderungen speichern aus.

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 das Menü App-Clients und wählen Sie einen App-Client aus der Liste aus.
5. Wählen Sie auf der Registerkarte Attributberechtigungen die Option 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.

Für jeden App-Client können Sie Attribute 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, zu dessen Schreiben sie nicht berechtigt ist, kehrt Amazon Cognito zurück`NotAuthorizedException`. 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. Dies ist möglich, aber nicht praktikabel, da sich der Benutzer nur einmal anmelden kann. Bei Anmeldeversuchen nach dem ersten Versuch lehnt Amazon Cognito den Versuch aufgrund der Zuordnungsregel zu einem jetzt nicht beschreibbaren Attribut ab.
- `AdminCreateUser`: Sie geben einen Wert für ein unveränderliches Attribut an.

Attributberechtigungen mit Geltungsbereichen

In Benutzerpools, die Sie mit einem AWS SDK oder CDK, der REST-API oder dem konfigurieren, können Sie den AWS CLI Lese- oder Schreibzugriff des App-Clients mit dem OIDC-Bereich konfigurieren. `oidc:profile` Das `oidc:profile` gewährt Lese- oder Schreibzugriff auf die folgenden Standardattribute:

- `name`
- `family_name`
- `given_name`
- `middle_name`
- `nickname`
- `preferred_username`
- `profile`
- `picture`
- `website`
- `gender`

- `birthdate`
- `zoneinfo`
- `locale`

Diese Liste besteht aus den OIDC-Standardattributen abzüglich `email``phone_number`,`sub`, und`address`, wie in Abschnitt 2.4 der OIDC-Spezifikation definiert. Informationen zu den Bereichen, die Sie Ihren App-Clients zuweisen können, finden Sie unter Bereiche, M2M und APIs mit Ressourcenservern.

Um Ihren App-Client so zu konfigurieren, dass er in die Attribute im `oidc:profile` Bereich schreibt, legen Sie in einer WriteAttributesoder `oidc:profile` API-Anfrage den Wert auf sowie alle anderen Attribute fest, deren Änderung Sie durch Ihre Anwendung zulassen möchten. CreateUserPoolClientUpdateUserPoolClient Um ebenfalls Lesezugriff auf diese Attribute zu gewähren, erhöhen Sie `oidc:profile` den Wert von ReadAttributes.

Sie können Attributberechtigungen und -bereiche nach dem Erstellen des Benutzerpools ändern.

JSON-Webtoken (JWTs) für den Benutzerpool verstehen

Token sind Authentifizierungsartefakte, die Ihre Anwendungen als Nachweis für die OIDC-Authentifizierung und zur Anforderung des Zugriffs auf Ressourcen verwenden können. 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 Behauptungen wie`scope`, mit denen der authentifizierte Benutzer auf Self-Service-API-Operationen von Drittanbietern APIs, Amazon Cognito Cognito-Benutzern zugreifen kann, und die. UserInfo-Endpunkt 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 Token als base64url -kodierte Zeichenfolgen aus. Sie können jede Amazon Cognito Cognito-ID oder jedes Zugriffstoken in Klartext-JSON `base64url` 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. Sie können die Token verwenden, um Ihren Benutzern Zugriff auf nachgelagerte Ressourcen APIs wie Amazon API Gateway zu 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 Bereiche der OAuth Version 2.0, Gruppenzugehörigkeit im Benutzerpool, Benutzerattribute und

andere. 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

- Grundlegendes zum Identitätstoken (ID)
- Das Zugriffstoken verstehen
- Tokens aktualisieren
- Benutzersitzungen mit Token-Widerruf beenden
- Überprüfung von JSON-Web-Tokens
- Verwaltung des Ablaufs und der Zwischenspeicherung von Benutzerpool-Tokens

Grundlegendes zum Identitätstoken (ID)

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 Überprüfung von 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 mit verwaltetem Login anmeldet, setzt Amazon Cognito Sitzungscookies, die für 1 Stunde gültig sind. Wenn Sie die verwaltete Anmeldung für die Authentifizierung in Ihrer Anwendung verwenden und eine Mindestdauer von weniger als 1 Stunde für Ihre Zugriffs- und ID-Token angeben, haben Ihre Benutzer weiterhin eine gültige Sitzung, bis das Cookie abläuft. 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 Ihren Benutzerpool-Signaturschlüssel IDs 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 RS256 kryptografischen Algorithmus, bei dem es sich um eine RSA-Signatur mit SHA-256 handelt.

Weitere Informationen über den `alg`-Parameter finden Sie unter dem Algorithmus-(alg)-Header-Parameter.

Standard-Payload für ID-Token

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 Kennung (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 aus einem gleichnamigen Parameter, den Sie zu Anfragen an Ihren OAuth 2.0-Endpunkt hinzufügen können. `authorize` 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. Überprüfung von JSON-Web-Tokens.

Das Zugriffstoken verstehen

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.

Mit dem Feature-Plan Essentials oder Plus können Sie auch einen Lambda-Trigger vor der Token-Generierung implementieren, der Ihren Zugriffstoken zur Laufzeit Bereiche hinzufügt. Weitere Informationen finden Sie unter Lambda-Auslöser für die Vorab-Generierung von Token.

Das Zugriffstoken eines Benutzers mit dem `openid` Gültigkeitsbereich ist die Erlaubnis, weitere Informationen zu den Attributen Ihres Benutzers von der anzufordern. UserInfo-Endpunkt Die Menge der Informationen vom `userInfo` Endpunkt ergibt sich aus den zusätzlichen Bereichen im Zugriffstoken: zum Beispiel `profile` für alle Benutzerdaten, `email` für deren E-Mail-Adresse.

Das Zugriffstoken eines Benutzers mit dem `aws.cognito.signin.user.admin` Gültigkeitsbereich ist die Erlaubnis, Benutzerattribute zu lesen und zu schreiben, Authentifizierungsfaktoren aufzulisten, Einstellungen für die Multi-Faktor-Authentifizierung (MFA) zu konfigurieren und gespeicherte Geräte zu verwalten. Die Zugriffsebene auf Attribute, die Ihr Zugriffstoken für diesen Bereich gewährt, entspricht den read/write Attributberechtigungen, die Sie Ihrem App-Client zuweisen.

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 Überprüfung von 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 nicht ein Minimum von weniger als einer Stunde an, wenn Sie die verwaltete Anmeldung verwenden. Bei der verwalteten Anmeldung werden Browser-Cookies gesetzt, die eine Stunde lang gültig sind. Wenn Sie eine Dauer des Zugriffstokens von weniger als einer Stunde konfigurieren, hat dies keine Auswirkungen auf die Gültigkeit des verwalteten Anmelde-Cookies und die Fähigkeit der Benutzer, sich nach der ersten Anmeldung eine Stunde lang ohne zusätzliche Anmeldeinformationen erneut zu authentifizieren.

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 Ihren Benutzerpool-Signaturschlüssel IDs am Endpunkt einsehen. `jwks_uri`

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 RS256 kryptografischen Algorithmus, bei dem es sich um eine RSA-Signatur mit SHA-256 handelt.

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 Kennung (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.us-east-1.amazonaws.com/us-east-1_EXAMPLE
```

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, validiert Amazon Cognito keine Zugriffs- und ID-Tokens mehr mit demselben `origin_jti` Wert.

token_use

Der vorgesehene Zweck des Tokens. In einem Zugriffstoken ist der Wert `access`.

scope

Eine Liste von OAuth 2.0-Bereichen, die für den angemeldeten Benutzer ausgestellt wurde. Bereiche definieren den Zugriff, den das Token auf externe APIs Self-Service-Operationen für Benutzer und Benutzerdaten auf dem Endpunkt gewährt. `userInfo` 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 des Benutzers im Benutzerpool.

Weitere -Quellen

- So passen Sie Zugriffstoken in Amazon Cognito Cognito-Benutzerpools an

Zugriffstoken-Signatur

Die Signatur des Zugriffstokens, signiert mit dem am `.well-known/jwks.json` Endpunkt angekündigten Schlüssel, bestätigt die Integrität des Token-Headers und der Nutzlast. Wenn Sie Zugriffstoken verwenden, um den Zugriff auf externe Daten zu autorisieren, konfigurieren Sie Ihren API-Autorisierer immer so APIs, dass er diese Signatur anhand des Schlüssels verifiziert, mit dem sie signiert wurde. Weitere Informationen finden Sie unter Überprüfung von JSON-Web-Tokens.

Tokens aktualisieren

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.

Abrufen neuer Zugriffs- und Identitätstoken mit einem Aktualisierungstoken

Amazon Cognito gibt Aktualisierungstoken als Reaktion auf eine erfolgreiche Authentifizierung mit dem verwalteten Anmeldeautorisierungscode-Flow und mit API-Operationen oder SDK-Methoden aus. Das Aktualisierungstoken gibt neue ID- und Zugriffstoken sowie optional ein neues Aktualisierungstoken zurück. Sie können Aktualisierungstoken auf folgende Weise verwenden.

GetTokensFromRefreshToken

Der GetTokensFromRefreshTokenAPI-Vorgang gibt neue ID- und Zugriffstoken aus einem gültigen Aktualisierungstoken aus. Sie erhalten auch ein neues Aktualisierungstoken, wenn Sie die Rotation des Aktualisierungstokens aktiviert haben.

InitiateAuth und AdminitiateAuth

Die InitiateAuthAPI-Operationen AdminInitiateAuthoder beinhalten den `REFRESH_TOKEN_AUTH` Authentifizierungsablauf. In diesem Ablauf übergeben Sie ein Aktualisierungstoken und erhalten neue ID- und Zugriffstoken. Sie können sich nicht mit `REFRESH_TOKEN_AUTH` In-App-Clients authentifizieren, wenn die Rotation des Aktualisierungstokens aktiviert ist.

OAuth Token-Endpunkt

Der Token-Endpunkt in Benutzerpools mit einer Domäne hat einen `refresh_token` Grant-Typ, der neue ID-, Zugriffs- und optional (bei Rotation des Aktualisierungstokens) Aktualisierungstoken von einem gültigen Aktualisierungstoken ausgibt.

Token-Rotation aktualisieren

Sie können optional die Rotation des Aktualisierungstokens in Ihrem App-Client konfigurieren. Mit der Rotation des Aktualisierungstokens kann Ihr Client das ursprüngliche Aktualisierungstoken für ungültig erklären und bei jeder Token-Aktualisierung ein neues Aktualisierungstoken ausgeben. Wenn diese Einstellung aktiviert ist, gibt jede erfolgreiche Anforderung in allen Formen der Token-Aktualisierung eine neue ID, einen neuen Zugriff und ein neues Aktualisierungstoken zurück. Wenn diese Einstellung deaktiviert ist, geben Token-Aktualisierungsanforderungen nur neue Zugriffs- und ID-Token zurück, und das ursprüngliche Aktualisierungstoken bleibt gültig. Das neue Aktualisierungstoken ist für die verbleibende Dauer des ursprünglichen Aktualisierungstokens gültig. Sie können App-Clients so konfigurieren, dass sie Aktualisierungstoken rotieren oder das

ursprüngliche Aktualisierungstoken übernehmen. Um für kurze Zeit Wiederholungsversuche zu ermöglichen, können Sie auch eine Übergangszeit von bis zu 60 Sekunden für das ursprüngliche Aktualisierungstoken konfigurieren.

Wissenswertes zur Rotation von Aktualisierungstoken

- Nachdem Sie die Rotation der Aktualisierungstoken aktiviert haben, werden neue Ansprüche in JSON-Webtoken aus Ihrem Benutzerpool hinzugefügt. Die `origin_jti`- und `jti`-Ansprüche werden zu Zugriffs- und ID-Token hinzugefügt. Diese Ansprüche erhöhen die Größe von JWTs.
- Die Rotation des Aktualisierungstokens ist nicht mit dem Authentifizierungsablauf kompatibel`REFRESH_TOKEN_AUTH`. Um die Rotation von Aktualisierungstoken zu implementieren, müssen Sie diesen Authentifizierungsablauf in Ihrem App-Client deaktivieren und Ihre Anwendung so gestalten, dass sie Token-Aktualisierungsanforderungen mit dem GetTokensFromRefreshTokenAPI-Vorgang oder der entsprechenden SDK-Methode sendet.
- Wenn die Rotation des Aktualisierungstokens inaktiv ist, können Sie Token-Aktualisierungsanforderungen entweder mit oder abschließen. `GetTokensFromRefreshToken` `REFRESH_TOKEN_AUTH`
- Wenn die Gerätespeicherung in Ihrem Benutzerpool aktiv ist, müssen Sie den Geräteschlüssel in Anfragen angeben. `GetTokensFromRefreshToken` Wenn Ihr Benutzer keinen bestätigten Geräteschlüssel hat, den Ihre Anwendung in der ersten Authentifizierungsanfrage übermittelt, gibt Amazon Cognito einen neuen aus. Um Token in dieser Konfiguration zu aktualisieren, müssen Sie einen Geräteschlüssel angeben, unabhängig davon, ob Sie einen in der Authentifizierungsantwort angegeben `AuthParameters` oder einen neuen erhalten haben.
- Sie können in Ihrer `GetTokensFromRefreshToken` Anfrage `ClientMetadata` an den Lambda-Trigger vor der Token-Generierung übergeben. Diese Daten, die an das Eingabeereignis für Ihren Trigger übergeben werden, liefern zusätzlichen Kontext, den Sie in der benutzerdefinierten Logik Ihrer Lambda-Funktion verwenden können.

Aus Sicherheitsgründen sollten Sie die Rotation der Aktualisierungstoken auf Ihren App-Clients aktivieren.

Enable refresh token rotation (console)

Mit dem folgenden Verfahren wird die Rotation des Aktualisierungstokens für Ihren App-Client ein- oder ausgeschaltet. Für dieses Verfahren ist ein vorhandener App-Client erforderlich. Weitere Informationen zum Erstellen eines App-Clients finden Sie unterAnwendungsspezifische Einstellungen mit App-Clients.

Um die Rotation des Aktualisierungstokens

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. Navigieren Sie zum Menü App-Clients und wählen Sie einen vorhandenen App-Client aus.
5. Wählen Sie auf der Seite im Bereich App-Client-Informationen die Option Bearbeiten aus.
6. Suchen Sie unter Erweiterte Sicherheitskonfigurationen nach der Option Aktualisierungstoken-Rotation aktivieren.
7. Um die Rotation zu aktivieren, aktivieren Sie das Kontrollkästchen. Um die Rotation zu deaktivieren, deaktivieren Sie das Kontrollkästchen.
8. Geben Sie unter Kulanzzeitraum für die Rotation des Aktualisierungstokens eine Anzahl von Sekunden (bis zu 60) ein, die Sie als Verzögerung festlegen möchten, bevor das rotierte Aktualisierungstoken gesperrt wird.

Enable refresh token rotation (API)

Konfigurieren Sie die Rotation des Aktualisierungstokens in einer CreateUserPoolClientoder API-Anfrage UpdateUserPoolClient. Der folgende Teil der Anfrage aktiviert die Rotation des Aktualisierungstokens und legt die Kulanzzeit auf zehn Sekunden fest.

```
"RefreshTokenRotation" : {
   "Feature" : "ENABLED,
   "RetryGracePeriodSeconds" : 10
}
```

Aktualisierung von API- und SDK-Tokens

Es gibt zwei Möglichkeiten, das Aktualisierungstoken zu verwenden, um neue IDs und Zugriffstoken mit der Benutzerpools-API abzurufen, je nachdem, ob die Rotation des Aktualisierungstokens aktiv ist. Verwenden Sie in App-Clients mit aktiver Aktualisierungstoken-Rotation den GetTokensFromRefreshTokenAPI-Vorgang. Verwenden Sie in App-Clients ohne Aktualisierungstoken-Rotation den `REFRESH_TOKEN_AUTH` Ablauf der AdminInitiateAuthoder InitiateAuthAPI-Operationen.

> **Note**
>
> Benutzer können sich mit Benutzerpools in verwalteten Logins oder in benutzerdefinierten Anwendungen, die Sie mit Amazon Cognito AWS SDKs API-Vorgängen erstellen, authentifizieren. Der `REFRESH_TOKEN_AUTH` Ablauf und beide `GetTokensFromRefreshToken` können die Token-Aktualisierung für verwaltete Login-Benutzer abschließen. Die Tokenaktualisierung in benutzerdefinierten Anwendungen hat keine Auswirkungen auf verwaltete Anmeldesitzungen. Diese Sitzungen werden in einem Browser-Cookie gespeichert und sind eine Stunde lang gültig. Die `GetTokensFromRefreshToken` Antwort gibt neue ID-, Zugriffs- und optionale Aktualisierungstoken aus, erneuert jedoch nicht das verwaltete Anmeldesitzungscookie. `REFRESH_TOKEN_AUTH`Hist in App-Clients mit aktivierter Aktualisierungstoken-Rotation nicht verfügbar.

GetTokensFromRefreshToken

GetTokensFromRefreshTokengibt neue ID-, Zugriffs- und Aktualisierungstoken aus einer Anfrage zurück, die Sie mit einem Aktualisierungstoken autorisieren. Im Folgenden finden Sie ein Beispiel für einen Anfragetext für`GetTokensFromRefreshToken`. In Anfragen für diesen Vorgang können Sie Client-Metadaten an Lambda-Trigger senden.

```
{
    "RefreshToken": "eyJjd123abcEXAMPLE",
    "ClientId": "1example23456789",
    "ClientSecret": "myappclientsecret123abc",
    "ClientMetadata": {
      "MyMetadataKey" : "MyMetadataValue"
   },
}
```

AdminInitiateAuth/InitiateAuth

Um das Aktualisierungstoken zu verwenden, wenn die Aktualisierungstoken-Rotation inaktiv ist, verwenden Sie die InitiateAuthAPI-Operationen AdminInitiateAuthoder. 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.

Im Folgenden finden Sie ein Beispiel für einen Anforderungstext für eine Token-Aktualisierung mit der `AdminInitiateAuth` API `InitiateAuth` oder.

```
{
    "AuthFlow": "REFRESH_TOKEN_AUTH",
    "ClientId": "1example23456789",
    "UserPoolId": "us-west-2_EXAMPLE",
    "AuthParameters": {
        "REFRESH_TOKEN": "eyJjd123abcEXAMPLE",
        "SECRET_HASH": "kT5acwCVrbD6JexhW3EQwnRSe6fLuPTRkEQ50athqv8="
    }
}
```

OAuth Token-Aktualisierung

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.

Anfragen an den Token-Endpunkt sind in App-Clients verfügbar, bei denen die Aktualisierungstoken-Rotation aktiv ist, und in App-Clients, bei denen sie inaktiv ist. Wenn die Aktualisierungstoken-Rotation aktiv ist, gibt der Token-Endpunkt ein neues Aktualisierungstoken zurück.

Im Folgenden finden Sie eine Beispielanforderung mit einem Aktualisierungstoken.

```
POST /oauth2/token HTTP/1.1
Host: auth.example.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw
Content-Length: **

client_id=1example23456789&grant_type=refresh_token&refresh_token=eyJjd123abcEXAMPLE
```

Widerrufen von Aktualisierungstokens

Sie können Aktualisierungstokens widerrufen, die einem Benutzer gehören. Weitere Informationen über Token finden Sie unter Benutzersitzungen mit Token-Widerruf beenden.

> **Note**
>
> Durch das Widerrufen des Aktualisierungstokens werden alle ID- und Zugriffstoken widerrufen, die Amazon Cognito aus Aktualisierungsanfragen mit diesem Token ausgestellt hat.

Um Benutzer von allen aktuell angemeldeten Sitzungen abzumelden, widerrufen Sie alle ihre Token mit GlobalSignOutoder AdminUserGlobalSignOutAPI-Anfragen. 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 Cookies für verwaltete Anmeldesitzungen nicht automatisch ablaufen, kann sich Ihr Benutzer mit einem Sitzungscookie erneut authentifizieren, ohne dass zusätzliche Anmeldeinformationen angefordert werden müssen. Nachdem Sie Ihre verwalteten Anmeldebenutzer abgemeldet haben, leiten Sie sie zu dem weiterLogout-Endpunkt, wo Amazon Cognito ihr Sitzungscookie löscht.

Mit Aktualisierungstoken können Sie Benutzersitzungen in Ihrer App für eine lange Zeit aufrechterhalten. Im Laufe der Zeit möchten Ihre Benutzer möglicherweise einige Anwendungen deautorisieren, bei denen sie mit ihren Aktualisierungstoken angemeldet geblieben sind. Um Ihren Benutzer von einer einzelnen Sitzung abzumelden, 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 unterListe der API-Operationen, gruppiert nach Autorisierungsmodell.

Benutzersitzungen mit Token-Widerruf beenden

Sie können Aktualisierungstoken und Endbenutzersitzungen mit den folgenden Methoden 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.

RevokeToken Betrieb

RevokeTokenwiderruft alle Zugriffstoken für ein bestimmtes Aktualisierungstoken, einschließlich des ersten Zugriffstokens bei der interaktiven Anmeldung. Dieser Vorgang hat keine Auswirkungen auf die anderen Aktualisierungstoken des Benutzers oder auf die ID- und Zugriffstoken, die diesen anderen Aktualisierungstoken untergeordnet sind.

Endpunkt der Sperrung

Der Widerrufsendpunkt widerruft ein bestimmtes Aktualisierungstoken und alle ID- und Zugriffstoken, die das Aktualisierungstoken generiert hat. Dieser Endpunkt widerruft auch das anfängliche Zugriffstoken für die interaktive Anmeldung. Anfragen an diesen Endpunkt wirken sich weder auf die anderen Aktualisierungstoken des Benutzers noch auf die ID- und Zugriffstoken aus, die diesen anderen Aktualisierungstoken untergeordnet sind.

GlobalSignOut Betrieb

GlobalSignOutist ein Self-Service-Vorgang, den ein Benutzer mit seinem Zugriffstoken autorisiert. Dieser Vorgang widerruft alle Aktualisierungs-, ID- und Zugriffstoken des anfragenden Benutzers.

AdminUserGlobalSignOut Vorgang

AdminUserGlobalSignOutist ein serverseitiger Vorgang, den ein Administrator mit IAM-Anmeldeinformationen autorisiert. Dieser Vorgang widerruft alle Aktualisierungs-, ID- und Zugriffstoken des Zielbenutzers.

Wissenswertes über den Widerruf von Tokens

- Ihre Widerrufsanforderung für ein Aktualisierungstoken muss dieselbe Client-ID beinhalten, die zum Abrufen des Tokens verwendet wurde.

- Der Benutzerpool ist JWTs eigenständig und verfügt über eine Signatur und eine Ablaufzeit, die bei der Erstellung des Tokens zugewiesen wurden. 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.
- Wenn Sie einen neuen Benutzerpool-Client erstellen, ist der Tokenwiderruf standardmäßig aktiviert.
- Sie können Aktualisierungstoken nur in App-Clients widerrufen, bei denen der Token-Widerruf aktiviert ist.
- 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.
- Wenn Sie den Token-Widerruf in einem App-Client deaktivieren, in dem er zuvor aktiviert war, werden widerrufene Token nicht wieder aktiv.
- Wenn Sie ein Benutzerkonto deaktivieren (wodurch Aktualisierungs- und Zugriffstoken gesperrt werden), werden die widerrufenen Token nicht aktiv, wenn Sie das Benutzerkonto erneut aktivieren.
- Wenn Sie einen neuen Benutzerpool-Client mithilfe der AWS Management Console, der oder der AWS API erstellen AWS CLI, ist der Token-Widerruf 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 API 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`.

Um einen App-Client mit aktiviertem Token-Widerruf mit der Amazon Cognito Cognito-API oder einem AWS SDK zu erstellen oder zu ändern, fügen Sie den folgenden Parameter in Ihre CreateUserPoolClientoder Ihre UpdateUserPoolClientAPI-Anfrage ein.

```
"EnableTokenRevocation": true
```

Um den Token-Widerruf in der Amazon Cognito Cognito-Konsole zu konfigurieren, wählen Sie einen App-Client aus dem App-Client-Menü in Ihrem Benutzerpool aus. Klicken Sie in den App-Client-Informationen auf die Schaltfläche Bearbeiten und aktivieren oder deaktivieren Sie den Token-Widerruf unter Erweiterte Konfiguration.

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.

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.

Überprüfung von JSON-Web-Tokens

JSON-Webtoken (JWTs) können einfach dekodiert, gelesen und geändert werden. Ein modifiziertes Zugriffstoken birgt das Risiko einer Rechteeskalation. Ein modifiziertes ID-Token birgt das Risiko

eines Identitätswechsels. Ihre Anwendung vertraut Ihrem Benutzerpool als Token-Aussteller, aber was ist, wenn ein Benutzer das Token während der Übertragung abfängt? Sie müssen sicherstellen, dass Ihre Anwendung dasselbe Token erhält, das Amazon Cognito ausgestellt hat.

Amazon Cognito gibt Token aus, die einige der Integritäts- und Vertraulichkeitsfunktionen der OpenID Connect (OIDC) -Spezifikation nutzen. Benutzerpool-Token geben anhand von Objekten wie der Ablaufzeit, dem Aussteller und der digitalen Signatur die Gültigkeit an. Die Signatur, das dritte und letzte Segment des durch `.` -getrennten JWT, ist die Schlüsselkomponente der Token-Validierung. Ein böswilliger Benutzer kann ein Token ändern, aber wenn Ihre Anwendung den öffentlichen Schlüssel abruft und die Signatur vergleicht, stimmt sie nicht überein. Jede Anwendung, die die OIDC-Authentifizierung verarbeitet JWTs , muss diesen Überprüfungsvorgang bei jeder Anmeldung durchführen.

Auf dieser Seite geben wir einige allgemeine und spezifische Empfehlungen zur Überprüfung von. JWTs Die Anwendungsentwicklung umfasst eine Vielzahl von Programmiersprachen und Plattformen. Da Amazon Cognito OIDC so nah an der öffentlichen Spezifikation implementiert, kann jede seriöse JWT-Bibliothek in der Entwicklungsumgebung Ihrer Wahl Ihre Überprüfungsanforderungen erfüllen.

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 übernehmen möglicherweise bereits die Aufgaben in diesem Abschnitt. AWS SDKs Stellen Sie Tools für die Handhabung und Verwaltung von Amazon Cognito Cognito-Benutzerpool-Tokens in Ihrer App bereit. 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;
- Diese Zugriffstoken-Ansprüche enthalten die richtigen OAuth 2.0-Bereiche.
- 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 unter aws-jwt-verifyweiter. GitHub

Tokens verstehen und überprüfen

Bevor Sie die Tokeninspektion in Ihre App integrieren, sollten Sie sich überlegen, wie Amazon Cognito zusammenbaut JWTs. 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 Verfahren zur manuellen Überprüfung von Amazon Cognito JWTs beschrieben, während Sie Ihre App vorbereiten.

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`. Dies `kid` ist ein gekürzter Verweis auf einen privaten 2048-Bit-RSA-Signaturschlüssel, der in Ihrem Benutzerpool gespeichert ist.

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 dekodierbar (base64url) wie der Header und die Payload. Es handelt sich RSA256 um einen Bezeichner, der aus einem Signaturschlüssel und Parametern abgeleitet wird und die Sie an Ihrer JWKS-URI beobachten können.

Der Header und die Nutzlast sind Base64URL-kodierte JSON-Dateien. Sie können sie anhand der Zeichen `eyJ` am Anfang erkennen, die zum Startzeichen `{` dekodiert werden. Wenn Ihr Benutzer Ihrer App ein base64url-codiertes JWT präsentiert und es nicht im Format ist`[JSON Header].[JSON Payload].[Signature]`, ist es kein gültiges Amazon Cognito Cognito-Token und Sie können es verwerfen.

Die folgende Beispielanwendung verifiziert Benutzerpool-Token mit. `aws-jwt-verify`

```
// cognito-verify.js
// Usage example: node cognito-verify.js eyJra789ghiEXAMPLE

const { CognitoJwtVerifier } = require('aws-jwt-verify');

// Replace with your Amazon Cognito user pool ID
const userPoolId = 'us-west-2_EXAMPLE';

async function verifyJWT(token) {
```

```
  try {
    const verifier = CognitoJwtVerifier.create({
      userPoolId,
      tokenUse: 'access', // or 'id' for ID tokens
      clientId: '1example23456789', // Optional, only if you need to verify the token
 audience
    });

    const payload = await verifier.verify(token);
    console.log('Decoded JWT:', payload);
  } catch (err) {
    console.error('Error verifying JWT:', err);
  }
}

// Example usage
if (process.argv.length < 3) {
  console.error('Please provide a JWT token as an argument.');
  process.exit(1);
}

const MyToken = process.argv[2];
verifyJWT(MyToken);
```

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 den Benutzerpool zu dekodieren. JWTs 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 RS256 kryptografischen Algorithmus, bei dem es sich um eine RSA-Signatur mit SHA-256 handelt. 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 Base64URL-codierter Wert dargestellt. UInt

RSA-Modulo (**`n`**)

Der `n`-Parameter enthält den Modulo-Wert für den öffentlichen RSA-Schlüssel. Er wird als Base64Url -codierter Wert dargestellt. UInt

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.

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.

Verwaltung des Ablaufs und der Zwischenspeicherung von Benutzerpool-Tokens

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.
- Beantragen Sie auf Ihren verwalteten Anmeldeseiten eine implizite Gewährung.
- 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. Um die Leistung und Verfügbarkeit Ihrer App sicherzustellen, verwenden Sie Amazon Cognito Cognito-Token für etwa 75% der Token-Lebensdauer 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. Zu den Amazon Cognito OAuth2.0-Endpunkten gehört der Token-Endpunkt, der Kundenanmeldedaten und verwaltete Anfragen von Anmeldeautorisierungscodes bearbeitet. 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 API-Gateway-Token-Caching kann Ihre App als Reaktion auf Ereignisse skaliert werden, die die standardmäßige Quote für die Anforderungsrate von Amazon Cognito OAuth Cognito-Endpunkten 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 bei der Übertragung und optional im

Ruhezustand verschlüsselt. Ein API-Gateway-Cache eignet sich ideal für die Gewährung von OAuth 2.0-Client-Anmeldeinformationen. Dabei handelt es sich um einen häufig großvolumigen Grant-Typ, der Zugriffstoken für Autorisierungs machine-to-machine - und Microservice-Sitzungen generiert. 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.

Bei dieser Lösung definieren Sie einen Cache in Ihrer API, um für jede Kombination aus OAuth Bereichen und App-Client, die Sie in Ihrer App anfordern möchten, ein separates Zugriffstoken zu speichern. 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 aus den OAuth Bereichen, die Sie im `scope` Parameter im Anfragetext anfordern, und dem `Authorization` Header in der Anfrage. 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 ElastiCache (Redis OSS) implementieren. 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.

      > **Note**
      > Diese Methode der Anforderungsvalidierung ist für die `client_secret_basic` Autorisierung bei Token-Anfragen vorgesehen, bei denen das Client-Geheimnis im Anforderungsheader kodiert ist. `Authorization` Um den Hauptteil der JSON-Anforderung bei der `client_secret_post` Autorisierung zu validieren, erstellen Sie stattdessen ein Datenmodell, das erfordert, dass client_secret vorhanden ist. In diesem Modell sollte Ihr Request-Validator den Text, die Parameter der Abfragezeichenfolge und die Header validieren.

   b. Konfigurieren Sie die Methode Request Validator so, dass die Parameter und Header der Abfragezeichenfolge validiert werden. Weitere Informationen zur Anforderungsvalidierung finden Sie unter Anforderungsvalidierung im Amazon API Gateway Developer Guide.
   c. Legen Sie den `scope`-Parameter und den `Authorization`-Header als Caching-Schlüssel fest.
      i. Fügen Sie den URL-Abfragezeichenfolgenparametern eine Abfragezeichenfolge hinzu. Geben Sie eine Abfragezeichenfolge „Name of“ ein `scope` und wählen Sie „Erforderlich“ und „Zwischenspeichern“ aus.
      ii. Fügen Sie den Headern der HTTP-Anfrage einen Header hinzu. Geben Sie den Namen des Anforderungsheaders ein `Authorization` und wählen Sie Erforderlich und Zwischenspeichern aus.
4. Konfigurieren Sie das Caching unter Stages (Phasen).

a. Wählen Sie die Phase aus, die Sie ändern möchten, und wählen Sie in den Stufendetails die Option Bearbeiten aus.

b. Aktivieren Sie unter Zusätzliche Einstellungen, Cache-Einstellungen die Option API-Cache bereitstellen.

c. Wählen Sie einen Wert für Cache capacity (Cache-Kapazität) aus. Eine höhere Cachekapazität verbessert die Leistung, ist jedoch mit zusätzlichen Kosten verbunden.

d. Deaktivieren Sie das Kontrollkästchen Autorisierung erforderlich. Wählen Sie Weiter aus.

e. API Gateway wendet Cache-Richtlinien nur auf GET-Methoden aus der Stufenebene an. Sie müssen eine Überschreibung der Cache-Richtlinie auf Ihre POST-Methode anwenden.

   Erweitern Sie die Phase, die Sie konfiguriert haben, und wählen Sie die `POST` Methode aus. Um Cache-Einstellungen für die Methode zu erstellen, wählen Sie Create Override aus.

f. Aktivieren Sie die Option Methoden-Cache aktivieren.

g. Geben Sie einen Cache time-to-live (TTL) von 3600 Sekunden ein. Wählen Sie Speichern.

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 erfolgreicher Anmeldung

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 JSON-Webtoken (JWTs) für den Benutzerpool verstehen.

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 Daten mit Zugriffstoken befinden.
- Autorisieren Sie den Zugriff auf Anwendungsressourcen, die auf dem Client oder Server gespeichert sind, mit Amazon Verified Permissions.

Weitere Informationen erhalten Sie unter Ein Beispiel für eine Authentifizierungssitzung und JSON-Webtoken (JWTs) für den Benutzerpool verstehen.

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 von Ihnen erstellte Anwendungen. 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 Backend-Service übergeben. API Gateway verfügt über 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 2.0-Bereichen in einem Zugriffstoken erhält. OAuth

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 ist gültig, nicht abgelaufen und 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 – Authentifzierungsablauf.

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-Pools – Authentifzierungsablauf.

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 ***<region>*** aus der Benutzerpool-ID ableiten. Wenn die Benutzerpool-ID beispielsweise lautet`us-east-1_EXAMPLE1`, dann *<region>* ist `us-east-1` die. Wenn die Benutzerpool-ID lautet`us-west-2_EXAMPLE2`, dann *<region>* ist die`us-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)
```

Bereiche, M2M und APIs mit Ressourcenservern

Nachdem Sie eine Domain für Ihren Benutzerpool konfiguriert haben, stellt Amazon Cognito automatisch einen OAuth 2.0-Autorisierungsserver und eine gehostete Weboberfläche mit Anmelde- und Anmeldeseiten bereit, die Ihre App Ihren Benutzern präsentieren kann. Weitere Informationen finden Sie unter Vom Benutzerpool verwaltete Anmeldung. 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 `scope` Anspruch auf Zugriffstoken enthalten. Weitere Informationen zu den Ansprüchen in Amazon-Cognito-Zugriffstoken finden Sie unter Das Zugriffstoken verstehen.

Mit Amazon Cognito können die Bereiche in Zugriffstoken den Zugriff auf externe Attribute APIs oder Benutzerattribute autorisieren. Sie können Zugriffstoken für lokale Benutzer, Verbundbenutzer oder Maschinenidentitäten ausgeben.

Themen

- API-Autorisierung
- Machine-to-machine (M2M) -Autorisierung
- Grundlegendes zu Bereichen
- Grundlegendes zu Ressourcenservern

API-Autorisierung

Im Folgenden finden Sie einige Möglichkeiten, Anfragen APIs mit Amazon Cognito Cognito-Token zu autorisieren:

Zugriffstoken

Wenn Sie einen Amazon Cognito Cognito-Autorisierer zu einer REST-API-Methodenanforderungskonfiguration hinzufügen, fügen Sie der Autorisierungskonfiguration Autorisierungsbereiche hinzu. Bei dieser Konfiguration akzeptiert Ihre API Zugriffstoken im `Authorization` Header und überprüft sie auf akzeptierte Bereiche.

ID-Token

Wenn Sie in Ihrer REST-API ein gültiges ID-Token an einen Amazon Cognito Cognito-Authorizer übergeben, akzeptiert API Gateway die Anfrage und leitet den Inhalt des ID-Tokens an das API-Backend weiter.

Amazon Verified Permissions

Unter Verifizierte Berechtigungen haben Sie die Möglichkeit, einen API-verknüpften Richtlinienspeicher zu erstellen. Verified Permissions erstellt und weist einen Lambda-Autorisierer zu, der ID- oder Zugriffstoken aus Ihrem Anforderungsheader verarbeitet. `Authorization` Dieser Lambda-Autorisierer leitet Ihr Token an Ihren Richtlinienspeicher weiter, wo Verified Permissions es mit Richtlinien vergleicht und eine Entscheidung über das Zulassen oder Verweigern an den Autorisierer zurückgibt.

Weitere -Quellen

- Steuern und Verwalten des Zugriffs auf eine REST-API in API Gateway
- Autorisierung mit Amazon Verified Permissions

Machine-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-Computern herstellen. APIs 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. Anwendungsspezifische Einstellungen mit App-Clients

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 Machine-to-Machine-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 für die Benutzerauthentifizierung Kosten pro aktivem Benutzer anfallen, spiegelt die M2M-Abrechnung die aktiven App-Clients mit Kundenanmeldedaten 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 Verwaltung des Ablaufs und der Zwischenspeicherung von Benutzerpool-Tokens Sie unter.

Informationen zur Optimierung von Amazon Cognito Cognito-Vorgängen, die Ihre AWS Rechnung mit zusätzlichen Kosten belasten, finden Sie unterVerwalten von Kosten.

Client-Metadaten für machine-to-machine (M2M-) Kundenanmeldedaten

Sie können Client-Metadaten in M2M-Anfragen übergeben. Client-Metadaten sind zusätzliche Informationen aus einer Benutzer- oder Anwendungsumgebung, die zu den Ergebnissen einer Lambda-Auslöser für die Vorab-Generierung von Token beitragen können. Bei Authentifizierungsvorgängen mit einem Benutzerprinzipal können Sie Client-Metadaten im Hauptteil von AdminRespondToAuthChallengeund RespondToAuthChallengeAPI-Anfragen an den Trigger vor der Token-Generierung übergeben. Da Anwendungen den Ablauf für die Generierung von Zugriffstoken für M2M mit direkten Anfragen an die durchführenToken-Endpunkt, haben sie ein anderes Modell. Übergeben Sie im POST-Hauptteil der Token-Anfragen für Client-Anmeldeinformationen einen `aws_client_metadata` Parameter mit dem Client-Metadatenobjekt URL-encoded () `x-www-form-urlencoded` an string. Eine Beispielanforderung finden Sie unter Kundenanmeldedaten mit Basisautorisierung. Im Folgenden finden Sie einen Beispielparameter, der die Schlüssel-Wert-Paare übergibt. `{"environment": "dev", "language": "en-US"}`

```
aws_client_metadata=%7B%22environment%22%3A%20%22dev%22,%20%22language%22%3A%20%22en-US
%22%7D
```

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. Amazon Cognito Cognito-Benutzerpools stellen Zugriffstoken mit dem reservierten API-Bereich, benutzerdefinierten Bereichen und OpenID Connect (OIDC) - Bereichen für Benutzerpools aus.

Der reservierte API-Bereich der Benutzerpools

Der `aws.cognito.signin.user.admin` Bereich autorisiert Self-Service-Operationen für den aktuellen Benutzer in der Amazon Cognito Cognito-Benutzerpools-API. Er autorisiert den Inhaber eines Zugriffstokens, alle Informationen über den Träger abzufragen und zu aktualisieren, z. B. mit den Operationen und der API. GetUserUpdateUserAttributes 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 externe APIs Objekte, 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.

OpenID Connect (OIDC) -Bereiche

Wenn Sie Benutzer mit Ihrem Benutzerpool-Autorisierungsserver authentifizieren, auch mit verwalteter Anmeldung, müssen Sie Bereiche anfordern. Sie können lokale Benutzerpool-Benutzer und externe Verbundbenutzer auf Ihrem Amazon-Cognito-Autorisierungsserver authentifizieren. OIDC-Bereiche autorisieren Ihre App, Benutzerinformationen aus Ihrem Benutzerpool zu lesen. UserInfo-Endpunkt 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 OIDC-Bereichen auszugeben.

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. Bei Anfragen `openid` und anderen OIDC-Bereichen wie `profile emailphone`, und ist der Inhalt des ID-Tokens und der UserInfo-Antwort auf die 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 Cognito-Zugriffstoken kann den Zugriff auf APIs diesen Support OAuth 2.0 autorisieren. Amazon API Gateway REST bietet integrierte Unterstützung für APIs die Autorisierung mit Amazon Cognito 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 beispielsweise über einen Ressourcenserver für Fotos verfügen, könnte dieser zwei Bereiche definieren: `photos.read` für den Lesezugriff auf die Fotos und `photos.write` für write/delete den Zugriff. 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 Überprüfung von 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 in Form von OAuth 2.0-Autorisierungscode-Erteilungen, impliziten Zuweisungen und Zuweisungen von Kundenanmeldedaten bei der an. Token-Endpunkt 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. Bei der verwalteten Anmeldung können Sie auch benutzerdefinierte Bereiche in einem Zugriffstoken aus einer impliziten Gewährung 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. Beispiel, `sunproximity.read`.
- Beschreibung: Eine Kurzbeschreibung des Bereichs. Beispiel, `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 die Bereiche für die Zugriffstoken Ihrer Benutzer während der Authentifizierungsabläufe mit dem OAuth 2.0-Autorisierungsserver, der die verwaltete Anmeldung beinhaltet, festlegen. 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 das Domain-Menü unter Branding und suchen Sie nach Ressourcenservern.
4. Wählen Sie Create a resource server (Ressourcenserver erstellen) aus.
5. Geben Sie einen Name für den Ressourcenserver ein. Beispiel, `Photo Server`.
6. Geben Sie eine ID für den Ressourcenserver ein. 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 Erstellen aus.

Ihre benutzerdefinierten Bereiche können im Menü Domäne unter Ressourcenserver in der Spalte Benutzerdefinierte Bereiche überprüft werden. Benutzerdefinierte Bereiche können für App-Clients im Menü App-Clients unter Anwendungen aktiviert werden. Wählen Sie einen App-Client aus, suchen Sie nach Anmeldeseiten und wählen Sie 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

Benutzerpoolfunktionen konfigurieren

In den vorherigen Kapiteln haben Sie wahrscheinlich einige Funktionen mithilfe der Amazon Cognito Cognito-Konsole konfiguriert. Die Seiten in diesem Abschnitt bieten einen tieferen Einblick in die detaillierten Konfigurationsanforderungen einiger Kernfunktionen von Benutzerpools. Es gibt wichtige Referenzinformationen zu Ihren Optionen in Bezug auf App-Clients, E-Mail- und SMS-Konfiguration, Speicherung von Benutzergeräten und mehr.

Themen

- Aktualisierung der Benutzerpool- und App-Client-Konfiguration
- Anwendungsspezifische Einstellungen mit App-Clients
- Verwendung von Benutzergeräten in Ihrem Benutzerpool
- Verwendung von Amazon Pinpoint für Benutzerpool-Analysen
- E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools
- Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools

Aktualisierung der Benutzerpool- und App-Client-Konfiguration

Wenn Sie eine Einstellung in einem Benutzerpool oder App-Client ändern möchten, können Sie das Update mit wenigen Klicks in der Amazon Cognito Cognito-Konsole anwenden. Sie navigieren durch die funktionsbasierten Registerkarten in Ihren Benutzerpool-Einstellungen und Aktualisierungsfeldern, wie in anderen Bereichen dieses Handbuchs beschrieben.

Viele Organisationen verwalten ihre Ressourcen programmgesteuert in Anwendungen AWS CloudFormation, die auf dem CDK AWS SDKs oder CDK basieren, und anderer Automatisierungssoftware. Wenn dies Ihr Ressourcenverwaltungsmodell ist, müssen Sie besonders vorsichtig sein, wenn Sie Änderungen an Ihren Ressourcen vornehmen.

Die API-Operationen UpdateUserPoolund die UpdateUserPoolClientAktualisierung eines vorhandenen Benutzerpools oder App-Clients. Für jedes Attribut gibt es eine Warnung in der API-Referenz: Wenn Sie keinen Wert für ein Attribut angeben, setzt Amazon Cognito es auf den

Standardwert. Wenn Sie eine Aktualisierungsanfrage mit nur einem Parameter einreichen, setzt Amazon Cognito diesen Parameter auf den Wert Ihrer Wahl und setzt alle anderen auf einen Standardwert. Dadurch können Konfigurationen wie Ihr Attributschema, Ihre Lambda-Trigger und Ihre E-Mail- und SMS-Nachrichtenkonfiguration zurückgesetzt werden.

Darüber hinaus sind einige Einstellungen gesperrt, nachdem Sie Ihren Benutzerpool oder App-Client erstellt haben. Sie können sie nur ändern, wenn Sie eine neue Ressource erstellen.

Themen

- Einstellungen, die Sie nicht ändern können
- SMS-Konfiguration
- Aktualisierung eines Benutzerpools mit einem AWS SDK oder einer REST-API AWS CDK

Einstellungen, die Sie nicht ändern können

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.

> **Note**
>
> Bisher konnten Sie den Namen eines Benutzerpools nicht ändern. Das hat sich geändert. Sie können Ihren Benutzerpools jetzt neue benutzerfreundliche Namen zuweisen.

Benutzerpool-ID

Name des API-Parameters: Id/ UserPoolId

Die Benutzerpool-ID wird `us-east-1_EXAMPLE` beispielsweise automatisch von Amazon Cognito generiert und kann nicht geändert werden.

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 Anwendungsspezifische Einstellungen mit App-Clients. 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 Mit Benutzerattributen arbeiten. Wenn Sie die erforderlichen Attribute ändern möchten, erstellen Sie einen neuen Benutzerpool.

Benutzerdefinierte Attribute (Löschung)

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 Mit Benutzerattributen arbeiten. 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 den vollständigen Parametersatz 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 AWS SDKssind 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
```

Anwendungsspezifische Einstellungen mit App-Clients

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 alle App-Clients IDs oder Secrets schützen, 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.

Sie können Einstellungen für die folgenden Benutzerpoolfunktionen auf App-Client-Ebene anwenden:

1. Analysen
2. Verwaltete Anmeldung IdPs, Arten von Zuschüssen URLs, Rückrufe und Anpassungen
3. Ressourcenserver und benutzerdefinierte Bereiche
4. Schutz vor Bedrohungen
5. Lese- und Schreibberechtigungen für Attribute
6. Ablauf und Widerruf von Tokens
7. Authentifizierungsabläufe

Arten von App-Clients

Wenn Sie einen App-Client in Amazon Cognito erstellen, können Sie Optionen auf der Grundlage der OAuth Standardclienttypen öffentlicher Client und vertraulicher Client vorab ausfüllen. 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.

> **Note**
>
> Die Amazon Cognito Cognito-Konsole erstellt App-Clients mit Client-Geheimnissen, wenn Sie die Optionen Traditionelle Webanwendung und achine-to-machineM-Anwendung als Anwendungstyp auswählen. Wählen Sie eine dieser Optionen, um einen geheimen Clientschlüssel zu generieren, oder erstellen Sie den Client programmgesteuert mit CreateUserPoolClientund stellen Sie ihn auf ein. `GenerateSecret true`

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 Typen 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 2.0 definierten Standards für Zugriffs- und Aktualisierungstoken hinzugefügt. OAuth 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 Grundlegendes zum Identitätstoken (ID).

Zugriffstoken

Eine überprüfbare Erklärung zu den Zugriffsrechten Ihres Benutzers. Das Zugriffstoken enthält Bereiche, eine Funktion von OIDC und 2.0. OAuth 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 Das Zugriffstoken verstehen.

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 Tokens aktualisieren.

Sie können den Ablauf dieser Token für jeden App-Client im App-Client-Menü Ihres Benutzerpools in der Amazon Cognito Cognito-Konsole festlegen.

App-Client-Bedingungen

Die folgenden Begriffe sind verfügbare Eigenschaften von App-Clients in der Amazon-Cognito-Konsole.

Rückruf erlaubt 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.

Siehe OAuth 2.0 — Endpunkt der Umleitung.

Amazon Cognito fordert `HTTPS` statt `HTTP`, außer nur für Testzwecke für `http://localhost`.

App-Callbacks URLs wie diese `myapp://example` werden ebenfalls unterstützt.

Abmelden erlaubt 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. Ihr App-Client kann Werte in unveränderliche Attribute schreiben, wenn Sie einen Benutzer erstellen oder registrieren. 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, E-Mail- und SMS-Nachricht, Passkey-Authentifikatoren OTPs, benutzerdefinierte Authentifizierung mit Lambda-Triggern und Token-Aktualisierung unterstützen. Verwenden Sie als bewährte Sicherheitspraxis die SRP-Authentifizierung für die Authentifizierung von Benutzernamen und Passwörtern in maßgeschneiderten Anwendungen.

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 Bereiche, M2M und APIs 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 Rückrufe 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 ein Callback URLs 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 zum IdP generieren und ihn auf Ihrer verwalteten Anmeldeseite anzeigen. Sie können mehrere zuweisen IdPs, müssen jedoch mindestens einen zuweisen. Weitere Informationen zur Verwendung von Extern IdPs finden Sie unterBenutzerpool-Anmeldung mit externen Identitätsanbietern.

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.

OAuth Arten von Zuschüssen

Ein OAuth Grant ist eine Authentifizierungsmethode, mit der Benutzerpool-Token abgerufen werden. Amazon Cognito unterstützt die folgenden Arten von Erteilungen. Um diese OAuth Zuschüsse 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 Ablauf erfolgt, wie die implizite Gewährung, in den Browsern Ihrer Benutzer. 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 Ablauf für die Gewährung von Autorisierungscodes aktivieren und Proof Key for Code Exchange (PKCE) implementieren, um den Token-Austausch 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 Erteilung von Kundenanmeldedaten ist für (M2M) -Kommunikation vorgesehen. machine-to-machine 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 Bereiche, M2M und APIs 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. Bei beiden Optionen werden Sie aufgefordert, einen App-Client mit anwendungsspezifischen Einstellungen zu konfigurieren.
4. Wählen Sie einen Anwendungstyp, der Ihrer Anwendungsarchitektur entspricht.
5. Benennen Sie Ihre Anwendung mit einer benutzerfreundlichen Kennung.
6. Geben Sie eine Rückgabe-URL ein.
7. Wählen Sie Create app client. Sie können erweiterte Optionen ändern, nachdem Sie Ihren App-Client erstellt haben.
8. Amazon Cognito führt Sie zu den App-Client-Details zurück. Um auf den Beispielcode für Ihre Anwendung zuzugreifen, wählen Sie auf der Registerkarte Kurzanleitung zur Einrichtung eine Plattform aus.

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 Logout 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 Cognito-Benutzerpool-API anmelden, können Sie die Aktivitätsprotokolle Ihrer Benutzer aus dem Bedrohungsschutz jedem ihrer Geräte zuordnen und Ihren Benutzern optional 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 Ihre Anwendung mit einem Authentifizierungsablauf für vertrauenswürdige Geräte konfigurieren. In diesem Ablauf kann Ihre Anwendung Benutzern die Möglichkeit bieten, sich bis zu einem späteren Zeitpunkt ohne MFA anzumelden, je nach den Sicherheitsanforderungen Ihrer App oder den Präferenzen Ihrer Benutzer. Am Ende dieses Zeitraums muss Ihre Anwendung den Gerätestatus auf Nicht gespeichert ändern und der Benutzer muss sich mit MFA anmelden, bis er bestätigt, dass er sich ein Gerät merken möchte. Ihre Anwendung könnte Ihre Benutzer beispielsweise dazu auffordern, einem Gerät 30, 60 oder 90 Tage lang zu vertrauen. Sie können dieses Datum in einem benutzerdefinierten Attribut speichern und an diesem Datum den gespeicherten Status ihres Geräts ändern. Sie müssen Ihren Benutzer dann erneut auffordern, einen MFA-Code einzureichen und das Gerät so einrichten, dass es nach erfolgreicher Authentifizierung wieder gespeichert wird.

1. 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 Ihren Benutzerpool im Anmeldemenü Ihres Benutzerpools unter Geräteverfolgung so, dass er sich Geräte merkt. 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 Einstellungen für gespeicherte Geräte auch in einer CreateUserPoolodder UpdateUserPoolAPI-Anfrage für Ihren Benutzerpool konfigurieren. Weitere Informationen finden Sie in der DeviceConfigurationUnterkunft.

Die Benutzerpool-API von Amazon Cognito bietet zusätzliche Funktionen für gespeicherte Geräte.

1. ListDevicesund AdminListDevicesgibt eine Liste der Geräteschlüssel und ihrer Metadaten für einen Benutzer zurück.
2. GetDeviceund AdminGetDevicegibt den Geräteschlüssel und die Metadaten für ein einzelnes Gerät zurück.
3. UpdateDeviceStatusund AdminUpdateDeviceStatuslegen Sie das Gerät eines Benutzers als gespeichert oder nicht gespeichert fest.
4. ForgetDeviceund das bestätigte Gerät eines Benutzers aus seinem Profil AdminForgetDeviceentfernen.

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 Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten.

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 ConfirmDeviceSie eine 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 in einer Anfrage mit der Auswahl Ihres Benutzers. UpdateDeviceStatus

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 den Bedrohungsschutz für Ihren App-Client aktivieren und in Ihrer Anfrage einen Gerätefingerabdruck kodieren, ordnet Amazon Cognito Benutzerereignisse dem bestätigten Gerät zu.

So rufen Sie einen neuen Geräteschlüssel ab

1. Starten Sie die Anmeldesitzung Ihres Benutzers mit einer InitiateAuthAPI-Anfrage.
2. Beantworten Sie alle Authentifizierungsherausforderungen so lange, RespondToAuthChallengebis 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 diesen SRP-Bibliotheken verfügbar.
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 ConfirmDeviceAPI-Anfrage an. 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 sie bestätigen, dass sie sich das Gerät merken möchten, generieren Sie eine UpdateDeviceStatusAPI-Anfrage mit dem Zugriffstoken, dem Geräteschlüssel und Ihrem Benutzer. `"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 Flow durchzuführen, der Lambda-Trigger für benutzerdefinierte Authentifizierungsherausforderungen verwendet, übergeben Sie einen `DEVICE_KEY` Parameter in Ihrer InitiateAuthAPI-Anfrage. 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 InitiateAuthAPI-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 RespondToAuthChallengeAPI-Anfrage ein ChallengeName von DEVICE_SRP_AUTH und Parameter für USERNAMEDEVICE_KEY, undSRP_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. Schließe zusätzliche Herausforderungen ab, bis du die des Benutzers erhältst JWTs.

Der folgende Pseudocode zeigt, wie Sie Werte für Ihre Antwort auf die DEVICE_PASSWORD_VERIFIER-Aufgabe berechnen. Generieren Sie für die SRP-Authentifizierung mit einem Gerät ein neues SRP-Geheimnis für Ihren Benutzer: ein neues Passwort mit hoher EntropieDeviceSecret, ein Salt und den zugehörigen Passwort-Verifier. Diese Werte unterscheiden sich von Passwort, Salt und Verifier, die für die SRP-Authentifizierung des Benutzers verwendet werden. Sie werden nur für die Geräteauthentifizierung verwendet und nur auf dem Gerät gespeichert. Funktionen zum Generieren der SRP-Geheimnisse für Benutzergeräte sind in verschiedenen SRP-Bibliotheken verfügbar. SDKs

```
PASSWORD_CLAIM_SECRET_BLOCK = SECRET_BLOCK
TIMESTAMP = "Tue May 7 00:09:40 UTC 2025"
k = SHA256(N || g) as a non-negative integer in big-endian
u = SHA256(SRP_A || SRP_B) as a non-negative integer in big-endian
x = SHA256(salt || SHA256(DeviceGroupKey || DeviceKey || ":" || DeviceSecret)) as a
 non-negative integer in big-endian
S_USER = (SRP_B - k * g^x)^(a + u * x) % N
K_USER = HKDF_HMAC_SHA256(salt=u, ikm=S_USER, info="Caldera Derived Key", length=16
 bytes)
PASSWORD_CLAIM_SIGNATURE = Base64(HMAC_SHA256(key=K_USER, message=(DeviceGroupKey ||
 DeviceKey || PASSWORD_CLAIM_SECRET_BLOCK || TIMESTAMP)))
```

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. Bei der verwalteten Anmeldung können Sie den `aws.cognito.signin.user.admin` Geltungsbereich anfordern, aber die verwaltete Anmeldung fügt automatisch Geräteinformationen zu den Benutzerprotokollen mit erweiterter Sicherheit hinzu und bietet nicht an, sich Geräte zu merken.

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, reichen Sie den Zugriffs- und Geräteschlüssel Ihres Benutzers in einer GetDeviceAPI-Anfrage ein.
- Um die Geräteinformationen Ihres Benutzers in einer vertraulichen Client-App anzuzeigen, signieren Sie eine AdminGetDeviceAPI-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.

- Reichen Sie in einer öffentlichen Client-App das Zugriffstoken Ihres Benutzers in einer API-Anfrage ein. ListDevices

- Unterschreiben Sie in einer App für vertrauliche Clients eine AdminListDevicesAPI-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.

- Reichen Sie in einer öffentlichen Client-App den Geräteschlüssel und das Zugriffstoken Ihres Benutzers in einer API-Anfrage ein. ForgetDevice
- Reichen Sie in einer vertraulichen Client-App den Geräteschlüssel und das Zugriffstoken Ihres Benutzers in einer API-Anfrage ein. AdminForgetDevice

Verwendung von Amazon Pinpoint für Benutzerpool-Analysen

> **Note**
>
> Hinweis zum Ende des Supports: Am 30. Oktober 2026 AWS wird der Support für Amazon Pinpoint eingestellt. Nach dem 30. Oktober 2026 können Sie nicht mehr auf die Amazon Pinpoint-Konsole oder die Amazon Pinpoint Pinpoint-Ressourcen (Endpunkte, Segmente, Kampagnen, Journeys und Analysen) zugreifen. Weitere Informationen finden Sie unter Ende des Supports von Amazon Pinpoint. Hinweis: APIs In Bezug auf SMS sind Sprach-, Handy-Push-, OTP- und Telefonnummernvalidierung von dieser Änderung nicht betroffen und werden von AWS End User Messaging unterstützt.

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 Analytics in Amazon Cognito Cognito-Benutzerpools können Sie Benutzerpool-Anmeldungen, Anmeldungen, fehlgeschlagene Authentifizierungen, täglich aktive Benutzer (DAUs) und monatlich aktive Benutzer (MAUs) in der Amazon Pinpoint Pinpoint-Konsole verfolgen. 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 in der Analytics-Konfiguration für Ihren App-Client im Menü App-Clients in der Amazon Cognito-Konsole Benutzerattributdaten mit Amazon Pinpoint teilen wählen, 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-Principal, der Ihre Analytics-Konfiguration hinzufügt, muss über Berechtigungen verfügen. CreateServiceLinkedRole Die mit dem Dienst verknüpfte 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 Zuordnungen 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 (Milan)
- Naher Osten (Bahrain)
- Asien-Pazifik (Osaka)
- Israel (Tel Aviv)
- Afrika (Kapstadt)
- Asien-Pazifik (Jakarta)
- Asien-Pazifik (Malaysia)

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

Region mit dem Amazon-Cognito-Benutzerpool	Region für das Amazon-Pinpoint-Projekt
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 außer denen in der obigen Tabelle kann Amazon Cognito nur ein Amazon Pinpoint Pinpoint-Projekt in derselben Region wie Ihr Benutzerpool verwenden. 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 das Menü App-Clients und wählen Sie den App-Client aus, den Sie aktualisieren möchten.
4. Wählen Sie auf der Registerkarte Analytics unter Pinpoint Analytics die Option Aktivieren aus.
5. Wählen Sie eine Pinpoint-Region aus.
6. 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.

7. 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.

8. Wählen Sie Änderungen speichern aus.

Angeben der Analyseeinstellungen (AWS CLI und der AWS API) von Amazon Pinpoint

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 mithilfe einer App mit einer benutzerdefinierten Benutzeroberfläche (UI) in ein AWS SDK integrieren können. Sie können keine Analyse-Metadaten für föderierte oder lokale Benutzer weitergeben, die sich über eine verwaltete Anmeldung 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 erscheinen meine Amazon Cognito Cognito-Benutzerpool-Analysen nicht auf meinem Amazon Pinpoint Pinpoint-Dashboard? im AWS Knowledge Center.

E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools

Bestimmte Ereignisse in Ihrer Anwendung können dazu führen, dass Amazon Cognito Ihren Benutzern E-Mails 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	Vorlage für eine Nachricht
Passwort vergessen	ForgotPassword, AdminResetUserPassword	E-Mail, SMS	Code	Ja	Bestätigungsnachricht
Einladung	AdminCreateUser	E-Mail, SMS	Code	Ja	Einladungsnachricht
Selbstregistrierung	SignUp, ResendConfirmationCode	E-Mail, SMS	Code, Link	Ja	Bestätigungsnachricht
Bestätigung der E-Mail-Adresse oder Telefonnummer	UpdateUserAttributes, AdminUpdateUserAttributes, GetUserAttributeVerificationCode	E-Mail, SMS	Code	Ja	Bestätigungsnachricht
Multi-Faktor-Authentifizierung (MFA)	AdminInitiateAuth, InitiateAuth	E-Mail[1], SMS, Authentifizierungs-App	Code	Ja[2]	MFA-Nachricht
Einmalige Passwortauthentifi	AdminInitiateAuth, InitiateAuth	E-Mail[1], SMS	Code	Ja	MFA-Nachricht [3]

Aktivität	API-Operation	Zustelloptionen	Formatierungsoptionen	Anpassbar	Vorlage für eine Nachricht
zierung (OTP)					

[1] Erfordert den Feature-Plan Essentials oder höher und die E-Mail-Konfiguration von Amazon SES.

[2] Für SMS- und E-Mail-Nachrichten.

[3] Sie können die MFA-Nachrichtenvorlage nur anpassen, wenn MFA in Ihrem Benutzerpool erforderlich oder optional ist. Wenn MFA inaktiv ist, sendet Amazon Cognito Einmalkennwörter mit der Standardvorlage.

Amazon SES berechnet Gebühren für E-Mail-Nachrichten. Weitere Informationen finden Sie unter Amazon SES – Preise.

Weitere Informationen zu E-Mail-MFA finden Sie unter SMS- und E-Mail-Nachricht MFA.

Amazon Cognito verhindert möglicherweise die Zustellung zusätzlicher E-Mail- oder SMS-Nachrichten an ein einzelnes Ziel in einem kurzen Zeitraum. Wenn Sie glauben, dass Ihr Benutzerpool betroffen ist, konfigurieren und überprüfen Sie die Protokolle auf Fehler bei der Nachrichtenzustellung und wenden Sie sich dann an Ihr Kundenbetreuungsteam.

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)
Kanada West (Calgary)	Alternative Region	Kanada (Zentral), USA West (Nordkalifornien) [1]
Mexiko (Zentral)	Alternative Region	USA Ost (Nord-Virginia), USA West (Oregon) [1]

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 (Hongkong)	Alternative Region	Asien-Pazifik (Singapur), Asien-Pazifik (Tokio) [1]
Asien-Pazifik (Seoul)	Abwärtskompatibel	Asien-Pazifik (Tokio), USA Ost (Nord-Virginia), USA West (Oregon), Europa (Irland)
Asien-Pazifik (Malaysia)	Alternative Region	Asien-Pazifik (Sydney), Asien-Pazifik (Singapur) [1]
Asien-Pazifik (Thailand)	Alternative Region	Asien-Pazifik (Singapur), Asien-Pazifik (Mumbai) [1]
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)

Region des Benutzerpools	Option Region	Von Amazon SES unterstützte Regionen
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)
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)

Region des Benutzerpools	Option Region	Von Amazon SES unterstützte Regionen
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 Standardadresse`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 es dem Amazon Cognito-Service Principal ermöglichen, E-Mail-Nachrichten zu senden. 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.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "arn:aws:ses:us-
east-1:111122223333:identity/support@example.com",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
east-1:111122223333:userpool/us-east-1_EXAMPLE"
                }
            }
        }
    ]
}
```

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 *me-central-1* Beispiel-Regionskennung nach Bedarf durch die erforderliche Regions-ID.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cognito-idp.me-central-1.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "arn:aws:ses:us-
east-1:111122223333:identity/support@example.com",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
east-1:111122223333:userpool/us-east-1_EXAMPLE"
                }
            }
        }
    ]
```

```
}
```

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 Domäne handelt, müssen Sie einen Wert für die FROM-E-Mail-Adresse angeben.

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 Anmeldeinformationen ein.
2. Wählen Sie User Pools (Benutzerpools) aus.
3. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus.
4. Wählen Sie das Menü Authentifizierungsmethoden, suchen Sie nach E-Mail-Konfiguration und wählen Sie 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. Amazon SNS wiederum übergibt SMS-Nachrichten an AWS Endbenutzer-Messaging SMS. Wenn Sie zum ersten Mal eine Textnachricht über Amazon Cognito senden, befinden Sie AWS Endbenutzer-Messaging SMS sich in einer Sandbox-Umgebung. In der Sandbox-Umgebung können Sie Ihre Anwendungen auf SMS-Textnachrichten testen. In der Sandbox können Sie nur das Senden von Nachrichten simulieren.

> **Note**
>
> Im November 2024 AWS wurde Amazon SNS SMS Messaging durch ersetzt. AWS Endbenutzer-Messaging SMS Derzeit bezieht sich die Amazon Cognito Cognito-Konsole auf Amazon SNS SNS-Ressourcen. Benutzerpools initiieren SMS-Nachrichten mit dem Amazon SNS Publish-Vorgang, bei dem es sich um eine Weiterleitung an handelt. AWS Endbenutzer-Messaging SMS Dementsprechend müssen Sie weiterhin Berechtigungen für `sns:Publish` konfigurieren, nicht. `sms-voice:SendTextMessage`

AWS Endbenutzer-Messaging SMS Gebühren für SMS-Textnachrichten. Weitere Informationen finden Sie unter AWS Endbenutzer-Messaging SMS Preise.

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	Vorlage für eine Nachricht
Passwort vergessen	ForgotPassword, AdminResetUserPassword	E-Mail, SMS	Code	Ja	Bestätigungsnachricht
Einladung	AdminCreateUser	E-Mail, SMS	Code	Ja	Einladungsnachricht
Selbstregistrierung	SignUp, ResendConfirmationCode	E-Mail, SMS	Code, Link	Ja	Bestätigungsnachricht
Bestätigung der E-Mail-Adresse oder Telefonnummer	UpdateUserAttributes, AdminUpdateUserAttributes, GetUserAttributeVerificationCode	E-Mail, SMS	Code	Ja	Bestätigungsnachricht
Multi-Faktor-Authentifizierung (MFA)	AdminInitiateAuth, InitiateAuth	E-Mail[1], SMS, Authentifizierungs-App	Code	Ja[2]	MFA-Nachricht

Aktivität	API-Operation	Zustelloptionen	Formatierungsoptionen	Anpassbar	Vorlage für eine Nachricht
Einmalige Passwortauthentifizierung (OTP)	AdminInitiateAuth, InitiateAuth	E-Mail[1], SMS	Code	Ja	MFA-Nachricht [3]

[1] Erfordert den Feature-Plan Essentials oder höher und die E-Mail-Konfiguration von Amazon SES.

[2] Für SMS- und E-Mail-Nachrichten.

[3] Sie können die MFA-Nachrichtenvorlage nur anpassen, wenn MFA in Ihrem Benutzerpool erforderlich oder optional ist. Wenn MFA inaktiv ist, sendet Amazon Cognito Einmalkennwörter mit der Standardvorlage.

AWS Endbenutzer-Messaging SMS Gebühren für SMS-Nachrichten. Weitere Informationen finden Sie unter AWS Endbenutzer-Messaging SMS Preise.

Weitere Informationen zu MFA finden Sie unter SMS- und E-Mail-Nachricht MFA.

Amazon Cognito verhindert möglicherweise die Zustellung zusätzlicher E-Mail- oder SMS-Nachrichten an ein einzelnes Ziel in einem kurzen Zeitraum. Wenn Sie glauben, dass Ihr Benutzerpool betroffen ist, konfigurieren und überprüfen Sie die Protokolle auf Fehler bei der Nachrichtenzustellung und wenden Sie sich dann an Ihr Kundenbetreuungsteam.

Bewährte Methoden

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 SMS-message-related Vorschriften in Ländern, in denen Ihre Benutzer möglicherweise leben, im Auge behalten und Ihre SMS-Nachrichtenkonfiguration auf dem neuesten Stand halten. Weitere Informationen finden Sie im AWS Endbenutzer-Messaging SMS Benutzerhandbuch unter Funktionen und Einschränkungen der jeweiligen Länder für SMS und MMS.

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.

Unter den folgenden Links finden Sie Informationen zur Sicherung Ihrer Architektur für die SMS-Nachrichtenzustellung.

- Reduzieren Sie mit Amazon Cognito Cognito-Benutzerpools das Risiko von Betrug bei Benutzeranmeldungen und SMS-Versand
- Schutz vor SMS-Pumping: Neue AWS Funktionen zur Bekämpfung von künstlich aufgeblähtem Traffic

Erstmaliges Einrichten von SMS-Nachrichten in Amazon-Cognito-Benutzerpools

Amazon Cognito verwendet Amazon SNS und indirekt AWS Endbenutzer-Messaging SMS, um SMS-Nachrichten aus Ihren Benutzerpools zu versenden. Sie können auch einen Benutzerdefinierter Lambda-Auslöser für SMS-Sender verwenden, um Ihre eigenen Ressourcen zum Senden von SMS-Nachrichten zu verwenden. Wenn Sie SMS-Textnachrichten zum ersten Mal in einer bestimmten Region einrichten AWS-Region, werden Ihre Textnachrichten AWS-Konto in der SMS-Sandbox für diese Region AWS Endbenutzer-Messaging SMS platziert. AWS Endbenutzer-Messaging SMS verwendet die Sandbox, um Betrug und Missbrauch zu verhindern und Compliance-Anforderungen zu erfüllen. Wenn Sie AWS-Konto sich in der Sandbox befinden, gelten einige AWS Endbenutzer-Messaging SMS Einschränkungen. Sie können beispielsweise Textnachrichten an maximal 10 verifizierte Zielnummern senden, wenn Sie eine Absenderidentität haben, oder Sie können das Senden von Nachrichten ohne Absenderidentität simulieren. Solange Sie in der Sandbox AWS-Konto bleiben, sollten Sie in der Produktion keine SMS-Nachrichten versenden. Wenn Sie sich in der Sandbox befinden, kann Amazon Cognito keine Nachrichten an die Telefonnummern Ihrer Benutzer senden.

Themen

- Bereiten Sie eine IAM-Rolle vor, mit der Amazon Cognito SMS-Nachrichten senden kann AWS Endbenutzer-Messaging SMS
- Wählen Sie AWS-Region für SMS-Nachrichten
- Eine Ursprungsidentität zum Senden von SMS-Nachrichten an US-Telefonnummern anfordern

- Bestätigen Sie, dass Sie sich in der SMS-Sandbox befinden.
- Verschieben Sie Ihr Konto aus der Sandbox
- Verwenden Sie Simulatornummern oder verifizierte Telefonnummern mit AWS Endbenutzer-Messaging SMS
- Die Benutzerpool-Einrichtung in Amazon Cognito abschließen

Bereiten Sie eine IAM-Rolle vor, mit der Amazon Cognito SMS-Nachrichten senden kann AWS Endbenutzer-Messaging SMS

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 Cognito-Konsole können Sie im Menü Authentifizierungsmethoden Ihres Benutzerpools unter SMS eine IAM-Rollenauswahl festlegen oder diese Auswahl während des Assistenten zur Erstellung 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 die folgenden Bedingungen erfüllt:

- Der Vorgang „Rolle übernehmen“ erfolgt im Namen des Benutzerpools in dieser Bedingung. `aws:SourceArn`
- Der Vorgang zur Rollenübernahme erfolgt im Namen eines Benutzerpools, der in der Bedingung AWS-Konto festgelegt ist. `aws:SourceAccount`
- Der Vorgang „Rolle übernehmen“ schließt die externe ID in die Bedingung ein. `sts:externalId`

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "cognito-idp.amazonaws.com"
            },
```

```
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "sts:ExternalId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
west-2:111122223333:userpool/us-west-2_EXAMPLE"
                }
            }
        }
    ]
}
```

Sie können einen genauen Benutzerpool-ARN oder ein Platzhalter-ARN im Wert der Bedingung `aws:SourceArn` angeben. Suchen Sie in AWS Management Console oder mit ARNs einer DescribeUserPoolAPI-Anfrage nach Ihren Benutzerpools.

Um SMS-Nachrichten für die Multi-Faktor-Authentifizierung zu senden, muss Ihre Vertrauensrichtlinie für IAM-Rollen eine `sts:ExternalId` Bedingung erfüllen. Der Wert dieser Bedingung muss der `ExternalId` Eigenschaft SmsConfigurationIhres Benutzerpools entsprechen. Wenn Sie während der Erstellung des Benutzerpools in der Amazon Cognito-Konsole eine IAM-Rolle erstellen, konfiguriert Amazon Cognito die externe ID für Sie in der Rolle und in den Benutzerpooleinstellungen. Dies ist nicht der Fall, wenn Sie eine vorhandene IAM-Rolle verwenden.

Sie müssen den `ExternalId` Benutzerpoolparameter in einer UpdateUserPoolAPI-Anforderung aktualisieren und die Vertrauensrichtlinie für die IAM-Rolle mit einer `sts:externalId` Bedingung mit demselben Wert aktualisieren. Informationen zur Verwendung der API zur Aktualisierung eines Benutzerpools unter Beibehaltung der ursprünglichen Konfiguration finden Sie unter. Aktualisierung der Benutzerpool- und App-Client-Konfiguration

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 SMS-Nachrichten

> **Note**
>
> SMS-Nachrichten in AWS werden jetzt in verwaltet AWS Endbenutzer-Messaging SMS.

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, außer im asiatisch-pazifischen Raum (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.

Wählen Sie eine Region für SMS-Ressourcen im Schritt Nachrichtenzustellung konfigurieren des Assistenten für neue Benutzerpools aus. Sie können auch im Menü Authentifizierungsmethoden eines vorhandenen Benutzerpools unter SMS die Option Bearbeiten 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 (via AWS Endbenutzer-Messaging SMS) 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 im Benutzerhandbuch unter Übergang von der AWS Endbenutzer-Messaging SMS MMS- und Voice-Sandbox zur Produktion.AWS Endbenutzer-Messaging SMS

Sie können SMS-Nachrichten für jede Amazon Cognito Cognito-Region in der folgenden Tabelle mit AWS Endbenutzer-Messaging SMS Ressourcen in der entsprechenden SMS-Nachrichtenregion senden.

Amazon-Cognito-Region	Region für SMS-Nachrichten
USA Ost (Ohio)	USA Ost (Ohio), USA Ost (Nord-Virginia)

Amazon-Cognito-Region	Region für SMS-Nachrichten
USA Ost (Nord-Virginia)	USA Ost (Nord-Virginia)
USA West (Nordkalifornien)	USA West (Nordkalifornien)
USA West (Oregon)	USA West (Oregon)
Kanada (Zentral)	Kanada (Zentral), USA Ost (Nord-Virginia)
Kanada West (Calgary)	Kanada West (Calgary)
Mexiko (Zentral)	Mexiko (Zentral)
Europa (Frankfurt)	Europa (Frankfurt), Europa (Irland)
Europa (London)	Europa (London), Europa (Irland)
Europa (Irland)	Europa (Irland)
Europa (Paris)	Europa (Paris)
Europa (Stockholm)	Europa (Stockholm)
Europa (Milan)	Europa (Milan)
Europa (Spain)	Europa (Spain)
Europa (Zürich)	Europa (Zürich)
Asien-Pazifik (Malaysia)	Asien-Pazifik (Singapur)
Asien-Pazifik (Thailand)	Asien-Pazifik (Mumbai)
Asien-Pazifik (Mumbai)	Asien-Pazifik (Mumbai), Asien-Pazifik (Singapur)
Asien-Pazifik (Hyderabad)	Asien-Pazifik (Hyderabad)
Asien-Pazifik (Hongkong)	Asien-Pazifik (Singapur)
Asia Pacific (Seoul)	Asien-Pazifik (Tokio)

Amazon-Cognito-Region	Region für SMS-Nachrichten
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)
Naher Osten (Bahrain)	Naher Osten (Bahrain)
Naher Osten (VAE)	Naher Osten (VAE)
Südamerika (São Paulo)	Südamerika (São Paulo)
Israel (Tel Aviv)	Israel (Tel Aviv)
Afrika (Kapstadt)	Afrika (Kapstadt)

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.

US-Mobilfunkanbieter benötigen eine Absenderidentität, um Nachrichten an US-Telefonnummern zu senden. Wenn Sie noch keine Ursprungsidentität besitzen, müssen Sie eine anfordern. Informationen dazu, wie Sie eine Absenderidentität erhalten, finden Sie im AWS Endbenutzer-Messaging SMS Benutzerhandbuch unter Telefonnummer anfordern.

Wenn Sie mehr als eine Absenderidentität in derselben Datei haben AWS-Region, AWS Endbenutzer-Messaging SMS wählt Sie einen Identitätstyp in der folgenden Prioritätsreihenfolge aus: Kurzcode, 10DLC, gebührenfreie Nummer. Sie können diese Prioritätsreihenfolge nicht ändern. Weitere Informationen finden Sie unter AWS Endbenutzer-Messaging SMS FAQs.

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 das Menü Authentifizierungsmethoden.
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:

   Konfigurieren Sie AWS-Service Abhängigkeiten, um die Einrichtung Ihrer SMS-Nachricht abzuschließen

   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 unter Zur Amazon SNS-Produktionsumgebung wechseln den Link Amazon SNS aus. 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 wie folgt an AWS-Region:

   ```
   This account is in the SMS sandbox in US East (N. Virginia).
   ```

Verschieben Sie Ihr Konto aus der Sandbox

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 in der eine Originationsidentität konfiguriert haben AWS-Region , die die AWS Endbenutzer-Messaging SMS Ressourcen enthält, die Amazon Cognito verwenden soll, können Sie US-Telefonnummern verifizieren, während Ihre in der AWS-Konto SMS-

Sandbox verbleiben. Wenn sich Ihre Umgebung in der Produktionsumgebung befindet, müssen Sie die Telefonnummern der Benutzer nicht verifizieren, bevor Sie SMS-Nachrichten an sie senden.

Sie können eine Anfrage zum Verlassen der Sandbox entweder von der AWS Endbenutzer-Messaging SMS Konsole oder der Amazon SNS SNS-Konsole aus erstellen. Eine ausführliche Anleitung finden Sie im Benutzerhandbuch unter Umstieg aus der SMS-Sandbox.AWS Endbenutzer-Messaging SMS

Verwenden Sie Simulatornummern oder verifizierte Telefonnummern mit AWS Endbenutzer-Messaging SMS

Wenn Sie Ihr Konto aus der SMS-Sandbox verschoben haben, überspringen Sie diesen Schritt.

Wenn Sie sich in der Sandbox befinden, aber eine Ausgangsnummer eingerichtet haben, können Sie Nachrichten an verifizierte Zielnummern senden. Informationen zum Einrichten verifizierter Ziele finden Sie im AWS Endbenutzer-Messaging SMS Benutzerhandbuch unter Hinzufügen einer verifizierten Zieltelefonnummer.

Sie können auch Nachrichten mit simulierten Absendern und Zielen senden. Simulatornachrichten erzeugen Protokolle, werden aber nicht über das Carrier-Netzwerk gesendet. Wählen Sie im Menü Kurzbefehle die Option SMS-Versand mit SMS-Simulator testen aus. Weitere Informationen finden Sie im AWS Endbenutzer-Messaging SMS Benutzerhandbuch unter Simulator-Telefonnummern.

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.

Verwendung der Sicherheitsfunktionen für Amazon-Cognito-Benutzerpools

Möglicherweise möchten Sie Ihre Anwendung vor Netzwerkangriffen, Kennwortraten, Identitätsmissbrauch und böswilliger Anmeldung und Anmeldung schützen. Ihre Konfiguration der Sicherheitsfunktionen von Amazon Cognito Cognito-Benutzerpools kann eine Schlüsselkomponente in Ihrer Sicherheitsarchitektur sein. Die Sicherheit Ihrer Anwendung liegt in der Verantwortung des Kunden. „Sicherheit in der Cloud“, wie im Modell der AWS gemeinsamen Verantwortung beschrieben.

Die Tools in diesem Kapitel tragen dazu bei, dass Ihr Anwendungssicherheitsdesign diesen Zielen entspricht.

Eine wichtige Entscheidung, die Sie bei der Konfiguration Ihres Benutzerpools treffen müssen, ist, ob Sie die öffentliche Registrierung und Anmeldung zulassen möchten. Einige Benutzerpool-Optionen wie vertrauliche Clients, administrative Erstellung und Bestätigung von Benutzern und Benutzerpools ohne Domäne sind in geringerem Maße Angriffen über das Internet ausgesetzt. Ein häufiger Anwendungsfall sind jedoch öffentliche Clients, die die Anmeldung von jedem Benutzer im Internet akzeptieren und alle Operationen direkt an Ihren Benutzerpool senden. In jeder Konfiguration, insbesondere aber bei diesen öffentlichen Konfigurationen, empfehlen wir, dass Sie Ihren Benutzerpool unter Berücksichtigung der Sicherheitsfunktionen planen und bereitstellen. Ungenügende Sicherheit kann sich auch auf Ihre AWS Rechnung auswirken, wenn durch unerwünschte Quellen neue aktive Benutzer entstehen oder versucht wird, bestehende Benutzer auszunutzen.

MFA und Bedrohungsschutz gelten für lokale Benutzer. Drittanbieter IdPs sind für den Sicherheitsstatus verbundener Benutzer verantwortlich.

Sicherheitsfunktionen werden in Benutzerpools zusammengefasst

Multi-Faktor-Authentifizierung (MFA)

Fordern Sie einen Code an, den Ihr Benutzerpool per E-Mail (mit dem Featureplan Essentials oder Plus) oder SMS-Nachricht oder über eine Authenticator-App sendet, um die Anmeldung für den Benutzerpool zu bestätigen.

Schutz vor Bedrohungen

Überwachen Sie die Anmeldung auf Risikoindikatoren und wenden Sie MFA an oder blockieren Sie die Anmeldung. Fügen Sie benutzerdefinierte Ansprüche und Bereiche für den Zugriff auf Tokens hinzu. Senden Sie MFA-Codes per E-Mail.

AWS WAF Netz ACLs

Untersuchen Sie den eingehenden Datenverkehr zu den Endpunkten Ihres Benutzerpools und zur Authentifizierungs-API auf unerwünschte Aktivitäten auf Netzwerk- und Anwendungsebene.

Sensitivität zwischen Groß- und Kleinschreibung

Verhindern Sie die Erstellung von Benutzern, deren E-Mail-Adresse oder bevorzugter Benutzername mit denen eines anderen Benutzers identisch ist, mit Ausnahme der Groß- und Kleinschreibung.

Löschschutz

Verhindern Sie, dass automatisierte Systeme Ihre Benutzerpools versehentlich löschen. Erfordern Sie eine zusätzliche Bestätigung für das Löschen des Benutzerpools in AWS Management Console.

Fehler beim Vorhandensein eines Benutzers

Schützen Sie sich vor der Offenlegung vorhandener Benutzernamen und Aliase in Ihrem Benutzerpool. Geben Sie als Antwort auf eine erfolglose Authentifizierung einen generischen Fehler zurück, unabhängig davon, ob der Benutzername gültig ist oder nicht.

Themen

- Hinzufügen der MFA zu einem Benutzerpool
- Erweiterte Sicherheit mit Bedrohungsschutz
- Ordnen Sie einem AWS WAF Benutzerpool eine Web-ACL zu
- Berücksichtigung der Groß-/Kleinschreibung im Benutzerpool
- Löschschutz für Benutzerpools
- Verwalten von Reaktionen auf Fehler bei vorhandenen Benutzern

Hinzufügen der MFA zu einem Benutzerpool

MFA fügt dem anfänglichen Etwas, das Sie wissen, einen Authentifizierungsfaktor hinzu, bei dem es sich normalerweise um einen Benutzernamen und ein Passwort handelt. Sie können SMS-Textnachrichten, E-Mail-Nachrichten oder zeitbasierte Einmalkennwörter (TOTP) als zusätzliche Faktoren wählen, um Ihre Benutzer anzumelden, deren primärer Authentifizierungsfaktor Kennwörter ist.

Die Multi-Faktor-Authentifizierung (MFA) erhöht die Sicherheit für die lokalen Benutzer in Ihrer Anwendung. Im Fall von Verbundbenutzern delegiert Amazon Cognito alle Authentifizierungsprozesse an den IdP und bietet ihnen keine zusätzlichen Authentifizierungsfaktoren an.

> **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 als Reaktion auf ein erhöhtes Risikoniveau ein zusätzlicher Authentifizierungsfaktor erforderlich ist. Weitere Informationen darüber, wie Sie Ihrem Benutzerpool eine adaptive Authentifizierung hinzufügen, finden Sie unter Erweiterte Sicherheit mit Bedrohungsschutz.

Wenn Sie die MFA für einen Benutzerpool auf `required` festlegen, müssen alle Benutzer MFA abschließen, um sich anzumelden. Um sich anzumelden, muss jeder Benutzer mindestens einen MFA-Faktor einrichten. Wenn MFA erforderlich ist, müssen Sie das MFA-Setup in das Benutzer-Onboarding einbeziehen, damit Ihr Benutzerpool es ihnen ermöglicht, sich anzumelden.

Die verwaltete Anmeldung fordert Benutzer auf, MFA einzurichten, wenn Sie MFA als erforderlich festlegen. Wenn Sie MFA in Ihrem Benutzerpool als optional festlegen, werden Benutzer bei der verwalteten Anmeldung nicht aufgefordert. 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.

Themen

- Wissenswertes über den Benutzerpool MFA
- Benutzer-MFA-Einstellungen
- Einzelheiten der MFA-Logik zur Benutzerlaufzeit
- Konfigurieren Sie einen Benutzerpool für die Multi-Faktor-Authentifizierung
- SMS- und E-Mail-Nachricht MFA
- TOTP-Software-Token-MFA

Wissenswertes über den Benutzerpool MFA

Berücksichtigen Sie Folgendes, bevor Sie MFA einrichten:

- Benutzer können entweder über MFA verfügen oder sich mit kennwortlosen Faktoren anmelden.

- In Benutzerpools, die Einmalkennwörter oder Hauptschlüssel unterstützen, können Sie MFA nicht auf erforderlich setzen.
- Sie können Ihrem Benutzerpool keine MFA hinzufügen `WEB_AUTHNEMAIL_OTP`, oder `SMS_OTP` hinzufügen, `AllowedFirstAuthFactors` wenn MFA erforderlich ist. In der Amazon Cognito Cognito-Konsole können Sie die Optionen für die wahlbasierte Anmeldung nicht so bearbeiten, dass sie kennwortlose Faktoren enthalten.
- Wahlbasierte Anmeldung bietet `PASSWORD` und `PASSWORD_SRP` berücksichtigt nur dann alle App-Clients, wenn MFA im Benutzerpool erforderlich ist. Weitere Informationen zu den Abläufen zwischen Benutzernamen und Passwörtern finden Sie unter Melden Sie sich mit dauerhaften Passwörtern an und Melden Sie sich mit dauerhaften Passwörtern und sicherer Payload an im Kapitel Authentifizierung dieses Handbuchs.
- In Benutzerpools, in denen MFA optional ist, können sich Benutzer, die einen MFA-Faktor konfiguriert haben, bei der wahlbasierten Anmeldung nur mit Benutzername-Kennwort-Authentifizierungsabläufen anmelden. Diese Benutzer sind für alle clientbasierten Anmeldeabläufe berechtigt.

In der folgenden Tabelle werden die Auswirkungen der MFA-Einstellungen des Benutzerpools und der Benutzerkonfiguration von MFA-Faktoren auf die Fähigkeit der Benutzer beschrieben, sich mit kennwortlosen Faktoren anzumelden.

MFA-Einstellung für Benutzerpool	MFA-Status des Benutzers	WebAuthn/OTP verfügbar	Nach der Passwortanmeldung zur Eingabe von MFA aufgefordert	Kann sich mit / OTP anmelden WebAuthn
Erforderlich	Configured	Nein	Ja	Nein
Erforderlich	Nicht konfiguriert	Nein	Nein (kann mich nicht anmelden)	Nein

MFA-Einstellung für Benutzerpool	MFA-Status des Benutzers	WebAuthn/OTP verfügbar	Nach der Passworta nmeldung zur Eingabe von MFA aufgeford ert	Kann sich mit / OTP anmelden WebAuthn
Optional	Configured	Kann eingerich tet werden WebAuthn , kann mich aber nicht mit dem Hauptschlüssel anmelden	Ja	Nein
Optional	Nicht konfiguriert	Ja	Nein	Ja
Aus	Any	Ja	Nein	Ja

- Die bevorzugte MFA-Methode eines Benutzers beeinflusst die Methoden, mit denen er sein Passwort wiederherstellen kann. Benutzer, deren bevorzugte MFA per E-Mail-Nachricht erfolgt, können keinen Code zum Zurücksetzen des Passworts per E-Mail erhalten. Benutzer, deren bevorzugte MFA per SMS-Nachricht erfolgt, können keinen Code zum Zurücksetzen des Passworts per SMS erhalten.

  Ihre Einstellungen für die Kennwortwiederherstellung müssen eine alternative Option bieten, wenn Benutzer nicht für Ihre bevorzugte Methode zum Zurücksetzen des Passworts in Frage kommen. Beispielsweise könnten Ihre Wiederherstellungsmechanismen E-Mail als erste Priorität haben und E-Mail-MFA könnte eine Option in Ihrem Benutzerpool sein. Fügen Sie in diesem Fall die Kontowiederherstellung per SMS-Nachricht als zweite Option hinzu oder verwenden Sie administrative API-Operationen, um die Passwörter für diese Benutzer zurückzusetzen.

  Amazon Cognito beantwortet Anfragen zum Zurücksetzen von Passwörtern von Benutzern, die keine gültige Wiederherstellungsmethode haben, mit einer Fehlerantwort. `InvalidParameterException`

Das Beispiel für eine `AccountRecoverySetting` Anfrage UpdateUserPoolzeigt, wie Benutzer auf die Wiederherstellung per SMS-Nachricht zurückgreifen können, wenn das Zurücksetzen des Kennworts für E-Mail-Nachrichten nicht verfügbar ist.

- Benutzer können MFA- und Passwort-Reset-Codes nicht an dieselbe E-Mail-Adresse oder Telefonnummer erhalten. Wenn sie Einmalpasswörter (OTPs) aus E-Mail-Nachrichten für MFA verwenden, müssen sie SMS-Nachrichten für die Kontowiederherstellung verwenden. Wenn sie OTPs SMS-Nachrichten für MFA verwenden, müssen sie E-Mail-Nachrichten für die Kontowiederherstellung verwenden. In Benutzerpools mit MFA können Benutzer die Self-Service-Kennwortwiederherstellung möglicherweise nicht abschließen, wenn sie Attribute für ihre E-Mail-Adresse, aber keine Telefonnummer haben, oder ihre Telefonnummer, aber keine E-Mail-Adresse.

  Um zu verhindern, dass Benutzer ihre Passwörter in Benutzerpools mit dieser Konfiguration nicht zurücksetzen können, legen Sie die `phone_number` Attribute `email` und nach Bedarf fest. Als Alternative können Sie Prozesse einrichten, bei denen diese Attribute immer erfasst und festgelegt werden, wenn sich Benutzer registrieren oder wenn Ihre Administratoren Benutzerprofile erstellen. Wenn Benutzer über beide Attribute verfügen, sendet Amazon Cognito automatisch Codes zum Zurücksetzen des Passworts an das Ziel, das nicht dem MFA-Faktor des Benutzers entspricht.
- Wenn Sie MFA in Ihrem Benutzerpool aktivieren und SMS-Nachricht oder E-Mail-Nachricht als zweiten Faktor wählen, können Sie Nachrichten an eine Telefonnummer oder ein E-Mail-Attribut senden, das Sie in Amazon Cognito nicht verifiziert haben. Nachdem Ihr Benutzer MFA abgeschlossen hat, setzt Amazon Cognito sein `phone_number_verified` oder `email_verified` Attribut auf. `true`
- Nach fünf erfolglosen Versuchen, einen MFA-Code zu präsentieren, beginnt Amazon Cognito mit dem unter Sperrverhalten bei fehlgeschlagenen Anmeldeversuchen beschriebenen exponentiellen Timeout-Kontosperrungsprozess.
- 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.
- Um den MFA-Status von Benutzern als Reaktion auf erkannte Ereignisse mit Bedrohungsschutz zu ändern, aktivieren Sie MFA und legen Sie ihn in der Amazon Cognito Cognito-Benutzerpool-Konsole als optional fest. Weitere Informationen finden Sie unter Erweiterte Sicherheit mit Bedrohungsschutz.

- E-Mail- und SMS-Nachrichten setzen voraus, dass Ihre Benutzer über E-Mail-Adressen- und Telefonnummernattribute verfügen. Sie können die Attribute `email` oder nach `phone_number` Bedarf in Ihrem Benutzerpool festlegen. In diesem Fall können Benutzer die Registrierung nur abschließen, wenn sie eine Telefonnummer angeben. Wenn Sie diese Attribute nicht wie erforderlich festlegen, aber MFA für E-Mail- oder SMS-Nachrichten verwenden möchten, fordern Sie Benutzer bei der Registrierung zur Eingabe ihrer E-Mail-Adresse oder Telefonnummer auf. Es hat sich bewährt, Ihren Benutzerpool so zu konfigurieren, dass Benutzer automatisch Nachrichten erhalten, um diese Attribute zu überprüfen.

  Amazon Cognito zählt eine Telefonnummer oder E-Mail-Adresse als verifiziert, wenn ein Benutzer erfolgreich einen temporären Code per SMS oder E-Mail-Nachricht erhalten und diesen Code in einer VerifyUserAttributeAPI-Anfrage zurückgegeben hat. Als Alternative kann Ihr Team Telefonnummern festlegen und diese mit einer Verwaltungsanwendung, die AdminUpdateUserAttributesAPI-Anfragen ausführt, als verifiziert markieren.
- Wenn Sie MFA als erforderlich festgelegt und mehr als einen Authentifizierungsfaktor aktiviert haben, fordert Amazon Cognito neue Benutzer auf, einen MFA-Faktor auszuwählen, den sie verwenden möchten. Benutzer benötigen eine Telefonnummer, um SMS-Nachrichten-MFA einzurichten, und eine E-Mail-Adresse, um E-Mail-Nachrichten-MFA einzurichten. Wenn ein Benutzer das Attribut nicht für eine verfügbare nachrichtenbasierte MFA definiert hat, fordert Amazon Cognito ihn auf, TOTP-MFA einzurichten. Die Aufforderung, einen MFA-Faktor (`SELECT_MFA_TYPE`) auszuwählen und einen ausgewählten Faktor (`MFA_SETUP`) einzurichten, erfolgt als Challenge-Antwort auf InitiateAuthAdminInitiateAuthAPI-Operationen.

Benutzer-MFA-Einstellungen

Benutzer können mehrere MFA-Faktoren einrichten. Nur einer kann aktiv sein. Sie können die effektive MFA-Präferenz für Ihre Benutzer in den Benutzerpooleinstellungen oder in Benutzeraufforderungen auswählen. Ein Benutzerpool fordert einen Benutzer zur Eingabe von MFA-Codes auf, wenn die Benutzerpooleinstellungen und ihre eigenen Einstellungen auf Benutzerebene die folgenden Bedingungen erfüllen:

1. Sie haben MFA in Ihrem Benutzerpool auf optional oder erforderlich gesetzt.
2. Der Benutzer hat ein gültiges `email` oder `phone_number` -Attribut oder hat eine Authenticator-App für TOTP eingerichtet.
3. Mindestens ein MFA-Faktor ist aktiv.
4. Ein MFA-Faktor ist als bevorzugt festgelegt.

Benutzerpool-Einstellungen und ihre Auswirkung auf MFA-Optionen

Die Konfiguration Ihres Benutzerpools beeinflusst die MFA-Methoden, die Benutzer wählen können. Im Folgenden sind einige Benutzerpooleinstellungen aufgeführt, die Einfluss darauf haben, ob Benutzer MFA einrichten können.

- In der Konfiguration der Multi-Faktor-Authentifizierung im Anmelde-Menü der Amazon Cognito Cognito-Konsole können Sie MFA auf optional oder erforderlich setzen oder deaktivieren. Das API-Äquivalent dieser Einstellung ist der MfaConfigurationParameter von`CreateUserPool`, und. `UpdateUserPool` `SetUserPoolMfaConfig`

  Auch in der Konfiguration für die Multi-Faktor-Authentifizierung bestimmt die Einstellung der MFA-Methoden die MFA-Faktoren, die Benutzer einrichten können. Das API-Äquivalent dieser Einstellung ist der Vorgang. SetUserPoolMfaConfig

- Im Anmeldemenü können Sie unter Benutzerkontenwiederherstellung konfigurieren, wie Ihr Benutzerpool Nachrichten an Benutzer sendet, die ihr Passwort vergessen haben. Die MFA-Methode eines Benutzers kann nicht dieselbe MFA-Übermittlungsmethode haben wie die Benutzerpool-Übermittlungsmethode für Codes mit vergessenen Passwörtern. Der API-Parameter für die Übermittlungsmethode „Passwort vergessen“ ist der Parameter von und. AccountRecoverySetting`CreateUserPool``UpdateUserPool`

  Beispielsweise können Benutzer E-Mail-MFA nicht einrichten, wenn Ihre Wiederherstellungsoption Nur E-Mail ist. Dies liegt daran, dass Sie E-Mail-MFA nicht aktivieren und die Wiederherstellungsoption auf Nur E-Mail im selben Benutzerpool festlegen können. Wenn Sie diese Option auf E-Mail, falls verfügbar, setzen, andernfalls auf SMS, ist E-Mail die bevorzugte Wiederherstellungsoption, aber Ihr Benutzerpool kann auf SMS-Nachrichten zurückgreifen, wenn ein Benutzer nicht für die Wiederherstellung von E-Mail-Nachrichten berechtigt ist. In diesem Szenario können Benutzer E-Mail-MFA als bevorzugt festlegen und nur dann eine SMS-Nachricht erhalten, wenn sie versuchen, ihr Passwort zurückzusetzen.

- Wenn Sie nur eine MFA-Methode als verfügbar festlegen, müssen Sie die Benutzer-MFA-Einstellungen nicht verwalten.

- Eine aktive SMS-Konfiguration macht SMS-Nachrichten automatisch zu einer verfügbaren MFA-Methode in Ihrem Benutzerpool.

  Eine aktive E-Mail-Konfiguration mit Ihren eigenen Amazon SES SES-Ressourcen in einem Benutzerpool und dem Feature-Plan Essentials oder Plus macht E-Mail-Nachrichten automatisch zu einer verfügbaren MFA-Methode in Ihrem Benutzerpool.

- Wenn Sie MFA in einem Benutzerpool auf erforderlich setzen, können Benutzer keine MFA-Methoden aktivieren oder deaktivieren. Sie können nur eine bevorzugte Methode festlegen.
- Wenn Sie MFA in einem Benutzerpool auf optional setzen, werden Benutzer bei der verwalteten Anmeldung nicht aufgefordert, MFA einzurichten, sondern Benutzer werden aufgefordert, einen MFA-Code einzugeben, wenn sie eine bevorzugte MFA-Methode haben.
- Wenn Sie den Bedrohungsschutz aktivieren und Antworten mit adaptiver Authentifizierung im Vollfunktionsmodus konfigurieren, muss MFA in Ihrem Benutzerpool optional sein. Eine der Antwortoptionen bei der adaptiven Authentifizierung besteht darin, MFA für einen Benutzer vorzuschreiben, dessen Anmeldeversuch daraufhin bewertet wird, ob er ein gewisses Risiko eindämmt.

  Die Einstellung Erforderliche Attribute im Anmeldemenü der Konsole bestimmt, ob Benutzer eine E-Mail-Adresse oder Telefonnummer angeben müssen, um sich in Ihrer Anwendung zu registrieren. E-Mail- und SMS-Nachrichten werden zu geeigneten MFA-Faktoren, wenn ein Benutzer über das entsprechende Attribut verfügt. Der Schema-Parameter von `CreateUserPool` legt die Attribute nach Bedarf fest.
- Wenn Sie MFA in einem Benutzerpool auf erforderlich setzen und sich ein Benutzer mit verwalteter Anmeldung anmeldet, fordert Amazon Cognito ihn auf, eine MFA-Methode aus den verfügbaren Methoden für Ihren Benutzerpool auszuwählen. Managed Login kümmert sich um die Erfassung einer E-Mail-Adresse oder Telefonnummer und die Einrichtung von TOTP. Das folgende Diagramm zeigt die Logik hinter den Optionen, die Amazon Cognito Benutzern bietet.

MFA-Einstellungen für Benutzer konfigurieren

Sie können MFA-Einstellungen für Benutzer in einem Self-Service-Modell mit Zugriffstoken-Autorisierung oder in einem vom Administrator verwalteten Modell mit administrativen API-Vorgängen konfigurieren. Diese Operationen aktivieren oder deaktivieren MFA-Methoden und legen eine von mehreren Methoden als bevorzugte Option fest. Nachdem Ihr Benutzer eine MFA-Einstellung festgelegt hat, fordert Amazon Cognito ihn bei der Anmeldung auf, einen Code für seine bevorzugte MFA-Methode einzugeben. Benutzer, die keine Präferenz festgelegt haben, werden aufgefordert, in einer Challenge eine bevorzugte Methode auszuwählen. `SELECT_MFA_TYPE`

- Legt in einem Benutzer-Self-Service-Modell oder einer öffentlichen Anwendung SetUserMfaPreference, autorisiert mit dem Zugriffstoken eines angemeldeten Benutzers, die MFA-Konfiguration fest.

- Legt in einer vom Administrator verwalteten oder vertraulichen Anwendung, die mit AWS Administratoranmeldedaten autorisiert ist AdminSetUserPreference, die MFA-Konfiguration fest.

Sie können die Benutzer-MFA-Einstellungen auch über das Benutzermenü der Amazon Cognito Cognito-Konsole festlegen. Weitere Informationen zu den öffentlichen und vertraulichen Authentifizierungsmodellen in der Amazon Cognito Cognito-Benutzerpools-API finden Sie unterGrundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten.

Einzelheiten der MFA-Logik zur Benutzerlaufzeit

Um die Schritte zu bestimmen, die bei der Anmeldung von Benutzern zu ergreifen sind, bewertet Ihr Benutzerpool Benutzer-MFA-Einstellungen, Benutzerattribute, die MFA-Einstellung des Benutzerpools, Maßnahmen zum Schutz vor Bedrohungen und Einstellungen für die Self-Service-Kontowiederherstellung. Anschließend werden Benutzer angemeldet, aufgefordert, eine MFA-Methode auszuwählen, sie werden aufgefordert, eine MFA-Methode einzurichten, oder sie werden zur Eingabe von MFA aufgefordert. Um eine MFA-Methode einzurichten, müssen Benutzer eine E-Mail-Adresse oder Telefonnummer angeben oder einen TOTP-Authentifikator registrieren. Sie können auch MFA-Optionen einrichten und eine bevorzugte Option im Voraus registrieren. In der folgenden Abbildung sind die detaillierten Auswirkungen der Konfiguration des Benutzerpools auf Anmeldeversuche unmittelbar nach der ersten Anmeldung aufgeführt.

Die hier dargestellte Logik gilt für SDK-basierte Anwendungen und die verwaltete Anmeldung, ist aber bei verwalteter Anmeldung weniger sichtbar. Gehen Sie bei der Fehlerbehebung für MFA von den Ergebnissen Ihrer Benutzer zurück zu den Benutzerprofil- und Benutzerpoolkonfigurationen, die zu der Entscheidung beigetragen haben.

Die folgende Liste entspricht der Nummerierung im Entscheidungslogikdiagramm und beschreibt jeden Schritt im Detail. A

steht für eine erfolgreiche Authentifizierung und den Abschluss des Datenflusses. A

steht für eine erfolglose Authentifizierung.

1. Ein Benutzer zeigt seinen Benutzernamen oder seinen Benutzernamen und sein Passwort auf Ihrem Anmeldebildschirm an. Wenn er keine gültigen Anmeldeinformationen vorlegt, wird seine Anmeldeanfrage abgelehnt.

2. Wenn die Benutzername-Passwort-Authentifizierung erfolgreich ist, stellen Sie fest, ob MFA erforderlich, optional oder deaktiviert ist. Wenn die Option deaktiviert ist, führen der richtige Benutzername und das richtige Passwort zu einer erfolgreichen Authentifizierung.

   a. Wenn MFA optional ist, stellen Sie fest, ob der Benutzer zuvor einen TOTP-Authentifikator eingerichtet hat. Wenn sie TOTP eingerichtet haben, fordern Sie TOTP MFA an. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

   b. Stellen Sie fest, ob der Benutzer aufgrund der adaptiven Authentifizierungsfunktion des Bedrohungsschutzes MFA einrichten musste. Wenn kein MFA zugewiesen wurde, ist der Benutzer angemeldet.

3. Wenn MFA erforderlich ist oder die adaptive Authentifizierung MFA zugewiesen hat, stellen Sie fest, ob der Benutzer einen MFA-Faktor als aktiviert und bevorzugt festgelegt hat. Falls ja, fordern Sie MFA mit diesem Faktor an. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

4. Wenn der Benutzer keine MFA-Präferenz festgelegt hat, stellen Sie fest, ob der Benutzer einen TOTP-Authentifikator registriert hat.

   a. Wenn der Benutzer einen TOTP-Authentifikator registriert hat, stellen Sie fest, ob TOTP-MFA im Benutzerpool verfügbar ist (TOTP-MFA kann deaktiviert werden, nachdem Benutzer zuvor Authentifikatoren eingerichtet haben).

   b. Stellen Sie fest, ob MFA für E-Mail-Nachrichten oder SMS-Nachrichten auch im Benutzerpool verfügbar ist.

   c. Wenn weder E-Mail- noch SMS-MFA verfügbar ist, fordern Sie den Benutzer zur Eingabe von TOTP-MFA auf. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

   d. Wenn E-Mail- oder SMS-MFA verfügbar sind, stellen Sie fest, ob der Benutzer über das entsprechende `email` `phone_number` OR-Attribut verfügt. Wenn ja, stehen ihnen alle Attribute zur Verfügung, die nicht die primäre Methode für die Self-Service-Kontowiederherstellung sind und für MFA aktiviert sind.

   e. Stellen Sie den Benutzer `SELECT_MFA_TYPE` vor eine Herausforderung mit `MFAS_CAN_SELECT` Optionen, die TOTP und die verfügbaren MFA-Faktoren für SMS oder E-Mail beinhalten.

f. Fordert den Benutzer zur Eingabe des Faktors auf, den er als Antwort auf die `SELECT_MFA_TYPE` Aufforderung ausgewählt hat. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

5. Wenn der Benutzer keinen TOTP-Authentifikator registriert hat oder wenn er einen registriert hat, TOTP MFA jedoch derzeit deaktiviert ist, stellen Sie fest, ob der Benutzer über ein Oder-Attribut verfügt. `email` `phone_number`
6. Wenn der Benutzer nur eine E-Mail-Adresse oder nur eine Telefonnummer hat, stellen Sie fest, ob dieses Attribut auch die Methode ist, die der Benutzerpool implementiert, um Nachrichten zur Kontowiederherstellung zum Zurücksetzen des Kennworts zu senden. Wenn der Wert wahr ist, können sie die Anmeldung nicht abschließen, wenn MFA erforderlich ist, und Amazon Cognito gibt einen Fehler zurück. Um die Anmeldung für diesen Benutzer zu aktivieren, müssen Sie ein Attribut hinzufügen, das nicht wiederhergestellt werden kann, oder einen TOTP-Authentifikator für ihn registrieren.

   a. Wenn sie über eine verfügbare E-Mail-Adresse oder Telefonnummer verfügen, die nicht wiederhergestellt werden kann, stellen Sie fest, ob der entsprechende E-Mail- oder SMS-MFA-Faktor aktiviert ist.
   b. Wenn sie über ein E-Mail-Adressattribut verfügen, das nicht wiederhergestellt werden kann und E-Mail-MFA aktiviert ist, fordern Sie sie auf, sie herauszufordern. `EMAIL_OTP` Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

   c. Wenn sie ein Telefonnummernattribut haben, das nicht wiederhergestellt werden kann und SMS-MFA aktiviert ist, fordern Sie sie zur Aufforderung auf. `SMS_MFA` Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

   d. Wenn sie kein Attribut haben, das für einen aktivierten E-Mail- oder SMS-MFA-Faktor in Frage kommt, stellen Sie fest, ob TOTP-MFA aktiviert ist. Wenn TOTP MFA deaktiviert ist, können sie die Anmeldung nicht abschließen, wenn MFA erforderlich ist, und Amazon Cognito gibt einen Fehler zurück. Um die Anmeldung für diesen Benutzer zu aktivieren, müssen Sie ein Attribut hinzufügen, das nicht wiederhergestellt werden kann, oder einen TOTP-Authentifikator für ihn registrieren.

> **Note**
>
> Dieser Schritt wurde bereits als Nein bewertet, wenn der Benutzer über einen TOTP-Authentifikator verfügt, TOTP MFA jedoch deaktiviert ist.

e. Wenn TOTP MFA aktiviert ist, stellen Sie den Benutzer `SOFTWARE_TOKEN_MFA` in den `MFAS_CAN_SETUP` Optionen `MFA_SETUP` vor eine Herausforderung. Um diese Aufforderung abzuschließen, müssen Sie separat einen TOTP-Authentifikator für den Benutzer registrieren und mit antworten. `"ChallengeName": "MFA_SETUP", "ChallengeResponses": {"USERNAME": "[username]", "SESSION": "[Session ID from VerifySoftwareToken]}"`

f. Nachdem der Benutzer auf die Aufforderung mit dem `MFA_SETUP` Sitzungstoken aus einer VerifySoftwareTokenAnfrage geantwortet hat, fordern Sie ihn zu einer `SOFTWARE_TOKEN_MFA` Aufforderung auf. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

7. Wenn der Benutzer sowohl eine E-Mail-Adresse als auch eine Telefonnummer hat, bestimmen Sie, welches Attribut, falls vorhanden, die primäre Methode für Kontowiederherstellungsnachrichten zum Zurücksetzen des Kennworts ist.

   a. Wenn die Self-Service-Kontowiederherstellung deaktiviert ist, kann jedes der Attribute für MFA verwendet werden. Stellen Sie fest, ob einer oder beide der E-Mail- und SMS-MFA-Faktoren aktiviert sind.

   b. Wenn beide Attribute als MFA-Faktor aktiviert sind, `SELECT_MFA_TYPE` fordern Sie den Benutzer mit den `MFAS_CAN_SELECT` Optionen `SMS_MFA` und `EMAIL_OTP` auf.

   c. Fordert sie zur Eingabe des Faktors auf, den sie als Antwort auf die `SELECT_MFA_TYPE` Aufforderung ausgewählt haben. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

   d. Wenn nur ein Attribut als MFA-Faktor in Frage kommt, fordern Sie sie auf, den verbleibenden Faktor herauszufordern. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

      Dieses Ergebnis tritt in den folgenden Szenarien auf.

      i. Wenn sie `phone_number` Attribute haben`email`, sind SMS und E-Mail-MFA aktiviert, und die primäre Methode zur Kontowiederherstellung erfolgt per E-Mail oder SMS-Nachricht.

   ii. Wenn sie über `phone_number` Attribute verfügen`email`, ist nur SMS-MFA oder E-Mail-MFA aktiviert und die Self-Service-Kontowiederherstellung ist deaktiviert.

8. Wenn der Benutzer keinen TOTP-Authentifikator registriert hat und auch kein `email` `phone_number` NOR-Attribut besitzt, fordern Sie ihn auf, ihn herauszufordern. `MFA_SETUP` Die Liste `MFAS_CAN_SETUP` enthält alle aktivierten MFA-Faktoren im Benutzerpool, die nicht die primäre Option für die Kontowiederherstellung sind. Sie können auf diese Herausforderung per E-Mail oder TOTP-MFA reagieren. `ChallengeResponses` Um SMS-MFA einzurichten, fügen Sie separat ein Telefonnummernattribut hinzu und starten Sie die Authentifizierung erneut.

   Für TOTP MFA antworten Sie mit. `"ChallengeName": "MFA_SETUP", "ChallengeResponses": {"USERNAME": "[username]", "SESSION": "[Session ID from VerifySoftwareToken]"}`

   Wenn Sie MFA per E-Mail erhalten möchten, antworten Sie mit`"ChallengeName": "MFA_SETUP", "ChallengeResponses": {"USERNAME": "[username]", "email": "[user's email address]"}`.

   a. Fragen Sie sie nach dem Faktor, den sie als Antwort auf die `SELECT_MFA_TYPE` Herausforderung ausgewählt haben. Wenn sie die MFA erfolgreich beantworten, sind sie angemeldet.

Konfigurieren Sie einen Benutzerpool für die Multi-Faktor-Authentifizierung

Sie können MFA in der Amazon Cognito Cognito-Konsole oder mit den SetUserPoolMfaConfigAPI-Operations- und SDK-Methoden 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 das Anmelde-Menü. Suchen Sie nach Multi-Faktor-Authentifizierung und wählen Sie Bearbeiten.
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 einer zusätzlichen SMS, E-Mail oder einem zeitbasierten Einmalkennwort (TOTP) als zusätzlichem Authentifizierungsfaktor anmelden.

b. Optionaler MFA. Sie können Ihren Benutzern die Möglichkeit geben, einen zusätzlichen Anmeldefaktor zu registrieren, aber dennoch Benutzern, die MFA nicht konfiguriert haben, die Anmeldung gestatten. Wählen Sie diese Option, wenn Sie die adaptive Authentifizierung verwenden. Weitere Informationen zur adaptiven Authentifizierung finden Sie unter Erweiterte Sicherheit mit Bedrohungsschutz.

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 E-Mail-Nachrichten, SMS-Nachrichten oder TOTP-generierende Authenticator-Apps als zweiten Faktor festlegen.

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 im Menü Authentifizierungsmethoden für Ihren Benutzerpool nach 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.

   Wenn Sie E-Mail-Nachrichten als zweiten Faktor verwenden und keine ursprüngliche Identität für die Verwendung mit Amazon Simple Email Service (Amazon SES) für E-Mail-Nachrichten konfiguriert haben, erstellen Sie eine in der Konsole. Sie müssen die Option E-Mail mit SES senden auswählen. Suchen Sie im Menü Authentifizierungsmethoden für Ihren Benutzerpool nach E-Mail und wählen Sie Bearbeiten. Wählen Sie eine ABSENDER-E-Mail-Adresse aus den verfügbaren verifizierten Identitäten in der Liste aus. Wenn Sie beispielsweise eine verifizierte Domain wählen`example.com`, müssen Sie auch einen FROM-Absendernamen in der verifizierten Domain konfigurieren. `admin-noreply@example.com`

8. Wählen Sie Änderungen speichern aus.

SMS- und E-Mail-Nachricht MFA

SMS- und E-Mail-MFA-Nachrichten bestätigen, dass Benutzer Zugriff auf ein Nachrichtenziel haben, bevor sie sich anmelden können. Sie bestätigen, dass sie nicht nur Zugriff auf ein Passwort haben, sondern auch auf die SMS-Nachrichten oder den E-Mail-Posteingang des ursprünglichen Benutzers.

Amazon Cognito fordert Benutzer auf, einen Kurzcode anzugeben, den Ihr Benutzerpool gesendet hat, nachdem sie erfolgreich einen Benutzernamen und ein Passwort eingegeben haben.

MFA für SMS und E-Mail-Nachrichten erfordert keine zusätzliche Konfiguration, nachdem Ihr Benutzer seinem Profil eine E-Mail-Adresse oder Telefonnummer hinzugefügt hat. Amazon Cognito kann Nachrichten an nicht verifizierte E-Mail-Adressen und Telefonnummern senden. Wenn ein Benutzer sein erstes MFA abschließt, markiert Amazon Cognito seine E-Mail-Adresse oder Telefonnummer als verifiziert.

Die MFA-Authentifizierung beginnt, wenn ein Benutzer mit MFA seinen Benutzernamen und sein Passwort in Ihre Anwendung eingibt. Ihre Anwendung übermittelt diese Anfangsparameter in einer SDK-Methode, die eine oder API-Anfrage aufruft. InitiateAuthAdminInitiateAuth Die `ChallengeParameters` API-Antwort enthält einen `CODE_DELIVERY_DESTINATION` Wert, der angibt, wohin der Autorisierungscode gesendet wurde. Zeigen Sie in Ihrer Anwendung ein Formular an, das den Benutzer auffordert, sein Telefon zu überprüfen, und das ein Eingabeelement für den Code enthält. Wenn sie ihren Code eingeben, reichen Sie ihn in einer Challenge-Response-API-Anfrage ein, um den Anmeldevorgang abzuschließen.

Nachdem sich ein Benutzer mit MFA auf den verwalteten Anmeldeseiten mit Benutzername und Passwort angemeldet hat, wird er automatisch zur Eingabe des MFA-Code aufgefordert.

Benutzerpools senden SMS-Nachrichten für MFA und andere Amazon Cognito Cognito-Benachrichtigungen mit Amazon Simple Notification Service (Amazon SNS) -Ressourcen in Ihrem. AWS-Konto In ähnlicher Weise senden Benutzerpools E-Mail-Nachrichten mit Amazon Simple Email Service (Amazon SES) -Ressourcen in Ihrem Konto. Für diese verknüpften Dienste fallen eigene Kosten für die Nachrichtenzustellung auf Ihrer AWS Rechnung an. Sie haben auch zusätzliche Anforderungen für den Versand von Nachrichten in Produktionsmengen. Weitere Informationen finden Sie unter den folgenden Links:

- Einstellungen für SMS-Nachrichten für Amazon-Cognito-Benutzerpools
- Weltweite SMS-Preise
- E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools
- Amazon SES SES-Preise

Überlegungen zur MFA für SMS- und E-Mail-Nachrichten

- Damit sich Benutzer mit E-Mail-MFA anmelden können, muss Ihr Benutzerpool über die folgenden Konfigurationsoptionen verfügen:

1. Sie haben den Plus- oder Essentials-Funktionsplan in Ihrem Benutzerpool. Weitere Informationen finden Sie unter Funktionspläne für Benutzerpools.
2. Ihr Benutzerpool sendet E-Mail-Nachrichten mit Ihren eigenen Amazon SES SES-Ressourcen. Weitere Informationen finden Sie unter E-Mail-Konfiguration von Amazon SES.

- Der MFA-Code ist für die Dauer der Authentifizierungsflusssitzung gültig, die Sie für Ihren App-Client festgelegt haben.

  Legen Sie die Dauer einer Authentifizierungsflow-Sitzung in der Amazon Cognito Cognito-Konsole im Menü App-Clients fest, wenn Sie Ihren App-Client bearbeiten. Sie können die Dauer der Authentifizierungsablaufsitzung auch in einer `CreateUserPoolClient`- oder `UpdateUserPoolClient`-API-Anforderung festlegen. Weitere Informationen finden Sie unter Ein Beispiel für eine Authentifizierungssitzung.
- Wenn ein Benutzer erfolgreich einen Code aus einer SMS oder E-Mail-Nachricht eingibt, die Amazon Cognito an eine nicht verifizierte Telefonnummer oder E-Mail-Adresse gesendet hat, markiert Amazon Cognito das entsprechende Attribut als verifiziert.
- Damit ein Benutzer eine Self-Service-Änderung am Wert einer Telefonnummer oder E-Mail-Adresse vornehmen kann, die mit MFA verknüpft ist, muss er sich anmelden und die Anfrage mit einem Zugriffstoken autorisieren. Wenn sie nicht auf ihre aktuelle Telefonnummer oder E-Mail-Adresse zugreifen können, können sie sich nicht anmelden. Ihr Team muss diese Werte mit AWS Administratoranmeldedaten in AdminUpdateUserAttributesAPI-Anfragen ändern.
- Nachdem Sie SMS in Ihrem Benutzerpool konfiguriert haben, können Sie SMS-Nachrichten nicht mehr als verfügbaren MFA-Faktor deaktivieren.

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 verwaltete Anmeldung verwendet, um Benutzer anzumelden, gibt Ihr Benutzer seinen Benutzernamen und sein Passwort ein und gibt dann das TOTP-Passwort auf einer zusätzlichen Anmeldeseite ein.

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 über die verwaltete Anmeldung für Ihre App registrieren, 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.

> **Important**
>
> 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 mit verwalteter Anmeldung führen. Informationen zum Erstellen einer Regel mit einer CAPTCHA-Aktion, die sich nicht auf das TOTP für verwaltete Logins auswirkt, finden Sie unter. Konfiguration Ihrer AWS WAF Web-ACL für die verwaltete Anmeldung TOTP MFA Weitere Informationen zu AWS WAF Web ACLs und Amazon Cognito finden Sie unterOrdnen Sie einem AWS WAF Benutzerpool eine Web-ACL zu.

Informationen zur Implementierung von TOTP MFA in einer maßgeschneiderten Benutzeroberfläche mit einem AWS SDK und der Amazon Cognito Cognito-Benutzerpools-API finden Sie unter. Konfiguration von TOTP MFA für einen Benutzer

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 mit dem `SignUp` API-Vorgang oder über die verwaltete Anmeldung in Ihrem Benutzerpool anmeldet, erhält Einmal-Token, wenn der Benutzer die Registrierung abgeschlossen hat.
   - Nachdem Sie einen Benutzer erstellt haben und der Benutzer sein erstes Passwort festgelegt hat, gibt Amazon Cognito Einmal-Token von der verwalteten Anmeldung 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 hat, aber TOTP MFA nicht eingerichtet hat, kann sich der Benutzer erst mit verwalteter Anmeldung anmelden, wenn er MFA eingerichtet hat. Anstelle des Zugriffstokens können Sie den `session` Antwortwert einer Anfrage InitiateAuthoder AdminInitiateAuthin einer `MFA_SETUP` 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 nur Apps TOTPs von Authenticator, die Codes mit der HMAC-Hash-Funktion generieren. SHA1 Mit SHA-256-Hashing generierte Codes geben einen `Code mismatch`-Fehler zurück.

Konfiguration von TOTP MFA für einen Benutzer

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 Ihre verwaltete Anmeldung präsentiert Amazon Cognito das TOTP in den Antworten auf MFA-Challenge-Antworten.

Unter bestimmten Umständen werden neue Benutzer bei der verwalteten Anmeldung aufgefordert, einen TOTP-Authentifikator einzurichten. Weitere Informationen finden Sie unter. Einzelheiten der MFA-Logik zur Benutzerlaufzeit

Themen

- Zuordnen des TOTP-Software-Tokens
- Bestätigen des TOTP-Tokens
- 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 Nutzer muss den Schlüssel in eine TOTP-generierende App wie Google Authenticator eingeben, indem er entweder den QR-Code scannt, den Ihre Anwendung anhand des privaten Schlüssels generiert, oder indem er den Schlüssel manuell eingibt.
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-Tokens

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 an, um Ihren Benutzer so SetUserMFAPreferencezu 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. SetUserMFAPreference
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 die verwaltete Anmeldung 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 mit verwalteter Anmeldung führen. AWS WAF CAPTCHA-Regeln haben diesen Effekt nur auf TOTP MFA in der verwalteten Anmeldung und in der klassischen gehosteten Benutzeroberfläche. 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. Um eine Regel mit einer CAPTCHA-Aktion zu erstellen, die sich nicht auf TOTP auf verwalteten Anmeldeseiten auswirkt, schließen Sie die `x-amzn-cognito-operation-name`

Header-Werte von `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`

Weitere Informationen zu AWS WAF Web ACLs und Amazon Cognito finden Sie unterOrdnen Sie einem AWS WAF Benutzerpool eine Web-ACL zu.

Erweiterte Sicherheit mit Bedrohungsschutz

Nachdem Sie Ihren Benutzerpool erstellt haben, haben Sie im Navigationsmenü der Amazon Cognito Cognito-Konsole Zugriff auf Bedrohungsschutz. Sie können die Funktionen zum Schutz vor Bedrohungen aktivieren und die Maßnahmen anpassen, die als Reaktion auf verschiedene Risiken ergriffen werden. Sie können auch den Prüfmodus verwenden, um Metriken zu erkannten Risiken zu erfassen, ohne Sicherheitsminderungen anzuwenden. Im Auditmodus veröffentlicht Threat Protection Kennzahlen an Amazon CloudWatch. Sie können Metriken sehen, nachdem Amazon Cognito sein erstes Ereignis generiert hat. Siehe Kennzahlen zum Schutz vor Bedrohungen anzeigen.

Der Bedrohungsschutz, früher als erweiterte Sicherheitsfunktionen bezeichnet, besteht aus einer Reihe von Tools zur Überwachung unerwünschter Aktivitäten in Ihrem Benutzerpool sowie aus Konfigurationstools zur automatischen Abschaltung potenziell bösartiger Aktivitäten. Der Bedrohungsschutz bietet verschiedene Konfigurationsoptionen für standardmäßige und benutzerdefinierte Authentifizierungsvorgänge. Möglicherweise möchten Sie beispielsweise eine Benachrichtigung an einen Benutzer mit einer verdächtigen benutzerdefinierten Authentifizierung senden, für den Sie zusätzliche Sicherheitsfaktoren eingerichtet haben, aber einen Benutzer mit derselben Risikostufe mit einfacher Benutzername/Kennwortauthentifizierung blockieren.

Der Bedrohungsschutz ist im Plus-Funktionsplan verfügbar. Weitere Informationen finden Sie unter Funktionspläne für Benutzerpools.

Die folgenden Benutzerpool-Optionen sind die Komponenten des Bedrohungsschutzes.

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 in username-and-password Standardauthentifizierungsabläufen in Benutzerpools nach kompromittierten Anmeldeinformationen suchen. Amazon Cognito erkennt keine kompromittierten Anmeldeinformationen in Secure Remote Password (SRP) oder benutzerdefinierter Authentifizierung.

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 zu dem Ereignis finden.

Weitere Informationen

Arbeiten mit der Erkennung kompromittierter Anmeldeinformationen

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. Sie können Benutzeraktivitäten überwachen und Reaktionen auf erkannte Risikostufen in Bezug auf Benutzername/Passwort und SRP sowie benutzerdefinierte Authentifizierung automatisieren.

Wenn Sie den Bedrohungsschutz 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.

Weitere Informationen

Arbeiten Sie mit adaptiver Authentifizierung

Liste der zugelassenen und verweigerten IP-Adressen

Mit dem Bedrohungsschutz von Amazon Cognito im Vollfunktionsmodus können Sie IP-Adressen erstellen: Immer blockieren und Ausnahmen immer zulassen. 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.

Wissenswertes über Zulassungs- und Blocklisten für IP-Adressen

- Sie müssen Always block und Always allow im CIDR-Format angeben`192.0.2.0/24`, z. B. eine 24-Bit-Maske oder `192.0.2.252/32` eine einzelne IP-Adresse.
- Geräte mit IP-Adressen im IP-Bereich „Immer blockieren“ können sich nicht mit SDK-basierten oder verwalteten Anmeldeanwendungen registrieren oder anmelden, sie können sich jedoch mit Drittanbieteranwendungen anmelden. IdPs
- Die Listen „Immer zulassen“ und „Immer blockieren“ wirken sich nicht auf die Token-Aktualisierung aus.
- Amazon Cognito wendet keine MFA-Regeln für die adaptive Authentifizierung auf Geräte aus dem IP-Bereich „Immer zulassen“ an, wendet jedoch Regeln für kompromittierte Anmeldeinformationen an.

Export von Protokollen

Der Bedrohungsschutz protokolliert detaillierte Details der Authentifizierungsanfragen der Benutzer in Ihrem Benutzerpool. Diese Protokolle enthalten Bedrohungsbeurteilungen, Benutzerinformationen und Sitzungsmetadaten wie Standort und Gerät. Sie können externe Archive dieser Protokolle zur Aufbewahrung und Analyse erstellen. Amazon Cognito Cognito-Benutzerpools exportieren Bedrohungsschutzprotokolle nach Amazon S3, CloudWatch Logs und Amazon Data Firehose. Weitere Informationen finden Sie unter Benutzerereignisverlauf anzeigen und exportieren.

Weitere Informationen

Benutzeraktivitätsprotokolle zum Schutz vor Bedrohungen exportieren

Themen

- Überlegungen und Einschränkungen zum Schutz vor Bedrohungen
- Den Bedrohungsschutz in Benutzerpools aktivieren
- Konzepte zur Durchsetzung des Bedrohungsschutzes
- Bedrohungsschutz für Standardauthentifizierung und benutzerdefinierte Authentifizierung
- Voraussetzungen für den Schutz vor Bedrohungen

- Schutz vor Bedrohungen einrichten
- Arbeiten mit der Erkennung kompromittierter Anmeldeinformationen
- Arbeiten Sie mit adaptiver Authentifizierung
- Erfassung von Daten für den Schutz vor Bedrohungen in Anwendungen

Überlegungen und Einschränkungen zum Schutz vor Bedrohungen

Die Optionen zum Schutz vor Bedrohungen unterscheiden sich je nach Authentifizierungsablauf

Amazon Cognito unterstützt sowohl die adaptive Authentifizierung als auch die Erkennung kompromittierter Anmeldeinformationen mit den Authentifizierungsabläufen und. `USER_PASSWORD_AUTH` `ADMIN_USER_PASSWORD_AUTH` Sie können nur die adaptive Authentifizierung für aktivieren. `USER_SRP_AUTH` Sie können den Bedrohungsschutz nicht mit föderierter Anmeldung verwenden.

Beiträge zur Anforderung von Kontingenten immer blockieren IPs

Blockierte Anfragen von IP-Adressen auf einer Ausnahmeliste Immer blockieren in Ihrem Benutzerpool trägt zu den Anforderungsratenkontingenten für Ihre Benutzerpools bei.

Beim Bedrohungsschutz gibt es keine Ratenbegrenzungen

Ein Teil des böswilligen Datenverkehrs zeichnet sich durch ein hohes Anforderungsvolumen aus, wie z. B. Distributed-Denial-of-Service (DDoS) -Angriffe. Die Risikoeinstufungen, die Amazon Cognito auf eingehenden Datenverkehr anwendet, gelten pro Anfrage und berücksichtigen nicht das Anforderungsvolumen. Einzelne Anfragen in einem Ereignis mit hohem Volumen können aus Gründen auf Anwendungsebene, die nichts mit ihrer Rolle bei einem volumetrischen Angriff zu tun haben, mit einer Risikobewertung und einer automatisierten Antwort versehen werden. Um Schutzmaßnahmen gegen volumetrische Angriffe in Ihren Benutzerpools zu implementieren, fügen Sie Web hinzu. AWS WAF ACLs Weitere Informationen finden Sie unter Ordnen Sie einem AWS WAF Benutzerpool eine Web-ACL zu.

Der Bedrohungsschutz wirkt sich nicht auf M2M-Anfragen aus

Die Gewährung von Kundenanmeldedaten ist für die machine-to-machine (M2M-) Autorisierung ohne Verbindung zu Benutzerkonten vorgesehen. Der Bedrohungsschutz überwacht nur Benutzerkonten und Passwörter in Ihrem Benutzerpool. Um Sicherheitsfunktionen in Ihre M2M-Aktivitäten zu integrieren, sollten Sie die Funktionen AWS WAF zur Überwachung von Anforderungsraten und

Inhalten in Betracht ziehen. Weitere Informationen finden Sie unter Ordnen Sie einem AWS WAF Benutzerpool eine Web-ACL zu.

Den Bedrohungsschutz in Benutzerpools aktivieren

Amazon Cognito user pools console

Um den Bedrohungsschutz für einen Benutzerpool zu aktivieren

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. Falls Sie dies noch nicht getan haben, aktivieren Sie den Plus-Funktionsplan im Menü Einstellungen.
5. Wählen Sie das Menü Bedrohungsschutz und dann Aktivieren aus.
6. Wählen Sie Änderungen speichern aus.

API

Stellen Sie Ihren Funktionsplan in einer CreateUserPooloder UpdateUserPoolAPI-Anfrage auf Plus ein. Im folgenden Teil des Anfragetextes wird der Bedrohungsschutz auf den Vollfunktionsmodus gesetzt. Ein vollständiges Beispiel für eine Anfrage finden Sie unter Beispiele.

```
"UserPoolAddOns": {
      "AdvancedSecurityMode": "ENFORCED"
   }
```

Bedrohungsschutz ist der Sammelbegriff für Funktionen, die Benutzervorgänge auf Anzeichen einer Kontoübernahme hin überwachen und automatisch auf sichere betroffene Benutzerkonten reagieren. Sie können Einstellungen für den Bedrohungsschutz auf Benutzer anwenden, wenn sie sich mit standardmäßigen und benutzerdefinierten Authentifizierungsabläufen anmelden.

Der Bedrohungsschutz generiert Protokolle, in denen die An- und Abmeldung der Benutzer sowie andere Aktivitäten detailliert beschrieben werden. Sie können diese Protokolle auf ein System eines

Drittanbieters exportieren. Weitere Informationen finden Sie unter Benutzerereignisverlauf anzeigen und exportieren.

Konzepte zur Durchsetzung des Bedrohungsschutzes

Der Bedrohungsschutz beginnt mit einem reinen Auditmodus, in dem Ihr Benutzerpool die Benutzeraktivitäten überwacht, Risikostufen zuweist und Protokolle generiert. Es hat sich bewährt, zwei Wochen oder länger den Modus nur für Audits zu verwenden, bevor Sie den Vollfunktionsmodus aktivieren. Der Vollfunktionsmodus umfasst eine Reihe automatischer Reaktionen auf erkannte riskante Aktivitäten und kompromittierte Passwörter. Im Modus „Nur Audit“ können Sie die Bedrohungsbeurteilungen überwachen, die Amazon Cognito durchführt. Sie können auch Feedback geben, um die Funktion anhand falscher positiver und negativer Ergebnisse zu schulen.

Sie können die Durchsetzung des Bedrohungsschutzes auf Benutzerpoolebene so konfigurieren, dass sie alle App-Clients im Benutzerpool abdeckt, und auf Ebene einzelner App-Clients. Konfigurationen zum Schutz vor Bedrohungen durch App-Clients haben Vorrang vor der Konfiguration des Benutzerpools. Um den Bedrohungsschutz für einen App-Client zu konfigurieren, navigieren Sie im App-Client-Menü Ihres Benutzerpools in der Amazon Cognito Cognito-Konsole zu den App-Client-Einstellungen. Dort können Sie Einstellungen auf Client-Ebene verwenden und die Erzwingung ausschließlich für den App-Client konfigurieren.

Darüber hinaus können Sie den Bedrohungsschutz für standardmäßige und benutzerdefinierte Authentifizierungstypen separat konfigurieren.

Bedrohungsschutz für Standardauthentifizierung und benutzerdefinierte Authentifizierung

Wie Sie den Bedrohungsschutz konfigurieren können, hängt von der Art der Authentifizierung ab, die Sie in Ihrem Benutzerpool und Ihren App-Clients vornehmen. Jede der folgenden Authentifizierungstypen kann ihren eigenen Erzwingungsmodus und ihre eigenen automatisierten Antworten haben.

Standardauthentifizierung

Die Standardauthentifizierung umfasst die Benutzeranmeldung, -abmeldung und Passwortverwaltung mit Benutzername-Passwort-Flows und bei verwalteter Anmeldung. Amazon Cognito Threat Protection überwacht Operationen auf Risikoindikatoren, wenn sie sich mit verwaltetem Login anmelden oder die folgenden `AuthFlow` API-Parameter verwenden:

InitiateAuth

`USER_PASSWORD_AUTH`,`USER_SRP_AUTH`. Die Funktion für kompromittierte Anmeldeinformationen hat bei der `USER_SRP_AUTH` Anmeldung keinen Zugriff auf Passwörter und überwacht und reagiert auch nicht auf Ereignisse mit diesem Ablauf.

AdminInitiateAuth

`ADMIN_USER_PASSWORD_AUTH`,. `USER_SRP_AUTH` Die Funktion für kompromittierte Anmeldeinformationen hat bei der `USER_SRP_AUTH` Anmeldung keinen Zugriff auf Passwörter und überwacht und reagiert auch nicht auf Ereignisse mit diesem Ablauf.

Sie können den Erzwingungsmodus für die Standardauthentifizierung auf Nur Prüfung oder Vollzugriff festlegen. Um die Bedrohungsüberwachung für die Standardauthentifizierung zu deaktivieren, setzen Sie den Bedrohungsschutz auf Keine Durchsetzung.

Benutzerdefinierte Authentifizierung

Bei der benutzerdefinierten Authentifizierung handelt es sich um eine Benutzeranmeldung mit benutzerdefinierten Challenge-Lambda-Triggern. Bei der verwalteten Anmeldung können Sie keine benutzerdefinierte Authentifizierung durchführen. Amazon Cognito Threat Protection überwacht Operationen auf Risikoindikatoren, wenn sie sich mit dem `AuthFlow` API-Parameter `CUSTOM_AUTH` `InitiateAuth` und `AdminInitiateAuth` anmelden.

Sie können den Erzwingungsmodus für die benutzerdefinierte Authentifizierung auf „Nur Prüfung“, „Vollständige Funktion“ oder „Keine Durchsetzung“ festlegen. Die Option Keine Durchsetzung deaktiviert die Bedrohungsüberwachung für die benutzerdefinierte Authentifizierung, ohne dass andere Funktionen zum Schutz vor Bedrohungen beeinträchtigt werden.

Voraussetzungen für den Schutz vor Bedrohungen

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.

Schutz vor Bedrohungen einrichten

Folgen Sie diesen Anweisungen, um den Bedrohungsschutz für den Benutzerpool einzurichten.

> **Note**
>
> Um eine andere Bedrohungsschutzkonfiguration für einen App-Client in der Amazon Cognito-Benutzerpools-Konsole einzurichten, wählen Sie den App-Client im Menü App-Clients aus und wählen Sie Einstellungen auf Client-Ebene verwenden.

AWS Management Console

Um den Bedrohungsschutz für einen Benutzerpool zu konfigurieren

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 das Menü Bedrohungsschutz und dann Aktivieren aus.
5. Wählen Sie die Methode zum Schutz vor Bedrohungen, die Sie konfigurieren möchten: Standardauthentifizierung und benutzerdefinierte Authentifizierung. Sie können verschiedene Erzwingungsmodi für die benutzerdefinierte und die Standardauthentifizierung festlegen, aber im Vollfunktionsmodus gilt dieselbe Konfiguration für automatische Antworten.
6. Wählen Sie Bearbeiten aus.
7. Wählen Sie einen Erzwingungsmodus. Um sofort auf erkannte Risiken zu reagieren, wählen Sie Vollständige Funktionalität und konfigurieren Sie die automatischen Reaktionen auf kompromittierte Anmeldeinformationen und die adaptive Authentifizierung. Um Informationen in Logs und Logs auf Benutzerebene zu sammeln CloudWatch, wählen Sie Nur Audit aus.

   Wir empfehlen, den Bedrohungsschutz zwei Wochen lang im Überwachungsmodus zu lassen, bevor Sie Aktionen aktivieren. Während dieser Zeit kann Amazon Cognito die

Nutzungsmuster Ihrer App-Benutzer kennenlernen und Sie können Feedback zu Ereignissen geben, um die Antworten anzupassen.

8. 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.

9. 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.

10. Erweitern Sie IP-Adressausnahmen, um eine Liste mit den Optionen „Immer zulassen“ oder „Immer blockieren“ mit IPv6 Adressbereichen zu erstellen, die unabhängig von der IPv4

Risikobewertung für den Bedrohungsschutz immer zugelassen oder blockiert werden. Geben Sie die IP-Adressbereiche in CIDR-Notation an (z. B. 192.168.100.0/24).

11. Wählen Sie Änderungen speichern aus.

API (user pool)

Um die Konfiguration des Bedrohungsschutzes für einen Benutzerpool festzulegen, senden Sie eine SetRiskConfigurationAPI-Anfrage, die einen `UserPoolId` Parameter, aber keinen Parameter enthält. `ClientId` Im Folgenden finden Sie ein Beispiel für einen Anfragetext für einen Benutzerpool. Diese Risikokonfiguration sieht je nach Schwere des Risikos eine Reihe von Maßnahmen vor und benachrichtigt Benutzer aller Risikostufen. Bei Anmeldevorgängen wird der Zugriff auf kompromittierte Anmeldeinformationen gesperrt.

Um diese Konfiguration zu erzwingen, müssen Sie dies `ENFORCED` in einer separaten CreateUserPoolAnfrage `AdvancedSecurityMode` oder einer API-Anfrage festlegen. UpdateUserPool Weitere Informationen zu den Platzhaltervorlagen wie `{username}` in diesem Beispiel finden Sie unterKonfiguration von MFA-, Authentifizierungs-, Überprüfungs- und Einladungsnachrichten.

```
{
   "AccountTakeoverRiskConfiguration": {
      "Actions": {
         "HighAction": {
            "EventAction": "MFA_REQUIRED",
            "Notify": true
         },
         "LowAction": {
            "EventAction": "NO_ACTION",
            "Notify": true
         },
         "MediumAction": {
            "EventAction": "MFA_IF_CONFIGURED",
            "Notify": true
         }
      },
      "NotifyConfiguration": {
         "BlockEmail": {
            "Subject": "You have been blocked for suspicious activity",
            "TextBody": "We blocked {username} at {login-time} from {ip-address}."
         },
         "From": "admin@example.com",
```

```
         "MfaEmail": {
            "Subject": "Suspicious activity detected, MFA required",
            "TextBody": "Unexpected sign-in from {username} on device {device-name}.
 You must use MFA."
         },
         "NoActionEmail": {
            "Subject": "Suspicious activity detected, secure your user account",
            "TextBody": "We noticed suspicious sign-in activity by {username} from
 {city}, {country} at {login-time}. If this was not you, reset your password."
         },
         "ReplyTo": "admin@example.com",
         "SourceArn": "arn:aws:ses:us-west-2:123456789012:identity/
admin@example.com"
      }
   },
   "CompromisedCredentialsRiskConfiguration": {
      "Actions": {
         "EventAction": "BLOCK"
      },
      "EventFilter": [ "SIGN_UP" ]
   },
   "RiskExceptionConfiguration": {
      "BlockedIPRangeList": [ "192.0.2.0/24","198.51.100.0/24" ],
      "SkippedIPRangeList": [ "203.0.113.0/24" ]
   },
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

API (app client)

Um die Bedrohungsschutzkonfiguration für einen App-Client festzulegen, senden Sie eine SetRiskConfigurationAPI-Anfrage, die einen `UserPoolId` Parameter und einen `ClientId` Parameter enthält. Im Folgenden finden Sie ein Beispiel für einen Anfragetext für einen App-Client. Diese Risikokonfiguration ist schwerwiegender als die Benutzerpoolkonfiguration und blockiert Einträge mit hohem Risiko. Außerdem werden bei der Registrierung und Anmeldung sowie beim Zurücksetzen des Kennworts Sperren kompromittierter Anmeldeinformationen blockiert.

Um diese Konfiguration zu erzwingen, müssen Sie in einer separaten Anfrage oder einer API-Anfrage auf festlegen`AdvancedSecurityMode`. `ENFORCED` CreateUserPoolUpdateUserPool Weitere Informationen zu den Platzhaltervorlagen wie `{username}` in diesem Beispiel finden Sie unterKonfiguration von MFA-, Authentifizierungs-, Überprüfungs- und Einladungsnachrichten.

```
{
   "AccountTakeoverRiskConfiguration": {
      "Actions": {
         "HighAction": {
            "EventAction": "BLOCK",
            "Notify": true
         },
         "LowAction": {
            "EventAction": "NO_ACTION",
            "Notify": true
         },
         "MediumAction": {
            "EventAction": "MFA_REQUIRED",
            "Notify": true
         }
      },
      "NotifyConfiguration": {
         "BlockEmail": {
            "Subject": "You have been blocked for suspicious activity",
            "TextBody": "We blocked {username} at {login-time} from {ip-address}."
         },
         "From": "admin@example.com",
         "MfaEmail": {
            "Subject": "Suspicious activity detected, MFA required",
            "TextBody": "Unexpected sign-in from {username} on device {device-name}.
 You must use MFA."
         },
         "NoActionEmail": {
            "Subject": "Suspicious activity detected, secure your user account",
            "TextBody": "We noticed suspicious sign-in activity by {username} from
 {city}, {country} at {login-time}. If this was not you, reset your password."
         },
         "ReplyTo": "admin@example.com",
         "SourceArn": "arn:aws:ses:us-west-2:123456789012:identity/
admin@example.com"
      }
   },
   "ClientId": "1example23456789",
   "CompromisedCredentialsRiskConfiguration": {
      "Actions": {
         "EventAction": "BLOCK"
      },
      "EventFilter": [ "SIGN_UP", "SIGN_IN", "PASSWORD_CHANGE" ]
```

```
    },
    "RiskExceptionConfiguration": {
       "BlockedIPRangeList": [ "192.0.2.1/32","192.0.2.2/32" ],
       "SkippedIPRangeList": [ "192.0.2.3/32","192.0.2.4/32" ]
    },
    "UserPoolId": "us-west-2_EXAMPLE"
}
```

Arbeiten mit der Erkennung kompromittierter 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 Benutzername und Passwort, mit verwalteter Anmeldung und mit der Amazon Cognito Cognito-API anmelden.

Im Bedrohungsschutz-Menü der Amazon Cognito Cognito-Konsole können Sie 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 Kennzahlen zum Schutz vor Bedrohungen anzeigen. 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.

Durch kompromittierte Anmeldeinformationen können Passwörter für die folgenden Benutzeraktivitäten überprüft werden.

Registrieren

Ihr Benutzerpool überprüft die Passwörter, die Benutzer während des SignUpVorgangs und auf der Anmeldeseite des verwalteten Logins übertragen, auf Anzeichen für kompromittierte Zugriffe.

Anmelden

Ihr Benutzerpool überprüft Passwörter, die Benutzer bei der kennwortbasierten Anmeldung eingeben, auf Anzeichen für eine Gefährdung. Amazon Cognito kann den `ADMIN_USER_PASSWORD_AUTH` Flow-In AdminInitiateAuth, den `USER_PASSWORD_AUTH` Flow-In InitiateAuthund die `PASSWORD USER_AUTH` Flow-Option in beiden Fällen überprüfen.

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.

Zurücksetzen des Passworts

Ihr Benutzerpool sucht bei Vorgängen, bei denen neue Benutzerkennwörter mit dem ConfirmForgotPasswordSelf-Service-Vorgang zum Zurücksetzen von Passwörtern festgelegt werden, nach Anzeichen für eine Kompromittierung. Der Code, der für diesen Vorgang erforderlich ist, wird von ForgotPasswordund AdminResetUserPasswordgeneriert.

Bei kompromittierten Anmeldeinformationen werden temporäre oder permanente, vom Administrator festgelegte Passwörter, die mit eingerichtet wurden, nicht überprüft. AdminSetUserPassword Bei temporären Kennwörtern überprüft Ihr Benutzerpool jedoch die Passwörter anhand der Antworten auf die `NEW_PASSWORD_REQUIRED` Abfrage in und. RespondToAuthChallengeAdminRespondToAuthChallenge

Weitere Informationen darüber, wie Sie Ihrem Benutzerpool einen Schutz gegen nicht mehr zuverlässige Anmeldeinformationen hinzufügen, finden Sie unter Erweiterte Sicherheit mit Bedrohungsschutz.

Arbeiten Sie mit adaptiver 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 Geräte- und Benutzerfaktoren, die Ihre Anwendung bereitstellt, und anderen Faktoren, die Amazon Cognito aus der Anfrage ableitet. Einige Faktoren, die zur Risikobewertung durch Amazon Cognito beitragen, sind die IP-Adresse, der Benutzeragent und die geografische Entfernung zu anderen Anmeldeversuchen. 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 Kennzahlen zu Anmeldeversuchen, deren Risikoniveau und fehlgeschlagenen Anfragen an Amazon. CloudWatch Weitere Informationen finden Sie unter Kennzahlen zum Schutz vor Bedrohungen anzeigen.

Weitere Informationen darüber, wie Sie Ihrem Benutzerpool eine adaptive Authentifizierung hinzufügen, finden Sie unter Erweiterte Sicherheit mit Bedrohungsschutz.

Themen

- Übersicht über die adaptive Authentifizierung
- Hinzufügen von Benutzergeräte- und Sitzungsdaten zu API-Anforderungen
- Benutzerereignisverlauf anzeigen und exportieren
- Bereitstellung von Ereignisfeedback
- Senden von Benachrichtigungsmeldungen

Übersicht über die adaptive Authentifizierung

Im Menü Bedrohungsschutz in der Amazon Cognito Cognito-Konsole können Sie Einstellungen für die adaptive Authentifizierung auswählen, einschließlich der Maßnahmen, die bei verschiedenen Risikostufen ergriffen werden sollen, und der Anpassung der Benachrichtigungen an Benutzer. Sie können allen Ihren App-Clients eine globale Konfiguration für den Schutz vor Bedrohungen zuweisen, aber auf einzelne App-Clients eine Konfiguration auf Client-Ebene 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 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 Amazon Cognito Threat Protection weitergeben, wenn Sie ihn mit der API registrieren, anmelden und sein Passwort zurücksetzen. 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 Threat Protection. Die Bibliothek zur Erfassung von Kontextdaten ist in der enthalten. AWS SDKs Weitere Informationen finden Sie unter Integration der Amazon-Cognito-Authentifizierung und -Autorisierung mit Web- und mobilen Apps. Sie können einreichen`ContextData`, wenn Sie über den Plus-Funktionsplan verfügen. Weitere Informationen finden Sie unter Schutz vor Bedrohungen einrichten.

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 Amazon Cognito Unauthenticated API Operations aufrufen, können Sie sich `UserContextData` an Amazon Cognito Threat Protection wenden. Diese Daten enthalten eine Geräteidentifikation im Parameter `EncodedData`. Sie können auch einen Parameter `IpAddress` in den `UserContextData` übergeben, wenn Sie die folgenden Bedingungen erfüllen:

- Ihr Benutzerpool ist im Plus-Funktionsplan enthalten. Weitere Informationen finden Sie unter Funktionspläne für Benutzerpools.
- Ihr App-Client verfügt über einen Client-Schlüssel. Weitere Informationen finden Sie unter Anwendungsspezifische Einstellungen mit 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.
- Sie möchten nur, dass Ihre nicht authentifizierten API-Operationen einen Gerätefingerabdruck, aber keine IP-Adresse an den Amazon Cognito Cognito-Bedrohungsschutz 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 das Menü App-Clients.
4. Wählen oder erstellen Sie einen App-Client. 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. Informationen darüber, wie Sie mit dem Bedrohungsschutz in Ihrer Web- oder Mobil-App arbeiten können, finden Sie unterErfassung von Daten für den Schutz vor Bedrohungen in Anwendungen. 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 verwendet`getData`.

```
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 zu Amazon Cognito SDKs finden Sie unter Installieren eines Benutzerpool-SDK. Amazon Cognito Threat Protection speichert nur Ereignisse und weist ihnen eine Risikobewertung zu, die Ihre App im richtigen Format übermittelt. Wenn Amazon Cognito eine Fehlerantwort zurückgibt, überprüfen Sie, ob Ihre Anfrage einen gültigen geheimen Hash enthält und IPv4 ob der `IPaddress` Parameter eine gültige IPv6 Oder-Adresse ist.

`ContextData`- und **`UserContextData`**-Ressourcen

- AWS Amplify SDK for Android: GetUserContextData
- AWS Amplify SDK for iOS: userContextData
- JavaScript: amazon-cognito-advanced-security-data.min.js

Benutzerereignisverlauf anzeigen und exportieren

Amazon Cognito generiert ein Protokoll für jedes Authentifizierungsereignis eines Benutzers, wenn Sie den Bedrohungsschutz aktivieren. Standardmäßig können Sie Benutzerprotokolle im Benutzermenü der Amazon Cognito Cognito-Konsole oder mit der AdminListUserAuthEventsAPI-Operation anzeigen. Sie können diese Ereignisse auch in ein externes System wie CloudWatch Logs, Amazon S3 oder Amazon Data Firehose exportieren. Mit der Exportfunktion können Sicherheitsinformationen über Benutzeraktivitäten in Ihrer Anwendung für Ihre eigenen Sicherheitsanalysesysteme leichter zugänglich gemacht werden.

Themen

- Den Verlauf der Benutzerereignisse anzeigen ()AWS Management Console
- Benutzerereignisverlauf anzeigen (API/CLI)
- Benutzerauthentifizierungsereignisse exportieren

Den Verlauf der Benutzerereignisse anzeigen ()AWS Management Console

Um den Anmeldeverlauf eines Benutzers zu sehen, können Sie den Benutzer aus dem Benutzermenü in der Amazon Cognito Cognito-Konsole auswählen. Der Ereignisverlauf für Benutzer wird von Amazon Cognito für zwei Jahre aufbewahrt.

Jedes Anmeldeereignis hat eine Ereignis-ID. Das Ereignis enthält auch entsprechende Kontextdaten wie dem Standort, Gerätdetails sowie den Risikoerkennungsergebnissen.

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 JSON-Webtoken (JWTs) für den Benutzerpool verstehen.

Benutzerereignisverlauf anzeigen (API/CLI)

Sie können den Verlauf von Benutzerereignissen mit der Amazon Cognito Cognito-API-Operation AdminListUserAuthEventsoder mit AWS Command Line Interface (AWS CLI) mit admin-list-user-auth-events abfragen.

AdminListUserAuthEvents request

Der folgende Anfragetext für `AdminListUserAuthEvents` gibt das neueste Aktivitätsprotokoll für einen Benutzer zurück.

```
{
  "UserPoolId": "us-west-2_EXAMPLE",
  "Username": "myexampleuser",
  "MaxResults": 1
}
```

admin-list-user-auth-events request

Die folgende Anfrage für `admin-list-user-auth-events` gibt das neueste Aktivitätsprotokoll für einen Benutzer zurück.

```
aws cognito-idp admin-list-user-auth-events --max-results 1 --username myexampleuser
 --user-pool-id us-west-2_EXAMPLE
```

Response

Amazon Cognito gibt auf beide Anfragen denselben JSON-Antworttext zurück. Im Folgenden finden Sie eine Beispielantwort für ein verwaltetes Anmeldeereignis, bei dem keine Risikofaktoren festgestellt wurden:

```
{
    "AuthEvents": [
        {
            "EventId": "[event ID]",
            "EventType": "SignIn",
            "CreationDate": "[Timestamp]",
```

```
            "EventResponse": "Pass",
            "EventRisk": {
                "RiskDecision": "NoRisk",
                "CompromisedCredentialsDetected": false
            },
            "ChallengeResponses": [
                {
                    "ChallengeName": "Password",
                    "ChallengeResponse": "Success"
                }
            ],
            "EventContextData": {
                "IpAddress": "192.168.2.1",
                "DeviceName": "Chrome 125, Windows 10",
                "Timezone": "-07:00",
                "City": "Bellevue",
                "Country": "United States"
            }
        }
    ],
    "NextToken": "[event ID]#[Timestamp]"
}
```

Benutzerauthentifizierungsereignisse exportieren

Konfigurieren Sie Ihren Benutzerpool so, dass Benutzerereignisse aus dem Bedrohungsschutz in ein externes System exportiert werden. Die unterstützten externen Systeme — Amazon S3, CloudWatch Logs und Amazon Data Firehose — können Ihre AWS Rechnung mit Kosten für Daten, die Sie senden oder abrufen, belasten. Weitere Informationen finden Sie unter Benutzeraktivitätsprotokolle zum Schutz vor Bedrohungen exportieren.

AWS Management Console

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 das Menü Log-Streaming. Wählen Sie Bearbeiten aus.
5. Markieren Sie unter Protokollierungsstatus das Kontrollkästchen neben Export von Benutzeraktivitätsprotokollen aktivieren.

6. Wählen Sie unter Logging-Ziel AWS-Service das aus, für das Sie Ihre Logs verwalten möchten: CloudWatch Log-Gruppe, Amazon Data Firehose-Stream oder S3-Bucket.
7. Durch Ihre Auswahl wird die Ressourcenauswahl mit dem entsprechenden Ressourcentyp gefüllt. Wählen Sie eine Protokollgruppe, einen Stream oder einen Bucket aus der Liste aus. Sie können auch auf die Schaltfläche Erstellen klicken, um AWS Management Console zum ausgewählten Service zu navigieren und eine neue Ressource zu erstellen.
8. Wählen Sie Änderungen speichern aus.

API

Wählen Sie einen Zieltyp für Ihre Benutzeraktivitätsprotokolle aus.

Im Folgenden finden Sie ein Beispiel für einen `SetLogDeliveryConfiguration` Anforderungstext, der einen Firehose-Stream als Protokollziel festlegt.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "FirehoseConfiguration": {
            "StreamArn": "arn:aws:firehose:us-west-2:123456789012:deliverystream/
example-user-pool-activity-exported"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Im Folgenden finden Sie ein Beispiel für einen `SetLogDeliveryConfiguration` Anfragetext, der einen Amazon S3 S3-Bucket als Protokollziel festlegt.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "S3Configuration": {
            "BucketArn": "arn:aws:s3:::amzn-s3-demo-logging-bucket"
         },
         "LogLevel": "INFO"
```

```
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Im Folgenden finden Sie ein Beispiel für einen `SetLogDeliveryConfiguration` Anfragetext, der eine CloudWatch Protokollgruppe als Protokollziel festlegt.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "CloudWatchLogsConfiguration": {
            "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-group:DOC-
EXAMPLE-LOG-GROUP"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

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 Cognito-Konsole im Benutzermenü einen Benutzer aus und wählen Sie Event-Feedback geben 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 in den Benutzerdetails im Benutzermenü auf. Wenn Sie einen Eintrag auswählen, können Sie das Ereignis als gültig oder ungültig markieren. Sie können Feedback

auch über die API-Operation 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 dem Bedrohungsschutz kann Amazon Cognito Ihre Benutzer über riskante 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.

> **Note**
>
> Amazon Cognito sendet nur dann Benachrichtigungen an Benutzer, wenn ihre Aktion eine automatisierte Risikoreaktion generiert: Anmeldung blockieren, Anmeldung zulassen, MFA auf optional setzen oder MFA erfordern. Einigen Anfragen wird möglicherweise eine Risikostufe zugewiesen, sie generieren jedoch keine automatisierten Risikoantworten mit adaptiver Authentifizierung. Bei diesen Anfragen sendet Ihr Benutzerpool keine Benachrichtigungen. Beispielsweise können falsche Passwörter mit einer Risikoeinstufung protokolliert werden, aber die Antwort von Amazon Cognito besteht darin, die Anmeldung fehlzuschlagen und keine adaptive Authentifizierungsregel anzuwenden.

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 E-Mail-Vorlagen

unter Nachrichten mit adaptiver Authentifizierung in Ihrer Bedrohungsschutzkonfiguration. Weitere Informationen über E-Mail-Vorlagen finden Sie unter Nachrichtenvorlagen.

Erfassung von Daten für den Schutz vor Bedrohungen in Anwendungen

Die adaptive Authentifizierung von Amazon Cognito bewertet die Risikostufen für versuchte Kontoübernahmen anhand von Kontextdetails der Anmeldeversuche von Benutzern. Ihre Anwendung muss den API-Anfragen Kontextdaten hinzufügen, damit Amazon Cognito Threat Protection das Risiko genauer einschätzen kann. Kontextdaten sind Informationen wie IP-Adresse, Browser-Agent, Geräteinformationen und Anforderungsheader, die kontextbezogene Informationen darüber liefern, wie ein Benutzer mit Ihrem Benutzerpool verbunden ist.

Die zentrale Verantwortung einer Anwendung, die diesen Kontext an Amazon Cognito übermittelt, ist ein `EncodedData` Parameter in Authentifizierungsanfragen an Benutzerpools. Um diese Daten zu Ihren Anfragen hinzuzufügen, können Sie Amazon Cognito mit einem SDK implementieren, das diese Informationen automatisch für Sie generiert, oder Sie können ein Modul für JavaScript iOS oder Android implementieren, das diese Daten sammelt. Anwendungen, die nur für Kunden bestimmt sind und direkte Anfragen an Amazon Cognito stellen, müssen implementiert werden. AWS Amplify SDKs Client-Server-Anwendungen, die über einen Zwischenserver oder eine API-Komponente verfügen, müssen ein separates SDK-Modul implementieren.

In den folgenden Szenarien verwaltet Ihr Authentifizierungs-Frontend die Erfassung von Benutzerkontextdaten ohne zusätzliche Konfiguration:

- Managed Login sammelt automatisch Kontextdaten und leitet sie an den Bedrohungsschutz weiter.
- Alle AWS Amplify Bibliotheken haben die Erfassung von Kontextdaten in ihre Authentifizierungsmethoden integriert.

Senden von Benutzerkontextdaten in Anwendungen, die nur für Kunden bestimmt sind, mit Amplify

Amplify SDKs unterstützt mobile Clients, die sich direkt bei Amazon Cognito authentifizieren. Kunden dieser Art stellen direkte API-Anfragen an öffentliche API-Operationen von Amazon Cognito. Amplify-Clients sammeln standardmäßig automatisch Kontextdaten zum Schutz vor Bedrohungen.

Amplify-Anwendungen mit JavaScript sind eine Ausnahme. Sie erfordern das Hinzufügen eines JavaScript Moduls, das Benutzerkontextdaten sammelt.

In der Regel verwendet eine Anwendung in dieser Konfiguration nicht authentifizierte API-Operationen wie InitiateAuthund. RespondToAuthChallenge Das UserContextDataObjekt hilft dabei, die Risiken für diese Operationen genauer einzuschätzen. Das Amplify SDKs fügt Geräte- und Sitzungsinformationen zu einem `EncodedData` Parameter von `UserContextData` hinzu.

Sammeln von Kontextdaten in Client-Server-Anwendungen

Einige Anwendungen haben eine Front-End-Stufe, die Benutzerauthentifizierungsdaten sammelt, und eine Anwendungs-Back-End-Stufe, die Authentifizierungsanfragen an Amazon Cognito sendet. Dies ist eine gängige Architektur für Webserver und Anwendungen, die durch Microservices unterstützt werden. In diesen Anwendungen müssen Sie eine öffentliche Bibliothek zur Erfassung von Kontextdaten importieren.

In der Regel verwendet ein Anwendungsserver in dieser Konfiguration authentifizierte API-Operationen wie AdminInitiateAuthund. AdminRespondToAuthChallenge Das ContextDataObjekt hilft Amazon Cognito dabei, Risiken für diese Operationen genauer einzuschätzen. Der Inhalt von besteht aus `ContextData` den codierten Daten, die Ihr Front-End an Ihren Server weitergegeben hat, und zusätzlichen Details aus der HTTP-Anfrage des Benutzers an Ihren Server. Diese zusätzlichen Kontextdetails, wie die HTTP-Header und die IP-Adresse, geben Ihrem Anwendungsserver die Merkmale der Benutzerumgebung wieder.

Ihr Anwendungsserver kann sich auch mit nicht authentifizierten API-Vorgängen wie und anmelden. InitiateAuthRespondToAuthChallenge Das UserContextDataObjekt dient als Grundlage für die Risikoanalyse des Bedrohungsschutzes bei diesen Vorgängen. Die Operationen in den verfügbaren Bibliotheken für die Datenerfassung im öffentlichen Kontext fügen dem `EncodedData` Parameter in Authentifizierungsanforderungen Sicherheitsinformationen hinzu. Konfigurieren Sie Ihren Benutzerpool außerdem so, dass er zusätzliche Kontextdaten akzeptiert, und fügen Sie dem `IpAddress` Parameter von die Quell-IP des Benutzers hinzu`UserContextData`.

Um Kontextdaten zu Client-Server-Anwendungen hinzuzufügen

1. Erfassen Sie in Ihrer Front-End-Anwendung codierte Kontextdaten vom Client mit einem iOS-, Android- oder JavaScript Modul.
2. Übergeben Sie die codierten Daten und die Details der Authentifizierungsanfrage an Ihren Anwendungsserver.
3. Extrahieren Sie auf Ihrem Anwendungsserver die IP-Adresse des Benutzers, die relevanten HTTP-Header, den angeforderten Servernamen und den angeforderten Pfad aus der HTTP-

Anfrage. Füllen Sie diese Werte in den ContextDataParameter Ihrer API-Anfrage an Amazon Cognito ein.

4. Füllen Sie den `EncodedData` Parameter von `ContextData` in Ihrer API-Anfrage mit den codierten Gerätedaten aus, die Ihr SDK-Modul gesammelt hat. Fügen Sie diese Kontextdaten zur Authentifizierungsanfrage hinzu.

Kontextdatenbibliotheken für Client-Server-Anwendungen

JavaScript

Das `amazon-cognito-advanced-security-data.min.js` Modul sammelt Daten`EncodedData`, die Sie an Ihren Anwendungsserver übergeben können.

Fügen Sie das `amazon-cognito-advanced-security-data.min.js` Modul zu Ihrer JavaScript Konfiguration hinzu. `<region>`Ersetzen Sie es durch einen AWS-Region aus der folgenden Liste: `us-east-1``us-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>
```

Um ein `encodedContextData` Objekt zu generieren, das Sie im `EncodedData` Parameter verwenden können, fügen Sie Ihrer JavaScript Anwendungsquelle Folgendes hinzu:

```
var encodedContextData = AmazonCognitoAdvancedSecurityData.getData(_username,
 _userpoolId, _userPoolClientId);
```

iOS/Swift

Um Kontextdaten zu generieren, können iOS-Anwendungen das Mobile SDK for iOS iOS-Modul AWSCognito IdentityProviderASF integrieren.

Um verschlüsselte Kontextdaten zum Schutz vor Bedrohungen zu sammeln, fügen Sie Ihrer Anwendung den folgenden Ausschnitt hinzu:

```
import AWSCognitoIdentityProviderASF

let deviceId = getDeviceId()
let encodedContextData = AWSCognitoIdentityProviderASF.userContextData(
```

```
                            userPoolId,
                            username: username,
                            deviceId: deviceId,
                            userPoolClientId: userPoolClientId)

/**
 * Reuse DeviceId from keychain or generate one for the first time.
 */
func getDeviceId() -> String {
    let deviceIdKey = getKeyChainKey(namespace: userPoolId, key:
 "AWSCognitoAuthAsfDeviceId")

   if let existingDeviceId = self.keychain.string(forKey: deviceIdKey) {
        return existingDeviceId
    }

    let newDeviceId = UUID().uuidString
    self.keychain.setString(newDeviceId, forKey: deviceIdKey)
    return newDeviceId
}

/**
 * Get a namespaced keychain key given a namespace and key
 */
func getKeyChainKey(namespace: String, key: String) -> String {
    return "\(namespace).\(key)"
}
```

Android

Um Kontextdaten zu generieren, können Android-Anwendungen das Mobile SDK for Android aws-android-sdk-cognitoidentityprovider -Modul -asf integrieren.

Um verschlüsselte Kontextdaten zum Schutz vor Bedrohungen zu sammeln, fügen Sie Ihrer Anwendung den folgenden Ausschnitt hinzu:

```
UserContextDataProvider provider = UserContextDataProvider.getInstance();
// context here is android application context.
String encodedContextData = provider.getEncodedContextData(context, username,
 userPoolId, userPoolClientId);
```

Ordnen Sie einem AWS WAF Benutzerpool eine Web-ACL zu

AWS WAF ist eine Firewall für Webanwendungen. Mit einer AWS WAF Web-Zugriffskontrollliste (Web ACL) können Sie Ihren Benutzerpool vor unerwünschten Anfragen an Ihre klassische gehostete Benutzeroberfläche, verwaltete Anmeldung und Amazon Cognito API-Serviceendpunkte schützen. Eine Web-ACL ermöglicht eine differenziertere Kontrolle aller HTTPS-Webanforderungen, auf die Ihr Benutzerpool reagiert. Weitere Informationen zum AWS WAF Web ACLs finden Sie unter Verwaltung und Verwendung 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

- Sie können Web-ACL-Regeln nicht so konfigurieren, dass sie mit personenbezogenen Daten (PII) in Benutzerpoolanfragen übereinstimmen, z. B. Benutzernamen, Passwörter, Telefonnummern oder E-Mail-Adressen. Diese Daten werden nicht verfügbar sein für. AWS WAF Konfigurieren Sie stattdessen Ihre Web-ACL-Regeln so, dass sie mit den Sitzungsdaten in den Headern, Pfaden und Textteilen wie IP-Adressen, Browser-Agenten und angeforderten API-Vorgängen übereinstimmen.
- Web-ACL-Regelbedingungen können nur benutzerdefinierte Blockantworten auf die erste Anfrage eines Benutzers an eine benutzerinteraktive verwaltete Anmeldeseite zurückgeben. Wenn nachfolgende Verbindungen einer benutzerdefinierten Blockantwortbedingung entsprechen, geben sie Ihren benutzerdefinierten Statuscode, Ihre Header- und Umleitungsantworten zurück, jedoch eine standardmäßige Blocknachricht.
- 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 jedem Benutzerpool 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. Die ATP-Funktion befindet sich in der `AWS-AWSManagedRulesATPRuleSet` verwalteten Regelgruppe. Bevor Sie eine Web-ACL einem Benutzerpool zuordnen, stellen Sie sicher, dass dieser diese verwaltete Regelgruppe nicht verwendet.
- 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 die verwaltete Anmeldung führen. Informationen zum Erstellen einer Regel mit einer CAPTCHA-Aktion, die sich nicht auf das TOTP für verwaltete Logins auswirkt, finden Sie unter. Konfiguration Ihrer AWS WAF Web-ACL für die verwaltete Anmeldung TOTP MFA

AWS WAF untersucht Anfragen an die folgenden Endpunkte.

Verwaltete Anmeldung und die klassische gehostete Benutzeroberfläche

Anforderungen aller Endpunkte in der Benutzerpool-Endpunkte und Referenz für verwaltete Logins.

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 Liste der API-Operationen, gruppiert nach Autorisierungsmodell.

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 verwalteter Anmeldeanfragen anpassen. Sie können Ihrem Benutzer nur bei der verwalteten Anmeldung ein CAPTCHA zur Lösung vorlegen.
> - 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 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- und AWS WAF Berechtigungen verfügen. Informationen zu AWS WAF Berechtigungen finden Sie unter AWS WAF API-Berechtigungen im AWS WAF Entwicklerhandbuch.

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowWebACLUserPool",
   "Effect": "Allow",
   "Action": [
    "cognito-idp:ListResourcesForWebACL",
    "cognito-idp:GetWebACLForResource",
    "cognito-idp:AssociateWebACL"
   ],
   "Resource": [
    "arn:aws:cognito-idp:*:123456789012:userpool/*"
```

```
    ]
   },
   {
    "Sid": "AllowWebACLUserPoolWAFv2",
    "Effect": "Allow",
    "Action": [
     "wafv2:ListResourcesForWebACL",
     "wafv2:AssociateWebACL",
     "wafv2:DisassociateWebACL",
     "wafv2:GetWebACLForResource"
    ],
    "Resource": "arn:aws:wafv2:*:123456789012:*/webacl/*/*"
   },
   {
    "Sid": "DisassociateWebACL1",
    "Effect": "Allow",
    "Action": "wafv2:DisassociateWebACL",
    "Resource": "*"
   },
   {
    "Sid": "DisassociateWebACL2",
    "Effect": "Allow",
    "Action": [
     "cognito-idp:DisassociateWebACL"
    ],
    "Resource": [
     "arn:aws:cognito-idp:*:123456789012:userpool/*"
    ]
   }
  ]
 }
```

Sie müssen zwar IAM-Berechtigungen gewähren, aber die aufgelisteten Aktionen sind nur für Genehmigungen bestimmt und entsprechen keinem API-Vorgang.

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 AWS WAFRegisterkarte im Bereich Sicherheit.

4. Wählen Sie Bearbeiten aus.
5. Wählen Sie 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 testen und protokollieren AWS WAF ACLs

Wenn Sie eine Regelaktion in Ihrer Web-ACL auf Count setzen, wird die Anfrage zu 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. Verwaltete Anmeldeanfragen enthalten nur den `x-amzn-cognito-client-id` Header. Weitere Informationen finden Sie unter Protokollieren des Web-ACL-Datenverkehrs im Entwicklerhandbuch zu AWS WAF .

AWS WAF Web ACLs sind in allen Featureplänen für Benutzerpools verfügbar. Die Sicherheitsfunktionen von AWS WAF ergänzen den Bedrohungsschutz von Amazon Cognito. Sie können beide Funktionen in einem Benutzerpool aktivieren. AWS WAF berechnet die Überprüfung von Benutzerpool-Anforderungen separat. Weitere Informationen finden Sie unter AWS WAF – Preise.

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.

Die Unterscheidung zwischen Groß- und Kleinschreibung gilt nicht nur für Attributeingaben, sondern auch für Ausgaben. Attributwerte mit gemischter Groß- und Kleinschreibung in Benutzerpools ohne Berücksichtigung der Groß- und Kleinschreibung werden in der Textausgabe des Benutzerpools auf Kleinbuchstaben reduziert. Beispiele für Textausgaben von Benutzerpools sind UserInfo-Antworten, Benutzerabfrageantworten wie die Ausgabe von GetUserund Eingabeereignisse für Lambda-Trigger.

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 Hinweise zur Berücksichtigung von Groß- und Kleinschreibung bei eindeutigen Bezeichnern und zu SAML IdPs finden Sie unter. Implementieren Sie die SP-initiierte 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`. Bei Benutzerpools, die Sie in der Amazon Cognito Cognito-Konsole erstellen, wird zwischen Groß- und Kleinschreibung unterschieden. Um einen Benutzerpool mit Berücksichtigung der Groß- und Kleinschreibung zu erstellen, müssen Sie den Vorgang verwenden. `CreateUserPool` Vor dem 12. Februar 2020 wurde bei Benutzerpools unabhängig von der Plattform standardmäßig Groß- und Kleinschreibung beachtet.

Im Anmeldemenü von AWS Management Console und in der `UsernameConfiguration` Eigenschaft von können Sie die Einstellungen zur DescribeUserPoolBerücksichtigung der Groß- und Kleinschreibung für jeden Benutzerpool in Ihrem Konto 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. Bei diesen Konflikten handelt es sich um Benutzernamen und E-Mail-Adressen, die in einem Benutzerpool, bei dem die Groß- und Kleinschreibung beachtet wird, eindeutig, in einem Benutzerpool, bei dem die Groß- und Kleinschreibung nicht berücksichtigt wird, jedoch identisch wären.

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 löschen AWS Management Console, 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 Cognito-API-Anfrage löschen möchten, müssen Sie zunächst `Inactive` in einer UpdateUserPoolAnfrage `DeletionProtection` zu wechseln. 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 DeleteUserPoolAnfrage 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. Um diese Funktion in Benutzerpools zu verwenden, die Sie mit dem AWS CLI oder einem AWS SDK erstellen, setzen Sie den `DeletionProtection` Parameter auf`True`.

Sie können den Löschschutzstatus im Löschschutzcontainer im Menü Einstellungen der Amazon Cognito Cognito-Konsole aktivieren oder deaktivieren.

So konfigurieren Sie Löschschutz

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 das Menü Einstellungen und navigieren Sie zur Registerkarte Löschschutz. Wählen Sie „Aktivieren“ oder „Deaktivieren“.
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, wenn Sie eine Anfrage für einen Benutzernamen einreichen, der nicht existiert:

- Amazon Cognito antwortet mit unspezifischen Informationen auf API-Anfragen, bei denen die Antwort andernfalls offenlegen könnte, dass ein gültiger Benutzer existiert.
- Amazon Cognito gibt auf Anfragen mit vergessenen Passwörtern und auf Authentifizierungsanfragen mit Authentifizierungsabläufen, außer bei wahlbasierter Authentifizierung ()*`USER_AUTH`*, eine generische Antwort auf Authentifizierungsfehler zurück

— zum Beispiel oder. `USER_SRP_AUTH` `CUSTOM_AUTH` Die Fehlerantwort gibt an, dass der Benutzername oder das Passwort falsch ist.

- Amazon Cognito beantwortet Anfragen nach einer wahlbasierten Authentifizierung mit einer zufälligen Auswahl aus den für den Benutzerpool zulässigen Challenge-Typen. Ihr Benutzerpool gibt möglicherweise einen Hauptschlüssel, ein Einmalpasswort oder eine Passwortabfrage zurück.
- Das Verhalten der Amazon Cognito Cognito-Kontobestätigung und der Kennwortwiederherstellung APIs wechselt zwischen der Rückgabe einer Antwort, die besagt, dass ein Code an ein simuliertes Liefermedium gesendet wurde, und der Rückgabe eines Fehlers. `InvalidParameterException`

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 in der Benutzername-Passwort-Authentifizierung und der Secure Remote Password (SRP) -Authentifizierung konfigurieren. Sie können die Fehler, die Sie zurückgeben, auch mit der benutzerdefinierten Authentifizierung anpassen. Die wahlbasierte Authentifizierung wird von Ihrer Konfiguration nicht beeinflusst. `PreventUserExistenceErrors`

Angaben zur Offenlegung der Existenz von Benutzern in Authentifizierungsabläufen

Wahlbasierte Authentifizierung

Im `USER_AUTH` wahlbasierten Authentifizierungsablauf gibt Amazon Cognito je nach Konfiguration Ihres Benutzerpools und Benutzerattributen eine Anfrage anhand der verfügbaren primären Authentifizierungsfaktoren zurück. Bei diesem Authentifizierungsablauf können Abfragen wie Passwort, Secure Remote Password (SRP), WebAuthn (Passkey), SMS-Einmalpasswort (OTP) oder E-Mail-OTP zurückgegeben werden. Wenn diese Option `PreventUserExistenceErrors` aktiviert ist, fordert Amazon Cognito nicht existierende Benutzer auf, eine oder mehrere der verfügbaren Authentifizierungsformen auszufüllen. Bei `PreventUserExistenceErrors` inaktiv gibt Amazon Cognito eine `UserNotFound` Ausnahme zurück.

Authentifizierung mit Benutzername und Passwort

Der Authentifizierungsfluss `ADMIN_USER_PASSWORD_AUTH` und der `PASSWORD` `USER_AUTH` Rückfluss a `NotAuthorizedException` mit der Meldung, `Incorrect username or password` wenn aktiv `PreventUserExistenceErrors` ist. `USER_PASSWORD_AUTH` Wenn inaktiv `PreventUserExistenceErrors` ist, kehren diese Flows zurück`UserNotFoundException`.

Secure-Remote-Password-(SRP)-basierte Authentifizierung

Es hat sich bewährt, die Implementierung nur `PreventUserExistenceErrors` mit `USER_SRP_AUTH` oder `USER_AUTH` im `PASSWORD_SRP` Flow von Benutzerpools ohne E-Mail-Adresse, Telefonnummer oder bevorzugte Alias-Attribute für Benutzernamen vorzunehmen. Benutzer mit Aliasattributen unterliegen möglicherweise nicht der Unterdrückung der Existenz von Benutzern im SRP-Authentifizierungsablauf. Benutzername-Kennwort-Authentifizierungsabläufe — `ADMIN_USER_PASSWORD_AUTHUSER_PASSWORD_AUTH`, und die `USER_AUTH PASSWORD` Herausforderung — verhindern vollständig die Existenz von Benutzern anhand von Aliasattributen.

Wenn jemand versucht, sich über SRP mit einem Benutzernamen anzumelden, der Ihrem App-Client nicht bekannt ist, gibt Amazon Cognito im ersten Schritt eine simulierte Antwort zurück, wie in RFC 5054 beschrieben. Amazon Cognito gibt dasselbe Salz und eine interne Benutzer-ID im UUID-Format 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. Weitere Informationen zur Implementierung der SRP-Authentifizierung finden Sie unter. Melden Sie sich mit dauerhaften Passwörtern und sicherer Payload an

> **Note**
>
> Sie können eine generische Antwort mit Benutzername und Kennwortauthentifizierung simulieren, wenn Sie verifizierungsbasierte Aliasattribute verwenden und der unveränderliche Benutzername nicht als UUID formatiert ist.

Lambda-Trigger für benutzerdefinierte Authentifizierungsanforderungen

Amazon Cognito ruft die benutzerdefinierte Authentifizierungsherausforderung auf, die Lambda auslöst, wenn Benutzer versuchen, sich mit dem `CUSTOM_AUTH` Authentifizierungsablauf anzumelden, ihr Benutzername jedoch nicht gefunden wird. Das Eingabeereignis enthält einen booleschen Parameter, der `UserNotFound` mit dem Wert für jeden nicht existierenden Benutzer benannt ist. `true` Dieser Parameter erscheint in den Anforderungsereignissen, die Ihr Benutzerpool an die Lambda-Funktionen Create, Define und Verify Auth Challenge sendet, die die benutzerdefinierte Authentifizierungsarchitektur bilden. Wenn Sie diesen

Indikator in der Logik Ihrer Lambda-Funktion untersuchen, können Sie benutzerdefinierte Authentifizierungsherausforderungen für einen Benutzer simulieren, der nicht existiert.

Lambda-Auslöser für die Vorab-Authentifizierung

Amazon Cognito ruft den Vorauthentifizierungs-Trigger auf, wenn Benutzer versuchen, sich anzumelden, ihr Benutzername jedoch nicht gefunden wird. Das Eingabeereignis enthält einen `UserNotFound` Parameter mit dem Wert `true` für jeden nicht existierenden Benutzer.

In der folgenden Liste werden die Auswirkungen von `PreventUserExistenceErrors` auf die Erstellung von Benutzerkonten beschrieben.

Einzelheiten zur Offenlegung der Existenz von Benutzern in Abläufen zur Benutzererstellung

SignUp

Die `SignUp` Operation 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 Von importierten Benutzer verlangen, dass sie ihre Passwörter 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 Importieren von Benutzern mit einem 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.

Benutzerpool-Endpunkte und Referenz für verwaltete Logins

Amazon Cognito bietet zwei Modelle der Benutzerpool-Authentifizierung: mit der Benutzerpools-API und mit dem OAuth 2.0-Autorisierungsserver. Verwenden Sie die API, wenn Sie OpenID Connect (OIDC) -Tokens mit einem AWS SDK in Ihrem Anwendungs-Backend abrufen möchten. Verwenden Sie den Autorisierungsserver, wenn Sie Ihren Benutzerpool als OIDC-Anbieter implementieren möchten. Der Autorisierungsserver fügt Funktionen wie Verbundanmeldung, API- und M2M-Autorisierung mit Gültigkeitsbereichen für Zugriffstoken und verwaltete Anmeldung hinzu. Sie können die API- und OIDC-Modelle jeweils einzeln oder zusammen verwenden, die auf Benutzerpoolebene oder App-Client-Ebene konfiguriert sind. Dieser Abschnitt ist eine Referenz für die Implementierung des OIDC-Modells. Weitere Informationen zu den beiden Authentifizierungsmodellen finden Sie unter. Grundlegendes zur Authentifizierung über API, OIDC und verwaltete Anmeldeseiten

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 verwaltete Anmeldung, bei der sich Ihre Benutzer registrieren und anmelden (Login-Endpunkt) und abmelden können (Logout-Endpunkt). Weitere Informationen zu diesen Ressourcen finden Sie unter Vom Benutzerpool verwaltete Anmeldung.

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 Um einen Benutzer mit einem Federated Identity Provider anzumelden, müssen Ihre Benutzer eine Anfrage an das interaktive verwaltete Login oder das OIDC senden. Login-Endpunkt Autorisieren des Endpunkts Der Authorize-Endpunkt leitet Ihre Benutzer entweder zu Ihren verwalteten Anmeldeseiten 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 verwalteten Anmeldung lässt sich Amazon Cognito auch SDKs für Android JavaScript, iOS und mehr integrieren. SDKs Sie bieten Tools zur Durchführung von Benutzerpool-API-Vorgängen 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 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 bei wechselnden AWS Zertifikaten ohne Vorankündigung fehlschlägt. 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

- Benutzerinteraktives verwaltetes Login und klassische Endpunkte für gehostete Benutzeroberflächen
- Identitätsanbieter und Endpunkte der vertrauenden Partei
- OAuth 2.0 Zuschüsse
- Verwendung von PKCE in Autorisierungscode-Zuschüssen
- Antworten auf verwaltete Anmelde- und Verbundfehler

Benutzerinteraktives verwaltetes Login und klassische Endpunkte für gehostete Benutzeroberflächen

Amazon Cognito aktiviert die verwalteten Anmeldeendpunkte in diesem Abschnitt, wenn Sie Ihrem Benutzerpool eine Domain hinzufügen. Dies sind Webseiten, auf denen Ihre Benutzer die wichtigsten

Authentifizierungsvorgänge eines Benutzerpools abschließen können. Sie enthalten Seiten für Passwortverwaltung, Multi-Faktor-Authentifizierung (MFA) und Attributüberprüfung.

Die Webseiten, aus denen sich die verwaltete Anmeldung zusammensetzt, sind Frontend-Webanwendungen für interaktive Benutzersitzungen mit Ihren Kunden. Ihre App muss die verwaltete Anmeldung in den Browsern Ihrer Benutzer aufrufen. Amazon Cognito unterstützt keinen programmatischen Zugriff auf die Webseiten in diesem Kapitel. Die Verbund-Endpunkte in der Identitätsanbieter und Endpunkte der vertrauenden Partei, die eine JSON-Antwort zurückgeben, können direkt in Ihrem App-Code abgefragt werden. Die Autorisieren des Endpunkts leitet entweder zur verwalteten Anmeldung oder zu einer IdP-Anmeldeseite weiter und muss auch in den Browsern der Benutzer geöffnet werden.

Alle Benutzerpool-Endpunkte akzeptieren Datenverkehr von IPv4 und IPv6 Quell-IP-Adressen.

In den Themen in diesem Handbuch werden häufig verwendete Endpunkte für verwaltete Anmeldung und klassische gehostete Benutzeroberflächen ausführlich beschrieben. Der Unterschied zwischen der verwalteten Anmeldung und der gehosteten Benutzeroberfläche ist sichtbar und funktioniert nicht. Mit Ausnahme `/passkeys/add` von: Alle Pfade werden von den beiden Versionen des Brandings für verwaltete Logins gemeinsam genutzt.

Amazon Cognito stellt die folgenden Webseiten bereit, wenn Sie Ihrem Benutzerpool eine Domain zuweisen.

Verwaltete Anmeldeendpunkte

Endpunkt-URL	Beschreibung	Zugriff darauf
https://*`Your user pool domain`*/login	Meldet lokale Benutzer und Verbundbenutzer im Benutzerpool an.	Weiterleitung von Endpunkten wie Autorisieren des Endpunkts, `/logout` und `/confirmforgotPassword` . Siehe Login-Endpunkt.
https://*`Your user pool domain`*/logout	Meldet Benutzer des Benutzerpools ab.	Direkt-Link. Siehe Logout-Endpunkt.
https:///ConfirmUser *`Your user pool domain`*	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.

Endpunkt-URL	Beschreibung	Zugriff darauf
https:///signup *Your user pool domain*	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:///confirm *Your user pool domain*	Nachdem Ihr Benutzerpool einen Bestätigungscode an einen Benutzer gesendet hat, der sich angemeldet hat, fordert er Ihren Benutzer auf, den Code einzugeben.	Nur Weiterleitung von `/signup`.
https://*Your user pool domain*/Passwort vergessen	Fordert die Benutzer zur Eingabe des Benutzern amens 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:///ConfirmForgotPassword *Your user pool domain*	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` .

Endpunkt-URL	Beschreibung	Zugriff darauf
https://*Your user pool domain*/resendcode	Sendet einen neuen Bestätigungscode an einen Benutzer, der sich in Ihrem Benutzerpool angemeldet hat.	Nur Weiterleitung von Neuen Code senden unter `/confirm`.
https:///passkeys/add *Your user pool domain*	Registriert einen neuen Hauptschlüssel. Nur bei verwalteter Anmeldung verfügbar.	• Im Anmeldevorgang nach der Bestätigung in App-Clients, die die Passkey-Authentifizierung unterstützen. • Direkter Link mit den gleichen Parametern wie `/oauth2/authorize` .

Themen

- Der Endpunkt für die verwaltete Anmeldung: /login
- Der Endpunkt für die verwaltete Anmeldung und Abmeldung: /logout

Der Endpunkt für die verwaltete Anmeldung: **`/login`**

Der Anmeldeendpunkt ist ein Authentifizierungsserver und ein Umleitungsziel vonAutorisieren des Endpunkts. Er ist der Einstiegspunkt für die verwaltete Anmeldung, 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 verwalteten Anmeldung. Rufen Sie in Ihrer App Verbundseiten und verwaltete Anmeldeseiten auf, die 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 zu `/login` von der weiterleitenAutorisieren des Endpunkts, werden alle Parameter weitergegeben, die Sie in Ihrer ersten 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.

Der Endpunkt für die verwaltete Anmeldung und Abmeldung: **`/logout`**

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 Endpunkt weiter. `/login` Die verfügbaren Parameter in einer GET-Anfrage an den `/logout` Endpunkt sind auf Anwendungsfälle der verwalteten Anmeldung mit Amazon Cognito zugeschnitten.

Der Abmelde-Endpunkt ist eine Front-End-Webanwendung für interaktive Benutzersitzungen mit Ihren Kunden. Ihre App muss diesen und andere verwaltete Anmeldeendpunkte in den Browsern Ihrer Benutzer aufrufen.

Um Ihren Benutzer zur verwalteten Anmeldung weiterzuleiten, 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`.

Um Ihren Benutzer auf eine von Ihnen gewählte Seite weiterzuleiten, fügen Sie Ihrem App-Client die Option Zulässige Abmeldung 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` einer der zulässigen Abmeldungen URLs für Ihren App-Client entspricht, 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 SAML-Benutzer mit Single Sign-Out abmelden.

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 Anwendungsspezifische Einstellungen mit App-Clients.

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 Anwendungsspezifische Einstellungen mit App-Clients.

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.

`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.

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 Parameter response_type.

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

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 ausgelassen werden, `logout_uri` aber ansonsten die Parameter angeben, die eine wohlgeformte Anfrage an den Autorisierungsendpunkt ausmachen, leitet Amazon Cognito Benutzer zur verwalteten Anmeldung weiter. Der Abmelde-Endpunkt hängt die Parameter in Ihrer ursprünglichen Anfrage an das Weiterleitungsziel an.

Die zusätzlichen Parameter, die Sie der Abmeldeanforderung hinzufügen, müssen in der Liste unter enthalten sein. Anforderungsparameter Beispielsweise unterstützt der Abmeldeendpunkt keine automatische IdP-Umleitung mit `identity_provider` oder `idp_identifier` -Parametern. Der Parameter `redirect_uri` in einer Anfrage an den Abmeldeendpunkt ist keine Abmelde-URL, sondern eine post-sign-in URL, die Sie an den Autorisierungsendpunkt weiterleiten möchten.

Im Folgenden finden Sie eine Beispielanforderung, die einen Benutzer abmeldet, zur Anmeldeseite weiterleitet und nach der Anmeldung einen Autorisierungscode bereitstellt. `https://www.example.com`

```
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
```

Identitätsanbieter und Endpunkte der vertrauenden Partei

Verbundendpunkte sind Benutzerpool-Endpunkte, die einem der von Benutzerpools verwendeten Authentifizierungsstandards dienen. Dazu gehören SAML-ACS URLs, OIDC-Erkennungsendpunkte und Dienstendpunkte für Benutzerpools, die sowohl als Identitätsanbieter als auch als vertrauende Partei fungieren. Verbundendpunkte initiieren Authentifizierungsabläufe, erhalten Authentifizierungsnachweise von Clients und stellen Token an diese aus IdPs. Sie interagieren mit IdPs Anwendungen und Administratoren, aber nicht mit Benutzern.

Die ganzseitigen Themen nach dieser Seite enthalten Informationen zu den OAuth 2.0- und OIDC-Provider-Endpunkten, die verfügbar werden, wenn Sie Ihrem Benutzerpool eine Domain hinzufügen. Die folgende Tabelle enthält eine Liste aller Verbundendpunkte.

Beispiele für Benutzerpool-Domänen sind:

1. Präfixdomäne: `mydomain.auth.us-east-1.amazoncognito.com`
2. Benutzerdefinierte Domain: `auth.example.com`

Benutzerpool-Verbund-Endpunkte

Endpunkt-URL	Beschreibung	Zugriff darauf
https://*Your user pool domain*/oauth2/authorize	Leitet einen Benutzer entweder zur verwaltet en Anmeldung oder zur Anmeldung mit seinem IdP weiter.	Wird im Kundenbrowser aufgerufen, um mit der Benutzerauthentifizierung zu beginnen. Siehe Autorisieren des Endpunkts.
https://*Your user pool domain*/oauth2/token	Gibt Token auf der Grundlage eines Autorisierungscodes oder einer Anforderung von Kundenanmeldeinformationen zurück.	Von der App zum Abrufen von Token angefordert. Siehe Token-Endpunkt.
*Your user pool domain*https://oauth2/UserInfo	Gibt Benutzerattribute basierend auf OAuth 2.0-Berei chen und Benutzeridentität in einem Zugriffstoken zurück.	Von der App zum Abrufen des Benutzerprofils angefordert. Siehe UserInfo-Endpunkt.

Endpunkt-URL	Beschreibung	Zugriff darauf
https://*Your user pool 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/ /.well-known/openid-configuration *your user pool ID*	Ein Verzeichnis der OIDC-Architektur Ihres Benutzerpools. [1]	Von der App angefordert, um Metadaten des Benutzerpool-Emittenten zu finden.
https://cognito-idp.*Region*.amazonaws.com/ /.well-known/jwks.json *your user pool ID*	Öffentliche Schlüssel, mit denen Sie Amazon Cognito Cognito-Token validieren können. [2]	Von der App zur Überprüfung angefordert JWTs.
https://*Your user pool domain*/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.
https://*Your user pool domain*/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. [3]
https*Your user pool 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] Das `openid-configuration` Dokument kann jederzeit mit zusätzlichen Informationen aktualisiert werden, damit der Endpunkt dem OIDC und OAuth2 den Spezifikationen entspricht.

[2] Die `jwks.json` JSON-Datei kann jederzeit mit neuen öffentlichen Token-Signaturschlüsseln aktualisiert werden.

[3] Weitere Informationen zur IDP-initiierten SAML-Anmeldung finden Sie unter. Implementieren Sie die IDP-initiierte SAML-Anmeldung

Weitere Informationen zu OpenID Connect und den OAuth Standards finden Sie unter OpenID Connect 1.0 und OAuth 2.0.

Themen

- Der Umleitungs- und Autorisierungsendpunkt
- Der Endpunkt des Token-Ausstellers
- Der Endpunkt der Benutzerattribute
- Der Endpunkt für den Token-Widerruf
- Der IdP-SAML-Assertion-Endpunkt

Der Umleitungs- und Autorisierungsendpunkt

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 Autorisierungsendpunkt leitet entweder zur verwalteten Anmeldung 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

Die Art der Antwort. 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 nach der Autorisierung des Benutzers durch Amazon Cognito weiterleitet.

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.

Siehe OAuth 2.0 — Endpunkt der Umleitung.

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 Parameter state hinzufügt, gibt Amazon Cognito seinen Wert an Ihre App zurück, wenn der Endpunkt `/oauth2/authorize` 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 in Base64 und dekodieren Sie sie dann in Ihrer App.

`identity_provider`

Optional.

Fügen Sie diesen Parameter hinzu, um die verwaltete Anmeldung zu umgehen und Ihren Benutzer auf die 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 für die Umleitung an einen Anbieter mit einem alternativen Namen für identity_provider hinzu. Sie können Identifikatoren für Ihre SAML 2.0 und OIDC IdPs im Menü Soziale Netzwerke und externe Anbieter der Amazon Cognito Cognito-Konsole eingeben.

`scope`

Optional.

Dabei kann es sich um eine Kombination aus für das System reservierten Bereichen oder benutzerdefinierten Bereichen, die einem Client zugeordnet sind, handeln. 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, das Sie zur Generierung der Herausforderung verwendet 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 PKCE-Abfrage (Proof of Key Code Exchange), die Sie anhand der generiert haben. `code_verifier` Weitere Informationen finden Sie unter Verwendung von PKCE in Autorisierungscode-Zuschüssen.

Nur erforderlich, wenn Sie einen `code_challenge_method`-Parameter angeben.

`nonce`

Optional.

Ein Zufallswert, den Sie der Anforderung 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.

`lang`

Optional.

Die Sprache, in der Sie benutzerinteraktive Seiten anzeigen möchten. Verwaltete Anmeldeseiten können lokalisiert werden, gehostete UI-Seiten (klassische Seiten) jedoch nicht. Weitere Informationen finden Sie unter Lokalisierung verwalteter Logins.

`login_hint`

Optional.

Eine Aufforderung zur Eingabe eines Benutzernamens, die Sie an den Autorisierungsserver weiterleiten möchten. Sie können einen Benutzernamen, eine E-Mail-Adresse oder eine Telefonnummer von Ihrem Benutzer erfassen und dem Zielanbieter erlauben, den Anmeldenamen des Benutzers vorab auszufüllen. Wenn Sie einen `login_hint` Parameter und keine `idp_identifier` oder `identity_provider` Parameter an den `oauth2/authorize` Endpunkt senden, füllt die verwaltete Anmeldung das Feld für den Benutzernamen mit Ihrem Hinweiswert aus. Sie können diesen Parameter auch an den übergeben Login-Endpunkt und den Wert für den Benutzernamen automatisch ausfüllen.

Wenn Ihre Autorisierungsanfrage eine Weiterleitung zu OIDC IdPs oder Google aufruft, fügt Amazon Cognito der Anfrage an diesen Drittanbieter-Autorisierer einen `login_hint` Parameter hinzu. Sie können Anmeldehinweise nicht an SAML, Apple, Login With Amazon oder Facebook (Meta) IdPs weiterleiten.

`prompt`

Optional.

Ein OIDC-Parameter, der das Authentifizierungsverhalten für bestehende Sitzungen steuert. Nur in der Branding-Version für verwaltete Anmeldung verfügbar, nicht in der klassischen gehosteten Benutzeroberfläche. Weitere Informationen aus der OIDC-Spezifikation finden Sie unter Authentifizierungsanfrage. Die Werte `none` und `login` wirken sich auf das Authentifizierungsverhalten des Benutzerpools aus.

Amazon Cognito leitet alle Werte von weiternone, `prompt` außer an Ihre, IdPs wenn Benutzer die Authentifizierung bei Drittanbietern auswählen. Dies ist der Fall, wenn die URL, auf die Benutzer zugreifen, einen `identity_provider` `idp_identifier` OR-Parameter enthält oder wenn der Autorisierungsserver sie an den weiterleitet Login-Endpunkt und sie einen IdP aus den verfügbaren Schaltflächen auswählen.

Parameterwerte abfragen

`prompt=none`

Amazon Cognito setzt die Authentifizierung für Benutzer, die über eine gültige authentifizierte Sitzung verfügen, im Hintergrund fort. Mit dieser Aufforderung können sich Benutzer unbeaufsichtigt zwischen verschiedenen App-Clients in Ihrem Benutzerpool authentifizieren. Wenn der Benutzer noch nicht authentifiziert ist, gibt der Autorisierungsserver einen Fehler zurück. `login_required`

`prompt=login`

Amazon Cognito verlangt von Benutzern, sich erneut zu authentifizieren, auch wenn sie eine bestehende Sitzung haben. Senden Sie diesen Wert, wenn Sie die Identität des Benutzers erneut überprüfen möchten. Authentifizierte Benutzer, die über eine bestehende Sitzung verfügen, können zur Anmeldung zurückkehren, ohne dass diese Sitzung ungültig wird. Wenn sich ein Benutzer, der eine bestehende Sitzung hat, erneut anmeldet, weist Amazon Cognito ihm ein neues Sitzungscookie zu. Dieser Parameter kann auch an Ihren weitergeleitet werden. IdPs IdPsdie diesen Parameter akzeptieren, fordern auch einen neuen Authentifizierungsversuch vom Benutzer an.

`prompt=select_account`

Dieser Wert hat keine Auswirkung auf die lokale Anmeldung und muss in Anfragen eingereicht werden, die zu IdPs umleiten. Wenn dieser Parameter in Ihrer Autorisierungsanfrage enthalten ist, wird er `prompt=select_account` zum URL-Pfad für das IdP-Umleitungsziel hinzugefügt.

Wenn dieser Parameter IdPs unterstützt wird, fordern sie die Benutzer auf, das Konto auszuwählen, mit dem sie sich anmelden möchten.

`prompt=consent`

Dieser Wert hat keine Auswirkung auf die lokale Anmeldung und muss in Anfragen eingereicht werden, die zu IdPs umleiten. Wenn dieser Parameter in Ihrer Autorisierungsanfrage enthalten ist, wird er `prompt=consent` zum URL-Pfad für das IdP-Umleitungsziel hinzugefügt. Wenn dieser Parameter IdPs unterstützt wird, fordern sie die Zustimmung des Benutzers an, bevor sie zurück zu Ihrem Benutzerpool weiterleiten.

Wenn Sie den `prompt` Parameter in Ihrer Anfrage weglassen, folgt die verwaltete Anmeldung dem Standardverhalten: Benutzer müssen sich anmelden, es sei denn, ihr Browser hat ein gültiges Sitzungscookie für die verwaltete Anmeldung. Sie können beispielsweise mehrere Werte für `prompt` mit einem Leerzeichen als Trennzeichen kombinieren. `prompt=login consent`

Beispiel: Erteilung des Autorisierungscodes

Dies ist ein Beispiel für eine Anfrage zur Erteilung eines Autorisierungscodes.

Die folgende Anfrage initiiert eine Sitzung zum Abrufen eines Autorisierungscodes, den Ihr Benutzer an Ihre App am `redirect_uri` Ziel 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
```

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
```

Beispiel: Erteilung eines Autorisierungscodes mit PKCE

In diesem Beispielablauf wird eine Autorisierungscode-Erteilung mit PKCE durchgeführt.

Diese Anfrage fügt einen `code_challenge` Parameter hinzu. Um den Austausch eines Codes gegen ein Token abzuschließen, müssen Sie den `code_verifier` Parameter in Ihre Anfrage an 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...
```

Der Autorisierungsserver leitet mit dem Autorisierungscode und dem Status zurück zu Ihrer Anwendung. Ihre Anwendung verarbeitet den Autorisierungscode und tauscht ihn gegen Token ein.

```
HTTP/1.1 302 Found
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111&state=abcdefg
```

Beispiel: erfordert eine erneute Authentifizierung mit **`prompt=login`**

Die folgende Anfrage fügt einen `prompt=login` Parameter hinzu, nach dem sich der Benutzer erneut authentifizieren muss, auch wenn er bereits eine Sitzung hat.

```
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&
prompt=login
```

Der Autorisierungsserver leitet zum Anmeldeendpunkt weiter und erfordert eine erneute Authentifizierung.

```
HTTP/1.1 302 Found Location: https://mydomain.auth.us-east-1.amazoncognito.com/
login?response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com&state=abcdefg&scope=openid+profile
+aws.cognito.signin.user.admin&prompt=login
```

Beispiel: unbeaufsichtigte Authentifizierung mit **`prompt=none`**

Die folgende Anfrage fügt einen `prompt=none` Parameter hinzu, der im Hintergrund überprüft, ob der Benutzer eine gültige Sitzung hat.

```
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&
prompt=none
```

Wenn keine gültige Sitzung vorhanden ist, gibt der Autorisierungsserver einen Fehler an die Umleitungs-URI zurück

```
HTTP/1.1 302 Found Location: https://www.example.com?error=login_required&state=abcdefg
```

Wenn eine gültige Sitzung besteht, gibt der Autorisierungsserver einen Autorisierungscode zurück.

```
HTTP/1.1 302 Found Location: https://www.example.com?
code=AUTHORIZATION_CODE&state=abcdefg
```

Beispiel: Token-Erteilung (implizit) ohne Gültigkeitsbereich **`openid`**

Dieser Beispielablauf generiert eine implizite Gewährung und kehrt JWTs direkt zur Sitzung des Benutzers zurück.

Die Anfrage bezieht sich auf eine implizite Erteilung von Ihrem Autorisierungsserver. Sie fordert Bereiche im Zugriffstoken an, die Self-Service-Vorgänge für Benutzerprofile autorisieren.

```
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
```

Der Autorisierungsserver leitet nur mit einem Zugriffstoken zurück zu Ihrer Anwendung. 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.

```
HTTP/1.1 302 Found
Location: https://example.com/
callback#access_token=eyJra456defEXAMPLE&token_type=bearer&expires_in=3600&state=STATE
```

Beispiel: Token-Erteilung (implizit) mit Gültigkeitsbereich **`openid`**

Dieser Beispielablauf generiert eine implizite Gewährung und gibt Token an den Browser des Benutzers zurück.

Die Anfrage bezieht sich auf eine implizite Erteilung durch Ihren Autorisierungsserver. Sie fordert Bereiche im Zugriffstoken an, die den Zugriff auf Benutzerattribute und Self-Service-Operationen autorisieren.

```
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
```

Der Autorisierungsserver leitet mit dem Zugriffstoken und dem ID-Token zurück zu Ihrer Anwendung (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
    ```

Der Endpunkt des Token-Ausstellers

Der OAuth 2.0-Token-Endpunkt unter `/oauth2/token` gibt JSON-Webtoken (JWTs) an Anwendungen aus, die die Gewährung von Autorisierungscode und Kundenanmeldedaten abschließen möchten. Diese Token sind das Endergebnis der Authentifizierung mit einem Benutzerpool. Sie enthalten Informationen über den Benutzer (ID-Token), die Zugriffsebene des Benutzers (Zugriffstoken) und die Berechtigung des Benutzers, seine angemeldete Sitzung

beizubehalten (Aktualisierungstoken). Rely-Party-Bibliotheken von OpenID Connect (OIDC) verarbeiten Anfragen an und Antwortnutzlasten von diesem Endpunkt aus. Tokens bieten einen überprüfbaren Authentifizierungsnachweis, Profilinformationen und einen Mechanismus für den Zugriff auf Backend-Systeme.

Ihr Benutzerpool OAuth 2.0-Autorisierungsserver 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. Machine-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 Autorisierungscode anmelden, der bei der verwalteten Anmeldung oder über den Verbund gewährt wurde, 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 Token mit dem entsprechenden API- oder SDK-Token-Aktualisierungsvorgang für Ihren App-Client.

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 Grant-Typen, Client-Geheimnissen, zulässigen Bereichen und Clients finden Sie unter. IDs Anwendungsspezifische Einstellungen mit App-Clients Weitere Informationen zur M2M-Autorisierung, zur Erteilung von

Kundenanmeldedaten und zur Autorisierung mit Zugriffstoken finden Sie unter. Bereiche, M2M und APIs mit Ressourcenservern

Um Informationen über einen Benutzer aus seinem Zugriffstoken abzurufen, geben Sie diese an Ihre Anfrage UserInfo-Endpunkt oder an eine GetUserAPI-Anfrage weiter. Das Zugriffstoken muss die entsprechenden Bereiche für diese Anfragen enthalten,

Formatieren Sie eine POST-Anforderung für den Token-Endpunkt

Der `/oauth2/token` Endpunkt unterstützt ausschließlich `HTTPS POST`. Dieser Endpunkt ist nicht benutzerinteraktiv. Behandeln Sie Token-Anfragen mit einer OpenID Connect (OIDC) -Bibliothek in Ihrer Anwendung.

Der Token-Endpunkt unterstützt `client_secret_basic`- und `client_secret_post`-Authentifizierung. Weitere Informationen zur OIDC-Spezifikation finden Sie unter Client-Authentifizierung. Weitere Informationen zum Token-Endpunkt aus der OpenID-Connect-Spezifikation finden Sie unter Token-Endpunkt.

Anfrageparameter im Header

Sie können die folgenden Parameter im Header Ihrer Anfrage an den Token-Endpunkt übergeben.

`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 den App-Client `djc98u3jiedmi283eu928` mit dem geheimen Clientschlüsselabcdef01234567890, 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

Die folgenden Parameter können Sie im `x-www-form-urlencoded` Format im Anforderungstext für den Token-Endpunkt anfordern.

`grant_type`

Pflichtfeld

Die Art 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.

> **Note**
>
> Der Token-Endpunkt gibt nur dann ein Aktualisierungstoken zurück, wenn dies der `grant_type` Fall ist`authorization_code`.

`client_id`

Optional. Nicht erforderlich, wenn Sie die App-Client-ID im *`Authorization`* Header angeben.

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. Nicht erforderlich, wenn Sie das geheime Client-Geheimnis im *`Authorization`* Header angeben und wenn der App-Client kein Geheimnis hat.

Das geheime Geheimnis des App-Clients, falls der App-Client über eines verfügt, für die `client_secret_post` Autorisierung.

`scope`

Optional.

Kann eine Kombination aus beliebigen Bereichen sein, die Ihrem App-Client zugeordnet sind. Amazon Cognito ignoriert Bereiche in der Anfrage, die für den angeforderten App-Client nicht zulässig sind. Wenn Sie diesen Anforderungsparameter nicht angeben, gibt der Autorisierungsserver einen `scope` Zugriffstoken-Anspruch mit allen Autorisierungsbereichen zurück, die Sie in Ihrer App-Client-Konfiguration aktiviert haben. Sie können alle Bereiche anfordern, die für den angeforderten App-Client zulässig sind: Standardbereiche, benutzerdefinierte Bereiche von Ressourcenservern und der `aws.cognito.signin.user.admin` Self-Service-Bereich für Benutzer.

`redirect_uri`

Optional. Für die Gewährung von Kundenanmeldedaten nicht erforderlich.

Es muss sich um dieselbe `redirect_uri` handeln, die verwendet wurde, um `authorization_code` in `/oauth2/authorize` zu bekommen.

Falls `grant_type` ja, müssen Sie diesen Parameter angeben. `authorization_code`

`refresh_token`

Optional. Wird nur verwendet, wenn der Benutzer bereits über ein Aktualisierungstoken verfügt und neue ID- und Zugriffstoken erhalten möchte.

Um neue Zugriffs- und ID-Tokens für die Sitzung eines Benutzers `refresh_token` zu generieren, setzen Sie den Wert auf ein gültiges Aktualisierungstoken, das der angeforderte App-Client ausgegeben hat.

Gibt ein neues Aktualisierungstoken mit neuer ID und Zugriffstoken zurück, wenn die Aktualisierungstoken-Rotation aktiv ist. Andernfalls werden nur ID- und Zugriffstoken zurückgegeben.

`code`

Optional. Nur bei Erteilung von Autorisierungscodes erforderlich.

Der Autorisierungscode aus einer Autorisierungscode-Erteilung. Sie müssen diesen Parameter angeben, wenn Ihre Autorisierungsanfrage ein `grant_type` of enthielt`authorization_code`.

`aws_client_metadata`

Optional.

Informationen, die Sie an die Lambda-Auslöser für die Vorab-Generierung von Token internen machine-to-machine (M2M) -Autorisierungsabläufe weitergeben möchten. Ihre Anwendung kann Kontextinformationen über die Sitzung sammeln und sie in diesem Parameter übergeben. Wenn Sie das URL-kodierte JSON-Format übergebenaws_client_metadata, nimmt Amazon Cognito es in das Eingabeereignis für Ihre Trigger-Lambda-Funktion auf. Ihre Pre-Token-Trigger-Event-Version oder globale Lambda-Trigger-Version muss für Version drei oder höher konfiguriert sein. Amazon Cognito akzeptiert zwar Anfragen an diesen Endpunkt in M2M-Datenströmen für Autorisierungscode und Kundenanmeldedaten, Ihr Benutzerpool wird jedoch nur von Anfragen mit Kundenanmeldedaten aws_client_metadata an den Trigger vor der Token-Generierung weitergeleitet.

`code_verifier`

Optional. Nur erforderlich, wenn Sie in Ihrer ersten Autorisierungsanfrage code_challenge Parameter angegeben code_challenge_method haben.

Der generierte Codeverifizierer, anhand dessen Ihre Anwendung in einer Anfrage zur code_challenge Erteilung eines Autorisierungscodes bei PKCE den Wert berechnet hat.

Austausch eines Autorisierungscodes für Token

Mit der folgenden Anfrage werden nach der Authentifizierung mit einem Autorisierungscode erfolgreich ID-, Zugriffs- und Aktualisierungstoken generiert. Die Anfrage übergibt das Client-Geheimnis im client_secret_basic Format im Header. Authorization

```
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
```

Die Antwort gibt dem Benutzer neue ID-, Zugriffs- und Aktualisierungstoken mit zusätzlichen Metadaten aus.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj3example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Kundenanmeldedaten mit Basisautorisierung

Die folgende Anfrage einer M2M-Anwendung fordert die Gewährung von Kundenanmeldedaten an. Da für Client-Anmeldeinformationen ein Client-Geheimnis erforderlich ist, wird die Anfrage mit einem `Authorization` Header autorisiert, der aus der App-Client-ID und dem geheimen Schlüssel abgeleitet wird. Die Anfrage führt zu einem Zugriffstoken mit den beiden angeforderten Bereichen. Die Anfrage enthält auch Client-Metadaten, die IP-Adressinformationen und ein Token enthalten, das an den Benutzer ausgegeben wird, für den die Gewährung erfolgt. Amazon Cognito leitet die Client-Metadaten an den Lambda-Trigger vor der Token-Generierung weiter.

```
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%2Fscope1%20resourceServerIdentifier2%2Fscope2&
&aws_client_metadata=%7B%22onBehalfOfToken%22%3A%22eyJra789ghiEXAMPLE%22,
%20%22ClientIpAddress%22%3A%22192.0.2.252%22%7D
```

Amazon Cognito leitet das folgende Eingabeereignis an den Lambda-Trigger vor der Token-Generierung weiter.

```
{
    version: '3',
    triggerSource: 'TokenGeneration_ClientCredentials',
    region: 'us-east-1',
    userPoolId: 'us-east-1_EXAMPLE',
    userName: 'ClientCredentials',
    callerContext: {
```

```
        awsSdkVersion: 'aws-sdk-unknown-unknown',
        clientId: '1example23456789'
    },
    request: {
        userAttributes: {},
        groupConfiguration: null,
        scopes: [
           'resourceServerIdentifier1/scope1',
           'resourceServerIdentifier2/scope2'
        ],
        clientMetadata: {
            'onBehalfOfToken': 'eyJra789ghiEXAMPLE',
            'ClientIpAddress': '192.0.2.252'
        }
    },
    response: { claimsAndScopeOverrideDetails: null }
}
```

Die Antwort gibt ein Zugriffstoken zurück. Die Erteilung von Kundenanmeldedaten dient der machine-to-machine (M2M-) Autorisierung und gibt nur Zugriffstoken zurück.

```
HTTP/1.1 200 OK
Content-Type: application/json
{
    "access_token": "eyJra1example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Kundenanmeldedaten mit POST-Body-Autorisierung

Die folgende Anfrage zur Gewährung von Kundenanmeldedaten enthält den `client_secret` Parameter im Hauptteil der Anfrage und keinen Header. `Authorization` Diese Anfrage verwendet die `client_secret_post` Autorisierungssyntax. Die Anfrage führt zu einem Zugriffstoken mit dem angeforderten Bereich. Die Anfrage enthält auch Client-Metadaten, die IP-Adressinformationen und ein Token enthalten, das an den Benutzer ausgegeben wird, für den diese Genehmigung erteilt wurde. Amazon Cognito leitet die Client-Metadaten an den Lambda-Trigger vor der Token-Generierung weiter.

```
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%2Fmy_custom_scope&
client_secret=9example87654321&
aws_client_metadata=%7B%22onBehalfOfToken%22%3A%22eyJra789ghiEXAMPLE%22,
%20%22ClientIpAddress%22%3A%22192.0.2.252%22%7D
```

Amazon Cognito leitet das folgende Eingabeereignis an den Lambda-Trigger vor der Token-Generierung weiter.

```
{
    version: '3',
    triggerSource: 'TokenGeneration_ClientCredentials',
    region: 'us-east-1',
    userPoolId: 'us-east-1_EXAMPLE',
    userName: 'ClientCredentials',
    callerContext: {
        awsSdkVersion: 'aws-sdk-unknown-unknown',
        clientId: '1example23456789'
    },
    request: {
        userAttributes: {},
        groupConfiguration: null,
        scopes: [
           'resourceServerIdentifier1/my_custom_scope'
        ],
        clientMetadata: {
            'onBehalfOfToken': 'eyJra789ghiEXAMPLE',
            'ClientIpAddress': '192.0.2.252'
        }
    },
    response: { claimsAndScopeOverrideDetails: null }
}
```

Die Antwort gibt ein Zugriffstoken zurück. Die Erteilung von Kundenanmeldedaten dient der machine-to-machine (M2M-) Autorisierung und gibt nur Zugriffstoken zurück.

```
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"
}
```

Erteilung des Autorisierungscodes mit PKCE

Die folgende Beispielanforderung vervollständigt eine Autorisierungsanfrage, die `code_challenge` Parameter `code_challenge_method` und Parameter in einer Autorisierungscode-Erteilungsanfrage mit PKCE beinhaltet.

```
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
```

In der Antwort werden ID-, Zugriffs- und Aktualisierungstoken aus der erfolgreichen PKCE-Überprüfung durch die Anwendung zurückgegeben.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj3example",
    "token_type": "Bearer",
```

```
    "expires_in": 3600
}
```

Token-Aktualisierung ohne Token-Rotation

Die folgende Beispielanforderung stellt ein Aktualisierungstoken für einen App-Client bereit, bei dem die Rotation des Aktualisierungstokens inaktiv ist. Da der App-Client über ein geheimes Client-Geheimnis verfügt, stellt die Anfrage einen `Authorization` Header bereit.

```
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
```

Die Antwort gibt neue IDs und Zugriffstoken zurück.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Token-Aktualisierung mit Rotation des Aktualisierungstokens

Die folgende Beispielanforderung stellt ein Aktualisierungstoken für einen App-Client bereit, für den die Rotation des Aktualisierungstokens aktiv ist. Da der App-Client über ein geheimes Client-Geheimnis verfügt, stellt die Anfrage einen `Authorization` Header bereit.

```
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
```

Die Antwort gibt neue ID-, Zugriffs- und Aktualisierungstoken zurück.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj4example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Beispiele für negative Antworten

Fehlerhafte Anfragen erzeugen Fehler vom Token-Endpunkt aus. Im Folgenden finden Sie eine allgemeine Übersicht über den Antworttext, wenn Token-Anfragen einen Fehler generieren.

```
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`.

Der Endpunkt der Benutzerattribute

Wo OIDC ID-Token ausgibt, die Benutzerattribute enthalten, implementiert OAuth 2.0 den `/oauth2/userInfo` Endpunkt. Ein authentifizierter Benutzer oder Client erhält ein Zugriffstoken mit einem Anspruch. `scopes` Dieser Anspruch bestimmt die Attribute, die der Autorisierungsserver zurückgeben soll. Wenn eine Anwendung dem `userInfo` Endpunkt ein Zugriffstoken vorlegt, gibt der Autorisierungsserver einen Antworttext zurück, der die Benutzerattribute enthält, die innerhalb der durch die Gültigkeitsbereiche der Zugriffstoken festgelegten Grenzen liegen. Ihre Anwendung kann Informationen über einen Benutzer vom `userInfo` Endpunkt abrufen, sofern sie über ein gültiges Zugriffstoken mit mindestens einem `openid` Gültigkeitsbereich verfügt.

Der Endpunkt `userInfo` ist ein OIDC-UserInfo-Endpunkt (OpenID-Connect). Sie reagiert mit Benutzerattributen, wenn Dienstanbieter Zugriffstoken vorlegen, die Ihr Token-Endpunkt ausgegeben hat. 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 OAuth 2.0-Dritt-Identity-Provider (IdP) hostet auch 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`

Die Bereiche im Zugriffstoken eines Benutzers werden durch den `scopes` Anforderungsparameter in Authentifizierungsanforderungen oder durch die Bereiche bestimmt, die der Lambda-Trigger vor der Token-Generierung hinzufügt. Sie können Zugriffstoken dekodieren und `scope` Ansprüche untersuchen, um herauszufinden, welche Bereiche der Zugriffskontrolle sie enthalten. Im Folgenden sind einige Bereichskombinationen aufgeführt, die sich auf die vom Endpunkt zurückgegebenen Daten auswirken. `userInfo` Der reservierte Amazon Cognito Cognito-Bereich `aws.cognito.signin.user.admin` hat keine Auswirkung auf die von diesem Endpunkt zurückgegebenen Daten.

Beispielbereiche im Zugriffstoken und ihre Auswirkung auf die Antwort **`userInfo`**

`openid`

Gibt eine Antwort mit allen Benutzerattributen zurück, die der App-Client lesen kann.

`openid profile`

Gibt die Benutzerattributename,`family_name`,`given_name`,`middle_name`,`nickname`,`preferred_username`, `zoneinfolocale`, und zurück`updated_at`. Gibt auch benutzerdefinierte Attribute zurück. In App-Clients, die keinen Lesezugriff auf jedes Attribut haben, entspricht die Antwort auf diesen Bereich allen Attributen innerhalb der Spezifikation, auf die Ihr App-Client Lesezugriff hat.

`openid email`

Gibt grundlegende Profilinformationen und die `email_verified` Attribute `email` und zurück.

`openid phone`

Gibt grundlegende Profilinformationen und die `phone_number_verified` Attribute `phone_number` und zurück.

GET /oauth2/userInfo

Ihre Anwendung generiert Anfragen an diesen Endpunkt direkt, 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 — falsches 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.

Der Endpunkt für den Token-Widerruf

Benutzer, die in ihrer Sitzung über ein Aktualisierungstoken verfügen, haben etwas Ähnliches wie ein Browser-Cookie. Sie können ihre bestehende Sitzung verlängern, solange das Aktualisierungstoken

gültig ist. Anstatt einen Benutzer aufzufordern, sich nach Ablauf seiner ID oder seines Zugriffstokens anzumelden, kann Ihre Anwendung das Aktualisierungstoken verwenden, um neue, gültige Token abzurufen. Sie können jedoch extern festlegen, dass die Sitzung eines Benutzers beendet werden soll, oder der Benutzer kann sich dafür entscheiden, seine aktuelle Sitzung zu vergessen. Zu diesem Zeitpunkt können Sie das Aktualisierungstoken widerrufen, sodass der Benutzer seine Sitzung nicht länger fortführen kann.

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 das Aktualisierungstoken selbst und alle nachfolgenden Zugriffs- und Identitätstoken desselben Aktualisierungstoken. Nachdem der Endpunkt die Token gesperrt hat, können Sie die widerrufenen Zugriffstoken nicht mehr verwenden, um auf die Amazon Cognito Cognito-Token zuzugreifen APIs , die authentifiziert werden.

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 Client-Schlüssel 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

Diese Sperranforderung widerruft ein Aktualisierungstoken für einen App-Client, der kein geheimes Clientgeheimnis hat. Notieren Sie sich den `client_id` Parameter im Hauptteil der Anfrage.

```
POST /oauth2/revoke HTTP/1.1
Host: mydomain.auth.us-east-1.amazoncognito.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded
token=2YotnFZFEjr1zCsicMWpAA&
client_id=1example23456789
```

Diese Sperranforderung widerruft ein Aktualisierungstoken für einen App-Client, der über einen geheimen Clientschlüssel verfügt. Notieren Sie sich den `Authorization` Header, der eine verschlüsselte Client-ID und einen geheimen Client-Schlüssel enthält, aber nicht `client_id` im Hauptteil der Anfrage.

```
POST /oauth2/revoke HTTP/1.1
Host: 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.

Der IdP-SAML-Assertion-Endpunkt

Der `/saml2/idpresponse` empfängt SAML-Assertionen. Bei der service-provider-initiated (SP-initiierten) Anmeldung interagiert Ihre Anwendung nicht direkt mit diesem Endpunkt — Ihr SAML 2.0-Identitätsanbieter (IdP) leitet Ihren Benutzer mit seiner SAML-Antwort hierher weiter. Für die SP-initiierte Anmeldung konfigurieren Sie Ihren IdP mit dem Pfad zu Ihrer `saml2/idpresponse` als Assertion Consumer Service (ACS) -URL. Weitere Informationen zur Sitzungsinitiierung finden Sie unter. Initiierung der SAML-Sitzung in Amazon-Cognito-Benutzerpools

Rufen Sie bei der IDP-initiierten Anmeldung Anfragen an diesen Endpunkt in Ihrer Anwendung auf, nachdem Sie sich bei Ihrem SAML 2.0-Anbieter als Benutzer angemeldet haben. Ihre Benutzer melden sich mit Ihrem IdP in ihrem Browser an, dann sammelt Ihre Anwendung die SAML-Assertion und sendet sie an diesen Endpunkt. Sie müssen SAML-Assertionen im Hauptteil einer Anfrage über HTTPS einreichen. `HTTP POST` Der Hauptteil Ihrer `POST` Anfrage muss aus einem `SAMLResponse` Parameter und einem `Relaystate` Parameter bestehen. Weitere Informationen finden Sie unter Implementieren Sie die IDP-initiierte SAML-Anmeldung.

Der `saml2/idpresponse` Endpunkt kann SAML-Assertionen mit einer Länge von bis zu 100.000 Zeichen akzeptieren.

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

SAMLResponse

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 Zuschusstyp 2.0.

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, an den Sie Ihren Benutzer weiterleiten 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 Bereichen OAuth 2.0 enthalten `openidemail`, und. `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 Zuschüsse

Der Amazon Cognito Cognito-Benutzerpool OAuth 2.0-Autorisierungsserver gibt Token als Antwort auf drei Arten von OAuth 2.0-Autorisierungsberechtigungen 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 Beispiel: Erteilung des Autorisierungscodes. Amazon Cognito unterstützt Proof Key for Code Exchange (PKCE) bei der Gewährung von 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 auch alle drei Tokentypen aus der Authentifizierung über die Amazon Cognito Cognito-Benutzerpools-API abrufen, aber die API gibt keine Zugriffstoken mit anderen Bereichen als aus. `aws.cognito.signin.user.admin`

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 Beispiel: Token-Erteilung (implizit) ohne Gültigkeitsbereich `openid`.

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 Kundenanmeldedaten ist ein Autorisierungsmechanismus, der 2.0-Bereiche umfasst OAuth . In der Regel enthält das Token benutzerdefinierte Bereichsansprüche, mit denen HTTP-Operationen zugriffsgeschützt autorisiert werden. APIs Weitere Informationen finden Sie unter Bereiche, M2M und APIs mit Ressourcenservern.

Bei Zuschüssen mit Kundenanmeldedaten fallen zusätzliche Kosten auf Ihre Rechnung an. AWS Weitere Informationen finden Sie unter Amazon Cognito – Preise.

Aktualisierungs-Token

Sie können einen Zuschuss für ein Aktualisierungstoken direkt bei der beantragenToken-Endpunkt. Diese Gewährung gibt neue ID- und Zugriffstoken im Austausch gegen ein gültiges Aktualisierungstoken zurück.

Weitere Informationen zu diesen Zuschüssen und ihrer Implementierung finden Sie im AWS Sicherheitsblog unter So verwenden Sie OAuth 2.0 in Amazon Cognito: Erfahren Sie mehr über die verschiedenen OAuth 2.0-Zuschüsse.

Verwendung von PKCE in Autorisierungscode-Zuschüssen

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 verwaltete Anmeldung zu konfigurieren.
   a. Wenn Sie einen neuen Benutzerpool erstellen, richten Sie während der geführten Einrichtung einen App-Client ein und konfigurieren Sie die verwaltete Anmeldung.
   b. Wenn Sie einen vorhandenen Benutzerpool konfigurieren, fügen Sie eine Domain 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 verwaltete Anmeldung mit einer Anfrage zur Erteilung eines Autorisierungscodes bei PKCE ab. Im Folgenden finden Sie ein Beispiel für eine URL:

```
https://mydomain.auth.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.auth.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 unterJSON-Webtoken (JWTs) für den Benutzerpool verstehen.

Antworten auf verwaltete Anmelde- und Verbundfehler

Ein Anmeldevorgang bei verwalteter Anmeldung oder föderierter Anmeldung 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 bei der verwalteten Anmeldung 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 standardkonformen 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 Fehlermeldungen 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 verifizieren, erkennt Amazon Cognito die JSON-Webschlüssel (JWKs), die Ihr IdP zum Signieren von Tokens 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.

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. Der externe Identitätsanbieter kann ein Consumer (Social) OAuth 2.0-Anbieter wie Apple oder Google, ein benutzerdefinierter SAML- oder OIDC-Identitätsanbieter oder ein benutzerdefiniertes Authentifizierungsschema, auch Entwickler-Anbieter genannt, sein, das Sie selbst entworfen haben.

Eigenschaften von Amazon-Cognito-Identitätspools

Signieren 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 Social-Provider-Token gegen Anmeldeinformationen OAuth 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- und IDC-Anbietern in eine AssumeRoleWithWebIdentityAPI-Anfrage für kurzfristige Anmeldeinformationen an. OAuth

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

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

- Übersicht über die Identity-Pools-Konsole
- Identitäten-Pools – Authentifzierungsablauf
- IAM-Rollen
- Bewährte Sicherheitsmethoden für Amazon Cognito Cognito-Identitätspools
- Verwenden von Attributen für Zugriffskontrolle
- Verwenden der rollenbasierten Zugriffskontrolle
- Abrufen von Anmeldeinformationen
- Zugriff AWS-Services mit temporären Anmeldeinformationen
- Identitätspools, Identitätsanbieter von Drittanbietern
- Entwicklerauthentifizierte Identitäten
- Umschalten von nicht authentifizierten Benutzern auf authentifizierte Benutzer

Übersicht über die Identity-Pools-Konsole

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 Identitätspool ist ein Speicher von Benutzerkennungen, die mit Ihren externen Identitätsanbietern verknüpft sind.

Eine Möglichkeit, die Funktionen und Optionen von Identitätspools zu verstehen, besteht darin, einen in der Amazon Cognito Cognito-Konsole zu erstellen. Sie können die Auswirkungen verschiedener Einstellungen auf Authentifizierungsabläufe, rollen- und attributbasierte Zugriffskontrolle und Gastzugriff untersuchen. Von dort aus können Sie zu späteren Kapiteln in diesem Handbuch übergehen und Ihrer Anwendung die entsprechenden Komponenten hinzufügen, sodass Sie die Identitätspool-Authentifizierung implementieren können.

Themen

- Erstellen eines Identitäten-Pools
- IAM-Rollen von Benutzern

- Authentifizierte und nicht authentifizierte Identitäten
- Gastzugang aktivieren oder deaktivieren
- Ändern der mit einem Identitätstyp verknüpften Rolle
- Identitätsanbieter bearbeiten
- Löschen eines Identitätspools
- Löschen einer Identität aus einem Identitätspool
- Verwenden von Amazon Cognito Sync mit Identitätspools

Erstellen eines Identitäten-Pools

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 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 und direkt IdPs anrufen. AssumeRoleWithWebIdentity Weitere Informationen finden Sie unter Identitäten-Pools – Authentifzierungsablauf.
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 Identitätspools, Identitätsanbieter von Drittanbietern.
- Informationen zum Konfigurieren Ihres eigenen Backend-Authentifizierungsprozesses finden Sie unter Entwicklerauthentifizierte Identitäten.

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 Identitätspools, Identitätsanbieter von Drittanbietern.

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 Implementierung von Amazon Cognito Sync-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 Implementierung von Amazon Cognito Sync-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 Anpassen von Workflows mit Amazon Cognito Events.

Weitere Informationen AWS Lambda dazu finden Sie unter. AWS Lambda

Identitäten-Pools – Authentifzierungsablauf

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

Der erweiterte (vereinfachte) Authentifizierungsablauf

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`

Der grundlegende (klassische) Authentifizierungsablauf

Wenn Sie den Standardauthentifizierungsablauf implementieren, wählt Ihre Anwendung die IAM-Rolle aus, die Benutzer annehmen sollen.

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.

In Benutzerpools, die keine Rollenzuordnungen haben, können Sie sich mit dem Basic-Authflow anmelden. Für diese Art von Identitätspool gibt es keine Standardrolle mit Authentifizierung oder ohne Authentifizierung und keine konfigurierte rollen- oder attributbasierte Zugriffskontrolle. Wenn Sie versuchen`GetOpenIdToken`, einen Identitätspool mit Rollenzuordnungen zu verwenden, wird die folgende Fehlermeldung angezeigt.

```
Basic (classic) flow is not supported with RoleMappings, please use enhanced flow.
```

Reihenfolge der Vorgänge bei der Standardauthentifizierung

1. `GetId`
2. `GetOpenIdToken`
3. `AssumeRoleWithWebIdentity`

Der vom Entwickler authentifizierte Authentifizierungsablauf

Bei der Verwendung verwendet Ihr Client einen anderen AuthflowEntwicklerauthentifizierte Identitäten, der Code außerhalb von Amazon Cognito enthält, um den Benutzer in Ihrem eigenen Authentifizierungssystem zu validieren. Aus der Sicht Ihres Identitätspools handelt es sich bei den Angaben, die Sie in Ihrer Identitätsanfrage angeben, um willkürliche Identifikatoren, und die Authentifizierung wird durch IAM-Anmeldeinformationen autorisiert, die Sie in Ihrer Anwendung codieren.

Reihenfolge der Vorgänge bei der erweiterten Authentifizierung bei einem Entwickler-Anbieter

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 Authentifizierungsablauf sollte ich implementieren?

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 und/oder Status vertrauen. `amr` authenticated/unauthenticated

Machen Sie sich auch darüber im Klaren, dass die Entwicklerauthentifizierung eine Abkürzung zur 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.

Überblick über die API-Operationen des Authentifizierungsflusses

GetId

Der GetId-API-Aufruf ist der erste zur Einrichtung einer neuen Identität in Amazon Cognito erforderliche Aufruf.

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. Das AWS Mobilgerät SDKs und das AWS SDK für JavaScript im Browser verfügen über Anbieter für Anmeldeinformationen, die dieses Caching für Sie übernehmen.

Authentifizierter Zugriff

Wenn Sie Ihre Anwendung mit Unterstützung für einen öffentlichen Login-Anbieter (Facebook, Google+, Login with Amazon oder Sign in with Apple) konfiguriert haben, können Benutzer auch Tokens (OAuth oder OpenID Connect) angeben, die sie bei diesen Anbietern 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 GetCredentialsForIdentity-API kann aufgerufen werden, nachdem eine Identitäts-ID eingerichtet wurde. Dieser Vorgang entspricht also funktionell dem Aufrufen GetOpenIdToken. AssumeRoleWithWebIdentity

Damit Amazon Cognito `AssumeRoleWithWebIdentity` für Sie aufrufen kann, müssen Ihrem Identitäten-Pool IAM-Rollen zugewiesen sein. Sie können dies über die Amazon-Cognito-Konsole oder manuell über die SetIdentityPoolRoles-Operation tun.

GetOpenIdToken

Stellen Sie eine GetOpenIdToken-API-Anfrage, nachdem Sie eine Identitäts-ID eingerichtet haben. Zwischenspeichern Sie die Identität IDs nach Ihrer ersten Anfrage und starten Sie nachfolgende (klassische) Basissitzungen 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 AssumeRoleWithWebIdentity-Anfrage 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 Identität parent/owner aller zugehörigen Logins und die andere ist deaktiviert. In diesem Fall wird die Identitäts-ID von parent/owner zurückgegeben. Sie müssen Ihren lokalen Cache aktualisieren, wenn dieser Wert abweicht. Die Anbieter im AWS Mobilgerät SDKs oder JavaScript im AWS SDK für den Browser führen diesen Vorgang für Sie aus.

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 Entwickleridentität IDs nach Ihrer ersten Anfrage und starten Sie nachfolgende (klassische) Basissitzungen 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

Bei entwicklerauthentifizierten Identitäten unterstützt Amazon Cognito sowohl die implizite Zusammenführung als auch die explizite Zusammenführung über die

MergeDeveloperIdentities-API. 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 folgendermaßen:
Wenn ein Benutzer sich bei einer App anmeldet, generiert Amazon Cognito temporäre AWS -Anmeldeinformationen für ihn. Diese temporären Anmeldeinformationen sind mit einer bestimmten IAM-Rolle verknüpft. Mit der IAM-Rolle können Sie eine Gruppe von Berechtigungen für den Zugriff auf Ihre AWS -Ressourcen definieren.

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
- Zugriffsrichtlinien
- Vertrauensstellungen und Berechtigungen für Rollen

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. Amazon Cognito vermittelt Verbindungen zwischen einem AWS STS Identitätspool IdPs.

Das Token, dem präsentiert AWS STS wird, wird von einem Identitätspool generiert, der einen Benutzerpool, ein Social- oder OIDC-Provider-Token oder eine SAML-Assertion in ein eigenes Token übersetzt. Das Identitätspool-Token enthält einen `aud`-Anspruch, der die Identitätspool-ID ist.

Wenn es sich bei `Principal` der Vertrauensrichtlinie einer IAM-Rolle um einen Dienstprinzipal für Identitätspools handelt`cognito-identity.amazonaws.com`, können Sie keine Rollenvertrauensrichtlinien so erstellen oder ändern, dass sie es jedem Identitätspool ermöglichen, die Rolle zu übernehmen. Beim Identitätspool-Prinzipal muss das `Action` Element über eine `Condition` Funktion verfügen, die nur von Ihren Identitätspools ausgeführt `AssumeRoleWithWebIndentity` werden muss, wie durch einen Bedingungsschlüssel wie `cognito-identity.amazonaws.com:aud` angegeben. Andere Bedingungsschlüssel sind verfügbar, aber `aud` erforderlich. Wenn Sie versuchen, eine Rollenvertrauensrichtlinie ohne eine Bedingung dieses Typs zu speichern, gibt IAM einen Fehler zurück.

Weitere Informationen zu OIDC-Verbundschlüsseln (Web-Identität) finden Sie unter Verfügbare Schlüssel für AWS den OIDC-Verbund.

Im Folgenden sind die OIDC-Verbundbedingungsschlüssel für Amazon Cognito verfügbar.

`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.

Das folgende Beispiel für eine Rollenvertrauensrichtlinie ermöglicht es dem Verbunddienstprinzipal, die API `cognito-identity.amazonaws.com` aufzurufen. 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.

JSON

```
{
    "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)

Übersicht

Identitätspools können im Standardauthentifizierungsablauf nur dann Rollen im Namen von Benutzern übernehmen, wenn die Vertrauensrichtlinie der Zielrolle die `aud` Bedingung enthält.

Bei der Standardauthentifizierung gelten dieselben Einschränkungen für unsichere Richtlinien zur Rollenvertrauensstellung wie für die erweiterte Authentifizierung: Sie können keine Rollenvertrauensrichtlinie speichern, die die unterstützten Identitätspools nicht einschränkt, wenn eine `aud` Bedingung erfüllt ist. Diese Einschränkung wurde beim Start des Dienstes nicht durchgesetzt. Vor der Durchsetzung dieser Anforderung konnten Sie Richtlinien für Rollenvertrauensstellungen erstellen, die keine zusätzlichen Sicherheitsbedingungen enthielten. Nach der Durchsetzung dieser Anforderung können AWS STS Web-Identitäten Rollen annehmen, die nicht durch Bedingungen gesichert sind. Diese Rollen können jedoch nicht geändert werden, ohne dass diese Bedingungen eingeführt werden.

Für die Enhanced-Flow-Authentifizierung muss sich die IAM-Rolle im selben AWS-Konto Identitätspool befinden. Bei der Standardauthentifizierung, bei der Ihre Anwendung die `AssumeRoleWithWebIdentity` Anfrage verfasst, kann Ihre Anwendung jedoch beantragen, eine Rolle in einem anderen Konto zu übernehmen. Ihre kontoübergreifende Anfrage zur Übernahme einer Rolle schlägt jedoch fehl, wenn für die Zielrolle eine veraltete Vertrauensrichtlinie gilt, die diese Bedingung nicht erzwingt. `aud`

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 solche

Bedingung enthalten ist. Wenn es sich bei der Anfrage nicht um eine kontoübergreifende Anfrage handelt, wird diese AWS STS Einschränkung nicht durchgesetzt. Es hat sich bewährt, immer eine Bedingung dieses Typs auf die Vertrauensrichtlinien Ihrer Identitätspool-Rollen anzuwenden.

Im Folgenden finden Sie ein Beispiel für eine Vertrauensrichtlinie, die die Mindestanforderungen für eine IAM-Rolle für die Standardauthentifizierung mit mehreren Identitätspools erfüllt. Es hat sich bewährt, auch nur authentifizierte Identitäten mit einer `"cognito-identity.amazonaws.com:amr": "authenticated"` Bedingung zuzulassen.

```
{
    "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",
                        "us-west-2:hijklmo-5678-9101-112b-0e4221776g96",
                        "us-west-2:pqrstuv-9101-1121-314c-0e2110887h97"
                    ]
                }
            }
        }
    ]
}
```

Zusätzliche Bedingungen für die Vertrauenspolitik

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 Aktionen IDs in Ihren Zugriffsrichtlinien auf die Amazon Cognito Cognito-Identität 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
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 Amazon Location Service
Entwicklung von Spielen	GameLift Amazon-Server
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 KI[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`
	`ReEncryptTo`
	`ReEncryptFrom`
	`GenerateDataKey`
	`GenerateDataKeyPair`
	`GenerateDataKeyPair`
	`GenerateDataKeyPairWithoutPlaintext`
	`GenerateDataKeyWithoutPlaintext`

Amazon SageMaker KI	`InvokeEndpoint`
Amazon Textract	`DetectDocumentText` `AnalyzeDocument`
Amazon Sumerian	`View*`

Amazon Location Service	`SearchPlaceIndex*`
	`GetPlace`
	`CalculateRoute*`
	`*Geofence`
	`*Geofences`
	`*DevicePosition*`

Um Zugriff zu gewähren, der AWS-Services über diese Liste hinausgeht, aktivieren Sie den grundlegenden (klassischen) Authentifizierungsablauf in Ihrem Identitätspool. Wenn Ihre Benutzer aufgrund der Richtlinien AWS-Services , die der IAM-Rolle für nicht authentifizierte Benutzer zugewiesen sind, `NotAuthorizedException` Fehler feststellen, sollten Sie prüfen, ob Sie diesen Dienst aus Ihrem Anwendungsfall entfernen können. Wenn dies nicht möglich ist, wechseln Sie zum Basisablauf.

Die Inline-Sitzungsrichtlinie für Gastbenutzer

Amazon Cognito wendet zunächst eine Inline-Richtlinie in der Anforderung von IAM-Anmeldeinformationen an. Die eingebundene Sitzungsrichtlinie beschränkt die effektiven Berechtigungen Ihres Benutzers auf die AWS-Services in der folgenden Liste. Sie müssen diesen auch AWS-Services in den Richtlinien, die Sie für die IAM-Rolle des Benutzers anwenden, Berechtigungen erteilen. 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. Einen Überblick über die Gesamtwirkung der Inline-Richtlinie und anderer Sitzungsrichtlinien finden Sie unterServices, auf die nicht authentifizierte Benutzer zugreifen können.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "logs:*",
                "dynamodb:*",
                "kinesis:*",
                "mobileanalytics:*",
                "s3:*",
                "ses:*",
                "sns:*",
                "sqs:*",
```

```
                "lambda:*",
                "machinelearning:*",
                "execute-api:*",
                "iot:*",
                "gamelift:*",
                "cognito-identity:*",
                "cognito-idp:*",
                "lex:*",
                "polly:*",
                "comprehend:*",
                "translate:*",
                "transcribe:*",
                "rekognition:*",
                "mobiletargeting:*",
                "firehose:*",
                "appsync:*",
                "personalize:*",
                "sagemaker:InvokeEndpoint",
                "cognito-sync:*",
                "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 für Gäste

Amazon Cognito wendet auch eine AWS verwaltete Richtlinie als Sitzungsrichtlinie auf Enhanced-Flow-Sitzungen von nicht authentifizierten Gästen an. Diese Richtlinie schränkt den Umfang der Berechtigungen nicht authentifizierter Benutzer mit der Richtlinie ein. `AmazonCognitoUnAuthedIdentitiesSessionPolicy`

Sie müssen diese Berechtigung auch in den Richtlinien gewähren, die Sie Ihrer nicht authentifizierten IAM-Rolle zuordnen. Die effektiven Berechtigungen eines Benutzers für eine Sitzung mit angenommener Rolle ergeben sich aus der Schnittmenge der seiner Rolle zugewiesenen IAM-Richtlinien und der Sitzungsrichtlinien. Weitere Informationen finden Sie unter Sitzungsrichtlinien im Benutzerhandbuch zu AWS Identity and Access Management .

Einen Überblick über die Gesamtwirkung dieser AWS verwalteten Richtlinie und anderer Sitzungsrichtlinien finden Sie unter. Services, auf die nicht authentifizierte Benutzer zugreifen können

Die verwaltete `AmazonCognitoUnAuthedIdentitiesSessionPolicy`-Richtlinie enthält die folgenden Berechtigungen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "rum:PutRumEvents",
            "polly:*",
            "comprehend:*",
            "translate:*",
            "transcribe:*",
            "rekognition:*",
            "mobiletargeting:*",
            "firehose:*",
            "personalize:*",
            "sagemaker:InvokeEndpoint",
            "geo:GetMap*",
            "geo:SearchPlaceIndex*",
            "geo:GetPlace",
            "geo:CalculateRoute*",
            "geo:*Geofence",
```

```
            "geo:*Geofences",
            "geo:*DevicePosition*",
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncryptTo",
            "kms:ReEncryptFrom",
            "kms:GenerateDataKey",
            "kms:GenerateDataKeyPair",
            "kms:GenerateDataKeyPairWithoutPlaintext",
            "kms:GenerateDataKeyWithoutPlaintext"
        ],
        "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.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/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:::amzn-s3-demo-bucket/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:::amzn-s3-demo-bucket`.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${cognito-
identity.amazonaws.com:sub}/*"]}}
```

```
    },
    {
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${cognito-
identity.amazonaws.com:sub}/*"]
    }
  ]
}
```

Ein ähnliches Zugriffsmodell wird mit Amazon S3 Access Grants erreicht.

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 .

JSON

```
{
  "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.

JSON

```
{
    "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.

JSON

```
{
    "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.

JSON

```
{
  "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:

JSON

```
{
  "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 des 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
- Bewährte Methoden zur Konfiguration des Identitätspo

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:

JSON

```
{
  "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.

JSON

```
{
  "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 – Authentifzierungsablauf.

Der grundlegende Ablauf kann die clientseitige Logik offenlegen, die für die Rollenauswahl und die Zusammenstellung der AWS STS-API-Anforderung nach 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. 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

- Verwenden von Attributen für die Zugriffskontrolle mit Amazon-Cognito-Identitätspools
- Verwenden von Attributen für die Zugriffskontrollrichtlinie (Beispiel)
- Attribute für die Zugriffskontrolle deaktivieren (Konsole)
- Standard-Anbietermappings

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 Konto AWS
- 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 die Erlaubnis, die Rolle bei der Aktion zu übernehmen AssumeRoleWithWebIdentity. 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 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

JSON

```
{
 "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

JSON

```
{
  "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 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“

Anbieter	Token-Typ	Prinzipal-Tagwerte	Beispiel
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" , "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name“	"auth0|5e28d196f8f55a0eaaa95de3", "user123@gmail.com"
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.AGHNIFJQMFSBG3XCPVB35G6ODER QAA“

Anbieter	Token-Typ	Prinzipal-Tagwerte	Beispiel
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)	„DfwifTtKEX1FiIBRn OTl R0CFK; XGJ5xB8xIR IVCPj Xg W7FXMWC FvNOK9 1y5z1", „1269003884292222976" LIdk JJr gwZkLexo
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:

JSON

```
{
  "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.

JSON

```
{
    "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 `cognito:roles` Anspruch ist eine durch Kommas getrennte Zeichenfolge, die eine Reihe von zulässigen Rollen enthält. 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 einer Rolle im `cognito:roles` Anspruch entspricht. 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`, 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 Mit Benutzerattributen arbeiten.

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.

Um einem Identitätsanbieter in der Amazon Cognito Cognito-Konsole eine regelbasierte Zuordnung hinzuzufügen, fügen Sie einen IdP hinzu oder aktualisieren Sie ihn und wählen Sie unter

Rollenauswahl die Option Rolle mit Regeln auswählen aus. Von dort aus können Sie Regeln hinzufügen, die Anbieteransprüche IAM-Rollen zuordnen.

Sie können die regelbasierte Zuordnung für Identitätsanbieter in der AWS CLI oder API mit dem `RulesConfiguration` Feld des Typs einrichten. RoleMapping Sie können dieses Feld im `RoleMappings` Parameter der SetIdentityPoolRolesAPI angeben.

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 Web-Identitätsanmeldedaten 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 unter Übernehmen der Rolle als Anbieter von Anmeldeinformationen im Referenzhandbuch AWS SDKs und im Tools-Referenzhandbuch.

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 für JavaScript
- Anbieter von Amazon Cognito Cognito-Anmeldeinformationen im AWS SDK für .NET Entwicklerhandbuch
- Geben Sie die Anmeldeinformationen im Entwicklerhandbuch programmgesteuert an AWS SDK für Go
- Geben Sie temporäre Anmeldeinformationen im Code im Entwicklerhandbuch ein AWS SDK for Java 2.x
- assumeRoleWithWebIdentityCredentialProviderAnbieter im AWS SDK für PHP Entwicklerhandbuch
- Übernehmen einer Rolle mit einem Web-Identitätsanbieter in der AWS SDK für Python (Boto3) - Dokumentation
- Geben Sie Ihre Anmeldeinformationen und die Standardregion im AWS SDK für Rust Entwicklerhandbuch an

Die folgenden Abschnitte enthalten Beispielcode aus älteren Versionen 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 eine Meldung aus, NetworkOnMainThreadExceptionwenn Sie im Hauptanwendungsthread ein Netzwerk I/O 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 SDK für .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im SDK für .NET AWS SDK für .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 SDK für .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im SDK für .NET AWS SDK für .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();
```

Zugriff AWS-Services mit temporären Anmeldeinformationen

Das Ergebnis einer erfolgreichen Authentifizierung mit einem Identitätspool ist ein Satz von AWS Anmeldeinformationen. Mit diesen Anmeldeinformationen kann Ihre Anwendung Anfragen an AWS Ressourcen stellen, die mit der IAM-Authentifizierung geschützt sind. Mit den verschiedenen API-Vorgängen AWS SDKs , die Sie Ihren Anwendungen für den Zugriff auf Identitätspools hinzufügen können, können Sie nicht authentifizierte API-Anfragen stellen, die temporäre Anmeldeinformationen erzeugen. Anschließend können Sie weitere SDKs AWS-Services zu Ihrem Client hinzufügen und Anfragen mit diesen temporären Anmeldeinformationen signieren. Die IAM-Berechtigungen, die Ihrer Rolle mit temporären Anmeldeinformationen gewährt wurden, müssen die Vorgänge zulassen, die Sie von anderen Diensten anfordern.

Nachdem Sie Ihren Amazon Cognito Cognito-Anmeldeinformationsanbieter konfiguriert und AWS Anmeldeinformationen abgerufen haben, erstellen Sie einen AWS-Service Client. Im Folgenden finden Sie einige Beispiele aus der AWS SDK-Dokumentation.

AWS SDK-Ressourcen zum Erstellen eines Clients

- AWS Die Client-Konfiguration im AWS SDK für C++ Developer Guide
- Verwenden von AWS SDK für Go V2 mit AWS-Services im AWS SDK für Go Entwicklerhandbuch
- Konfiguration von HTTP-Clients im AWS SDK for Java 2.x Entwicklerhandbuch
- Serviceobjekte im AWS SDK für JavaScript Developer Guide erstellen und aufrufen
- Clients in der AWS SDK für Python (Boto3) Dokumentation erstellen
- Einen Service-Client im AWS SDK für Rust Developer Guide erstellen
- Verwendung von Clients im AWS SDK für 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 SDK für .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im SDK für .NET AWS SDK für .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 SDK für .NET. Informationen zu den ersten Schritten mit Amazon Cognito finden Sie im SDK für .NET AWS SDK für .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.

Identitätspools, Identitätsanbieter von Drittanbietern

Mit Amazon Cognito Cognito-Identitätspools können Sie eine Vielzahl von externen Identitätsanbietern (IdPs) integrieren, um temporäre AWS Anmeldeinformationen über eine Verbundauthentifizierung in Ihrer Anwendung bereitzustellen. Indem Sie Ihren Identitätspool so

konfigurieren, dass er mit diesen externen IdPs Geräten funktioniert, können Sie Ihren Benutzern den Zugriff auf AWS Back-End-Ressourcen mit Authentifizierung durch Amazon Cognito Cognito-Benutzerpools, soziale Anbieter, OIDC-Anbieter oder SAML-Anbieter autorisieren. In diesem Abschnitt werden die Schritte zur Einrichtung und Integration IdPs Ihres Amazon Cognito Cognito-Identitätspools beschrieben.

Mit der `logins`-Eigenschaft können Sie Anmeldeinformationen einrichten, die Sie von einem Identitätsanbieter (IdP) erhalten haben. Sie können einen Identitätspool auch 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
- Login with Amazon als Identitätspools (IdP) einrichten
- Google als Identitätspool-IdP einrichten
- Mit Apple anmelden als Identitätspool-IdP einrichten
- Einrichtung eines OIDC-Anbieters als Identitätspool-IdP
- Einrichtung eines SAML-Anbieters als Identitätspool-IdP

Facebook als Identitätspools einrichten IdP

Amazon Cognito Cognito-Identitätspools arbeiten mit Facebook zusammen, um eine Verbundauthentifizierung für Ihre Anwendungsbenutzer bereitzustellen. 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 interagieren. APIs

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:

> **Note**
>
> Der Amazon Cognito Cognito-Identitätspoolverbund ist nicht mit Facebook Limited Login kompatibel. Weitere Informationen darüber, wie Sie die Facebook-Anmeldung für iOS einrichten, ohne die für die eingeschränkte Anmeldung festgelegten Berechtigungen zu überschreiten, finden Sie unter Facebook-Anmeldung für iOS — Schnellstart bei Meta for Developers.

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 mit der Facebook-Anmeldung finden Sie im Facebook-Leitfaden „Erste Schritte“.

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 `get` an `refresh` oder 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.

> **Note**
>
> Der Amazon Cognito Cognito-Identitätspoolverbund ist nicht mit Facebook Limited Login kompatibel. Weitere Informationen darüber, wie Sie die Facebook-Anmeldung für iOS einrichten, ohne die für die eingeschränkte Anmeldung festgelegten Berechtigungen zu überschreiten, finden Sie unter Facebook-Anmeldung für iOS — Schnellstart bei Meta for Developers.

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

Amazon Cognito Cognito-Identitätspools arbeiten mit Login with Amazon zusammen, um eine Verbundauthentifizierung für Ihre Mobil- und Web-App-Benutzer 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 zur Integration in Ihren Identitätspool einrichten, die Dokumentation „Login with Amazon" suchen und herunterladen SDKs. 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 bei Login with Amazon 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 auf folgende requestDidSucceed Weise an den Amazon Cognito-Anmeldeinformationsanbieter übergeben: AMZNAccess TokenDelegate

```
- (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'
   }
});
```

Google als Identitätspool-IdP einrichten

Amazon Cognito Cognito-Identitätspools arbeiten mit Google zusammen, um eine Verbundauthentifizierung für die Nutzer Ihrer mobilen Anwendungen bereitzustellen. 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 für eine bessere Integration alle erstellten Kunden IDs als zusätzliche Zielgruppenwerte hinzu. 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 und dann den OAuth Zustimmungsbildschirm. 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 Kunden-ID. 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 unter Nutzer mit Anmeldung bei Google authentifizieren in der Google Identity-Dokumentation.

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 Platform erstellt haben. Weitere Informationen finden Sie in der Google Cloud Platform Console-Hilfe unter Einrichtung von OAuth 2.0.
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 für eine bessere Integration alle erstellten Kunden IDs als zusätzliche Zielgruppenwerte hinzu. 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 und dann den OAuth Zustimmungsbildschirm. 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 Kunden-ID. 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 Platform erstellt haben. Weitere Informationen finden Sie in der Google Cloud Platform Console-Hilfe unter Einrichtung von OAuth 2.0.
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 für eine bessere Integration alle erstellten Kunden IDs als zusätzliche Zielgruppenwerte hinzu. 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 und dann den OAuth Zustimmungsbildschirm. 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 Kunden-ID. 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 für eine bessere Integration alle erstellten Kunden IDs als zusätzliche Zielgruppenwerte hinzu. 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 & Dienste und dann den 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 Kunden-ID. 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 Platform erstellt haben. Weitere Informationen finden Sie in der Google Cloud Platform Console-Hilfe unter Einrichtung von OAuth 2.0.

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.
     });
  }
}
```

Mit Apple anmelden als Identitätspool-IdP einrichten

Amazon Cognito Cognito-Identitätspools arbeiten mit Sign in with Apple zusammen, um eine Verbundauthentifizierung 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 im linken Navigationsbereich 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 Identifikatoren das IDsApp-Menü und dann Dienste IDs aus. Klicken Sie auf das Pluszeichen (+).
11. Wählen Sie auf der Seite Neue Kennung registrieren die Option Dienste 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 URLs unter Website das Pluszeichen aus. Geben Sie für Domänen und Subdomänen den Domänennamen Ihrer App ein. Geben Sie als Antwort die Rückruf-URL ein URLs, an die die Autorisierung den Benutzer weiterleitet, nachdem er sich über Sign in with Apple authentifiziert hat.
    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 Ihrer Android-Benutzeroberfläche eine Schaltfläche „Mit Apple anmelden“ hinzuzufügen, folgen Sie in der Apple-Dokumentation den Abschnitt „Mit Apple anmelden“ -Buttons im Internet anzeigen.
- 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 eine Schaltfläche „Mit Apple anmelden“ hinzufügst, erfährst du in der Apple-Dokumentation unter „Mit Apple anmelden“ -Buttons im Internet anzeigen.
- 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.

```
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.
     });
}
```

Einrichtung eines OIDC-Anbieters als Identitätspool-IdP

OpenID Connect ist ein offener Standard für die Authentifizierung, der von einer Reihe von Anmeldungsanbietern unterstützt wird. Mit Amazon Cognito können Sie Identitäten mit OpenID Connect-Anbietern verknüpfen, über die Sie die Konfiguration vornehmen. AWS Identity and Access Management

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

In der Dokumentation Ihres Anbieters finden Sie Informationen dazu, wie Sie sich anmelden und ein ID-Token erhalten.

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 hier (https://tools.ietf.org/html/rfc7523) angegeben, bietet Amazon Cognito eine Übergangszeit von 5 Minuten, um jegliche Zeitversetzung zwischen den Systemen zu bewältigen.

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.

> **Note**
>
> Identitätspools verwalten für einen kurzen Zeitraum einen Cache mit dem OIDC-IdP-Signaturschlüssel. Wenn Ihr Anbieter seinen Signaturschlüssel ändert, gibt Amazon Cognito möglicherweise einen `NoKeyFound` Fehler zurück, bis dieser Cache aktualisiert wird. Wenn dieser Fehler auftritt, warten Sie etwa zehn Minuten, bis Ihr Identitätspool den Signaturschlüssel aktualisiert hat.

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 `azp` Parameter vorhanden ist, überprüfen Sie diesen Wert anhand des aufgelisteten Clients IDs in Ihrem OIDC-Anbieter.
5. Wenn der `azp` Parameter nicht vorhanden ist, vergleichen Sie den `aud` Parameter mit dem aufgelisteten Client IDs in Ihrem OIDC-Anbieter.

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 }
```

JavaScript

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
```

```
  IdentityPoolId: 'IDENTITY_POOL_ID',
  Logins: {
     'login.provider.com': token
  }
 });
```

Einrichtung eines SAML-Anbieters als Identitätspool-IdP

Mit Amazon Cognito Cognito-Identitätspools können Sie Benutzer mit Identitätsanbietern (IdPs) über SAML 2.0 authentifizieren. 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, bestimmt Ihr Identitätspool bei der Anmeldung, dass die SAML-Assertion gültig ist, wenn 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 Amazon Cognito verwenden, APIs um die resultierende SAML-Assertion für Amazon Cognito Identity bereitzustellen.

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 Map akzeptieren können, gehören,, und. `Logins`

GetIdGetCredentialsForIdentityGetOpenIdTokenGetOpenIDTokenForDeveloperIdentity 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.

Entwicklerauthentifizierte Identitäten

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 von Entwicklern authentifizierten Identitäten können Sie Benutzer über Ihren eigenen bestehenden Authentifizierungsprozess registrieren und authentifizieren und gleichzeitig Amazon Cognito verwenden, um Benutzerdaten zu synchronisieren und auf Ressourcen zuzugreifen. AWS Die Verwendung von entwicklerauthentifizierten Identitäten beinhaltet die Interaktion zwischen dem Endbenutzergerät, Ihrem Backend für die Authentifizierung und Amazon Cognito. Weitere Informationen finden Sie im Blog unter Understanding Amazon Cognito Authentication Part 2: Developer Authenticated Identities. AWS

Erläuterungen zum Authentifizierungsfluss

Der GetOpenIdTokenForDeveloperIdentityAPI-Vorgang kann die Entwicklerauthentifizierung sowohl für die erweiterte als auch für die Standardauthentifizierung initiieren. Diese API authentifiziert eine Anfrage mit Administratoranmeldedaten. Bei der `Logins` Map handelt es sich um den Namen eines Entwickler-Providers für Identitätspools, `login.mydevprovider` gepaart mit einer benutzerdefinierten ID.

Beispiel:

```
"Logins": {
        "login.mydevprovider": "my developer identifier"
    }
```

Verbesserte Authentifizierung

Rufen Sie den GetCredentialsForIdentityAPI-Vorgang mit einer `Logins` Map mit dem Namen `cognito-identity.amazonaws.com` und einem Wert des Tokens von aufGetOpenIdTokenForDeveloperIdentity.

Beispiel:

```
"Logins": {
        "cognito-identity.amazonaws.com": "eyJra12345EXAMPLE"
    }
```

`GetCredentialsForIdentity`mit vom Entwickler authentifizierten Identitäten werden temporäre Anmeldeinformationen für die standardmäßig authentifizierte Rolle des Identitätspools zurückgegeben.

Standardauthentifizierung

Rufen Sie den AssumeRoleWithWebIdentityAPI-Vorgang auf und fordern Sie eine beliebige IAM-Rolle an, für die eine entsprechende Vertrauensbeziehung definiert wurde. `RoleArn` Stellen Sie den Wert von `WebIdentityToken` auf das Token ein, von `GetOpenIdTokenForDeveloperIdentity` dem Sie abgerufen wurden.

Informationen zum Authflow für vom Entwickler authentifizierte Identitäten und wie sie sich von Identitäten externer Anbieter unterscheiden, finden Sie unter. Identitäten-Pools – Authentifzierungsablauf

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.

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 GetOpenIdTokenForDeveloperIdentityToken, indem Sie anrufen. Diese API muss von Ihrem Backend aus mit AWS Entwickleranmeldedaten 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 Dauer des Tokens 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 sozialen Identität eines Benutzers zu verbinden (Mit Amazon anmelden, Mit Apple, Facebook oder Google anmelden), fügen Sie das Identitätsanbieter-Token der Login-Map hinzu, wenn Sie

anrufen GetOpenIdTokenForDeveloperIdentity. 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 Login-Map basierend auf dem aktuellen Identity Provider 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 Login-Map basierend auf dem aktuellen Identity Provider 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 Vorgehensweise in Unity besteht darin, Ihren Identitätsanbieter von AmazonCognitoEnhancedIdentityProvide statt von zu erweitern und die übergeordnete RefreshAsync Methode anstelle Ihrer eigenen aufzurufen AbstractCognitoIdentityProvider, falls der Benutzer nicht mit Ihrem eigenen 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.

Umschalten von nicht authentifizierten Benutzern auf authentifizierte Benutzer

Amazon-Cognito-Identitäten-Pools unterstützen authentifizierte und nicht authentifizierte Benutzer. Nicht authentifizierte Benutzer erhalten Zugriff auf Ihre AWS Ressourcen, auch wenn sie bei keinem Ihrer Identitätsanbieter angemeldet sind (). IdPs 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. Irgendwann entscheiden sie sich vielleicht dafür, sich mit einem der unterstützten anzumelden. IdPs 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 könnten sie sich entscheiden, sich mit einer der unterstützten Optionen anzumelden IdPs. 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 könnten sie sich entscheiden, sich mit einer der unterstützten Optionen anzumelden IdPs. 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 zu dem `CognitoIdentityCredentials` Objekt finden Sie unter AWS.CognitoIdentityCredentialsin der AWS SDK für JavaScript API-Referenz.

Unity

Benutzer können sich als nicht authentifizierte Gäste bei Ihrer Anwendung anmelden. Irgendwann könnten sie beschließen, sich mit einem der unterstützten anzumelden IdPs. 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 könnten sie sich entscheiden, sich mit einer der unterstützten Optionen anzumelden IdPs. 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 AWS AppSync ist es ein Dienst zur geräteübergreifenden Synchronisation von Anwendungsdaten.
> 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 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.

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
- Synchronisieren von Daten zwischen Clients
- Bearbeitung von Rückrufen bei Ereignissen
- Push-Synchronisierung implementieren
- Implementierung von Amazon Cognito Sync-Streams
- Anpassen von Workflows mit Amazon Cognito Events

Erste Schritte mit Amazon Cognito Sync

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync AWS AppSync ist es ein Dienst zur geräteübergreifenden Synchronisation von Anwendungsdaten.
> 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, die die geräteübergreifende Synchronisierung anwendungsbezogener Benutzerdaten ermöglicht. 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 zwischen Clients.

Synchronisieren von Daten zwischen Clients

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync AWS AppSync ist es ein Dienst zur geräteübergreifenden Synchronisation von Anwendungsdaten.

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 Anbieter von Anmeldeinformationen erfasst temporäre AWS Anmeldeinformationen, damit Ihre App auf Ihre Ressourcen zugreifen kann. AWS 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 verwenden CocoaPods, ersetzen Sie es `<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 Bearbeitung von Rückrufen bei Ereignissen.

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 Bearbeitung von Rückrufen bei Ereignissen.

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 Bearbeitung von Rückrufen bei Ereignissen.

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 Bearbeitung von Rückrufen bei Ereignissen.

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 Bearbeitung von Rückrufen bei Ereignissen.

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 Bearbeitung von Rückrufen bei Ereignissen.

Bearbeitung von Rückrufen bei Ereignissen

> Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync AWS AppSync ist es ein Dienst zur geräteübergreifenden Synchronisation von Anwendungsdaten.
> 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.

Als Amazon Cognito Sync-Entwickler können Sie verschiedene Callbacks implementieren, um verschiedene Synchronisierungsereignisse und -szenarien zu behandeln. Die `SyncCallback` Schnittstelle im Android-SDK konfiguriert Benachrichtigungen über die Datensatzsynchronisierung, z. B. `onSuccess()` wenn ein Datensatz erfolgreich heruntergeladen wurde, `onFailure()` wenn eine Ausnahme auftritt und `onConflict()` um Konflikte zwischen lokalen und entfernten Daten zu lösen.

Im iOS-SDK können Sie sich für ähnliche Benachrichtigungen registrieren `AWSCognitoDidStartSynchronizeNotification` und Handler wie den

`AWSCognitoRecordConflictHandler` für die Konfliktlösung einrichten. Die Plattformen Unity JavaScript, Xamarin und Unity verfügen über analoge Callback-Mechanismen. Wenn Sie diese Callbacks implementieren, kann Ihre Anwendung die verschiedenen Synchronisierungsereignisse und -szenarien, die bei der Verwendung von Amazon Cognito Sync auftreten können, problemlos 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 enthält `userInfo` den Schlüsseldatensatz, der den Namen des zu synchronisierenden Datensatzes enthält, und die Schlüsselschlüssel, die eine Anzahl NSArray von Datensatzschlüsseln enthalten, die übertragen wurden.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Wird aufgerufen, wenn ein lokaler Wert sich aufgrund eines Synchronisationsvorgangs ändert. Das `userInfo` wird den Schlüsseldatensatz enthalten, der den Namen der zu synchronisierenden Datenmenge enthält, und die Schlüsselschlüssel, die einen Datensatz enthalten, NSArray der sich geändert hat.

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 Implementierung und Zuweisung eines können AWSCognito RecordConflictHandler Sie die standardmäßige Konfliktlösung ändern. Der AWSCognito Konflikt mit den Eingabeparametern Conflict enthält ein AWSCognito Record-Objekt sowohl für die lokal zwischengespeicherten Daten als auch für den widersprüchlichen Datensatz im Synchronisierungsspeicher. Mithilfe von AWSCognito Conflict können Sie den Konflikt mit dem lokalen Datensatz: [Conflict resolveWithLocal Record], dem Remote-Datensatz: [Conflict resolveWithRemote Record] oder einem brandneuen Wert: [resolveWithValueconflict:value] lösen. 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. Der enthält `userInfo` den Schlüsseldatensatz, der den Namen des zu synchronisierenden Datensatzes enthält, und die Schlüsselschlüssel, die eine Anzahl NSArray von Datensatzschlüsseln enthalten, die gedrückt wurden.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Wird aufgerufen, wenn ein lokaler Wert sich aufgrund eines Synchronisationsvorgangs ändert. Das `userInfo` wird den Schlüsseldatensatz enthalten, der den Namen der zu synchronisierenden Datenmenge enthält, und die Schlüsselschlüssel, die einen Datensatz enthalten, NSArray der sich geändert hat.

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. Mit dem können `AWSCognitoConflict` Sie den Konflikt mit dem lokalen Datensatz lösen: [conflict resolveWithLocal Record], dem Remote-Datensatz: [conflict resolveWithRemote Record] oder einem brandneuen Wert: [conflict:value resolveWithValue]. 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 implementieren

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync AWS AppSync ist es ein Dienst zur geräteübergreifenden Synchronisation von Anwendungsdaten.
> 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**
>
> Die Push-Synchronisierung wird für Unity oder JavaScript Xamarin nicht 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 vollen Amazon SNS SNS-Zugriff haben, oder erstellen Sie eine neue Rolle mit vollem Amazon SNS SNS-Zugriff. 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.

JSON

```
{
    "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 BroadcastReceiverAndroid-Objekts können Sie die neueste Version des geänderten Datensatzes überprüfen und entscheiden, ob Ihre App 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. Dies ist für den openOrCreate Dataset-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. Sobald Sie das Geräte-Token als NSData Objekt von erhalten haben APNs, müssen Sie das Gerät mit der `registerDevice:` Methode des Synchronisierungsclients bei Amazon Cognito 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;
      }
    ];
```

Im Debug-Modus registriert sich Ihr Gerät bei der APNs Sandbox; im Release-Modus registriert es sich bei. APNs 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. Sobald Sie das Geräte-Token als NSData Objekt von erhalten haben APNs, müssen Sie das Gerät mit der Methode registerDevice: des Synchronisierungsclients bei Amazon Cognito registrieren, wie unten dargestellt:

```
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
})
```

Im Debug-Modus registriert sich Ihr Gerät bei der APNs Sandbox; im Release-Modus registriert es sich bei. APNs 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.

Implementierung von Amazon Cognito Sync-Streams

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync AWS AppSync ist es ein Dienst zur geräteübergreifenden Synchronisation von Anwendungsdaten.
> 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.

Anpassen von Workflows mit Amazon Cognito Events

> ⚠ Wenn Amazon Cognito Sync für Sie neu ist, verwenden Sie zuerst AWS AppSync.
> Wie Amazon Cognito Sync AWS AppSync ist es ein Dienst zur geräteübergreifenden Synchronisation von Anwendungsdaten.
> 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 Events 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 Ihnen eine Lambda-Funktion zugeordnet ist, Sie diese jedoch `UpdateRecords` mit AWS Kontoanmeldedaten (Entwickleranmeldedaten) aufrufen, wird Ihre Lambda-Funktion nicht aufgerufen.

Eine Funktion erstellen 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. Suchen Sie auf dem Bildschirm „Blueprint auswählen“ nach "“ und wählen Sie es auscognito-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 über die Lambda-Konsole (siehe Verwenden ressourcenbasierter Richtlinien für AWS Lambda) oder mithilfe des Lambda-Vorgangs hinzufügen. 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.

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 währenddessen Ihre Lambda-Funktion aufruft UpdateRecords, muss die Funktion innerhalb von 5 Sekunden antworten. 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);
};
```

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
- Identity and Access Management für Amazon Cognito
- Protokollierung und Überwachung in Amazon Cognito
- Compliance-Validierung für Amazon Cognito
- Ausfallsicherheit bei Amazon Cognito
- Sicherheit der Infrastruktur in Amazon Cognito
- Konfigurations- und Schwachstellenanalyse in Amazon-Cognito-Benutzerpools
- AWS verwaltete Richtlinien für Amazon Cognito

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 AWS gesamte Cloud läuft. 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).
- Wird verwendet SSL/TLS , um mit AWS Ressourcen zu kommunizieren.
- 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 Liste der API-Operationen, gruppiert nach Autorisierungsmodell.

> **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
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- Funktionsweise von der Amazon Cognito mit IAM
- Beispiele für identitätsbasierte Richtlinien für Amazon Cognito
- Fehlerbehebung für Amazon-Cognito-Identität und -Zugriff
- Verwendung von serviceverknüpften Rollen für Amazon Cognito

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 Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anfordern 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. Anschließend müssen Sie Anforderungen an Ihren IAM-Administrator senden, um die Berechtigungen der Servicebenutzer 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 für die Selbstsignierung von Anforderungen finden Sie unter AWS Signature Version 4 für API-Anforderungen im IAM-Benutzerhandbuch.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen bereitstellen. AWS Empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center -Benutzerhandbuch und AWS Multi-Faktor-Authentifizierung (MFA) in IAM im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-

Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen. Verwenden Sie diese nur, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie 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 eine Gruppe benennen IAMAdminsund dieser Gruppe Berechtigungen zur Verwaltung 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 Anwendungsfälle für IAM-Benutzer im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb von Ihnen AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, jedoch nicht mit einer bestimmten Person verknüpft. Um vorübergehend eine IAM-Rolle in der zu übernehmen AWS Management Console, können Sie von einer Benutzer- zu einer IAM-Rolle (Konsole) 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 Methoden für die Übernahme einer Rolle 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 (Verbund) 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 Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

- Serviceübergreifender Zugriff — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Service aufrufen, ist es üblich, dass dieser Service Anwendungen in Amazon ausführt EC2 oder Objekte in Amazon S3 speichert. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Forward Access Sessions (FAS) — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.
  - Dienstbezogene Rolle — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- Auf Amazon ausgeführte Anwendungen EC2 — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2 Instance ausgeführt werden und AWS API-Anfragen stellen AWS CLI . Dies ist dem Speichern von Zugriffsschlüsseln innerhalb der EC2 Instance vorzuziehen. Um einer EC2 Instanz eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instanzprofil, das an die Instanz angehängt ist. Ein Instanzprofil enthält die Rolle und ermöglicht Programmen, die auf der EC2 Instanz ausgeführt werden, temporäre Anmeldeinformationen abzurufen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Verwenden einer

IAM-Rolle, um Berechtigungen für Anwendungen zu gewähren, die auf EC2 Amazon-Instances ausgeführt werden.

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. Die Berechtigungen in den Richtlinien legen fest, ob eine Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen 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 heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

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 Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten 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 Inline-Richtlinie wählen, finden Sie unter Auswählen 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.

Zugriffskontrolllisten () ACLs

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

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

Weitere Richtlinientypen

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

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist 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.

- Dienststeuerungsrichtlinien (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie alle Funktionen in einer Organisation aktivieren, können Sie Richtlinien zur Servicesteuerung (SCPs) auf einige oder alle Ihre Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Root-Benutzer des AWS-Kontos Entitäten. Weitere Informationen zu Organizations und SCPs finden Sie unter Richtlinien zur Servicesteuerung im AWS Organizations Benutzerhandbuch.
- Ressourcenkontrollrichtlinien (RCPs) — RCPs sind JSON-Richtlinien, mit denen Sie die maximal verfügbaren Berechtigungen für Ressourcen in Ihren Konten festlegen können, ohne die IAM-Richtlinien aktualisieren zu müssen, die jeder Ressource zugeordnet sind, deren Eigentümer Sie sind. Das RCP schränkt die Berechtigungen für Ressourcen in Mitgliedskonten ein und kann sich auf die effektiven Berechtigungen für Identitäten auswirken, einschließlich der Root-Benutzer des AWS-Kontos, unabhängig davon, ob sie zu Ihrer Organisation gehören. Weitere Informationen zu Organizations RCPs, einschließlich einer Liste AWS-Services dieser Support-Leistungen RCPs, finden Sie unter Resource Control Policies (RCPs) im AWS Organizations Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im 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 Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten 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 Kontoübergreifender Ressourcenzugriff in IAM 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 Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

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

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

Eine Liste der 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"
         ]
```

Signiert oder unsigniert APIs

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 es sich bei Amazon Cognito um ein Produkt zur Kundenidentität für Apps handelt, die Sie der Öffentlichkeit zugänglich machen möchten, haben Sie Zugriff auf die folgenden APIs unsignierten Dateien. Ihre App nimmt diese API-Anforderungen für Ihre Benutzer und Ihre potenziellen Benutzer vor. Bei einigen APIs ist keine vorherige Autorisierung erforderlich, z. B. `InitiateAuth` um eine neue Authentifizierungssitzung zu starten. Einige APIs verwenden Zugriffstoken oder Sitzungsschlüssel für die Autorisierung, z. B. `VerifySoftwareToken` um die MFA-Setup für einen Benutzer abzuschließen, der bereits über eine authentifizierte Sitzung verfügt. 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 Liste der API-Operationen, gruppiert nach Autorisierungsmodell.

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 heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

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

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

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

Amazon-Ressourcennamen (ARNs)

ARNs für Amazon Cognito Federated Identities

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 finden Sie unter ARNs IAM-Identifikatoren.

```
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 diesen Betrieb auf einem Identitätspool ist das ARN-Format des Identitätspools dasselbe wie für Amazon Cognito Federated Identities, außer dass der Dienstname anstelle von: `cognito-sync` `cognito-identity`

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

APIs Dazu können Sie mit einer einzigen Identität arbeiten, z. B. `RegisterDevice` können Sie mit dem folgenden ARN-Format auf die individuelle Identität verweisen:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/
identity/IDENTITY_ID
```

Um mit Datensätzen wie `UpdateRecords` und `ListRecords` zu arbeiten, können Sie mit dem folgenden ARN-Format auf den einzelnen Datensatz verweisen: APIs

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/
identity/IDENTITY_ID/dataset/DATASET_NAME
```

ARNs für Amazon Cognito 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 Eigenschaften finden Sie unter Von Amazon Cognito definierte Ressourcen in der Service Authorization Reference. 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 Richtlinienbedingungsschlüssel: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was 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.

Zugriffskontrolllisten (ACLs) in Amazon Cognito

Unterstützt ACLs: Nein

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

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 Definieren von Berechtigungen mit ABAC-Autorisierung 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 von einer Benutzerrolle zu einer IAM-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 implementieren.

> **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 serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

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 (Konsole) im IAM-Benutzerhandbuch.

Einzelheiten zu den von Amazon Cognito definierten Aktionen und Ressourcentypen, einschließlich des Formats ARNs für die einzelnen Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Cognito in der Service Authorization Reference.

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:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um Ihren Benutzern und Workloads zunächst Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden

Sie 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 mit 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 Sicherer API-Zugriff mit MFA 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

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:ListIdentityPools"
```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:*"
            ],
            "Resource": "arn:aws:cognito-identity:us-
east-1:111122223333:identitypool/us-east-1:1a1a1a1a-ffff-1111-9999-12345678"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cognito-sync:*"
            ],
            "Resource": "arn:aws:cognito-sync:us-
east-1:111122223333:identitypool/us-east-1:1a1a1a1a-ffff-1111-9999-12345678"
        }
    ]
}
```

Erteilen von Zugriff auf einen bestimmten Datensatz für alle Identitäten in einem Pool

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-sync:ListRecords",
                "cognito-sync:UpdateRecords"
            ],
            "Resource": "arn:aws:cognito-sync:us-
east-1:111122223333: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 den Zugriff auf Amazon Cognito zu ermöglichen, müssen Sie den Personen oder Anwendungen, die Zugriff benötigen, die Erlaubnis erteilen. Wenn Sie Personen und Anwendungen verwalten, weisen Sie Benutzern oder Gruppen Berechtigungssätze zu, um deren Zugriffsebene zu definieren. AWS IAM Identity Center Mit Berechtigungssätzen werden automatisch IAM-Richtlinien erstellt und den IAM-Rollen zugewiesen, die der Person oder Anwendung zugeordnet sind. Weitere Informationen finden Sie im AWS IAM Identity Center Benutzerhandbuch unter Berechtigungssätze.

Wenn Sie IAM Identity Center nicht verwenden, müssen Sie IAM-Entitäten (Benutzer oder Rollen) für die Personen oder Anwendungen erstellen, die Zugriff benötigen. Anschließend müssen Sie der Entität eine Richtlinie anfügen, die dieser die korrekten Berechtigungen in Amazon Cognito gewährt. Nachdem die Berechtigungen erteilt wurden, stellen Sie dem Benutzer oder Anwendungsentwickler die Anmeldeinformationen zur Verfügung. Sie werden diese Anmeldeinformationen für den Zugriff verwenden AWS. Weitere Informationen zum Erstellen von IAM-Benutzern, -Gruppen, -Richtlinien und -Berechtigungen finden Sie im IAM-Benutzerhandbuch unter IAM-Identitäten sowie Richtlinien und Berechtigungen in IAM.

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. Für Dienste, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (ACLs) unterstützen, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen 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 in AWS-Konten Ihrem Besitz gewähren können, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen AWS-Konto , dem Sie gehören.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM 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: AWSService RoleForAmazonCognitoIdpEmailService

- 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 AWSService RoleForAmazonCognitoIdp 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 AWSService RoleForAmazonCognitoIdp

- 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 mit dem AmazonCognitoIdpEmailService Dienst verknüpften 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 und Verwaltung der Kosten
- Exportieren von Protokollen aus Amazon Cognito Cognito-Benutzerpools
- Kontingente und Nutzung in CloudWatch und Service Quotas verfolgen
- Amazon Cognito anmelden AWS CloudTrail

Überwachung und Verwaltung der Kosten

Wie bei allen anderen ist es wichtig AWS-Service, die Auswirkungen Ihrer Amazon Cognito Cognito-Konfiguration und -Nutzung auf Ihre AWS Rechnung zu verstehen. Im Rahmen Ihrer Vorbereitungen für die Bereitstellung von Benutzerpools in der Produktion sollten Sie Überwachungs- und Schutzmaßnahmen für Aktivitäten und Ressourcenverbrauch einrichten. Wenn Sie wissen, wo Sie suchen müssen und welche Maßnahmen zusätzliche Kosten verursachen, können Sie Vorkehrungen treffen, um zu verhindern, dass Ihre Rechnung unerwartet wird.

Amazon Cognito berechnet Gebühren für die folgenden Dimensionen Ihrer Nutzung.

- Benutzerpool monatlich aktive Nutzer (MAUs) — Die Rate variiert je nach Funktionsplan
- Der Benutzerpool ist mit einem OIDC- oder SAML-Verbund MAUs angemeldet
- Aktive Benutzer bündeln App-Clients und fordern Volumen für die Machine-to-Machine-Autorisierung (M2M) an, wobei die Client-Anmeldeinformationen gewährt werden
- Gekaufte Nutzung über den Standardkontingenten für einige Kategorien von Benutzerpools 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

Großveranstaltungen wie Produkteinführungen und die Öffnung neuer Nutzer können die Anzahl Ihrer MAUs erhöhen und sich auf die Kosten auswirken. Schätzen Sie die Anzahl neuer Benutzer im Voraus ab und beobachten Sie die Aktivitäten, während sie passieren. Möglicherweise möchten Sie das Volumen durch den Kauf zusätzlicher Kontingentkapazität anpassen oder das Volumen mit zusätzlichen Sicherheitsmaßnahmen kontrollieren.

In der AWS Fakturierung und Kostenmanagement 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. Anfragen zu Kundenanmeldedaten werden beispielsweise 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 bald ablaufen.

Es hat sich bewährt, Token nach etwa 75% der Token-Lebensdauer zu aktualisieren. Diese Vorgehensweise maximiert die Tokendauer und gewährleistet gleichzeitig die Benutzerkontinuität in Ihrer Anwendung.

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 Server mit einer längeren Lebensdauer APIs und Servern, die die Häufigkeit von Anfragen nach Anmeldeinformationen dauerhaft verwalten können.

ListUsers, nicht AdminGetUser

Verwenden Sie, wenn möglich, die ListUsersAPI-Operation und die zugehörigen SDK-Methoden, um die Attribute von Benutzern in Ihrem Benutzerpool abzufragen. AdminGetUsermarkiert einen Benutzer für den Monat als aktiv und trägt zur Anzahl der monatlich aktiven Benutzer (MAUs) bei, anhand derer Ihre Rechnung für Benutzerpools berechnet wird.

Löschen Sie ungenutzte App-Clients mit Kundenanmeldedaten

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 unterAnwendungsspezifische Einstellungen mit App-Clients.

Funktionspläne verwalten

Wenn Sie einen Funktionsplan in einem Benutzerpool auswählen, gilt der Abrechnungstarif für alle MAUs Mitglieder des Benutzerpools. Wenn Sie Benutzer haben, die keine Funktionen benötigen, die in einem übergeordneten Funktionsplan enthalten sind, teilen Sie sie in einen anderen Benutzerpool auf.

Exportieren von Protokollen aus Amazon Cognito Cognito-Benutzerpools

Sie können Ihren Benutzerpool so konfigurieren, dass detaillierte Protokolle zusätzlicher Aktivitäten an einen anderen Benutzer gesendet werden AWS-Service, z. B. an eine CloudWatch Protokollgruppe. Diese Protokolle haben eine feinere Granularität als die darin enthaltenen und können nützlich sein AWS CloudTrail, um Fehler in Ihrem Benutzerpool zu beheben und Benutzeranmeldeaktivitäten mit Bedrohungsschutz zu analysieren. Wenn Sie Protokolle

von SMS- und E-Mail-Benachrichtigungsfehlern streamen möchten, sendet Ihr Benutzerpool Protokolle `ERROR` auf Ebene 1 an eine Protokollgruppe. CloudWatch Wenn Sie Protokolle der Benutzeranmeldeaktivitäten streamen möchten, sendet Ihr Benutzerpool Protokolle `INFO` auf der Ebene von Benutzern an eine Protokollgruppe, einen Amazon Data Firehose-Stream oder einen Amazon S3 S3-Bucket. Sie können beide Optionen in einem Benutzerpool kombinieren.

Themen

- Wissenswertes über den Log-Export
- Fehler beim Exportieren von E-Mail- und SMS-Nachrichten
- Benutzeraktivitätsprotokolle zum Schutz vor Bedrohungen exportieren

Wissenswertes über den Log-Export

Auswirkung auf die Kosten

Für Amazon Data Firehose, Amazon S3 und CloudWatch Logs fallen Kosten für die Datenaufnahme und den Datenabruf an. Ihre Protokollierungskonfiguration könnte sich auf Ihre Rechnung auswirken. AWS Weitere Informationen finden Sie hier:

- Verkaufte Logs zu CloudWatch Amazon-Preisen.
- Preise für Amazon Data Firehose
- Amazon S3 S3-Preise

Die Exporte von Benutzeraktivitätsprotokollen enthalten Sicherheitsbeurteilungen und sind Teil des Bedrohungsschutzes für den Benutzerpool. Amazon Cognito generiert diese Protokolle nur, wenn sich der Bedrohungsschutz im Modus Nur Prüfung oder Vollfunktionsmodus befindet und Ihr Benutzerpool im Plus-Funktionsplan ist.

Die Benutzeraktivitätsprotokolle entsprechen der Stufe **`INFO`**

Exportierte Benutzeraktivitätsprotokolle weisen nur die `INFO` Fehlerebene auf und bieten Informationen für statistische und Sicherheitsanalysen der Authentifizierungsaktivitäten. Meldungen auf der `ERROR` Fehlerebene `WARNING` und, z. B. Drosselungsfehler, sind nicht in den exportierten Protokollen enthalten.

Lieferung nach bestem Wissen

Die Bereitstellung von Protokollen von Amazon Cognito ist das optimale Szenario. Die Menge der Protokolle, die Ihr Benutzerpool liefert, und Ihre Servicekontingente für CloudWatch Logs, Amazon S3 und Firehose können sich auf die Übermittlung von Protokollen auswirken.

Bestehende externe Protokolle sind davon nicht betroffen

Diese Protokollierungsoptionen ersetzen oder ändern nicht die folgenden Protokollfunktionen von Benutzerpools.

1. CloudTrail Protokolle von routinemäßigen Benutzeraktivitäten wie Registrierung und Anmeldung.
2. Analyse der Benutzeraktivitäten im großen Maßstab mit CloudWatch Metriken.

Unabhängig davon finden Sie auch Protokolle von Die Importergebnisse des Benutzerpools in der CloudWatch Konsole anzeigen und Anpassen von Benutzerpool-Workflows mit Lambda-Auslösern in CloudWatch Protokollen. Amazon Cognito und Lambda speichern diese Protokolle in anderen Protokollgruppen als denen, die Sie für Benutzeraktivitätsprotokolle angeben.

Gilt nur für Benutzerpools

Für Identitätspools gibt es keine Protokollexportfunktionen.

Erfordert Benutzerberechtigungen und eine dienstbezogene Rolle

Der AWS Prinzipal, der den Protokollexport einrichtet, muss über Berechtigungen zum Ändern der Zielressourcen verfügen, wie in den folgenden Themen beschrieben. Amazon Cognito erstellt in Ihrem Namen eine serviceverknüpfte Rolle und übernimmt die Rolle, Protokolle an die Zielressource zu liefern.

Weitere Informationen zum Autorisierungsmodell für das Senden von Protokollen von Amazon Cognito finden Sie unter Protokollierung aktivieren AWS-Services im Amazon CloudWatch Logs-Benutzerhandbuch.

Die Protokollebene ist ausschließlich vom Protokolltyp abhängig

Bei der Nachrichtenzustellung handelt es sich um Protokolle des `userNotification` Typs und der `ERROR` Fehlerstufe. Benutzeraktivitätsprotokolle für erweiterte Sicherheitsanforderungen entsprechen dem `userAuthEvents` Typ und der Fehlerstufe. `INFO` Sie können zwei Mitglieder von kombinieren`LogConfigurations`, eines für `userNotification` to CloudWatch Logs und eines für `userAuthEvents` to Firehose, Amazon S3 oder CloudWatch Logs.

Sie können Benutzeraktivitätsprotokolle nicht an mehrere Ziele senden. Sie können Benutzerbenachrichtigungsprotokolle nur an Logs senden. CloudWatch

Verschiedene Konfigurationsoptionen

Sie können Benutzerbenachrichtigungsprotokolle nur mit der Amazon Cognito Cognito-Benutzerpools-API oder einem AWS SDK konfigurieren. Sie können erweiterte

Sicherheitsprotokolle für Benutzeraktivitäten mit der API oder in der Amazon Cognito Cognito-Konsole konfigurieren. Um beide festzulegen, verwenden Sie die API, wie in der Beispielanforderung unter gezeigt. SetLogDeliveryConfiguration

Bei großen ressourcenbasierten Richtlinien ist eine zusätzliche Konfiguration erforderlich

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.

Automatische Erstellung eines Ordners in Amazon S3

Wenn Sie den Export von Bedrohungsschutzprotokollen in einen Amazon S3 S3-Bucket konfigurieren, erstellt Amazon Cognito möglicherweise einen `AWSLogs` Ordner in Ihrem Bucket. Dieser Ordner wird nicht in allen Fällen erstellt, und die Konfiguration kann erfolgreich sein, ohne ihn zu erstellen.

Fehler beim Exportieren von E-Mail- und SMS-Nachrichten

Bei Fehlern bei der Zustellung von E-Mail- und SMS-Nachrichten können Sie Benutzerbenachrichtigungsprotokolle auf Fehlerebene aus Ihrem Benutzerpool bereitstellen. Wenn Sie diese Funktion aktivieren, können Sie die Protokollgruppe auswählen, an die Amazon Cognito Protokolle senden soll. Die Protokollierung von Benutzerbenachrichtigungen ist nützlich, wenn Sie den Status von E-Mail- und SMS-Nachrichten ermitteln möchten, die Ihr Benutzerpool mit Amazon SNS und Amazon SES zugestellt hat. Für diese Option zum Exportieren von Protokollen ist im Gegensatz zum Export von Benutzeraktivitäten der Plus-Funktionsplan nicht erforderlich.

Sie können detaillierte Benachrichtigungsprotokolle mit der Amazon Cognito Cognito-Benutzerpools-API in einer SetLogDeliveryConfigurationAPI-Anfrage konfigurieren. Sie können die Protokollierungskonfiguration eines Benutzerpools in einer GetLogDeliveryConfigurationAPI-Anfrage einsehen. Im Folgenden finden Sie ein Beispiel für einen Anfragetext.

```
{
   "LogConfigurations": [
      {
         "CloudWatchLogsConfiguration": {
            "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-group:example-user-
pool-exported"
         },
         "EventSource": "userNotification",
```

```
        "LogLevel": "ERROR"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Sie müssen diese Anfragen mit AWS Anmeldeinformationen autorisieren, die über die folgenden Berechtigungen verfügen.

JSON

```
{
    "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"
    }
}
```

Benutzeraktivitätsprotokolle zum Schutz vor Bedrohungen exportieren

Benutzerpools mit dem Plus-Funktionsplan und dem Bedrohungsschutz protokollieren Benutzeraktivitätsereignisse: Einzelheiten und Sicherheitsbeurteilungen von Benutzeranmeldungen, Abmeldungen und anderen Authentifizierungsvorgängen mit Ihrem Benutzerpool. Möglicherweise möchten Sie die Benutzeraktivitätsprotokolle in Ihrem eigenen Protokollverwaltungssystem überprüfen oder ein Archiv erstellen. Sie können diese Daten in eine Amazon CloudWatch Logs-Protokollgruppe, einen Amazon Data Firehose-Stream oder einen Amazon Simple Storage Service (Amazon S3) -Bucket exportieren. Von dort aus können Sie diese Daten in andere Systeme aufnehmen, die Daten so analysieren, normalisieren oder anderweitig verarbeiten, dass sie zu Ihren betrieblichen Abläufen passen. Um Daten dieses Typs zu exportieren, muss Ihr Benutzerpool den Plus-Funktionsplan nutzen und in Ihrem Benutzerpool muss der Bedrohungsschutz aktiv sein.

Anhand der Informationen in diesen Benutzeraktivitätsprotokollen können Sie sich ein Profil der Benutzeranmelde- und Kontoverwaltungsaktivitäten anzeigen lassen. Standardmäßig erfasst Amazon Cognito diese Ereignisse in einem Speicher, der in Ihrem Benutzerpool basiert. Das folgende Beispiel ist ein Beispielereignis für einen Benutzer, der sich angemeldet hat und bei dem festgestellt wurde, dass er keine Risikofaktoren aufweist. Sie können diese Informationen mit der `AdminListUserAuthEvents` API-Operation abrufen. Im Folgenden finden Sie ein Beispiel für eine Ausgabe:

```
{
    "AuthEvents": [
        {
            "EventId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "EventType": "SignIn",
            "CreationDate": "2024-06-27T10:49:59.139000-07:00",
            "EventResponse": "Pass",
            "EventRisk": {
                "RiskDecision": "NoRisk",
                "CompromisedCredentialsDetected": false
            },
            "ChallengeResponses": [
                {
                    "ChallengeName": "Password",
                    "ChallengeResponse": "Success"
                }
            ],
            "EventContextData": {
                "IpAddress": "192.0.2.1",
                "DeviceName": "Chrome 126, Windows 10",
                "Timezone": "-07:00",
                "City": "null",
                "Country": "United States"
            }
        }
    ],
    "NextToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222#2024-06-27T17:49:59.139Z"
}
```

Sie können den Protokollexport für Benutzeraktivitäten in der Amazon Cognito Cognito-Konsole oder mit dem SetLogDeliveryConfigurationAPI-Vorgang aktivieren.

AWS Management Console

1. Wenn Sie noch keinen haben, den Sie verwenden möchten, erstellen Sie einen S3-Bucket, einen Firehose-Stream oder eine CloudWatchProtokollgruppe.
2. Melden Sie sich bei der Amazon Cognito-Konsole an.
3. Wählen Sie User Pools (Benutzerpools) aus.
4. Wählen Sie einen vorhandenen Benutzerpool aus der Liste aus oder erstellen Sie einen neuen Benutzerpool.
5. Wählen Sie die Registerkarte Erweiterte Sicherheit. Suchen Sie nach Benutzeraktivitätsprotokolle exportieren und wählen Sie Bearbeiten
6. Markieren Sie unter Protokollierungsstatus das Kontrollkästchen neben Export von Benutzeraktivitätsprotokollen aktivieren.
7. Wählen Sie unter Logging-Ziel AWS-Service das aus, für das Sie Ihre Logs verwalten möchten: CloudWatch Log-Gruppe, Amazon Data Firehose-Stream oder S3-Bucket.
8. Ihre Auswahl füllt die Ressourcenauswahl mit dem entsprechenden Ressourcentyp. Wählen Sie eine Protokollgruppe, einen Stream oder einen Bucket aus der Liste aus. Sie können auch auf die Schaltfläche Erstellen klicken, um AWS Management Console zum ausgewählten Service zu navigieren und eine neue Ressource zu erstellen.
9. Wählen Sie Änderungen speichern aus.

API

Wählen Sie einen Zieltyp für Ihre Benutzeraktivitätsprotokolle.

Im Folgenden finden Sie ein Beispiel für einen `SetLogDeliveryConfiguration` Anforderungstext, der einen Firehose-Stream als Protokollziel festlegt.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "FirehoseConfiguration": {
            "StreamArn": "arn:aws:firehose:us-west-2:123456789012:deliverystream/
example-user-pool-activity-exported"
         },
         "LogLevel": "INFO"
      }
```

```
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Im Folgenden finden Sie ein Beispiel für einen `SetLogDeliveryConfiguration` Anfragetext, der einen Amazon S3 S3-Bucket als Protokollziel festlegt.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "S3Configuration": {
            "BucketArn": "arn:aws:s3:::amzn-s3-demo-logging-bucket"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Im Folgenden finden Sie ein Beispiel für einen `SetLogDeliveryConfiguration` Anfragetext, der eine CloudWatch Protokollgruppe als Protokollziel festlegt.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "CloudWatchLogsConfiguration": {
            "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-group:DOC-
EXAMPLE-LOG-GROUP"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Der Benutzer, der die Protokollzustellung konfiguriert, muss ein Benutzerpooladministrator sein und über die folgenden zusätzlichen Berechtigungen verfügen:

Amazon S3

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageUserPoolLogs",
            "Action": [
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "ManageLogsS3",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "s3:PutBucketPolicy",
                "s3:GetBucketPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

CloudWatch Logs

JSON

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

```
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "ManageLogsCWL",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Amazon Data Firehose

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageUserPoolLogs",
            "Action": [
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration"
            ],
            "Resource": [
```

```
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "ManageUserPoolLogsFirehose",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "iam:CreateServiceLinkedRole",
                "firehose:TagDeliveryStream"
            ],
            "Resource": "*"
        }
    ]
}
```

Es folgt ein Beispielereignis in einem Benutzerpool. Dieses Protokollschema kann sich ändern. Einige Felder werden möglicherweise mit Nullwerten protokolliert.

```
{
    "eventTimestamp": "1687297330677",
    "eventSource": "USER_ACTIVITY",
    "logLevel": "INFO",
    "message": {
        "version": "1",
        "eventId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "eventType": "SignUp",
        "userSub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "userName": "test-user",
        "userPoolId": "us-west-2_EXAMPLE",
        "clientId": "1example23456789",
        "creationDate": "Wed Jul 17 17:25:55 UTC 2024",
        "eventResponse": "InProgress",
        "riskLevel": "",
        "riskDecision": "PASS",
        "challenges": [],
        "deviceName": "Other, Other",
        "ipAddress": "192.0.2.1",
        "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
        "idpName": "",
        "compromisedCredentialDetected": "false",
```

```
        "city": "Seattle",
        "country": "United States",
        "eventFeedbackValue": "",
        "eventFeedbackDate": "",
        "eventFeedbackProvider": "",
        "hasContextData": "true"
    },
    "logSourceId": {
        "userPoolId": "us-west-2_EXAMPLE"
    }
}
```

Kontingente und Nutzung in CloudWatch und Service Quotas verfolgen

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 Kontoübergreifende Beobachtbarkeit von CloudWatch .

Themen

- Benutzerpool-Metriken in CloudWatch
- Metriken in Service Quotas

Benutzerpool-Metriken in CloudWatch

Benutzerpools melden Statistiken zur Benutzeraktivität in Form von CloudWatch Metriken. Von CloudWatch dort aus können Sie das Volumen der Authentifizierungsaktivitäten und die Kontingentnutzung in Ihren Benutzerpools analysieren. Anhand der Informationen in diesen Kennzahlen können Sie Alarme für wichtige Ereignisse einrichten und die Konfiguration Ihres Benutzerpools nach Bedarf anpassen. Während die Protokollierung der Benutzeraktivitäten detaillierte Aufzeichnungen der Benutzeraktivitäten in Ihren Benutzerpools enthält, enthalten CloudWatch Metriken aggregierte Statistiken und Leistungsindikatoren.

In der folgenden Tabelle sind die Metriken aufgeführt, die für Amazon-Cognito-Benutzerpools verfügbar sind. Amazon Cognito veröffentlicht Metriken in den `AWS/Cognito` Namespaces und. `AWS/Usage` Weitere Informationen finden Sie unter Namespaces im CloudWatch Amazon-Benutzerhandbuch.

Weitere Informationen zur Verfolgung von Kontingenten und Nutzung finden Sie unter und. Verfolgen der Kontingentnutzung Verfolgen Sie die monatlich aktiven Benutzer (MAUs)

> **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	Namespace
`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 `SampleCount` 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	`AWS/Cognito`

Metrik	Beschreibung	Namespace
	CloudWatch `Math` Ausdruck und subtrahieren Sie die `Sum` Statistik von der `Sample Count` Statistik. 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 (Benutzer migration) nicht ausgegeben wird. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl	

Metrik	Beschreibung	Namespace
`SignUpThrottles`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten gedrosselten Benutzerr egistrierungsanforderungen an. Sobald eine Benutzerr egistrierungsanforderung gedrosselt wird, wird der Wert 1 veröffentlicht. Verwenden Sie den statistis chen Wert `Sum` für diese Metrik, um die Gesamtzahl der gedrosselten Benutzerr egistrierungsanforderungen zu ermitteln. Diese Metrik wird für jeden Benutzerpool für jeden Client veröffentlicht. Falls die Anforderung, die gedrosselt wurde, von einem Administr ator stammt, wird die Metrik mit dem Benutzerpool-Client als veröffentlicht `Admin`. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl	`AWS/Cognito`

Metrik	Beschreibung	Namespace
`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	`AWS/Cognito`

Metrik	Beschreibung	Namespace
	. Um die Gesamtzahl der fehlgeschlagenen Benutzera uthentifizierungsanfragen 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-Clien t mit einer Anfrage versorgt wird, enthält der entsprech ende Benutzerpool-Client-Wert in der Metrik einen festen Wert `Invalid` anstelle des tatsächlich in der Anfrage gesendeten ungültigen Werts. Beachten Sie, dass Anforderu ngen 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	Namespace
`SignInThrottles`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichteten gedrosselten Benutzera uthentifizierungsanforderun gen an. Der Wert 1 wird veröffentlicht, wann immer eine Authentifizierungs anforderung gedrosselt wird. Verwenden Sie den statistis chen Wert `Sum` für diese Metrik, um die Gesamtzahl der gedrosselten Benutzera uthentifizierungsanforderun gen zu ermitteln. Diese Metrik wird für jeden Benutzerpool für jeden Client veröffentlicht. Falls ein ungültiger Benutzerpool-Clien t mit einer Anfrage versorgt wird, enthält der entsprech ende Benutzerpool-Client-Wert in der Metrik einen festen Wert `Invalid` anstelle des tatsächlich in der Anfrage gesendeten ungültigen Werts. Anforderungen zum Aktualisi eren des Amazon-Cognito-Tokens sind in dieser Metrik nicht enthalten. Es gibt eine separate Metrik mit dem	`AWS/Cognito`

Metrik	Beschreibung	Namespace
	statistischen Wert für die Token-`Refresh`. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl	

Metrik	Beschreibung	Namespace
`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	`AWS/Cognito`

Metrik	Beschreibung	Namespace
	. 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. `Sample Count` 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`. Metrikdimension: `UserPool`, `UserPoolClient` Einheiten: Anzahl	

Metrik	Beschreibung	Namespace
`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`	`AWS/Cognito`

Metrik	Beschreibung	Namespace
	Einheiten: Anzahl	

Metrik	Beschreibung	Namespace
`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	`AWS/Cognito`

Metrik	Beschreibung	Namespace
	. 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	Namespace
`FederationThrottles`	Gibt die Gesamtzahl der an den Amazon-Cognito-Benutzerpool gerichtet en gedrosselten Identität sverbundanforderungen an. Sobald eine Identität sverbundanforderung gedrosselt wird, wird der Wert 1 veröffentlicht. Verwenden Sie den statistis chen Wert `Sum` für diese Metrik, um die Gesamtzah l der gedrosselten Identität sverbündungsanforderungen zu ermitteln. Metrikdimension: `UserPool`, `UserPoolClient` , `IdentityProvider` Einheiten: Anzahl	`AWS/Cognito`

Metrik	Beschreibung	Namespace
`CallCount`	Gibt die Gesamtzahl der Anrufe an, die Kunden in Bezug auf eine Kategorie getätigt haben. Diese Metrik umfasst alle Anrufe, wie gedrosselte Anrufe, fehlgesch lagene Anrufe und erfolgreiche Anrufe. Das Kategorienkontingent wird für jedes AWS Konto in allen Benutzerpools eines Kontos und einer Region durchgese tzt. 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	`AWS/Usage`

Metrik	Beschreibung	Namespace
`ThrottleCount`	Gibt die Gesamtzahl der gedrosselten Anrufe in Bezug auf eine Kategorie an. Diese Metrik wird auf Kontoebene veröffentlicht. 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	`AWS/Usage`

Kennzahlen zum Schutz vor Bedrohungen anzeigen

Die von Ihrem Benutzerpool veröffentlichten Kennzahlen enthalten statistische Informationen darüber, wie sich Ihre Einstellungen für den Schutz vor Bedrohungen auf die Benutzerauthentifizierung auswirken. Möglicherweise möchten Sie wissen, wie viele Benutzer versuchen, sich mit kompromittierten Anmeldeinformationen anzumelden. Sie können auch herausfinden, bei welchem Prozentsatz der Anmeldeaktivitäten ein gewisses Risiko festgestellt wurde. Amazon Cognito veröffentlicht Kennzahlen für Bedrohungsschutzfunktionen in Ihrem Konto bei Amazon CloudWatch. Amazon Cognito gruppiert die Bedrohungsschutzkennzahlen nach Risikostufe und auch nach Anforderungsstufe.

Um Ihrer Risikoanalyse mehr Kontext zu verleihen, können Sie Informationen zu einzelnen Benutzeranmeldeversuchen entweder in Ihrem Benutzerpool oder in einer exportierten Datenquelle einsehen.

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	Namespace
CompromisedCredentialRisk	Anforderungen, bei denen Amazon Cognito kompromittierte Anmeldeinformationen erkannt hat	Vorgang: Die Arten des Vorgangs `PasswordChange` , `SignIn` oder `SignUp` sind die einzigen Dimensionen. UserPoolId: Die Kennung des Benutzerpools. RiskLevel: hoch (Standard), mittel oder niedrig.	`AWS/Cognito`
AccountTakeoverRisk	Anforderungen, bei denen Amazon Cognito ein Risiko	Vorgang: Die Arten des Vorgangs `PasswordChange` , `SignIn` oder `SignUp`	`AWS/Cognito`

Metrik	Beschreibung	Metrikdimensionen	Namespace
	zur Kontoübernahme erkannt hat.	sind die einzigen Dimensionen. UserPoolId: Die Kennung des Benutzerpools. RiskLevel: hoch, mittel oder niedrig.	
OverrideBlock	Anforderungen, die von Amazon Cognito aufgrund der vom Entwickler vorgenommenen 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.	`AWS/Cognito`
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.	`AWS/Cognito`

Metrik	Beschreibung	Metrikdimensionen	Namespace
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.	`AWS/Cognito`

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

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-

Dimension	Beschreibung
	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-Nutzungsmetriken ist API.
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 Sie die CloudWatch Konsole, um Metriken zu verfolgen

Sie können mithilfe CloudWatch von Amazon Cognito-Benutzerpool-Metriken verfolgen und sammeln. Das CloudWatch Dashboard zeigt Metriken 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.

Metriken in Service Quotas

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 zu verfolgen. 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.

Amazon Cognito anmelden 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 CloudWatch -Alarme einrichten, um bei Änderung einer Identitäten-Pool-Konfiguration einen Alarm auszulösen. Weitere Informationen finden Sie unter CloudWatch Alarme für CloudTrail Ereignisse erstellen: Beispiele.

Themen

- Informationen, an die Amazon Cognito sendet CloudTrail
- Analysieren von Amazon Cognito CloudTrail Cognito-Ereignissen mit Amazon CloudWatch Logs Insights
- Beispiele für Amazon Cognito Cognito-Ereignisse

Informationen, an die Amazon Cognito sendet 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 unter:

- 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 CloudTrail -Element userIdentity.

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 Beispiele für Amazon Cognito Cognito-Ereignisse. 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.

Ereignisse im Benutzerpool

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 verwaltete Anmeldeseiten bereitstellt. API-Anfragen haben den `eventType` `AwsApiCall` und Endpunktanfragen den `eventType` `AwsServiceEvent`.

Amazon Cognito protokolliert die folgenden Anfragen an Ihre verwalteten Anmeldedienste als Ereignisse in CloudTrail.

Hosted UI (classic) events

Gehostete UI-Ereignisse (klassisch) 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.

Operation	Beschreibung
`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.
`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.

Operation	Beschreibung
`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.
`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 sendet Amazon Cognito Feedback zu einem Bedrohungsschutzereignis.

Managed login events

Verwaltete Anmeldeereignisse in CloudTrail

Operation	Beschreibung
`login_POST`	Ein Benutzer sendet Anmeldeinformationen an IhreLogin-Endpunkt.
`login_continue_POST`	Ein Benutzer, der sich bereits einmal angemeldet hat, entscheidet sich dafür, sich erneut anzumelden.

Operation	Beschreibung
`forgotPassword_POST`	Ein Benutzer setzt sein Passwort zurück.
`selectChallenge_POST`	Ein Benutzer reagiert auf eine Authentif izierungsanfrage, nachdem er seinen Benutzernamen oder seine Anmeldein formationen eingegeben hat.
`confirmUser_GET`	Ein Benutzer öffnet den Link in einer Bestätigungs- oder Bestätigungs-E-Mail-Nachricht.
`mfa_back_POST`	Ein Benutzer wählt nach einer MFA-Auffo rderung die Schaltfläche Zurück.
`mfa_options_POST`	Ein Benutzer wählt eine MFA-Option aus.
`mfa_phone_register_POST`	Ein Benutzer reicht eine Telefonnummer ein, um sich als MFA-Faktor zu registrieren. Dieser Vorgang veranlasst Amazon Cognito, einen MFA-Code an ihre Telefonnummer zu senden.
`mfa_phone_verify_POST`	Ein Benutzer sendet einen MFA-Code, der an seine Telefonnummer gesendet wurde.
`mfa_phone_resendCode_POST`	Ein Benutzer sendet eine Anfrage, um einen MFA-Code erneut an seine Telefonnummer zu senden.
`mfa_totp_POST`	Ein Benutzer reicht einen TOTP-MFA-Code ein.
`signup_POST`	Ein Benutzer sendet Informationen an Ihre `/signup` verwaltete Anmeldeseite.
`signup_confirm_POST`	Ein Benutzer sendet einen Bestätigungscode aus einer E-Mail- oder SMS-Nachricht.

Operation	Beschreibung
`verifyCode_POST`	Ein Benutzer übermittelt ein Einmalkennwort (OTP) für die kennwortlose Authentifizierung.
`passkeys_add_POST`	Ein Benutzer sendet eine Anfrage zur Registrierung eines neuen Hauptschlüssels.
`passkeys_add_GET`	Ein Benutzer navigiert zu der Seite, auf der er einen Hauptschlüssel registrieren kann.
`login_passkey_POST`	Ein Benutzer meldet sich mit einem Hauptschl üssel an.

> **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.

Identitätspools von Ereignissen

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-Identitätspools-API-Operationen, bei denen Amazon Cognito protokolliert CloudTrail, finden Sie in der Amazon Cognito Cognito-Identitätspools-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.

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
```

Beispiele für Amazon Cognito Cognito-Ereignisse

Amazon Cognito protokolliert Informationen AWS CloudTrail über Benutzerauthentifizierungsaktivitäten und administrative Verwaltungsaktivitäten. Dies gilt sowohl für Benutzerpools als auch für Identitätspools. Sie `GetId` können beispielsweise `UpdateIdentityPool` Ereignisse in derselben Spur oder Ereignisse `UpdateAuthEventFeedback` und `SetRiskConfiguration` Ereignisse anzeigen. Außerdem werden Benutzerpoolprotokolle für gehostete Benutzeroberflächenaktivitäten angezeigt, die nicht den Vorgängen in der Benutzerpools-API entsprechen. In diesem Abschnitt finden Sie einige Beispiele für Protokolle, die Ihnen möglicherweise angezeigt werden. Um das CloudTrail Ereignisschema für einen Vorgang zu verstehen, generieren Sie eine Anfrage für diesen Vorgang und überprüfen Sie die Ereignisse, die dadurch in Ihrem Trail erzeugt werden.

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
- CloudTrail Beispielereignis für eine SAML-Anfrage
- CloudTrail Beispielereignisse für Anfragen an den Token-Endpunkt
- CloudTrail Beispielereignis für CreateIdentityPool
- CloudTrail Beispielereignis für GetCredentialsForIdentity
- CloudTrail Beispielereignis für GetId
- CloudTrail Beispielereignis für GetOpenIdToken
- CloudTrail Beispielereignis für GetOpenIdTokenForDeveloperIdentity
- CloudTrail Beispielereignis für UnlinkIdentity

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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "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"
}
```

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

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.

AWS verwaltete IAM-Richtlinien, die Zugriff auf Amazon Cognito gewähren

- `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 Cognito-Team verwaltet. Selbst wenn neue hinzugefügt APIs werden, haben Ihre Benutzer weiterhin die gleiche Zugriffsebene.

> **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 auf nicht authentifizierte Benutzer an. 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.

AWS verwaltete IAM-Richtlinien, die Amazon Cognito Gastbenutzern gewährt

- `AmazonCognitoUnAuthedIdentitiesSessionPolicy`— Beschränkt in Kombination mit einer Inline-Sitzungsrichtlinie die Berechtigungen, die IAM-Administratoren Identitätspool-Gastbenutzern gewähren können. Amazon Cognito wendet diese Richtlinie automatisch auf Gastsitzungen an. Weitere Informationen finden Sie unter Die Richtlinie für AWS verwaltete Sitzungen für Gäste.

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
`AmazonCognitoPowerUser` —Änderung	Amazon Cognito hat neue Aktionen hinzugefügt, um die Verwendung des AWS Endbenutzer-Messaging SMS API-Vorgangs DescribeAccountAttributesfür Administrator-Poweruser von Amazon Cognito Cognito-Benutzerpools zu ermöglichen.	27. Februar 2025
`AmazonCognitoUnAuthedIdentitiesSessionPolicy` —Veränderung	Amazon Cognito hat neue Aktionen hinzugefügt, um die Verwendung von AWS Key Management Service für nicht authentifizierte (Gast-) Benutzer in Identitätspools zu ermöglichen.	30. Oktober 2024
`AmazonCognitoUnAuthedIdentitiesSessionPolicy` —Ändern	Amazon Cognito hat neue Aktionen hinzugefügt, um die Nutzung von Amazon Location Service für nicht authentifizierte (Gast-) Benutzer in	9. August 2024

Änderung	Beschreibung	Datum
	Identitätspools zu ermöglich en.	
`AmazonCognitoUnAut hedIdentitiesSessi onPolicy` —Neue Richtlinie	Es wurde eine AWS verwaltet e Richtlinie für die Einschrän kung des Rechtebereichs von Gastbenutzern in Identität spools hinzugefügt.	14. Juli 2023
`AmazonCognitoPower User` und — Ändern `AmazonCognitoRead0 nly`	Neue Berechtigungen wurden hinzugefügt, um es Power-Use rn zu ermöglichen, Verknüpfu ngen von AWS WAF Web-ACLs zu Amazon Cognito Cognito-Benutzerpools anzuzeigen und zu verwalten. Neue Berechtigungen wurden hinzugefügt, um Benutzern mit Lesezugriff das Anzeigen von Verknüpfungen zwischen AWS WAF Web- und Amazon Cognito Cognito-Benutzerpo ols ACLs zu ermöglichen.	19. Juli 2022

Änderung	Beschreibung	Datum
`AmazonCognitoPowerUser` — Ändern	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 Autorisierungsrichtlinien zu senden. Darüber hinaus können Amazon-SES-Konfigurationssets angewendet werden, wenn Sie das Versenden von E-Mails in Ihrem Benutzerpool konfigurieren.	17. November 2021

Änderung	Beschreibung	Datum
`AmazonCognitoPowerUser` —Ändern	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 einer AWS Ressource AWS zuweisen oder 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 Sie Ihre AWS Ressourcen. Viele AWS Dienste unterstützen Tagging, sodass Sie Ressourcen aus verschiedenen Diensten dasselbe Tag zuweisen können. Dies hilft Ihnen, anzugeben, welche Ressourcen dazugehören. Sie können beispielsweise dasselbe Tag einem Amazon–Cognito-Benutzerpool und einer Amazon-DynamoDB-Tabelle zuweisen.
- Verfolgen Sie Ihre AWS Kosten. Sie können diese Tags im AWS Fakturierung und Kostenmanagement Dashboard aktivieren. AWS verwendet Tags zur Kostenzuweisung, um Ihre Kosten zu kategorisieren und Ihnen einen monatlichen Kostenverteilungsbericht zu liefern. 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 Cognito-API verwenden, um Tags für Benutzer- und Identitätspools hinzuzufügen, zu bearbeiten oder zu löschen. Für Benutzerpools können Sie die Tags zudem mit der Amazon-Cognito-Konsole verwalten.

Tipps zur Verwendung von Tags finden Sie unter AWS Markierungsstrategien 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. 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 das Menü Einstellungen und suchen Sie den Tab Tags.
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).

Um einen Identitätspool zu taggen, navigieren Sie zum Menü Identitätspools und wählen Sie einen Identitätspool aus oder erstellen Sie einen. Suchen Sie auf der Registerkarte Eigenschaften des Identitätspools nach Tags. Wählen Sie Add tag.

AWS CLI Beispiele

Das AWS CLI bietet Befehle, mit denen Sie die Tags verwalten können, die Sie Ihren Amazon Cognito Cognito-Benutzerpools und Identitätspools zuweisen.

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
- Verwalten von API-Anforderungsratenkontingenten
- API-Betriebskategorien und Anforderungsratenkontingente von Amazon-Cognito-Benutzerpools
- Amazon Cognito-Identitätspools (verbundene Identitäten) API-Vorgangsanforderungsratenkontingente
- Kontingente für die Anzahl und Größe der Ressourcen

Informationen zu API-Anforderungsratenkontingenten

Kategorisierung von Kontingenten

Amazon Cognito erzwingt eine maximale Anforderungsrate für API-Operationen. Weitere Informationen zu den API-Vorgängen, die Amazon Cognito zur Verfügung stellt, finden Sie in den API-Referenzhandbüchern für Benutzerpools und Identitätspools. Für Benutzerpools sind diese Operationen in Kategorien gängiger Anwendungsfälle wie `UserAuthentication` oder `UserCreation` unterteilt. Eine Liste der API-Operationen für Benutzerpools nach Kategorien finden Sie unterAPI-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. Wenn Sie Verbundbenutzer mithilfe eines SAML- oder OIDC-Verbunds mit lokalen Benutzern verknüpfen, zählt der lokale Benutzer als MAU oder im Unternehmensverzeichnis`EnterpriseMAU`, unabhängig davon, ob sich der Benutzer direkt oder über einen Verbund anmeldet. Weitere Informationen finden Sie unter Amazon Cognito – Preise.

- Registrierung oder administrative Erstellung eines Benutzers. Der CSV-Import von Benutzern trägt nicht zu Ihrer MAU-Anzahl bei.
- Bestätigung des Benutzerkontos oder Überprüfung von Attributen.
- Melden Sie sich an und fordern Sie die Antwort heraus. Vorgänge, die Sie mit dem Zugriffstoken des aktuell angemeldeten Benutzers autorisieren, tragen nicht zu Ihrer MAU-Anzahl bei. Da bei der Anmeldung jedoch Zugriffstoken erzeugt werden, weisen diese Vorgänge darauf hin, dass es sich bei dem zugehörigen Benutzer um eine MAU handelt.
- Abmeldung und Token-Widerruf.

- Self-Service-Zurücksetzen von Passwörtern und Einstellung von Benutzerkennwörtern als Administrator. Das Zurücksetzen von Benutzerkennwörtern als Administrator (AdminResetUserPassword) trägt nicht zu Ihrer MAU-Anzahl bei.
- Ändern Sie Benutzerattribute oder Gruppenzugehörigkeit.
- Fragen Sie als Administrator detaillierte Attribute eines Benutzers ab.

> **Note**
>
> Die Kategorie Detaillierte Attribute eines Benutzers als Administrator 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 Kosten zu vermeiden, sollten Sie Benutzerdaten mit einer externen Datenbank sammeln `ListUsers` oder Benutzerinformationen in einer externen Datenbank speichern.

Zusätzliche Sitzungen von aktiven Benutzern oder Benutzern, die innerhalb eines Kalendermonats nicht aktiv waren, werden Ihnen nicht in Rechnung gestellt. In einem Monat, in dem Sie Ihren Benutzerpool-Funktionsplan zwischen den verfügbaren Optionen Lite, Essentials und Plus gewechselt haben, wird Ihre Rechnung für diesen Monat anhand der Summe der monatlich aktiven Benutzer (MAUs) in jeder Stufe berechnet, wobei jede MAU der teuersten zugewiesenen Stufe zugewiesen wird, als der Benutzer aktiv war. Zum Beispiel:

1. Zu Beginn des Monats gilt für Ihren Benutzerpool der Plus-Funktionsplan.
2. Benutzer A meldet sich am ersten Tag des Monats an.
3. Benutzer B meldet sich am ersten und letzten Tag des Monats an.
4. Am zehnten Tag des Monats stellen Sie Ihren Feature-Plan auf Essentials um.
5. Benutzer C meldet sich am letzten Tag des Monats an.

In diesem Szenario sind Benutzer A und Benutzer B Plus MAUs und Benutzer C ist eine Essentials-MAU.

Leichte MAU

Ein Benutzer, der mindestens einmal im Monat aktiv war, als der Benutzerpool im Lite-Feature-Plan war, und nie aktiv war, als der Benutzerpool im Essentials- oder Plus-Plan war.

Essentials MAU

Ein Benutzer, der mindestens einmal im Monat aktiv war, als der Benutzerpool im Essentials-Funktionsplan war, und nie aktiv war, als der Benutzerpool im Plus-Plan war.

Plus MAU

Ein Benutzer, der mindestens einmal im Monat aktiv war, als der Benutzerpool im Plus-Plan war.

Weitere Informationen finden Sie unter Funktionspläne für Benutzerpools.

Verwalten von API-Anforderungsratenkontingenten

Kontingentanforderungen identifizieren

> **Important**
>
> Wenn Sie die Amazon Cognito Cognito-Kontingente für Kategorien wie `UserAuthenticationUserCreation`, 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 hat 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.

Zwischenspeichern JWTs

Verwenden Sie Zugriffstoken wieder, bis sie ablaufen. Ein Beispiel-Framework mit Token-Caching in einem API Gateway finden Sie unterVerwaltung des Ablaufs und der Zwischenspeicherung von Benutzerpool-Tokens. 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 ansehen.

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 die monatlich aktiven 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 zum Beispiel 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 nächste Million Nutzer mit einem angepassten Kontingent von 160 RPS unterzubringen.

Um deine zu überprüfen MAUs

Rufen Sie die AWS Billing Konsole auf und überprüfen Sie eine aktuelle Rechnung. Unter Gebühren nach Service kannst du nach Cognito filtern, um dir eine Aufschlüsselung deiner Gebühren MAUs für diesen Abrechnungszeitraum anzusehen.

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 jede Kategorie wird in monatlich aktiven Benutzern (MAUs) gemessen. AWS-Konten mit weniger als zwei Millionen MAUs können innerhalb des Standardkontingents arbeiten. Wenn Sie weniger als eine Million haben MAUs und Amazon Cognito Anfragen drosselt, sollten Sie erwägen, Ihre App zu optimieren. 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` , `InitiateAuth` , `RespondToAuthChallenge`	10
Schreiben eines Benutzerprofils Beispiele: `UpdateUserAttributes` , `SetUserSettings`	10

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 dem Token-Endpunkt • RespondToAuthChallenge[1] • AdminInitiateAuth • AdminRespondToAuthChallenge[1] • Verwaltete Anmeldung oder klassische Hosted UI-Anmeldung und MFA im Autorisierungscode oder implizite Zuweisungen [2]	Operationen, die einen Benutzer authentifizieren (anmelden). Diese Operationen unterliegen API-Operationen von Amazon-Cognito-Benutzerpools mit spezieller Handhabung der Anforderungsrate .	120	Ja
`UserCreation` • SignUp • ConfirmSignUp • AdminCreateUser • AdminConfirmSignUp	Operationen, die einen lokalen Amazon-Cognito-Benutzer erstellen oder bestätigen. Dies ist ein Benutzer, der direkt von Ihren Amazon-Cognito-Ben	50	Ja

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
	utzerpools 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] • Zurücksetzen des verwalteten Anmeldekennworts	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 • AdminSetUserMFAPreference • AdminSetUserSettings • AdminUpdateDeviceStatus • UpdateDeviceStatus • UpdateAuthEventFeedback • ConfirmDevice • SetUserMFAPreference • 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 • Holen CSVHeader • GetGroup • GetSigningCertificate • GetIdentityProviderByIdentifier • GetUserPoolMfaConfig • ListGroups • ListIdentityProviders • ListResourceServers • ListTagsForResource • ListUserImportJobs • DescribeRiskConfiguration • Holen UICustomization	Vorgänge, die Informationen über Ressourcen wie Gruppen oder Ressourcenserver aus einem Benutzerpool abrufen.[3]	20	Nein

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
• DescribeManagedLoginBranding • DescribeManagedLoginBrandingByClient			

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 • Einstellen UICustomization • TagResource • UntagResource • CreateManagedLoginBranding • UpdateManagedLoginBranding • DeleteManagedLoginBranding			
`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

Kategorie	Beschreibung	Standardkontingent (RPS)	Einstellbar
`ClientAuthentication` Anfragen des Erteilungstyps `client_credentials` an den Token-Endpunkt.	Operationen, die Anmeldeinformationen generieren, die bei der Autorisierung von Anfragen machine-to-machine verwendet werden	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] Jede verwaltete Anmeldung oder jeder klassische Hosted UI-Vorgang 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.

Grenzwerte für die Rate von Massenanfragen für Benutzerpool-Domänen

Die folgenden Kontingente gelten für das Gesamtvolumen der Anfragen an eine Benutzerpool-Domain.

Operation	Beschreibung	Standardkontingent (RPS)	Einstellbar
Pro IP	Volumen der Anfragen von einer IP-Adresse	300	Nein

Operation	Beschreibung	Standardkontingent (RPS)	Einstellbar
Pro App-Client	Volumen der Anfragen für eine App-Client-ID	300	Nein
Pro Domain	Gesamtvolumen der Anfragen für die Dienste einer Benutzerpool-Domäne	500	Nein

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`	Rufen Sie eine Identitätsliste IDs 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.

Amazon Cognito Cognito-Benutzerpools, Ressourcenkontingente

Die folgenden Kontingente beschreiben die maximale Anzahl oder Länge von Elementen, die Sie in Benutzerpools erstellen können.

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.
Verwaltete Login-Branding-Stile pro Benutzerpool	10	Nein	N/A
Benutzerdefinierte Attribute pro Benutzerpool	50	Nein	N/A
Zeichenanzahl pro Attribut	2048 Bytes	Nein	N/A

Ressource	Kontingent	Einstellbar	Höchstkontingent
Zeichen im benutzerdefinierten Attributnamen	20	Nein	N/A
Erforderliche Mindestkennwortzeichen in Passwortrichtlinie.	6–99	Nein	N/A
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 Identitätsanbieters	32	Nein	N/A
Zeichen in einer SAML-Antwort	100 000	Nein	N/A
Identifikatoren pro Identitätsanbieter	50	Nein	N/A

Ressource	Kontingent	Einstellbar	Höchstkontingent
Identitäten, die mit einem Benutzer verbunden sind	5	Nein	N/A
Passkey/Authentifikatoren pro Benutzer WebAuthn	20	Nein	N/A
Rückruf pro App-Client URLs	100	Nein	N/A
Abmeldung URLs pro App-Client	100	Nein	N/A
Bereiche pro Ressourcenserver	100	Nein	N/A
Bereiche pro App-Client	50	Nein	N/A
Benutzerdefinierte Domains pro Region	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 Geltungsbereiche von Zugriffs- und Identitätstoken in einer Transaktion muss eine Zahl ergeben, die kleiner oder gleich dieser Quote ist. 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. Diese Beschränkung wird täglich um 09:00 Uhr UTC zurückgesetzt. Weitere Informationen finden Sie unter E-Mail-Einstellungen für Amazon-Cognito-Benutzerpools.

Gültigkeitsparameter für Amazon-Cognito-Benutzerpools

Die folgenden Kontingente beschreiben die verfügbaren Einstellungen für die Dauer von Authentifizierungsartefakten und Benutzersitzungen in 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)

Die folgenden Kontingente beschreiben die verfügbaren Zeiträume im Zusammenhang mit Codes für die Anmeldung, Registrierung und Kennwortzurücksetzung.

Ressource	Kontingent
Gültigkeitszeitraum für Anmeldebestätigung scode	24 Stunden
Gültigkeitszeitraum des Verifizierungscodes für Benutzerattribute	24 Stunden

Ressource	Kontingent
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

Die folgenden Kontingente beschreiben die Ressourcen und Beschränkungen, die für Benutzerimportaufträge zur Verfügung stehen.

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 Cognito-Identitätspools (föderierte Identitäten)

Die folgenden Kontingente beschreiben die maximale Anzahl oder Länge von Elementen, die Sie in Identitätspools erstellen können.

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

Die folgenden Kontingente beschreiben die maximale Anzahl oder Länge von Elementen, die Sie in Amazon Cognito Sync erstellen können.

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.

Ressource	Kontingent	Einstellbar	Höchstkontingent
Zeichenanzahl im Datensatz-Namen	128 Byte	Nein	N/A
Wartezeit für eine Massen-Veröffentlichung nach einer erfolgreichen Anfrage	24 Stunden	Nein	N/A

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
Aktualisierungen der Anforderungen für IAM-Richtlinien zur Rollenvertrauensstellung für Identitätspools.	Sie können jetzt nur dann Änderungen an den Rollenvertrauensrichtlinien für den Amazon Cognito Cognito-Identitätspools-Dienstprinzipal vornehmen, wenn ein Bedingungsschlüssel die Zielgruppe für den OIDC-Verbund (Web Identity) auf einen oder mehrere Identitätspools beschränkt.	1. August 2025
Amazon Cognito ist jetzt in Mexiko (Zentral) AWS-Region verfügbar.	Sie können jetzt Amazon Cognito Cognito-Ressourcen in der Region Mexiko (Zentral) erstellen.	24. Juli 2025
Amazon Cognito ist jetzt im asiatisch-pazifischen Raum (Thailand) AWS-Region verfügbar.	Sie können jetzt Amazon Cognito Cognito-Ressourcen in der Region Asien-Pazifik (Thailand) erstellen.	24. Juli 2025
AWS WAF Web ACLs im verwalteten Login.	Sie können jetzt AWS WAF Web-ACL-Regeln auf Benutzerpool-App-Clients anwenden, die über die Branding-Version für verwaltete Logins verfügen.	24. Juni 2025

Beispiele für Lambda-Trigger wurden aktualisiert.	Die Beispielfunktion für Lambda-Trigger für benutzerdefinierte E-Mail- und SMS-Absender wurde aktualisiert, sodass sie mit Node.js 2.x kompatibel ist. Das Beispiel ist jetzt auch für Tests leichter zugänglich.	19. Mai 2025
Neuer `prompt` Parameter.	Mit dem `prompt` Parameter haben Sie jetzt mehr Kontrolle über die Neuauthentifizierung vorhandener verwalteter Anmeldesitzungen. Sie können Werte für diesen Parameter auch an Drittanbieter weitergeben.	15. Mai 2025
Client-Metadaten für M2M-Anfragen.	Sie können jetzt Client-Metadaten in Client-Anmeldeinformationen oder machine-to-machine (M2M) -Anfragen übergeben. Amazon Cognito leitet M2M-Client-Metadaten an den Lambda-Trigger vor der Token-Generierung weiter.	29. April 2025
Aktualisieren Sie die Token-Rotation.	Sie können jetzt neue Aktualisierungstoken abrufen und die Originale in Aktualisierungstoken-Anfragen für ungültig erklären.	22. April 2025
Amazon Cognito ist jetzt im asiatisch-pazifischen Raum (Malaysia) AWS-Region verfügbar.	Sie können jetzt Amazon Cognito Cognito-Ressourcen in der Region Asien-Pazifik (Malaysia) erstellen.	7. März 2025

Anpassung von Zugriffstoken für Maschinenidentitäten.	Der Lambda-Trigger vor der Token-Generierung verfügt jetzt über ein Ereignis der Version 3, das die Ansprüche auf Zugriffstoken und den Umfang der Zugriffstoken-Zuweisungen für machine-to-machine die (M2M) -Autorisierung ändert.	03. März 2025
Die Informationen zur verwalteten Richtlinie wurden aktualisiert. `AmazonCognitoPowerUser` AWS	In der AWS verwalteten Richtlinie für Amazon Cognito Cognito-Benutzerpools mit Power-Usern wurde ein AWS Endbenutzer-Messaging SMS Vorgang hinzugefügt.	27. Februar 2025
Aktualisierter Überblick über die OpenID Connect (OIDC) -Integration.	Es wurde ein Diagramm hinzugefügt, das veranschaulicht, wie Amazon Cognito sich bei OIDC-Identitätsanbietern authentifiziert.	25. Februar 2025
Es wurden Informationen zur MFA-Logik hinzugefügt.	Es wurde ein Diagramm hinzugefügt, das veranschaulicht, wie Amazon Cognito die Multi-Faktor-Authentifizierungseinstellungen (MFA) Ihres Benutzerpools zur Laufzeit auf Benutzer anwendet.	25. Februar 2025

Bewährte Sicherheitsmethoden für Amazon Cognito Cognito-Benutzerpools wurden hinzugefügt.	Es wurde eine Seite zur Sicherung von Geheimnissen und zur sonstigen Einhaltung von bewährten Sicherheitsmethoden bei der Konfiguration von Benutzerpools hinzugefügt.	25. Februar 2025
Aktualisierungen der Ressourcen für die ersten Schritte für Benutzerpools.	Die ersten Schritte mit Amazon Cognito Cognito-Benutzerpools haben ein neues Konsolendesign und neue Anwendungsoptionen.	21. November 2024
Neues Preismodell mit Funktionsplänen.	Das Abrechnungsmodell für Benutzerpools wurde aktualisiert. Erweiterte Sicherheitsfunktionen schützen jetzt vor Bedrohungen. Die Komponenten der Lizenz für erweiterte Sicherheitsfunktionen sind jetzt in den Featureplänen Essentials und Plus enthalten.	21. November 2024
Neue verwaltete Anmeldefunktion.	Managed Login gestartet, ein Update für die gehostete Benutzeroberfläche.	21. November 2024
Eine neue Authentifizierungsmethode und neue Authentifizierungsabläufe.	Sie können sich jetzt mit Hauptschlüsseln und Einmalkennwörtern bei Amazon Cognito Cognito-Benutzerpools anmelden.	21. November 2024

Aktualisierte Informationen über. `AmazonCognitoUnAuthedIdentitiesSessionPolicy`	Die AWS Key Management Service Vorgänge in der AWS verwalteten Richtlinie zur Erfassung des Gültigkeitsbereichs nicht authentifizierter Identitäten wurden von der Inline-Richtlinie in die verwaltete Richtlinie verschoben. AWS	1. November 2024
Parameter hinzugefügt. `login_hint`	Sie können jetzt Autorisierungsanfragen für die gehostete Benutzeroberfläche, OIDC und Google mit einem Hinweis auf einen Nutzernamen IdPs versehen. IdPs	3. Oktober 2024
Neue erweiterte Sicherheitsfunktionen für E-Mail-MFA.	Sie können jetzt Multi-Faktor-Authentifizierungscodes (MFA) per E-Mail-Nachricht mit erweiterten Sicherheitsfunktionen senden.	12. September 2024
Neue Inhalte und Seitenänderungen.	Titel wurden geändert, nicht benötigte Inhalte entfernt, szenariobasierte Intros hinzugefügt, Benutzerpools (OIDC) und die Referenz der gehosteten Benutzeroberflächenendpunkte in den Abschnitt Benutzerpools verschoben.	9. September 2024

`AmazonCognitoUnAuthedIdentitiesSessionPolicy` Aktualisierte Informationen über.	Die AWS verwaltete Richtlinie für die Erfassung nicht authentifizierter Identitäten in Identitätspools ermöglicht jetzt Amazon Location Service.	9. August 2024
Neue Bedrohungsabwehr für benutzerdefinierte Authentifizierung mit Lambda-Triggern und verbesserter Bedrohungserkennung.	Sie können jetzt die benutzerdefinierte Authentifizierungsanmeldung mit Bedrohungsschutz analysieren und adaptive Authentifizierungsantworten anwenden. Der Bedrohungsschutz analysiert jetzt auch den Anmeldeverkehr auf unmögliche geografische Entfernungen zwischen den Versuchen.	8. August 2024
Neue erweiterte Sicherheitsfunktionen zur Verhinderung der Wiederverwendung von Passwörtern und zum Export von Benutzeraktivitätsprotokollen.	Sie können jetzt Benutzeraktivitätsprotokolle exportieren und eine Passwort-Verlaufsrichtlinie mit erweiterten Sicherheitsfunktionen in Amazon Cognito Cognito-Benutzerpools einrichten.	6. August 2024
Amazon Cognito ist jetzt in Kanada West (Calgary) und Asien-Pazifik (Hongkong) verfügbar. AWS-Regionen	Sie können jetzt Amazon Cognito Cognito-Ressourcen in den Regionen Kanada West (Calgary) und Asien-Pazifik (Hongkong) erstellen.	9. Juli 2024
Verbesserte Beschreibung des Anwendungsverhaltens für erweiterte Sicherheit	Die Informationen zu den Gerätekontextdaten wurden für die erweiterte adaptive Sicherheitsauthentifizierung aktualisiert.	10. Juni 2024

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 bündeln Amazon Cognito Cognito-Benutzer E-Mails zum Lastenausgleich zwischen zwei entfernten Regionen.	10. Mai 2024
Es wurden Informationen zur M2M-Autorisierung und zur Kostenverwaltung hinzugefügt.	Erfahren Sie, wie Sie Zuwendungen für 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 (Hyderabad) verfügbar. AWS-Regionen	Sie können jetzt Amazon Cognito Cognito-Ressourcen in den Regionen Europa (Spanien) und Asien-Pazifik (Hyderabad) erstellen.	15. April 2024
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
Kontingenterhöhungen 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
Eine neue Funktion zur Anpassung des Inhalts von Zugriffstoken wurde 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 Anwendungsspezifische Einstellungen mit App-Clients und Bereiche, M2M und APIs 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 hervorgeh oben, 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 Zugriffse ntscheidungen von einer zentralen Stelle aus zu treffen.	1. August 2023
Eine neue Funktion zur Protokollierung detaillie rter 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 Berechtig ungen für Identitätspool-Gas tbenutzer 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-Benutzerpo ols hinzugefügt.	Amazon-Cognito-Benutzerpool s 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-Co gnito-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 Benutzerh andbuch für die gehostete Benutzeroberfläche und Informationen zu TOTP MFA in der gehosteten Benutzero berfläche hinzugefügt.	Ihre Benutzer können jetzt ein TOTP-MFA-Gerät in der von Amazon Cognito gehosteten Benutzeroberfläche registrie ren. Sie können jetzt eine Vorschau der standardm äßigen gehosteten Benutzero berfläche 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 zum Hinzufügen von Lesezeichen für Anwendungen hinzugefügt, sodass Amazon Cognito Benutzer automatisch 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 Sperrendpunkt	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 von der AWS US-Regierung unterstützt CLoud	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 in Ihre Web- oder Mobil-App mithilfe AWS Amplify SDKs von Bibliotheken hinzugefügt. Die vorherigen Informationen zur Verwendun g von Amazon Cognito SDKs wurden entfernt. AWS Amplify	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 wurden eine anpassbare integrierte App-Benutzeroberfläche und OAuth 2.0-Unterstützung mit benutzerdefinierten Ansprüchen hinzugefügt.	10. August 2017
Änderungen an 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 Benutzerverzeichnisse erstellen und verwalten sowie Registrierungs- und Anmeldeinformationen hinzufügen.	28. Juli 2016
SAML-Unterstützung	Unterstützung der Authentifizierung mit Identitätsanbietern über Security Assertion Markup Language 2.0 (SAML 2.0) wurde hinzugefügt.	23. Juni 2016
CloudTrail Integration	Integration mit AWS CloudTrail hinzugefügt.	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-Unterstützung	Ermöglicht die Unterstützung von OpenID Connect-Anbietern.	23. November 2014
Synchronisation per Push	Ermöglicht die Unterstützung der automatischen Push-Synchronisierung.	6. November 2014

Unterstützung für vom Entwickler authentifizierte Identitäten hinzugefügt	Entwickler mit eigenen Authentifizierungs- und Identitätsmanagementsystemen 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.