使用者指南

AWS Identity and Access Management



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Identity and Access Management: 使用者指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任從何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能附屬於 Amazon，或與 Amazon 有合作關係，亦或受到 Amazon 贊助。

Table of Contents

什麼是 IAM？ 1
為什麼要使用 IAM？ 2
共用存取您的 AWS 帳戶 2
精密許可 3
安全存取在 Amazon EC2 上執行的應用程式 AWS 的資源 3
多重要素驗證 (MFA) 3
聯合身分 3
身分資訊保證 3
PCI DSS 合規 3
IAM 的使用時機為何？ 4
執行不同工作職能時 4
當您獲得存取 AWS 資源的授權時 4
以 IAM 使用者身分登入時 5
擔任 IAM 角色時 5
建立政策和許可時 5
如何管理 IAM？ 6
使用 AWS 管理主控台 6
AWS 命令列工具 7
使用 AWS SDKs 9
使用 IAM 查詢 API 9
IAM 的運作方式 9
請求的元件 11
如何對主體進行身分驗證 11
授權和許可政策基本概念 12
..... 12
比較 IAM 身分和憑證 13
條款 13
IAM 使用者和 IAM Identity Center 之間的不同之處 15
來自現有身分來源的聯合身分使用者 16
提供使用者存取權的不同方法 17
支援程式設計使用者存取 19
許可和政策如何提供存取管理 21
政策和帳戶 21
政策與使用者 21

政策和 IAM 群組 22
聯合身分使用者工作階段和角色 22
以身分為基礎和以資源為基礎的政策 22
使用 ABAC 授權定義許可 23
ABAC 與傳統 RBAC 模型的比較 24
開始使用 26
設定您的 AWS 帳戶 26
檢視您的 AWS 帳戶 ID 28
檢視您的 AWS 帳戶 ID 28
為您的 AWS 帳戶 ID 使用別名 30
建立帳戶別名 31
刪除帳戶別名 31
計劃存取您的帳戶 AWS 33
身分來源 33
存取管理 33
IAM 使用者的使用案例 34
對身分使用多重要素驗證 45
為最低權限許可做好準備 46
檢閱您 AWS 帳戶的上次存取資訊 46
根據存取活動產生政策 51
使用搜尋尋找 IAM 資源 54
安全最佳實務和使用案例 57
安全最佳實務 57
要求人類使用者使用聯合身分提供者，以 AWS 使用臨時憑證存取 58
要求工作負載使用具有 IAM 角色的臨時憑證來存取 AWS 58
需要多重要素驗證 (MFA) 59
對於需要長期憑證的使用案例，請視需要更新存取金鑰 60
遵循最佳實務以保護您的根使用者憑證 60
套用最低權限許可 61
開始使用 AWS 受管政策，並朝最低權限的許可邁進 61
使用 IAM Access Analyzer 根據存取活動產生最低權限政策 61
定期檢閱並移除未使用的使用者、角色、許可、政策和憑證 61
使用 IAM 政策中的條件進一步限制存取權 61
使用 IAM Access Analyzer 驗證對資源的公開與跨帳戶存取權 62
使用 IAM Access Analyzer 驗證您的 IAM 政策是否有安全且功能正常的許可 62
建立跨多個帳戶的許可防護機制 62

使用許可界限委派帳戶內的許可管理 62
根使用者最佳實務 63
保護您的根使用者憑證，以防止未經授權使用 64
使用強式根使用者密碼來協助保護存取 64
使用多重要素驗證 (MFA) 保護您的根使用者登入 64
不要為根使用者建立存取金鑰 65
盡可能為根使用者登入使用多人核准 65
使用群組電子郵件地址作為根使用者憑證 65
限制存取帳戶復原機制 65
保護 AWS Organizations 您的帳戶根使用者登入資料 66
監控存取和用量 67
商業使用案例 68
範例企業的初始設定 69
搭配 Amazon EC2 的 IAM 使用案例 70
搭配 Amazon S3 的 IAM 使用案例 71
教學課程 73
AWS 帳戶 使用角色委派跨 的存取權 73
考量事項 74
先決條件 75
在目的地帳戶中建立角色 75
授予角色存取權 79
透過切換角色測試存取 81
其他資源 85
摘要 85
建立客戶受管政策 85
先決條件 86
步驟 1：建立政策 86
步驟 2：連接政策 87
步驟 3：測試使用者存取許可 88
相關資源 88
摘要 88
使用以屬性為基礎的存取控制 (ABAC) 88
教學課程概觀 89
先決條件 91
步驟 1：建立測試使用者 91
步驟 2：建立 ABAC 政策 93

步驟 3：建立角色 97
步驟 4：測試建立秘密 97
步驟 5：測試檢視秘密 100
步驟 6：測試可擴展性 102
步驟 7：測試更新和刪除秘密 103
摘要 105
相關資源 105
使用 ABAC 的 SAML 工作階段標籤 106
允許使用者管理其憑證和 MFA 設定 110
先決條件 110
步驟 1：建立政策以實施 MFA 登入 111
步驟 2：將政策連接到您的測試使用者群組 112
步驟 3：測試您的使用者存取權限 112
相關資源 114
使用 建立 SAML IdP CloudFormation 115
先決條件 115
建立 IdP 115
驗證 IdP 118
刪除資源 118
範本詳細資訊 118
範本 119
使用 建立 SAML 聯合角色 CloudFormation 123
先決條件 123
建立聯合身分角色 123
測試聯合身分角色 126
刪除資源 126
範本詳細資訊 127
範本 127
使用 建立 SAML IdP 和聯合角色 CloudFormation 133
先決條件 134
建立資源 134
測試聯合身分 137
刪除資源 138
範本詳細資訊 138
範本 138
身分 148

IAM 根使用者 148
IAM 使用者 148
IAM 使用者群組 149
IAM 角色 149
AWS 帳戶根使用者 149
集中管理成員帳戶的根存取權 150
集中根存取權 150
執行特權任務 154
針對根使用者的 MFA 159
變更密碼 166
重設遺失或忘記的根使用者密碼 168
為根使用者建立存取金鑰 169
刪除根使用者的存取金鑰 171
需要根使用者憑證的任務 173
其他資源 175
使用者 175
如何 AWS 識別 IAM 使用者 175
IAM 使用者和憑證 176
IAM 使用者和許可 177
IAM 使用者和帳戶 178
IAM 使用者做為服務帳戶 178
IAM 使用者如何登入 AWS 178
建立使用者 182
檢視 IAM 使用者 183
重新命名使用者 185
移除使用者 187
控制使用者對主控台的存取 191
變更使用者許可 192
使用者密碼 199
管理存取金鑰 216
多重要素驗證 242
服務特定憑證 314
尋找未使用的憑證 323
產生憑證報告 326
CodeCommit 的 IAM 憑證 333
管理伺服器憑證 335

使用者群組 340
建立 IAM 群組 342
檢視 IAM 群組 344
編輯 IAM 群組中的使用者 345
將政策連接至使用者群組 347
重新命名使用者群組 347
刪除 IAM 群組 349
角色 350
何時建立 IAM 使用者 (而不是角色) 352
角色術語和概念 352
其他資源 355
混淆代理人問題 356
常用案例 366
角色建立 376
角色管理 434
擔任角色的方法 465
身分提供者和 的聯合身分 AWS 621
使用 IAM Identity Center 的聯合 622
使用 IAM 的聯合 622
使用 Amazon Cognito 身分池的聯合 623
其他資源 623
常用案例 624
OIDC 聯合身分 628
SAML 2.0 聯合身分 650
將 AWS 身分聯合到外部服務 686
常用案例 686
運作方式 687
開始使用傳出聯合身分 688
了解字符宣告 693
使用 IAM 政策控制存取 698
暫時安全憑證 700
AWS STS 和 AWS 區域 700
暫時性憑證的常見案例 701
使用臨時憑證的應用程式範例 702
暫時憑證的其他資源 703
比較 AWS STS 登入資料 704

服務持有人權杖 707
請求臨時安全憑證 708
搭配 AWS 資源使用臨時登入資料 717
臨時安全憑證的許可 721
在 AWS STS 中管理 AWS 區域 755
AWS STS 區域和端點 759
啟用自訂身分經紀人主控台存取 763
IAM 資源的標籤 777
選擇 AWS 標籤命名慣例 778
在 IAM 和 中標記的規則 AWS STS 779
標記 IAM 使用者 781
標記 IAM 角色 784
標記客戶管理政策 787
標記 OIDC 身分提供者 789
標記 IAM SAML 身分提供者 792
標記執行個體設定檔 795
標記伺服器憑證 797
標記虛擬 MFA 裝置 799
標記帳戶存取管理員執行個體 801
傳遞工作階段標籤 804
存取管理 818
存取管理資源 819
政策和許可 819
政策類型 820
政策和根使用者 825
JSON 政策概觀 825
授予最低權限 830
受管政策與內嵌政策 831
資料周邊 840
許可界限 845
身分與資源 857
使用政策控制存取權 860
使用標籤控制對 IAM 使用者和角色的存取 871
使用標籤控制對 AWS 資源的存取 873
跨帳戶資源存取 878
轉送存取工作階段 883

政策範例 886
管理 IAM 政策 961
其他資源 961
建立 IAM 政策 962
政策驗證 970
政策測試 971
新增或移除身分許可 983
版本控制 IAM 政策 993
編輯 IAM 政策 997
刪除 IAM 政策 1007
使用存取資訊精簡許可 1012
政策摘要 1594
政策摘要 (服務清單) 1595
政策摘要中的存取層級 1604
服務摘要 (動作清單) 1607
動作摘要 (資源清單) 1611
政策摘要範例 1615
存取 IAM 資源所需的許可 1625
管理 IAM 身分的許可 1625
在 中工作的許可 AWS 管理主控台 1627
跨 AWS 帳戶授予許可 1627
由一個服務來存取另一個服務的許可 1628
必要的動作 1628
用於 IAM 的政策範例 1629
臨時委派 1632
概觀 1632
暫時委派的運作方式 1633
啟動暫時委派請求 1635
檢閱暫時委派請求 1637
撤銷暫時委派存取權 1638
管理委派請求的許可 1639
通知 1640
CloudTrail 1641
合作夥伴的暫時委派 1642
概觀 1642
運作方式 1633

合作夥伴資格 1643
加入程序 1643
合作夥伴問卷 1644
了解您的整合 1645
了解許可 1648
政策評估準則 1650
政策範本 1652
建置您的整合 1663
帳戶存取管理員 1671
什麼是帳戶存取管理員？ 1671
為什麼要使用帳戶存取管理員？ 1671
關鍵功能 1672
開始使用 1673
先決條件和考量事項 1674
啟用帳戶存取管理員 1674
更新防火牆和閘道 1675
刪除帳戶存取管理員 1677
管理對 的存取 AWS 帳戶 1678
委派的管理 1678
準備您的 IAM 角色 1680
指派和移除存取權 1682
檢視或搜尋指派 1686
工作階段和存取控制 1688
人力存取 1689
使用者入口網站 1689
帳戶存取入口網站入門 1690
使用者存取 AWS 帳戶 1691
安全 1695
帳戶存取管理員的身分和存取管理 1695
帳戶存取管理員中的資料保護 1703
在帳戶存取管理員中記錄和監控 1703
配額 1704
群組指派配額 1704
調節限制 1704
程式碼範例 1705
IAM 1711

基本概念 1717
案例 2497
AWS STS 2987
基本概念 2988
案例 3019
安全 3270
AWS 安全登入資料 3271
安全考量 3272
程式設計存取權 3273
AWS 安全稽核準則 3275
何時執行安全性稽核 3276
稽核準則 3277
檢閱 AWS 您的帳戶登入資料 3277
檢閱 IAM 使用者 3277
檢閱 IAM 群組 3278
檢閱 IAM 角色 3278
檢閱 SAML 和 OpenID Connect (OIDC) 的 IAM 提供者 3278
檢閱行動應用程式 3278
檢閱 IAM 政策的要訣 3279
資料保護 3280
IAM 和 中的資料加密 AWS STS 3281
IAM 和 中的金鑰管理 AWS STS 3281
IAM 和 中的網際網路流量隱私權 AWS STS 3281
日誌記錄和監控 3282
使用 CloudTrail 記錄事件 3282
在 CloudTrail 中追蹤特權任務 3308
法規遵循驗證 3311
恢復能力 3312
IAM 彈性的最佳實務 3314
基礎設施安全性 3314
組態與漏洞分析 3314
AWS 受管政策 3315
IAMReadOnlyAccess 3315
IAMUserChangePassword 3315
IAMAccessAnalyzerFullAccess 3316
IAMAccessAnalyzerReadOnlyAccess 3317

AccessAnalyzerServiceRolePolicy 3318
IAMAuditRootUserCredentials 3319
IAMCreateRootUserPassword 3320
IAMDeleteRootUserCredentials 3321
S3UnlockBucketPolicy 3323
SQSUnlockQueuePolicy 3324
AccountAccessManagerServiceRolePolicy 3325
政策更新 3325
IAM 外部的安全功能 3330
IAM Access Analyzer 3332
識別與外部實體共用的資源 3332
識別對業務關鍵型資源的內部存取 3334
識別授予 IAM 使用者和角色的未使用存取權 3335
根據 AWS 最佳實務驗證政策 3335
根據您指定的安全標準驗證政策 3335
產生政策 3336
IAM Access Analyzer 定價 3336
調查結果 3336
調查結果的運作方式 3338
IAM Access Analyzer 入門 3341
調查結果儀表板 3356
檢閱調查結果 3360
篩選調查結果 3363
將問題清單存檔 3368
解決調查結果 3369
錯誤調查結果 3371
支援的資源類型 3376
委派的管理員 3383
封存規則 3385
使用 EventBridge 進行監控 3386
Security Hub CSPM 整合 3397
使用 CloudTrail 進行記錄 3405
篩選金鑰 3407
使用服務連結角色 3419
預覽存取 3421
預覽 Amazon S3 主控台中的存取 3421

使用 IAM Access Analyzer API 預覽存取 ... 3422
驗證政策檢查 ... 3425
自訂政策檢查的運作方式 ... 3425
檢查新存取權的參考政策範例 ... 3426
檢查失敗的自訂政策檢查 ... 3427
使用基本政策檢查進行驗證 ... 3427
政策檢查參考 ... 3429
使用自訂政策檢查進行驗證 ... 3561
IAM Access Analyzer 政策產生 ... 3563
政策產生的運作方式 ... 3563
服務與動作層級資訊 ... 3564
須知事項 ... 3564
必要許可 ... 3565
根據 CloudTrail 活動 (主控台) 產生政策 ... 3568
使用另一個帳戶中 AWS CloudTrail 的資料產生政策 ... 3571
根據 CloudTrail 活動 (AWS CLI) 產生政策 ... 3574
根據 CloudTrail 活動 (AWS API) 產生政策 ... 3575
IAM Access Analyzer 政策產生服務 ... 3575
IAM Access Analyzer 配額 ... 4132
對 IAM 進行疑難排解 ... 4135
我無法登入我的 AWS 帳戶 ... 4135
如果遺失存取金鑰 ... 4135
政策變數無法運作 ... 4136
我所做的變更不一定都會立刻生效 ... 4136
我未獲得授權，不得執行：iam:DeleteVirtualMFADevice ... 4137
如何安全地建立 IAM 使用者？ ... 4138
其他資源 ... 4138
拒絕存取錯誤訊息 ... 4139
向 AWS 服務提出請求時，我收到「存取遭拒」 ... 4139
當我使用臨時安全憑證來發出請求時，出現「存取遭拒」 ... 4140
拒絕存取範例 ... 4141
使用授權 ID 存取遭拒錯誤訊息 （預覽） ... 4148
概觀 ... 4148
概念和定義 ... 4149
使用 AWS 管理主控台進行故障診斷 ... 4150
使用 AWS API 或 進行故障診斷 AWS CLI ... 4151

當您未收到授權 ID 時 4151
其他考量 4151
支援的服務 4152
根使用者問題 4152
IAM 政策 4153
使用視覺化編輯器進行故障排除 4154
使用政策摘要進行疑難排解 4158
故障排除政策管理 4167
JSON 政策文件故障排除 4168
通行密鑰和 FIDO 安全金鑰 4174
我無法啟用 FIDO 安全性金鑰 4174
我無法使用 FIDO 安全性金鑰登入 4175
我的 FIDO 安全性金鑰遺失或損壞 4175
其他問題 4175
IAM 角色 4175
我無法擔任角色 4176
我的 AWS 帳戶中出現新角色 4178
我無法在 中編輯或刪除角色 AWS 帳戶 4178
我未獲授權執行： iam:PassRole 4178
為何我無法擔任具 12 小時工作階段的角色？ (AWS CLI， AWS API) 4179
當我嘗試在 IAM 主控台中切換角色時收到錯誤 4179
我的角色具有允許我執行動作的政策，但我收到「存取遭拒」 4180
服務未建立角色的預設政策版本 4180
主控台中沒有服務角色的使用案例 4181
IAM 和 Amazon EC2 4182
嘗試啟動執行個體時，我在 Amazon EC2 主控台 IAM 角色清單中看不到角色 4182
我的執行個體上的憑證針對錯誤的角色 4183
當我嘗試呼叫 AddRoleToInstanceProfile 時，出現 AccessDenied 錯誤 4183
Amazon EC2：嘗試使用角色啟動執行個體時，出現 AccessDenied 錯誤。 4184
我無法存取 EC2 執行個體上的暫時安全憑證 4184
IAM 子目錄中的 info 文件的錯誤是什麼意思？ 4185
IAM 和 Amazon S3 4186
如何授與對 Amazon S3 儲存貯體的匿名存取權？ 4186
我以 AWS 帳戶 根使用者身分登入。為何無法存取我帳戶下的 Amazon S3 儲存貯體？ 4186
SAML 2.0 聯合身分 4187
無效的 SAML 回應 4188

RoleSessionName 是必要項目 4188
未授權 AssumeRoleWithSAML 4188
無效的 RoleSessionName 字元 4189
無效的來源身分字元 4189
無效的回應簽章 4189
無效的私有金鑰 4190
無法移除私有金鑰 4190
無法移除私有金鑰，因為金鑰 ID 與私有金鑰不符 4190
無法擔任角色 4190
無法剖析中繼資料 4190
無法更新身分提供者 4191
無法將聲明加密模式設定為「必要」，因為未提供私有金鑰 4191
無法在相同的請求中新增和移除私有金鑰 4191
指定的提供者不存在。 4192
DurationSeconds 超過 MaxSessionDuration 4192
已達到私有金鑰限制 2 4192
回應沒有包括必要的對象 4192
IAM 如何與其他 AWS 服務搭配使用 4193
使用 建立 IAM 資源 CloudFormation 4193
IAM 和 CloudFormation 範本 4194
進一步了解 CloudFormation 4194
AWS CloudShell 搭配 IAM 使用 4194
取得 的 IAM 許可 AWS CloudShell 4194
與 IAM 互動 4195
使用 AWS SDKs 4197
參考資料 4199
使用 Amazon Resource Name (ARNs識別 AWS 資源 4199
ARN 格式 4199
查詢資源的 ARN 格式 4200
ARN 中的路徑 4201
IAM 識別碼 4202
易用名稱和路徑 4203
IAM ARN 4203
唯一識別碼 4210
了解唯一 ID 字首 4211
獲取唯一識別碼 4212

IAM AWS STS 和配額 4213
IAM 名稱需求 4213
IAM 物件配額 4214
IAM Access Analyzer 配額 4215
IAM Roles Anywhere 配額 4215
STS 請求配額 4216
IAM 和 STS 字元限制 4217
雙堆疊端點支援 4221
IAM 雙堆疊端點支援 4221
AWS STS 雙堆疊端點支援 4221
介面 VPC 端點 4222
VPC 端點可用性 4223
建立 IAM 的 VPC 端點 4224
建立 的 VPC 端點 AWS STS 4225
為 OIDC 探索建立 VPC AWS STS 端點 4225
AWS STS 使用 VPC 端點政策控制對 的存取 4228
使用 VPC 條件金鑰來控制聯合存取 4235
可搭配 IAM 運作的服務 4241
可搭配 IAM 運作的服務 4242
其他資訊 4269
AWS Signature 第 4 版 4272
How AWS SigV4 運作方式 4273
如何運作 AWS SigV4a 4273
簽署請求的時機 4274
為什麼要簽署請求 4274
其他資源 4275
SigV4 請求元素 4275
身分驗證方法 4278
建立已簽署請求 4283
請求簽章範例 4295
SigV4 疑難排解 4296
政策參考 4301
JSON 元素參考 4302
政策評估邏輯 4402
政策文法 4426
AWS 任務函數的 受管政策 4434

全域條件鍵 4448
IAM 條件鍵 4521
動作、資源及條件金鑰 4570
Resources 4571
身分 4571
憑證 (密碼、存取金鑰和 MFA 裝置) 4571
許可和政策 4572
聯合與委派 4572
IAM 和其他 AWS 產品 4572
搭配使用 IAM 與 Amazon EC2 4572
搭配使用 IAM 與 Amazon S3 4573
搭配使用 IAM 與 Amazon RDS 4573
搭配使用 IAM 與 Amazon DynamoDB 4573
一般安全實務 4573
一般資源 4574
提出 HTTP 查詢請求 4575
端點 4575
必要的 HTTPS 4576
簽署 IAM API 請求 4576
文件歷史紀錄 4577
..... mmmmdxcvi

什麼是 IAM？

Follow us on Twitter

AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 資源的 AWS 存取。透過 IAM，您可以管理許可，以控制使用者可以存取哪些 AWS 資源。您可以使用 IAM 來控制能通過身分驗證 (登入) 和授權使用資源的 (具有許可) 的人員。IAM 提供控制 AWS 帳戶身分驗證和授權所需的基礎設施。

如需設定 AWS 帳戶 和建立身分的說明，請參閱 IAM 入門。

身分

當您建立 時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分具有對所有 AWS 服務 和 資源的完整存取權。強烈建議不要使用根使用者來執行日常任務。有關需要根使用者憑證的任務，請參閱《IAM 使用者指南》中的需要根使用者憑證的任務。

使用 IAM 來設定根使用者以外的其他身分，例如管理員、分析師和開發人員，並授予他們在任務中取得成功所需的資源存取權。

存取管理

在 IAM 中設定使用者後，他們會使用其登入憑證對 AWS進行驗證。身分驗證是透過將登入憑證比對至信任的委託人 (IAM 使用者、 AWS STS 聯合委託人、IAM 角色或應用程式） 來提供 AWS 帳戶。接下來，系統會提出請求來向主體授與資源的存取權。如果使用者已獲得資源許可，則會授予存取權以回應授權請求。舉例來說，當您第一次登入主控台且位於主控台首頁頁面時，您並未存取特定服務。選取某項服務時，系統會將授權請求傳送至該服務，該項服務會檢查您的身分是否在授權使用者清單上、強制執行了哪些政策來控制授與的存取層級，以及任何可能生效中的其他政策。授權請求可由您內部的委託人 AWS 帳戶 或您信任 AWS 帳戶 的委託人提出。

獲得授權後，主體即可對您 AWS 帳戶中的資源採取動作或執行操作。例如，委託人可以啟動新的 Amazon Elastic Compute Cloud 執行個體、修改 IAM 群組成員資格或刪除儲存 Amazon Simple Storage Service 貯體。

> **Tip**
>
> AWS Skill Builder 提供 10 分鐘的 IAM 影片簡介：
> AWS Skill Builder 網站上的 簡介 AWS Identity and Access Management。

服務可用性

IAM 與許多 AWS 其他服務一樣，最終是一致的。IAM 可跨 Amazon 全球資料中心內多部伺服器複寫資料，來達到高可用性。如果變更一些資料的請求成功完成，則該變更經認可並安全儲存。然而，該變更必須跨 IAM 複寫，這可能需要一些時間。此類變更包括建立或更新使用者、群組、角色或政策。在應用程式的關鍵、高可用性代碼路徑中，我們不建議進行此類 IAM 變更。而應在不常運作的、單獨的初始化或設定常式中進行 IAM 變更。另外，在生產工作流程套用這些變更之前，請務必確認變更已完成傳播。如需詳細資訊，請參閱我所做的變更不一定都會立刻生效。

服務成本資訊

AWS Identity and Access Management (IAM)、 AWS IAM Identity Center 和 AWS Security Token Service (AWS STS) 是您 AWS 帳戶的功能，無需額外付費。只有在您使用 IAM 使用者或 AWS STS 臨時安全登入資料存取其他服務時 AWS ，才會向您收取費用。

會提供 IAM Access Analyzer 外部存取分析，無需額外付費。不過，您需支付未使用的存取分析和客戶政策檢查的費用。如需 IAM Access Analyzer 的完整費用與定價清單，請參閱 IAM Access Analyzer 定價。

如需其他 AWS 產品定價的資訊，請參閱 Amazon Web Services 定價頁面。

與其他 AWS 服務的整合

IAM 與許多 AWS 服務整合。如需使用 AWS IAM 的服務清單，以及服務支援的 IAM 功能，請參閱 AWS 使用 IAM 的 服務。

為什麼要使用 IAM？

AWS Identity and Access Management 是一種強大的工具，可安全地管理對 AWS 資源的存取。使用 IAM 的主要優點之一是能夠授予 AWS 您帳戶的共用存取權。此外，IAM 可讓您指派精細的許可，讓您精確控制不同使用者可以對特定資源執行的動作。此層級的存取控制對於維護 AWS 環境的安全性至關重要。IAM 也提供數個其他安全功能。可以新增多重要素驗證 (MFA) 以獲得額外的保護，並利用聯合身分無縫整合來自公司網路或其他身分提供程式的使用者。IAM 也與 整合 AWS CloudTrail，提供詳細的記錄和身分資訊，以支援稽核和合規要求。透過利用這些功能，您可以協助確保對關鍵 AWS 資源的存取受到嚴格控制且安全。

共用存取您的 AWS 帳戶

您可以授與其他人管理和使用 AWS 帳戶資源的許可，而無需共用您的密碼或存取金鑰。

精密許可

您可以對不同的人員授與不同資源的不同許可。例如，您可以允許某些使用者完整存取 Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Simple Storage Service (Amazon S3)、Amazon DynamoDB、Amazon Redshift 和其他 AWS 服務。對於其他使用者，您可以只允許對一些 Amazon S3 儲存貯體進行唯讀存取，或僅具有管理一些 Amazon EC2 執行個體的許可，或存取您的帳單資訊，但不能進行任何其他動作。

安全存取在 Amazon EC2 上執行的應用程式 AWS 的資源

您可以使用 IAM 功能為在 EC2 執行個體上執行的應用程式安全地提供憑證。這些登入資料為您的應用程式提供存取其他 AWS 資源的許可。範例包括 S3 儲存貯體和 DynamoDB 表格。

多重要素驗證 (MFA)

您可以為帳戶和個別使用者新增雙重要素身分驗證，以提高安全性。使用 MFA，您或您的使用者不僅必須提供密碼或存取金鑰才能使用您的帳戶，還必須提供來自特殊設定裝置的代碼。如果您已將 FIDO 安全金鑰與其他 服務搭配使用，且具有 AWS 支援的組態，則可以使用 WebAuthn 確保 MFA 安全。如需詳細資訊，請參閱使用通行密鑰和安全金鑰的支援組態

聯合身分

您可以允許已經在其他地方擁有密碼的使用者 (例如，在您的公司網路中或使用網際網路身分提供程式) 存取您的 AWS 帳戶。這些使用者會獲得符合 IAM 最佳實務建議的臨時憑證。使用聯合身分可增強 AWS 帳戶的安全性。

身分資訊保證

如果您使用 AWS CloudTrail，則會收到日誌記錄，其中包含有關在您的帳戶中請求資源的資訊。該資訊是根據 IAM 身分。

PCI DSS 合規

IAM 支援處理、儲存、傳輸商家或服務供應商的信用卡資料，並且已驗證合規於支付卡產業 (PCI) 資料安全標準 (DSS)。如需 PCI DSS 的詳細資訊，包括如何請求 AWS PCI 合規套件的副本，請參閱 PCI DSS 第 1 級。

IAM 的使用時機為何？

AWS Identity and Access Management 是一種核心基礎設施服務，可提供根據內部身分進行存取控制的基礎 AWS。您每次存取 AWS 帳戶時都會使用 IAM。使用 IAM 的方式將取決於組織中的具體責任和工作職責。 AWS 服務的使用者使用 IAM 存取day-to-day工作所需的 AWS 資源，管理員會授予適當的許可。另一方面，IAM 管理員負責管理 IAM 身分並撰寫政策，以控制對 資源的存取。無論您的角色為何，每當您驗證和授權存取 AWS 資源時，都會與 IAM 互動。這可能包括以 IAM 使用者身分登入、充當 IAM 角色，或利用聯合身分進行無縫存取。了解各種 IAM 功能和使用案例對於有效管理 AWS 環境的安全存取至關重要。在建立政策和許可時，IAM 可提供靈活且精細的方法。除了指定使用者或角色可存取的動作和資源的身分型政策之外，還可以定義信任政策來控制哪些主體可以擔任某個角色。透過設定這些 IAM 政策，可以協助確保使用者和應用程式具有適當的許可層級來執行其所需任務。

執行不同工作職能時

AWS Identity and Access Management 是一種核心基礎設施服務，可提供根據內部身分進行存取控制的基礎 AWS。每次存取 AWS 帳戶時都會使用 IAM。

使用 IAM 的方式會有所不同，具體取決於您在 AWS中所執行的工作。

- 服務使用者 – 如果您使用 AWS 服務來執行任務，您的管理員會為您提供所需的登入資料和許可。隨著您為了執行作業而使用更多的進階功能，您可能會需要額外的許可。了解存取許可的管理方式可協助您向管理員請求正確的許可。
- 服務管理員 – 如果您負責公司 AWS 的資源，您可能擁有 IAM 的完整存取權。您的任務是判斷服務使用者應存取的 IAM 功能及資源。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。
- IAM 管理員：如果您是 IAM 管理員，您可以透過管理 IAM 身分和寫入政策來管理 IAM 存取權。

當您獲得存取 AWS 資源的授權時

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色身分進行身分驗證。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的如何登入您的 AWS 帳戶。

對於程式設計存取， AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的 API 請求的AWS 第 4 版簽署程序。

以 IAM 使用者身分登入時

IAM 使用者https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html是一種身分具備單人或應用程式的特定許可權。建議以臨時憑證取代具備長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 。

IAM 群組會指定 IAM 使用者集合，使管理大量使用者的許可權更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 使用者的使用案例。

擔任 IAM 角色時

IAM 角色https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html的身分具有特定許可權，其可以提供臨時憑證。您可以透過從使用者切換到 IAM 角色 （主控台） 或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的擔任角色的方法。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權與跨服務存取，以及在 Amazon EC2 執行的應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

建立政策和許可時

您能夠建立政策，進而將許可授與給使用者。該政策文件中會列出使用者可執行的動作，以及會受到這些動作影響的資源。根據預設，未明確允許的任何動作或資源將被拒絕。可建立政策並連接至主體 (使用者、使用者群組、使用者擔任的角色以及資源)。

可搭配使用這些政策與 IAM 角色：

- 信任政策：定義哪些主體可以在哪些條件下擔任該角色。信任政策是一項專屬於 IAM 角色的資源型政策。一個角色只能由一項信任政策。
- 身分型政策 (內嵌和受管)：這些政策會定義角色使用者能夠執行 (或無法執行) 的許可，以及在哪些資源上執行。

使用 以身分為基礎的 IAM 政策範例 可協助您定義 IAM 身分的許可。在您找到所需的政策後，選擇「查看政策」以查看政策的 JSON。您可以將 JSON 政策文件用作自己政策的範本。

> **Note**
>
> 如果使用 IAM Identity Center 來管理使用者，您可以在 IAM Identity Center 指派許可集，而不是將許可政策連接至主體。當您將許可集指派給群組或 時 AWS IAM Identity Center 的使用者，IAM Identity Center 會在每個帳戶中建立對應的 IAM 角色，並將許可集中指定的政策連接

到這些角色。IAM Identity Center 會負責管理角色，並允許您定義的授權使用者擔任該角色。修改許可集後，IAM Identity Center 會確保有據此更新對應的 IAM 政策和角色。
您也可以使用帳戶存取管理員 — IAM 功能，可讓您將建立和管理的現有 IAM 角色指派給 IAM Identity Center 使用者和群組，讓您存取完整的 IAM 角色功能集。您可以搭配許可集或單獨使用。
如需有關 IAM Identity Center 的詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

如何管理 IAM？

AWS Identity and Access Management 在 AWS 環境中進行管理涉及利用各種工具和界面。最常見的方法是透過 Web 型界面 AWS 管理主控台，可讓您執行各種 IAM 管理任務，從建立使用者和角色到設定許可。

對於更熟悉命令列界面的使用者， AWS 提供兩組命令列工具： AWS Command Line Interface 和 AWS Tools for Windows PowerShell。它們可讓您直接從終端發出 IAM 相關命令，通常比導覽主控台更有效率。此外， AWS CloudShell 可讓您使用與主控台登入相關聯的許可，直接從 Web 瀏覽器執行 CLI 或 SDK 命令。

除了主控台和命令列之外， AWS 還提供各種程式設計語言的軟體開發套件 (SDKs)，可讓您將 IAM 管理功能直接整合到您的應用程式。或者，您可以使用 IAM 查詢 API (允許您直接向服務發出 HTTPS 請求)，以程式設計方式存取 IAM。利用這些不同的管理方法，可靈活地將 IAM 整合到現有的工作流程和程序中。

使用 AWS 管理主控台

AWS 管理主控台是一種 Web 應用程式，包含並參考各種用於管理 AWS 資源的服務主控台。若是首次登入，這時主控台頁面將會顯示。首頁提供每個服務主控台的存取權，並提供單一位置來存取執行 AWS 相關任務的資訊。登入 主控台後，您可以使用哪些服務和應用程式，取決於您有權存取 AWS 的資源。您可以透過擔任某個角色、加入已取得許可的群組或明確取得相關許可，從而獲得對資源的許可。對於獨立的 AWS 帳戶，根使用者或 IAM 管理員會設定資源的存取權。對於 AWS Organizations，管理帳戶或委派的管理員會設定資源的存取權。

如果您計劃讓人員使用 AWS 管理主控台來管理 AWS 資源，我們建議您使用暫時登入資料來設定使用者，做為安全最佳實務。已擔任角色的 IAM 使用者、聯合身分主體和 IAM Identity Center 中的使用者擁有臨時憑證，而 IAM 使用者和根使用者擁有長期憑證。根使用者登入資料提供 的完整存取權 AWS 帳戶，而其他使用者則擁有登入資料，可存取 IAM 政策授予他們的資源。

登入體驗因不同類型的 AWS 管理主控台 使用者而異。

- IAM 使用者和根使用者從主要登入 URL (https://signin.aws.amazon.com：//) AWS 登入。一旦登入，他們便可存取已取得許可的帳戶中的資源。

  若要以根使用者身分登入，您必須有根使用者電子郵件地址和密碼。

  若要以 IAM 使用者身分登入，您必須擁有 AWS 帳戶 號碼或別名、IAM 使用者名稱和 IAM 使用者密碼。

  我們建議您將帳戶中的 IAM 使用者限定於需要長期憑證的特定情形，例如緊急存取，而且您只會在需要根使用者憑證的任務中使用根使用者。

  為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住 IAM 使用者名稱和帳戶資訊。下次使用者前往 中的任何頁面時 AWS 管理主控台，主控台會使用 Cookie 將使用者重新導向至帳戶登入頁面。

  完成您的工作階段後，請登出主控台以防止重複使用您先前的登入。
- IAM Identity Center 使用者使用其組織獨有的特定 AWS 存取入口網站登入。登入後，他們可以選擇要存取的帳戶或應用程式。如果選擇存取某個帳戶，他們會選擇想要在管理工作階段中使用哪個許可集。
- 外部身分提供者管理的 OIDC 和 SAML 聯合身分主體會連結至使用自訂企業存取入口網站的 AWS 帳戶 登入。使用者可以使用的 AWS 資源取決於其組織所選取的政策。

> **Note**
>
> 為了提供額外層級的安全性，根使用者、IAM 使用者和 IAM Identity Center 中的使用者可以在授予 AWS 資源存取權 AWS 之前，由 驗證多重要素驗證 (MFA)。當啟用 MFA 時，您還必須能夠存取要登入的 MFA 裝置。

若要進一步了解不同使用者如何登入 管理主控台，請參閱 登入使用者指南中的登入 AWS 管理主控台。 AWS

AWS 命令列工具

您可以使用 AWS 命令列工具，在系統的命令列發出命令，以執行 IAM 和 AWS 任務。使用命令列可以比主控台更快，也更便利。如果您想要建置執行 AWS 任務的指令碼，命令列工具也很有用。

AWS 提供兩組命令列工具： AWS Command Line Interface(AWS CLI) 和 AWS Tools for Windows PowerShell。如需安裝和使用 的詳細資訊 AWS CLI，請參閱AWS Command Line Interface 《使用者指南》。如需安裝和使用 Tools for Windows PowerShell 的詳細資訊，請參閱 AWS Tools for PowerShell 使用者指南。

登入 主控台後，您可以從 AWS CloudShell 瀏覽器使用 來執行 CLI 或 SDK 命令。存取 AWS 資源的許可取決於您用來登入 主控台的登入資料。根據您的經驗，您可能會發現 CLI 是管理 AWS 帳戶的更有效率的方法。如需詳細資訊，請參閱使用 AWS CloudShell 搭配 AWS Identity and Access Management

AWS 命令列界面 (CLI) 和軟體開發套件 (SDKs)

當 IAM Identity Center 和 IAM 使用者透過相關 SDK 中的 CLI 或應用程式介面 (API) 驗證身分時，他們會使用不同方法來驗證其憑證。

登入資料和組態設定位於多個位置，例如系統或使用者環境變數、本機 AWS 組態檔案，或在命令列上明確宣告為參數。某些位置的優先順序高於其他位置。

IAM Identity Center 和 IAM 均提供可與 CLI 或 SDK 搭配使用的存取金鑰。IAM Identity Center 存取金鑰是可自動整理的臨時憑證，並且優於與 IAM 使用者相關聯的長期存取金鑰。

若要 AWS 帳戶 使用 CLI 或 SDK 管理 ，您可以從 AWS CloudShell 瀏覽器使用 。如果您使用 CloudShell 執行 CLI 或 SDK 命令，必須先登入主控台。存取 AWS 資源的許可取決於您用來登入 主控台的登入資料。根據您的經驗，您可能會發現 CLI 是管理 AWS 帳戶的更有效率的方法。

對於應用程式開發，您可以將 CLI 或 SDK 下載到您的電腦，然後從命令提示或 Docker 視窗進行登入。在這種情況下，您要將身分驗證和存取憑證設定為 CLI 指令碼或 SDK 應用程式的一部分。您可以採用不同的方式設定對資源的程式設計存取，具體取決於適用於您的環境和存取權。

- 使用 AWS 服務驗證本機程式碼的建議選項是 IAM Identity Center 和 IAM Roles Anywhere
- 要驗證在 AWS 環境中執行的程式碼，建議的選項是使用 IAM 角色或使用 IAM Identity Center 憑證。

使用 AWS 存取入口網站登入時，您可以從選擇許可集的開始頁面取得短期憑證。這些憑證具有定義的持續時間，不會自動重新整理。如果您想要使用這些登入資料，請在登入 AWS 入口網站後，選擇 ，AWS 帳戶 然後選擇許可集。選取命令列或程式設計存取，以檢視可用於以程式設計方式或從 CLI 存取 AWS 資源的選項。如需有關這些方法的詳細資訊，請參閱《IAM Identity Center 使用者指南》中的取得與重新整理臨時憑證。這些憑證經常在應用程式開發期間使用，以便快速測試程式碼。

我們建議您使用 IAM Identity Center 登入資料，在自動化對 AWS 資源的存取時自動重新整理。如果您已在 IAM Identity Center 中設定使用者和許可集，您可以使用 `aws configure sso` 命令以便利用命令列精靈來協助您識別可用的憑證，並且將它們存放在設定檔中。如需有關設定您的設定檔的更多資訊，請參閱《AWS 命令列介面使用者指南 (適用於版本 2)》中的使用 `aws configure sso` 精靈設定您的設定檔。

> **Note**
>
> 許多範例應用程式使用與 IAM 使用者或根使用者相關聯的長期存取金鑰。您應該只在沙盒環境中使用長期憑證，這是您學習練習的一部分。檢閱長期存取金鑰的替代方案並制定轉換程式碼計畫，使其儘快使用替代憑證，例如 IAM Identity Center 憑證或 IAM 角色。在轉換程式碼後，請刪除存取金鑰。

若要進一步了解如何設定 CLI，請參閱《 第 2 AWS 版的命令列界面使用者指南》中的安裝或更新最新版本的 CLI，以及《 AWS 命令列界面使用者指南》中的身分驗證和存取登入資料。 AWS

若要詳細了解如何設定軟體開發套件，請參閱《AWS SDK 和工具參考指南》中的 IAM Identity Center 身分驗證和《AWS 軟體開發套件和工具參考指南》中的 IAM Roles Anywhere。

使用 AWS SDKs

AWS 提供 SDKs（軟體開發套件），其中包含適用於各種程式設計語言和平台 (Java、Python、Ruby、.NET、iOS、Android 等) 的程式庫和範本程式碼。SDKs提供便捷的方式來建立對 IAM 和 的程式設計存取 AWS。例如，開發套件會負責的工作諸如以密碼演算法簽署請求、管理錯誤以及自動重試請求。如需 AWS SDKs 的相關資訊，包括如何下載和安裝，請參閱適用於 Amazon Web Services 的工具頁面。

使用 IAM 查詢 API

您可以使用 IAM 查詢 API 以 AWS 程式設計方式存取 IAM 和 ，這可讓您直接向 服務發出 HTTPS 請求。使用查詢 API 時，您必須加入程式碼，才能夠使用您的登入資料以數位方式簽署請求。如需詳細資訊，請參閱 使用 HTTP 查詢請求呼叫 IAM API 和 IAM API 參考。

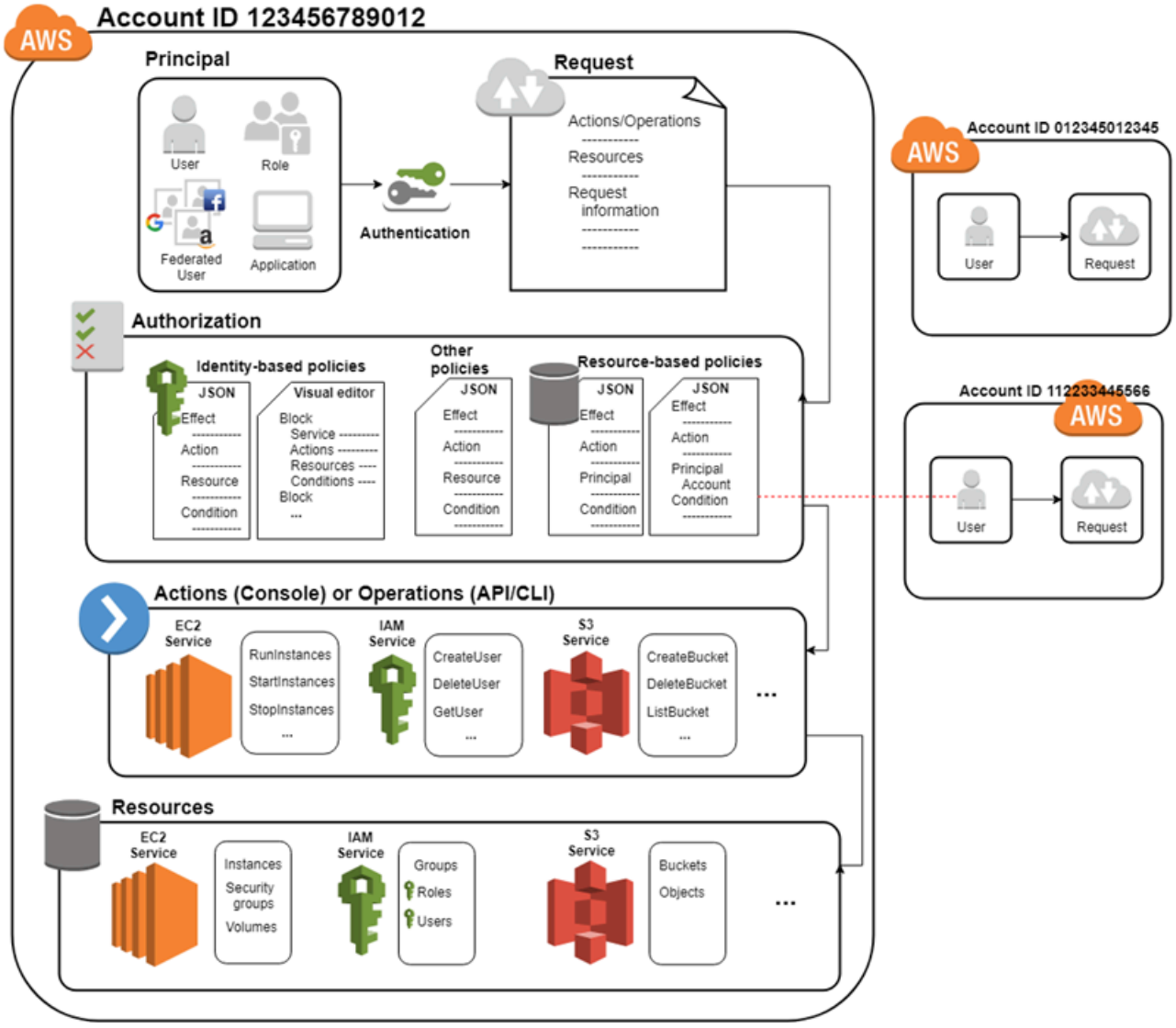
IAM 的運作方式

AWS Identity and Access Management 提供必要的基礎設施，以控制 的身分驗證和授權 AWS 帳戶。

首先，人類使用者或應用程式會使用其登入憑證來向 AWS進行身分驗證。IAM 會將登入憑證比對至 信任的委託人 (IAM 使用者、 AWS STS 聯合身分使用者委託人、IAM 角色或應用程式），AWS 帳戶 並驗證委託人以存取 AWS。

接下來，IAM 會提出請求，向主體授予資源的存取權。IAM 根據授權請求來授予或拒絕存取權。舉例來說，當您第一次登入主控台且位於主控台首頁頁面時，您並未存取特定服務。當您選取服務時，會針對該服務將授權請求傳送至 IAM。IAM 會驗證您的身分是否在授權使用者清單中，確定哪些政策控制授予的存取層級，並評估可能有效的任何其他政策。您內部 AWS 帳戶 或您 AWS 帳戶 信任之其他 的委託人可以提出授權請求。

獲得授權後，主體即可對 AWS 帳戶中的資源執行動作或操作。例如，委託人可以啟動新的 Amazon Elastic Compute Cloud 執行個體、修改 IAM 群組成員資格或刪除儲存 Amazon Simple Storage Service 貯體。下圖透過 IAM 基礎設施說明此流程：



請求的元件

當委託人嘗試使用 AWS 管理主控台、 AWS API 或 時 AWS CLI，該委託人會傳送請求至 AWS。請求包含下列資訊：

- 動作或操作 – 委託人想要執行的動作或操作，例如 中的 動作 AWS 管理主控台，或 AWS CLI 或 AWS API 中的 操作。
- 資源 – 委託人請求執行動作或操作 AWS 的資源物件。
- 主體 – 使用實體 (使用者或角色) 來傳送請求的人員或應用程式。主體的相關資訊包括許可政策。
- 環境資料：IP 地址、使用者代理程式、啟用 SSL 的狀態以及時間戳記的相關資訊。
- 資源資料：與所請求資源相關的資料，例如 DynamoDB 資料表名稱或 Amazon EC2 執行個體上的標籤。

AWS 會將請求資訊收集到請求內容中，IAM 會評估這些內容來授權請求。

如何對主體進行身分驗證

委託人 AWS 使用 IAM 驗證的登入資料登入 ，以允許委託人傳送請求 AWS。有些服務，例如 Amazon S3 和 AWS STS，允許匿名使用者提出特定請求。不過，這些服務是此規則的例外。每種類型的使用者都會經過身分驗證。

- 根使用者 – 用於身分驗證的登入憑證是您用來建立 AWS 帳戶 和您當時指定的密碼的電子郵件地址。
- 聯合委託人 – 您的身分提供者會驗證您的身分，並將您的登入資料傳遞給 AWS，您不需要直接登入 AWS。IAM Identity Center 和 IAM 都支援聯合身分。
- 中的使用者 AWS IAM Identity Center 目錄 （ 非聯合 ） – 直接在 IAM Identity Center 預設目錄中建立的使用者使用 AWS 存取入口網站登入，並提供您的使用者名稱和密碼。
- IAM 使用者：您可以透過提供您的帳戶 ID 或別名、使用者名稱以及密碼來登入。若要從 API 或 驗證工作負載 AWS CLI，您可以透過擔任角色來使用暫時登入資料，也可以透過提供存取金鑰和私密金鑰來使用長期登入資料。

  若要進一步了解 IAM 實體，請參閱 IAM 使用者 和 IAM 角色。

AWS 建議您對所有使用者使用多重驗證 (MFA)，以提高帳戶的安全性。若要進一步了解 MFA，請參閱 AWS IAM 中的多重要素驗證。

授權和許可政策基本概念

授權是指擁有完成其請求所需許可的主體。在授權期間，IAM 使用請求內容中的值來識別適用於請求的政策。接著使用政策以決定是否允許或拒絕請求。IAM 會將大多數許可政策儲存為 JSON 文件，它們可指定主體實體的許可。

有數種類型的政策會影響授權請求。若要提供使用者存取帳戶中 AWS 資源的許可，您可以使用身分型政策。資源型政策可授予跨帳戶存取權。若要在不同的帳戶中發出請求，在其他帳戶中的政策必須可讓您存取資源，而且您用來發出請求的 IAM 實體必須具備允許該請求的身分型政策。

IAM 會檢查適用於請求內容的每個政策。IAM 政策評估使用明確拒絕，這表示如果單一許可政策包含拒絕的動作，則 IAM 會拒絕整個請求並停止評估。由於預設拒絕請求，因此適用的許可政策必須允許 IAM 請求的每個部分，以授權您的請求。用於單一帳戶中請求的評估邏輯遵循以下基本規則：

- 根據預設，所有的請求一律拒絕。(一般而言，一律允許使用帳戶中的資源的 AWS 帳戶根使用者 登入資料提出請求)。
- 任何許可政策中的明確允許 (以身分為基礎或以資源為基礎的政策) 都會覆寫此預設值。
- AWS Organizations 服務控制政策 (SCP) 或資源控制政策 (RCP)、IAM 許可界限或工作階段政策的存在會覆寫允許。如果這些政策類型中有一或多個存在，則它們必須全部允許該請求。否則，它會被隱含拒絕。如需 SCP 和 RCP 的詳細資訊，請參閱 AWS Organizations User Guide 中 Authorization policies in AWS Organizations。
- 任何政策中的明確拒絕會覆寫任何政策中的任何允許。

如需詳細資訊，請參閱 政策評估邏輯。

在 IAM 驗證並授權主體之後，IAM 會透過評估適用於主體的許可政策來核准其請求中的動作或操作。每個 AWS 服務都會定義其支援的動作 （操作），並包含您可以對資源執行的動作，例如檢視、建立、編輯和刪除該資源。適用於主體的許可政策必須包含執行操作所需的動作。若要要進一步了解 IAM 如何評估許可政策，請參閱 the section called “政策評估邏輯”。

該服務定義了主體可對每個資源執行的一組動作。建立許可政策時，務必包含您希望使用者執行的動作。例如，IAM 支援超過 40 個使用者資源動作，包括下列基本動作：

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

此外，可以在許可政策中指定條件，以便在請求符合指定條件時提供對資源的存取。例如，您可能想要政策陳述式在特定日期之後生效，或者在 API 請求中出現特定值時控制存取。若要指定條件，請使用政策陳述式的 `Condition` 元素。

在 IAM 核准請求中的操作之後，主體可以使用帳戶中的相關資源。資源是存在於服務內的物件。範例包括 Amazon EC2 執行個體、IAM 使用者和 Amazon S3 儲存貯體。如果主體建立請求，以對未包含在許可政策中的資源執行動作，則服務會拒絕該請求。例如，如果您擁有刪除 IAM 角色的許可，但您請求刪除 IAM 群組，則如果您沒有刪除 IAM 群組的許可，則請求會失敗。若要進一步了解不同 AWS 服務支援的動作、資源和條件索引鍵，請參閱 AWS 服務的動作、資源和條件索引鍵。

比較 IAM 身分和憑證

在 中管理的身分 AWS Identity and Access Management 是 IAM 使用者、IAM 角色和 IAM 群組。這些身分是與您的 一起 AWS 建立的根使用者以外的身分 AWS 帳戶。

強烈建議您不要以根使用者處理日常作業，即使是管理作業。反之，應佈建其他使用者，並授予他們執行必要任務所需的許可。您可以將人員新增至 IAM Identity Center 目錄、將外部身分提供者與 IAM Identity Center 或 IAM 聯合，或建立最低權限的 IAM 使用者，以新增使用者。

為了提高安全性，我們建議您集中根存取，以協助您集中保護 AWS 帳戶 受管 的根使用者憑證 AWS Organizations。 集中管理成員帳戶的根存取權可讓您集中移除和防止長期根使用者憑證復原，防止大規模的意外根存取。啟用集中式根存取權後，您可以擔任特權工作階段，對成員帳戶執行動作。

設定使用者之後，您可以將 的存取權授予 AWS 帳戶 特定人員，並提供他們存取 資源的許可。

作為最佳實務， AWS 建議您要求人類使用者擔任 IAM 角色來存取 ， AWS 以便他們使用臨時登入資料。如果您要管理 IAM Identity Center 目錄中的身分，或與身分提供者聯合使用身分，請遵循以下最佳實務。

條款

使用 IAM 身分時通常會使用這些術語：

IAM 資源

IAM 服務會存放這些資源。您可以從主控台中新增、編輯和移除這些資源。

- IAM 使用者
- IAM 群組

- IAM 角色
- 許可政策
- 身分提供者物件

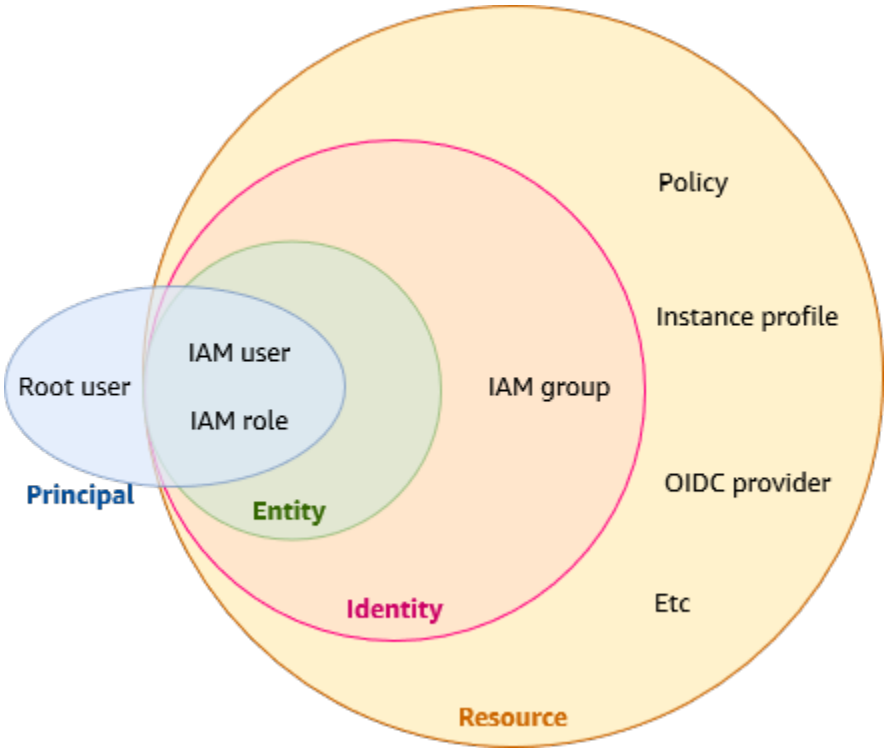
IAM 實體

AWS 用於身分驗證的 IAM 資源。在資源型政策中將實體指定為主體。

- IAM 使用者
- IAM 角色

IAM 身分

在政策中授權以執行動作和存取資源的 IAM 資源。身分包括 IAM 使用者、IAM 群組及 IAM 角色。



主體

可對 AWS 資源提出動作或操作請求的 AWS 帳戶根使用者、IAM 使用者或 IAM 角色。主體包括人類使用者、工作負載、聯合身分主體及擔任的角色。身分驗證之後，IAM 會根據委託人類型 AWS，授予委託人永久或暫時的登入資料來提出請求。

人類使用者也稱為人類身分，例如相關人員、管理員、開發人員、操作員和應用程式消費者。

工作負載是提供商業價值的資源和程式碼集合，例如應用程式、流程、操作工具和其他元件。

聯合主體是身分和登入資料由另一個身分提供者管理的使用者，例如 Active Directory、Okta 或 Microsoft Entra。

IAM 角色是您可以在帳戶中建立的 IAM 身分，具有特定許可，可決定身分可以和不可以執行的動作。但是，角色的目的是讓需要它的任何人可代入，而不是單獨地與某個人員關聯。

IAM 授予 IAM 使用者和根使用者長期憑證和 IAM 角色臨時憑證。 AWS IAM Identity Center 的使用者、OIDC 和 SAML 聯合身分主體在登入時擔任 IAM 角色 AWS，這會授予他們臨時憑證。最佳實務是，建議您要求人類使用者和工作負載使用臨時登入資料存取 AWS 資源。

IAM 使用者和 IAM Identity Center 之間的不同之處

IAM 使用者不是單獨的帳戶；他們是您帳戶內的個別使用者。每個使用者都有自己的密碼來存取 AWS 管理主控台。您也可以為每個使用者建立個別的存取金鑰，讓使用者能利用程式設計方式提出請求，進而能使用您帳戶中的資源。

IAM 使用者及其存取金鑰具有 AWS 資源的長期登入資料。IAM 使用者的主要用途是讓無法使用 IAM 角色的工作負載能夠使用 API 或 CLI 向 AWS 服務提出程式設計請求。

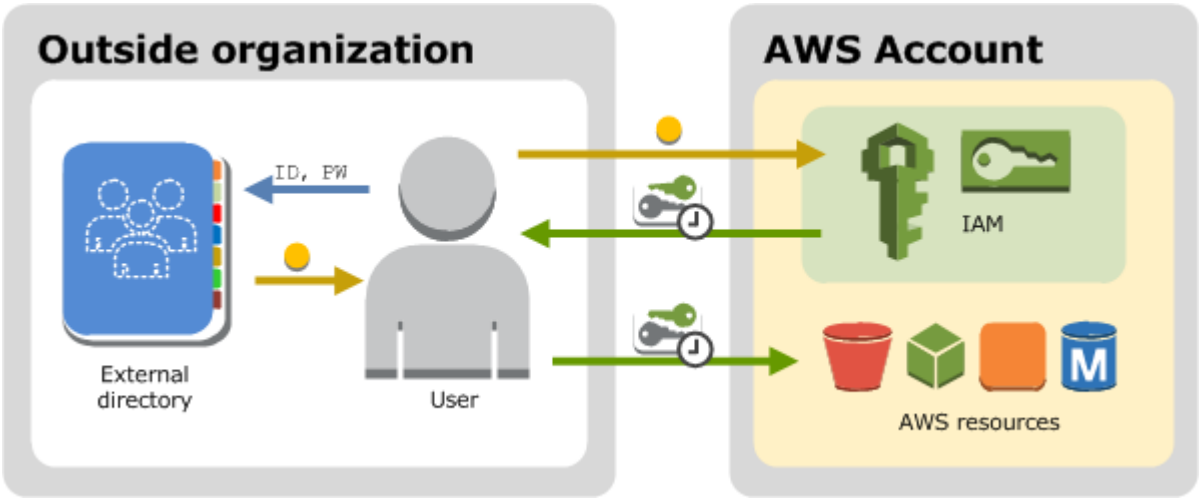
> **Note**
>
> 對於需要具有程式化存取和長期憑證的 IAM 使用者的情況，我們建議您視需要更新存取金鑰。如需詳細資訊，請參閱更新存取金鑰。

人力身分 （ 人員 ） 具有不同的許可需求AWS IAM Identity Center 的使用者，取決於他們正在執行的角色，並且可以 AWS 帳戶 跨組織在各種 中工作。如果您有需要存取金鑰的使用案例，您可以使用支援這些使用案例 AWS IAM Identity Center 的使用者。透過 AWS 存取入口網站登入的人員可以取得具有 AWS 資源短期憑證的存取金鑰。如果是集中式存取管理，我們建議您使用 AWS IAM Identity Center (IAM Identity Center) 管理他人對您帳戶的存取權和這些帳戶內的許可。IAM Identity Center 會自動設定 Identity Center 目錄做為您的預設身分來源，您可以在其中新增人員和群組，並將他們的存取層級指派給您的 AWS 資源。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 AWS IAM Identity Center。

這兩種使用者的主要區別在於，IAM Identity Center 中的使用者在存取管理主控台或 AWS 資源 AWS 之前，會在登入 時自動擔任 IAM 角色。每次使用者登入時，IAM 角色都會授予臨時登入資料 AWS。若要讓 IAM 使用者使用 IAM 角色登入，他們必須具有擔任和切換角色的許可，而且必須在存取 AWS 帳戶後明確選擇切換到想要擔任的角色。

來自現有身分來源的聯合身分使用者

如果組織中的使用者已在登入公司網路時進行身分驗證，則您不需要為他們建立個別 IAM 使用者，或建立 IAM Identity Center 使用者。反 AWS 之，您可以使用 IAM 或 將這些使用者身分聯合到 AWS IAM Identity Center。OIDC 和 SAML 聯合身分主體擔任 IAM 角色，該角色可為其提供存取特定資源的許可。如需角色的詳細資訊，請參閱角色術語和概念。



在下列情況下，聯合很有用：

- 您的使用者已存在於公司目錄。

  如果您的公司目錄與安全性聲明標記語言 2.0 (SAML 2.0) 相容，您可以設定公司目錄， AWS 管理主控台 為您的使用者提供 的單一登入 (SSO) 存取權。如需詳細資訊，請參閱暫時性憑證的常見案例。

  如果您的公司目錄與 SAML 2.0 不相容，您可以建立身分代理程式應用程式， AWS 管理主控台 為您的使用者提供 的單一登入 (SSO) 存取權。如需詳細資訊，請參閱啟用 AWS 主控台的自訂身分代理程式存取。

  如果您的公司目錄是 Microsoft Active Directory，您可以使用 AWS IAM Identity Center 來連接 Active Directory 中的自我管理目錄或 中的目錄AWS Directory Service，以在公司目錄與 之間建立信任 AWS 帳戶。

  如果您使用 Okta 或 Microsoft Entra 等外部身分提供者 (IdP) 來管理使用者，您可以使用 AWS IAM Identity Center 來建立 IdP 與 之間的信任 AWS 帳戶。如需詳細資訊，請參閱《AWS IAM Identity Center 使用者指南》 中的 連結至外部身分供應商。

- 您的使用者已經有網際網路身分。

如果您正在建立行動應用程式或 Web 為基礎的應用程式，讓使用者透過網際網路身分提供者 (例如，Login with Amazon、Facebook、Google 或任何與 OpenID Connect (OIDC) 相容的身分提供者) 識別自己的身分，則該應用程式可以使用聯合功能存取 AWS。如需詳細資訊，請參閱OIDC 聯合身分。

> **Tip**
>
> 若要搭配網際網路身分提供者使用聯合身分，建議您使用 Amazon Cognito。

提供使用者存取權的不同方法

以下是您可以提供 AWS 資源存取權的方法。

使用者存取類型	何時使用？	在哪裡可以獲得更多資訊？
針對相關人員 (例如您的人力使用者)，使用 IAM Identity Center 的單一登入存取來存取 AWS 資源	IAM Identity Center 提供集中位置，可集中管理使用者及其對 AWS 帳戶 和雲端應用程式的存取。 您可以在 IAM Identity Center 中設定身分存放區，或者您也可以使用現有的身分提供者 (IdP) 設定聯合。安全最佳實務建議將有限登入資料授予您的人類使用者 AWS 。 相關人員擁有更簡單的登入體驗，而您也能從單個系統持續控制他們的資源存取權。IAM Identity Center 支援多重要素驗證 (MFA)，以提高對帳戶安全的保護。	如需有關設定 IAM Identity Center 的詳細資訊，請參閱《AWS IAM Identity Center 使用者指南》中的入門。 如需有關使用 IAM Identity Center 中的 MFA 的詳細資訊，請參閱《AWS IAM Identity Center 使用者指南》中的多重要素驗證。

使用者存取類型	何時使用？	在哪裡可以獲得更多資訊？
針對人類使用者 (例如您的人力使用者)，使用 IAM 身分提供者 (IdP) 的聯合存取來存取 AWS 服務	IAM 支援 IdP，其與 OpenID Connect (OIDC) 或 SAML 2.0 (安全性聲明標記語言 2.0) 相容。建立 IAM 身分提供者後，可建立一個或多個 IAM 角色，這些角色可以動態指派給聯合身分主體。	如需有關 IAM 身分提供者與聯合的詳細資訊，請參閱 身分提供者和 的聯合身分 AWS。
之間的跨帳戶存取 AWS 帳戶	您想要與其他 中的使用者共用特定 AWS 資源的存取權 AWS 帳戶。 角色是授予跨帳戶存取權的主要方式。不過，一些 AWS 服務支援資源型政策，允許您直接將政策連接到資源 (而不是使用角色做為代理)。	如需關於 IAM 角色的詳細資訊，請參閱 IAM 角色。 如需服務連結角色的詳細資訊，請參閱建立服務連結角色。 如需有關哪些服務支援使用服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務。尋找服務連結角色欄中顯示為是的服務。若要檢視該服務的服務連結角色文件，請選擇該欄中與 Yes (是) 相關聯的連結。

使用者存取類型	何時使用？	在哪裡可以獲得更多資訊？
您 中指定 IAM 使用者的長期登入資料 AWS 帳戶	您可能有特定的使用案例，需要具有 IAM 使用者的長期登入資料 AWS。您可以使用 IAM 在 中建立這些 IAM 使用者 AWS 帳戶，並使用 IAM 管理其許可。以下是部分使用案例： • 無法使用 IAM 角色的工作負載 • 需要透過存取金鑰進行程式設計存取的第三方 AWS 用戶端 • AWS CodeCommit 或 Amazon Keyspaces 的服務特定登入資料 • AWS IAM Identity Center 不適用於您的帳戶，而且您沒有其他身分提供者 對於需要具有程式化存取和長期憑證的 IAM 使用者的情況，我們建議的最佳實務是視需要更新存取金鑰。如需詳細資訊，請參閱更新存取金鑰。	如需有關設定 IAM 使用者的詳細資訊，請參閱在 中建立 IAM 使用者 AWS 帳戶。 如需有關 IAM 使用者存取金鑰的詳細資訊，請參閱 管理 IAM 使用者的存取金鑰。 如需 AWS CodeCommit 或 Amazon Keyspaces 的服務特定登入資料的詳細資訊，請參閱 CodeCommit 的 IAM 登入資料：Git 登入資料、SSH 金鑰和 AWS 存取金鑰和 將 IAM 與 Amazon Keyspaces (適用於 Apache Cassandra) 搭配使用。

支援程式設計使用者存取

如果使用者想要與 AWS 外部互動，則需要程式設計存取 AWS 管理主控台。授予程式設計存取權的方式取決於正在存取的使用者類型 AWS：

- 如果您在 IAM Identity Center 中管理身分， AWS APIs需要設定檔，而 AWS Command Line Interface 需要設定檔或環境變數。

- 如果您有 IAM 使用者， AWS APIs和 AWS Command Line Interface 需要存取金鑰。如有可能，請建立臨時憑證，其中包含存取金鑰 ID、私密存取金鑰，以及指出憑證何時到期的安全記號。

若要授予使用者程式設計存取權，請選擇下列其中一個選項。

哪個使用者需要程式設計存取權？	選項	其他資訊
人力資源身分 (IAM Identity Center 中管理的人員和使用者)	使用短期登入資料簽署對 AWS CLI 或 AWS APIs程式設計請求（直接使用或使用 AWS SDKs)。	對於 AWS CLI，請遵循《使用者指南》中的取得 CLI 存取的 IAM 角色登入資料中的指示。AWS IAM Identity Center 對於 AWS APIs，請遵循 AWS SDKs 和工具參考指南中 SSO 登入資料的指示。
IAM 使用者	使用短期登入資料簽署對 AWS CLI 或 AWS APIs程式設計請求（直接使用或使用 AWS SDKs)。	請遵循將臨時登入資料與 AWS 資源搭配使用中的指示。
IAM 使用者	使用長期登入資料簽署對 AWS CLI 或 AWS APIs程式設計請求（直接使用或使用 AWS SDKs)。 (不建議使用)	請遵循管理 IAM 使用者的存取金鑰中的指示。
聯合身分主體	使用 AWS STS API 操作建立具有暫時性安全登入資料的新工作階段，其中包含存取金鑰對和工作階段字符。	如需 API 操作的說明，請參閱 the section called “請求臨時安全憑證”

許可和政策如何提供存取管理

AWS Identity and Access Management (IAM) 的存取管理部分可協助您定義委託人實體在帳戶中可執行的操作。主體實體是指使用 IAM 實體 (IAM 使用者或角色) 執行身分驗證的人員或應用程式。存取管理通常稱為授權。您可以透過 AWS 建立政策並將其連接至 IAM 身分 (IAM 使用者、IAM 群組或 IAM 角色） 或 AWS 資源來管理 中的存取。政策是 中的物件， AWS 當與身分或資源相關聯時， 會定義其許可。當委託人使用 IAM 實體 (IAM 使用者或 IAM 角色） 提出請求時， 會 AWS 評估這些政策。政策中的許可決定是否允許或拒絕請求。大多數政策會以 JSON 文件 AWS 的形式存放在 中。如需有關政策類型及其使用的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

政策和帳戶

如果您在 中管理單一帳戶 AWS，則會使用 政策定義該帳戶中的許可。如果您跨多個帳戶管理許可，您的 IAM 使用者會更難管理許可。對於跨帳戶許可，您可以使用 IAM 角色、以資源為基礎的政策或存取控制清單 (ACL)。不過，如果您擁有多個帳戶，我們建議您改用 AWS Organizations 服務來協助您管理這些許可。如需詳細資訊，請參閱什麼是 AWS Organizations？ AWS Organizations 《使用者指南》中的 。

政策與使用者

IAM 使用者是 AWS 帳戶中的身分。當您建立 IAM 使用者，在您提供許可前他們無法存取您帳戶中的任何項目。可透過建立連接到 IAM 使用者或該 IAM 使用者所屬 IAM 群組的身分型政策，為 IAM 使用者提供許可。下列範例顯示 JSON 政策，其允許 IAM 使用者執行 `us-east-2` 區域內 `123456789012` 帳戶之 `Books` 資料表上的所有 Amazon DynamoDB 動作 (`dynamodb:*`)。

JSON

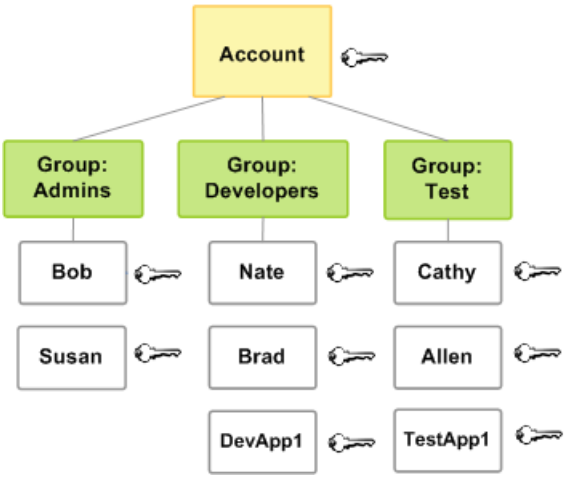
```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

將此政策連接至 IAM 使用者後，使用者有權執行 DynamoDB 執行個體的 `Books` 資料表中的所有動作。大多數 IAM 使用者都有多個政策，這些政策可合併以代表其授予的總許可。

根據預設，政策未明確允許的動作或資源會被拒絕。例如，如果上述政策是連接到使用者的單一政策，則該使用者可以對 Books 資料表執行 DynamoDB 動作，但無法對其他資料表執行動作。同樣地，不允許使用者在 Amazon EC2、Amazon S3 或任何其他 AWS 服務中執行任何動作，因為使用這些服務的許可不包含在政策中。

政策和 IAM 群組

您可以將 IAM 使用者組織到 IAM 群組並將政策連接到 IAM 群組。在這種情況下，各個 IAM 使用者仍有自己的憑證，但是 IAM 群組中的所有 IAM 使用者都擁有連接到 IAM 群組的許可。使用 IAM 群組可更輕鬆地進行許可管理。



IAM 使用者或 IAM 群組可以有多個政策與之連接，分別授予不同的許可。在這種情況下，政策組合會決定主體的有效許可。如果委託人沒有動作和資源的明確Allow許可，則委託人沒有這些許可。

聯合身分使用者工作階段和角色

聯合主體不會像 IAM 使用者 AWS 帳戶 那樣擁有永久身分。若要向聯合身分主體指派許可，可以建立稱為角色的實體，並為角色定義許可。當 SAML 或 OIDC 聯合身分主體登入 時 AWS，使用者會與角色相關聯，並獲授予角色中定義的許可。如需詳細資訊，請參閱為第三方身分提供者建立角色。

以身分為基礎和以資源為基礎的政策

以身分為基礎的政策是指您連接到 IAM 身分的許可政策，這些身分包括像是 IAM 使用者、群組或角色。以資源為基礎的政策是您連接到資源的許可政策，這些資源包括像是 Amazon S3 儲存貯體或 IAM 角色信任政策。

以身分為基礎政策可控制身分在何種條件下對哪些資源執行哪些動作。身分型政策可進一步分類：

- 受管政策：您可以連接到 中多個使用者、群組和角色的獨立身分型政策 AWS 帳戶。您可以使用兩種類型受管政策：
  - AWS 受管政策 – 由 建立和管理的受管政策 AWS。如果您是初次使用 政策，建議您先使用 AWS 受管政策。
  - 客戶管理政策：您在 AWS 帳戶中建立和管理的受管政策。客戶受管政策提供比 AWS 受管政策更精確的政策控制。您可以在視覺化編輯器建立和編輯和驗證 IAM 政策，或直接建立 JSON 政策文件。如需更多詳細資訊，請參閱 使用客戶管理政策定義自訂 IAM 許可 及 編輯 IAM 政策。
- 內嵌政策：由您建立和管理並直接嵌入到單一使用者、群組或角色的政策。在大多數情況下，我們建議您不要使用內嵌政策。

以資源為基礎的政策可控制指定的主體在何種條件下對該資源執行哪些動作。資源為基礎的政策是內嵌政策，而且沒有受管的以資源為基礎的政策。如需啟用跨帳戶存取權，您可以在其他帳戶內指定所有帳戶或 IAM 實體作為資源型政策的主體。

IAM 服務支援一種稱為角色信任政策的資源型政策，它可連接到 IAM 角色。由於 IAM 角色是支援資源型政策的身分也是資源，所以您必須將信任政策和身分型政策同時連接到 IAM 角色。信任政策會定義哪些委託人實體 （帳戶、使用者、角色和聯合身分使用者委託人） AWS STS 可以擔任該角色。若要了解 IAM 角色與其他以資源為基礎之政策之間的差別，請參閱 IAM 中的跨帳戶資源存取。

若要查看哪些服務支援以資源為基礎的政策，請參閱 AWS 使用 IAM 的 服務。若要進一步了解以資源為基礎的政策的詳細資訊，請參閱 以身分為基礎和以資源為基礎的政策。

使用 ABAC 授權根據屬性定義許可

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性定義許可。 會 AWS 呼叫這些屬性標籤。您可以將標籤連接至 IAM 資源，包括 IAM 實體 (IAM 使用者或 IAM 角色） 和 AWS 資源。您可以為您的 IAM 主體建立單一 ABAC 政策或是一組政策。您可以設計 ABAC 政策，當主體的標籤與資源標籤相符時允許操作。ABAC 的屬性系統可提供較高的使用者內容和精細的存取控制。由於 ABAC 是以屬性為基礎，因此可以對實時授予或撤銷存取權的資料或應用程式執行動態授權。ABAC 在擴展環境中以及身分或資源政策管理變得複雜的情況下很有用。

例如，您可以使用 `access-project` 標籤鍵建立三個 IAM 角色。將第一個 IAM 角色的標籤值設為 `Heart`，第二個設為 `Star`，並將第三個設為 `Lightning`。然後，您可以使用單一政策，在 IAM 角色和資源 AWS 具有標籤值 時允許存取`access-project`。如需示範如何在 AWS中使用 ABAC 的詳細教學，請參閱 IAM 教學課程：定義根據標籤存取 AWS 資源的許可。若要瞭解支援 ABAC 的服務，請參閱 AWS 使用 IAM 的 服務。

此圖表說明套用至主體的標籤必須符合套用至資源的標籤，才能為使用者授予資源許可。標籤可套用至 IAM 群組、資源群組、個別使用者和個別資源。

ABAC 與傳統 RBAC 模型的比較

IAM 中使用的傳統授權模型為角色型存取控制 (RBAC)。RBAC 會根據人員的任務函數或角色來定義許可，這與 IAM 角色不同。IAM 確實包括在 RBAC 模型中將許可對齊到任務功能的用於任務角色的受管政策。

在 IAM 中，您可以為不同的任務角色建立不同的政策，來實作 RBAC。然後，您可以將政策連接到身分 (IAM 使用者、IAM 群組或 IAM 角色)。根據最佳實務，您應授與任務角色所需要的最低許可。這會導致最低權限存取。每個工作職能政策都會列出該身分指派該政策可存取的特定資源。使用傳統 RBAC 模型的缺點是當您或您的使用者新增新的資源至環境時，您必須更新政策以允許存取這些新資源。

例如，假設您有三個員工正在進行的專案，分別名為 Heart、Star 和 Lightning。您為每個專案建立 IAM 角色。接著將政策連接到各 IAM 角色，定義允許擔任 IAM 角色的人員所能存取的資源。若一名員工在您的公司內變更了職位，您便會將他們指派至不同的 IAM 角色。可將人員或程式指派給多個 IAM 角色。但是，Star 專案可能需要其他資源，例如，新的 Amazon EC2 容器。在這種情況下，您必須更新連接到 Star IAM 角色的政策，指定新的容器資源。否則，Star 專案的成員便無法存取新的容器。
此圖表說明角色型存取控制要求為每個身分指派一個以特定工作職能為基礎的政策，以存取不同的資源。

與傳統 RBAC 模型相較，ABAC 提供了下列優點：

- ABAC 許可隨創新擴展。管理員不再需要更新現有政策來允許存取新的資源。例如，假設您使用 access-project 標籤設計您的 ABAC 策略。開發人員可將 IAM 角色與 access-project = Heart 標籤搭配使用。當 Heart 專案的人員需要額外 Amazon EC2 資源時，開發人員便可以使用 access-project = Heart 標籤建立新的 Amazon EC2 執行個體。接著，Heart 專案的任何人員便可以開始和停止這些執行個體，因為他們的標籤值相符。
- ABAC 需要的政策較少。由於您不需要為不同的任務功能建立不同政策，您建立的政策也會較少。這些政策較易於管理。
- 使用 ABAC 時，團隊可以動態回應變化和增長。由於新資源的許可會根據屬性自動授予，因此不需要手動將政策指派給身分。例如，若您的公司已使用 ABAC 支援 Heart 和 Star，新增新的 Lightning 專案也相當容易。IAM 管理員可以使用 access-project = Lightning 標籤建立新的 IAM 角色。不需要變更政策來支援新的專案。有權承擔 IAM 角色的任何人員都可建立和檢視標記有 access-project = Lightning 的執行個體。另一個情況是團隊成員從 Heart 專案移至

Lightning 專案時。若要提供團隊成員對 Lightning 專案的存取權，IAM 管理員可將他們指派給不同的 IAM 角色。而不需要變更許可政策。

- 使用 ABAC 可實現精密許可。在您建立政策時，最佳實務是授與最低權限。使用傳統 RBAC 時，需要撰寫允許存取特定資源的政策。但是，當您使用 ABAC 時，如果資源標籤與主體標籤相符，可以允許對所有資源執行動作。
- 搭配 ABAC 使用您企業目錄的員工屬性。您可以設定 SAML 或 OIDC 提供者，將工作階段標籤傳遞至 IAM。當您的員工聯合到 時 AWS，IAM 會將其屬性套用至其產生的委託人。您接著可以使用 ABAC 來根據這些屬性允許或拒絕許可。

如需示範如何在 中使用 ABAC 的詳細教學課程 AWS，請參閱 IAM 教學課程：定義根據標籤存取 AWS 資源的許可。

IAM 入門

AWS Identity and Access Management (IAM) 可協助您安全地控制對 Amazon Web Services (AWS) 和您帳戶資源的存取。IAM 還可以讓您的登入認證保持私有。您不需專門註冊即可使用 IAM。使用 IAM 免費。

IAM 的使用方式取決於您的任務函數。如需角色特定的指引IAM 的使用時機為何？，請參閱 。

使用 IAM 為使用者和角色等身分提供您的帳戶中資源的存取權。例如，您可以將 IAM 與您在公司目錄中管理外部的現有使用者搭配使用， AWS 也可以 AWS 使用 建立使用者 AWS IAM Identity Center。聯合身分會擔任定義的 IAM 角色，以存取他們所需的資源。如需有關 IAM Identity Center 的詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

> **Note**
>
> IAM 已與數個 AWS 產品整合。如需支援 IAM 的服務清單，請參閱 AWS 使用 IAM 的 服務。

若要了解如何開始使用 AWS、建立管理使用者、 AWS Organizations和使用多個 服務來解決建置和啟動第一個專案等問題，請參閱 入門資源中心。

如需常見 IAM 任務的實作實務，請參閱 IAM 教學課程。任務包括建立跨帳戶存取的角色，以及使用屬性型存取控制的標籤。

設定帳戶後，請規劃人員和工作負載存取帳戶的方式。如需詳細資訊，請參閱計劃存取您的帳戶 AWS。

然後使用 為最低權限許可做好準備遵循最低權限原則。

設定您的 AWS 帳戶

開始使用 IAM 之前，請確定您已完成 AWS 環境的初始設定。

AWS 會在註冊程序完成後傳送確認電子郵件給您。您可以隨時登錄 https://aws.amazon.com/ 並選擇我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

當您註冊服務時，您使用 AWS 帳戶 電子郵件地址和密碼建立 。這些是您的 AWS 根使用者登入資料。最佳實務是，您不使用您的根使用者登入資料來存取 AWS 每日任務。僅將根使用者憑證用於執

行需要根使用者憑證的任務。另外，不要與任何人共用您的憑證。反之，將人員新增至您的目錄，並授予他們存取您的 AWS 帳戶的權限。

保護您的 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者AWS 管理主控台身分登入 。在下一頁中，輸入您的密碼。

   如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的以根使用者身分登入。

2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

   如需說明，請參閱《IAM 使用者指南》中的為您的 AWS 帳戶 根使用者 （主控台） 啟用虛擬 MFA 裝置。

將存取權授予帳單主控台

AWS 帳戶 內的 IAM 使用者和角色預設無法存取帳單和成本管理主控台。即使他們具有授予特定帳單功能之存取權的 IAM 政策，也是如此。若要授予存取權， AWS 帳戶 根使用者必須先啟用 IAM 存取權。

> **Note**
>
> 我們的安全最佳實務是建議您透過聯合身分與 AWS IAM Identity Center來提供資源存取權限。當您搭配 啟用 IAM Identity Center 時 AWS Organizations，預設會啟用 Billing and Cost Management 主控台，其中包含 AWS 帳戶 組織中所有 的合併帳單。如需詳細資訊，請參閱《帳單與成本管理使用者指南》中的合併 AWS Organizations的帳單。

1. AWS 管理主控台 使用您的根使用者登入資料登入 （特別是您用來建立 AWS 帳戶的電子郵件地址和密碼）。
2. 在導覽列上選取您的帳戶名稱，然後選取帳戶。
3. 向下捲動頁面，直至找到帳單資訊的 IAM 使用者和角色存取權部分，然後選取編輯。
4. 選取啟用 IAM 存取核取方塊，以啟用對帳單和成本管理主控台頁面的存取。
5. 選擇更新。

   頁面會顯示已啟用 IAM 使用者/角色對帳單資訊的存取權訊息。

> **Important**
>
> 單獨啟用 IAM 存取並不會授予使用者或角色任何檢視帳單和成本管理主控台頁面的權限。還必須將必要的身分型政策連接至 IAM 角色，以授予對帳單主控台的存取權。角色會提供使用者在需要時可以使用的臨時憑證。

6. 使用 AWS 管理主控台 建立角色，使用者可以擔任該角色來存取帳單主控台。
7. 在角色的新增許可頁面上，新增許可以列出並檢視 AWS 帳戶中有關帳單資源的詳細資訊。

   AWS 受管政策 Billing 授予使用者檢視和編輯 Billing and Cost Management 主控台的許可。這包括檢視帳戶使用情況、修改預算和付款方式。如需可連接至 IAM 角色以控制對帳戶帳單資訊的存取的更多政策範例，請參閱 Billing and Cost Management User Guide 中的 AWS Billing policy examples。

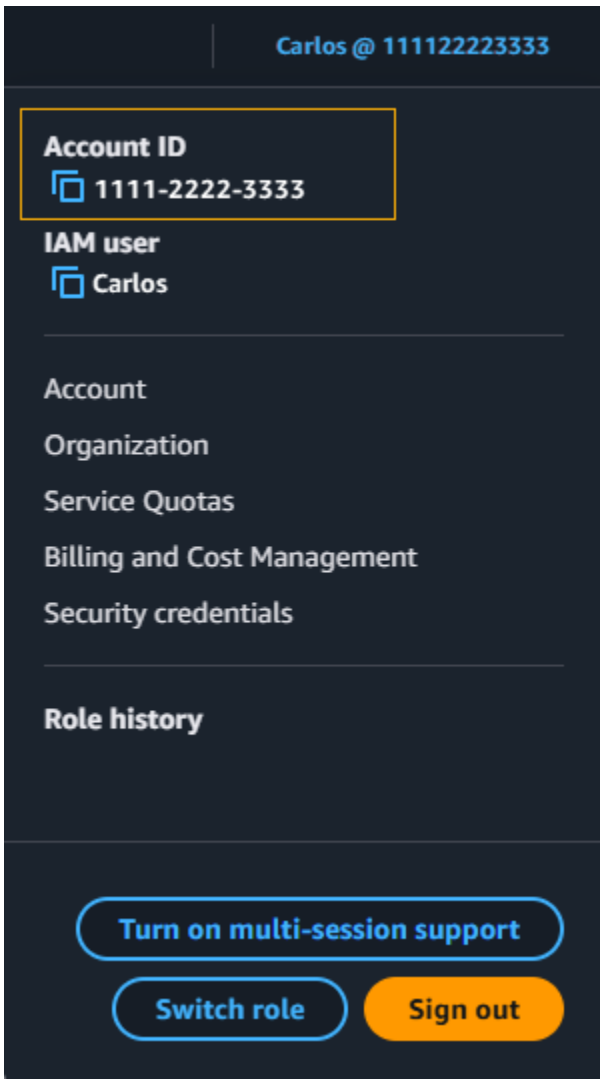
檢視您的 AWS 帳戶 ID

如果已登入主控台，您可以使用下列方法檢視 AWS 帳戶 的帳戶 ID。

檢視您的 AWS 帳戶 ID

Console

當您前往 AWS 帳戶 區段中的 IAM 儀表板時，會顯示 AWS 帳戶 ID。您也可以在右上角的導覽列中檢視帳戶 ID。選擇您的使用者名稱，帳戶 ID 即會顯示在使用者名稱上方。



AWS CLI

使用以下命令來檢視您的使用者 ID、帳戶 ID 和您的使用者 ARN：

- aws sts get-caller-identity

API

使用以下 API 來檢視您的使用者 ID、帳戶 ID 及您的使用者 ARN：

- GetCallerIdentity

為您的 AWS 帳戶 ID 使用別名

帳戶 ID 是一個 12 位數的號碼，可唯一識別您的帳戶。預設情況下，帳戶中的 IAM 使用者使用包含帳戶 ID 的 Web URL 登入。如果他們沒有 URL，則可以在登入時在 AWS 登入頁面上提供帳戶 ID。

預設情況下，您的登入頁面 URL 有以下格式。

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

許多人發現單字比數字更容易記住，因此為帳戶 ID 建立別名可以協助您的 IAM 使用者更輕鬆地登入。

如果您為 AWS 帳戶 ID AWS 帳戶 建立別名，您的登入頁面 URL 如下所示。

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

建立帳戶別名之前的考量

- 您的 只能 AWS 帳戶 有一個別名。如果您為 AWS 帳戶建立新的別名，新的別名會覆寫先前的別名，而包含先前別名的 URL 會停止運作。
- 帳戶別名必須僅包含數字、小寫字母和連字號。如需 AWS 帳戶實體限制的詳細資訊，請參閱 IAM AWS STS 和配額。
- 帳戶別名在指定網路分割區內所有 Amazon Web Services 產品中必須是唯一的。

  分割區是一組 AWS 區域。每個 AWS 帳戶的範圍都是一個分割區。

  以下是支援的分割區：

  - `aws` - AWS 區域
  - `aws-cn` - 中國區域
  - `aws-us-gov` - AWS GovCloud (US) 區域

> **Note**
>
> 帳戶別名並非秘密，它們會顯示在您的面向公眾的登入頁面 URL 中。請勿在您的帳戶別名中包含任何敏感資訊。
> 包含您 AWS 帳戶 ID 的原始 URL AWS 帳戶 會保持作用中狀態，並可在您建立別名後使用。

如需設定或移除別名的指示，請參閱 建立帳戶別名和 刪除帳戶別名。

建立帳戶別名

若要執行下列步驟，您至少必須擁有下列 IAM 許可：

- `iam:ListAccountAliases`
- `iam:CreateAccountAlias`

建立 AWS 帳戶 別名

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Dashboard (儀表板)。
3. 在 AWS 帳戶區段中，選擇帳戶別名旁邊的建立。如果別名已存在，請選擇 Edit (編輯)。
4. 在對話方塊中，輸入要用作別名的名稱，然後選擇儲存變更。

AWS CLI

執行以下命令：

- `aws iam create-account-alias`

API

若要為 AWS 管理主控台 登入頁面 URL 建立別名，請呼叫下列操作：

- `CreateAccountAlias`

刪除帳戶別名

若要執行下列步驟，您至少必須擁有下列 IAM 許可：

- `iam:ListAccountAliases`
- `iam:DeleteAccountAlias`

刪除帳戶別名

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Dashboard (儀表板)。
3. 在 AWS 帳戶區段中，選擇帳戶別名旁邊的刪除。

AWS CLI

若要刪除 AWS 帳戶 ID 別名，請執行下列命令：

- `aws iam delete-account-alias`

若要確認已刪除帳戶別名，請執行下列命令，嘗試顯示您的 AWS 帳戶 ID 別名：

- `aws iam list-account-aliases`

API

若要刪除 AWS 帳戶 ID 別名，請呼叫下列操作：

- `DeleteAccountAlias`

若要確認已刪除帳戶別名，請呼叫下列操作來嘗試顯示您的 AWS 帳戶 ID 別名：

- `ListAccountAliases`

> **Note**
>
> 刪除帳戶別名後，帳戶的唯一登入 URL 取決於帳戶 ID。連線至別名 URL 的任何嘗試都會失敗並且不會重新導向。

計劃存取您的帳戶 AWS

設定 時 AWS，規劃您希望人員如何存取 AWS 您的帳戶和資源。精心設計的身分管理解決方案可確保您的帳戶安全。

身分來源

根據 IAM 最佳實務，我們建議人類使用者和工作負載使用臨時登入資料來存取您的 AWS 資源。您可以透過 IAM 角色授予臨時登入資料。聯合到 IAM 的使用者和 IAM Identity Center 中的使用者 (聯合的或者在 IAM Identity Center 目錄中建立的) 都使用 IAM 角色來存取資源。

開始使用 之前 AWS，請規劃如何使用下列其中一種方法來設定您的身分：

- 使用 啟用 IAM Identity Center， AWS Organizations 並將 IAM Identity Center 中的使用者直接新增至組織目錄。

  如需將使用者直接新增至 IAM Identity Center 組織目錄的說明，請參閱新增使用者
- 將您的現有外部身分提供者與 IAM Identity Center 或 IAM 聯合。

  如需將外部身分提供者聯合至 IAM Identity Center 組織目錄的指示，請參閱適當的入門教學課程。

存取管理

識別您的使用者所需的 AWS 資源和服務。然後定義每個使用者、群組或角色的存取許可和政策。

- 如果您使用 IAM Identity Center，則會在組織中的每個 AWS 帳戶中建立 IAM 身分提供者、IAM 角色和許可政策。這些角色和許可與您將人員或群組指派給應用程式或 AWS 帳戶時指定的相符。

  如需詳細資訊，請參閱 Assign user access 和 Set up single sign-on access to your applications。
- 您可以使用帳戶存取管理員 — IAM 功能，可讓您將建立和管理的現有 IAM 角色指派給 IAM Identity Center 使用者和群組。帳戶存取管理員可讓您存取完整的 IAM 角色功能集。您可以搭配許可集或單獨使用。
- 如果您直接將身分提供者與 中的 IAM 聯合 AWS 帳戶，則必須為使用者建立角色。此角色需要信任政策和許可政策。信任政策指出誰可以擔任該角色。許可政策說明允許或拒絕存取角色 AWS 的動作和資源。

  如需詳細資訊，請參閱身分提供者和 的聯合身分 AWS

如需為登入新增額外安全層的說明，請參閱 對身分使用多重要素驗證。

IAM 使用者的使用案例

您在 中建立的 IAM 使用者 AWS 帳戶 具有您直接管理的長期登入資料。

在 中管理存取權時 AWS，IAM 使用者通常不是最佳選擇。應該避免在大多數使用案例中依賴 IAM 使用者，主要原因有如下幾個。

首先，IAM 使用者是為個別帳戶而設計，因此不會隨著組織的成長而擴展。管理大量 IAM 使用者的許可和安全性可能很快就成為挑戰。

IAM 使用者也缺乏您透過其他 AWS 身分管理解決方案取得的集中式可見性和稽核功能。這可能加大維護安全性與合規性的難度。

最後，透過可擴展性更好的身分管理方法，實作安全最佳實務 (例如多重要素驗證、密碼政策和角色分離) 將更容易。

我們建議您使用更強大的解決方案，例如 IAM Identity Center 搭配 或來自外部供應商的聯合身分 AWS Organizations，而不是依賴 IAM 使用者。隨著 AWS 環境的增長，這些選項將為您提供更好的控制、安全和營運效率。

因此，建議您僅將 IAM 使用者用於聯合身分使用者不支援的使用案例。

下列清單識別需要 IAM 使用者長期登入資料的特定使用案例 AWS。您可以使用 IAM 在 AWS 帳戶的保護下建立這些 IAM 使用者，並使用 IAM 管理其許可。

- 緊急存取您的帳戶 AWS
- 無法使用 IAM 角色的工作負載
  - AWS CodeCommit 存取
  - Amazon Keyspaces (適用於 Apache Cassandra)
- 第三方 AWS 用戶端
- AWS IAM Identity Center 不適用於您的帳戶，而且您沒有其他身分提供者

建立 IAM 使用者以進行緊急存取

IAM 使用者是 中的身分 AWS 帳戶 ，具有單一人員或應用程式的特定許可。

讓 IAM 使用者進行緊急存取是建立 IAM 使用者的建議原因之一，以便在身分提供者無法存取 AWS 帳戶 時存取您的 。

> **Note**
>
> 我們的安全最佳實務是建議您透過聯合身分來提供資源存取權限，而不是建立 IAM 使用者。若要了解需要 IAM 使用者的特定情形，請參閱建立 IAM 使用者 (而非角色) 的時機。

若要建立 IAM 使用者以進行緊急存取

> **最低許可**
>
> 若要執行下列步驟，您至少必須擁有下列 IAM 許可：
>
> - `access-analyzer:ValidatePolicy`
> - `iam:AddUserToGroup`
> - `iam:AttachGroupPolicy`
> - `iam:CreateGroup`
> - `iam:CreateLoginProfile`
> - `iam:CreateUser`
> - `iam:GetAccountPasswordPolicy`
> - `iam:GetLoginProfile`
> - `iam:GetUser`
> - `iam:ListAttachedGroupPolicies`
> - `iam:ListAttachedUserPolicies`
> - `iam:ListGroupPolicies`
> - `iam:ListGroups`
> - `iam:ListGroupsForUser`
> - `iam:ListPolicies`
> - `iam:ListUserPolicies`
> - `iam:ListUsers`

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。

2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中選取使用者，然後選取建立使用者。

> **Note**
>
> 如果您已啟用 IAM Identity Center，則 AWS 管理主控台 會顯示提醒，指出最好在 IAM Identity Center 中管理使用者的存取權。在此程序中，您建立的 IAM 使用者專門用於身分提供者無法使用的時機。

4. 在指定使用者詳細資訊頁面使用者詳細資訊下方的使用者名稱中，輸入新使用者的名稱。這是新使用者的 AWS登入名稱。在此範例中，請輸入 **`EmergencyAccess`**。

> **Note**
>
> 使用者名稱可以是長達 64 個字母、數字以及這些字元的組合：加號 (+)、等號 (=)、逗號 (,)、句號 (.)、@ 符號、底線 (_) 以及連字號 (-)。名稱在帳戶中必須是唯一的。它們無法透過大小寫進行區分。例如，您不可以建立兩個名為 TESTUSER 和 testuser 的使用者。使用者名稱用在政策中或作為 ARN 的一部分時，名稱區分大小寫。當主控台中的客戶顯示使用者名稱時 (例如在登入程序期間)，使用者名稱不區分大小寫。

5. 選取為使用者提供 AWS 管理主控台存取權 (選用) 旁的核取方塊，然後選擇我想要建立 IAM 使用者。
6. 在主控台密碼下選取自動產生的密碼。
7. 清除使用者必須在下次登入時建立新密碼 (建議) 旁的核取方塊。由於此 IAM 使用者是用於緊急存取，因此受信任的管理員會保留密碼，並僅在必要時提供密碼。
8. 在 設定許可 頁面的 許可選項 下方選取 新增使用者至群組。然後，在 使用者群組 下方選取 建立群組。
9. 在 建立使用者群組 頁面的 使用者群組名稱 中輸入**`EmergencyAccessGroup`**。然後，在許可政策下方選取 AdministratorAccess。
10. 選取建立使用者群組以返回設定許可頁面。
11. 在使用者群組下方選取先前建立的 **`EmergencyAccessGroup`** 的名稱。
12. 選取下一步以繼續前往檢閱和建立頁面。
13. 在檢閱和建立頁面上檢閱要新增至新使用者的使用者群組成員資格清單。準備好繼續時，請選取建立使用者。

14. 在擷取密碼頁面上選取下載 .csv 檔案，以儲存含有使用者憑證資訊 (連線 URL、使用者名稱和密碼) 的 .csv 檔案。
15. 如果您需要登入 IAM 且無法存取身分提供者，請儲存此檔案以供使用。

新的 IAM 使用者會顯示在使用者清單中。選取使用者名稱連結以檢視使用者詳細資訊。

AWS CLI

1. 建立名為 **EmergencyAccess** 的使用者。

   - aws iam create-user

   ```
   aws iam create-user \
      --user-name EmergencyAccess
   ```

2. (選用) 提供使用者 AWS 管理主控台存取權。這需要密碼。若要為 IAM 使用者建立密碼，可以使用 `--cli-input-json` 參數來傳遞包含密碼的 JSON 檔案。還必須為使用者提供您的帳戶登入頁面的 URL。

   - aws iam create-login-profile

   ```
   aws iam create-login-profile \
      --generate-cli-skeleton > create-login-profile.json
   ```

   - 在文字編輯器中開啟 `create-login-profile.json` 檔案並輸入符合密碼政策的密碼，然後儲存檔案。例如：

   ```
   {
    "UserName": "EmergencyAccess",
    "Password": "Ex@3dRA0djs",
    "PasswordResetRequired": false
   }
   ```

   - 再次使用 `aws iam create-login-profile` 命令，傳遞 `--cli-input-json` 參數以指定您的 JSON 檔案。

```
aws iam create-login-profile \
   --cli-input-json file://create-login-profile.json
```

> **Note**
>
> 如果在 JSON 檔案中提供的密碼違反帳戶的密碼政策，您會收到 `PasswordPolicyViolation` 錯誤。如果發生這種情況，請檢閱您帳戶的密碼政策，並依循要求更新 JSON 檔案中的密碼。

3. 建立 **`EmergencyAccessGroup`**、將 AWS 受管政策連接至 `AdministratorAccess` 群組，並將**`EmergencyAccess`**使用者新增至 群組。

> **Note**
>
> 「AWS 受管政策」為獨立的政策，由 AWS建立並管理。每個政策都有自己的 Amazon Resource Name (ARN)，其中包含政策名稱。例如，`arn:aws:iam::aws:policy/IAMReadOnlyAccess` 是 AWS 受管政策。如需有關 ARN 的詳細資訊，請參閱 IAM ARN。如需 受 AWS 管政策的清單 AWS 服務，請參閱 AWS 受管政策。

- aws iam create-group

```
aws iam create-group \
   --group-name EmergencyAccessGroup
```

- aws iam attach-group-policy

```
aws iam attach-group-policy \
   --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
   --group-name >EmergencyAccessGroup
```

- aws iam add-user-to-group

```
aws iam add-user-to-group \
   --user-name EmergencyAccess \
   --group-name EmergencyAccessGroup
```

- 執行 aws iam get-group 命令會列出 **EmergencyAccessGroup** 及其成員。

```
aws iam get-group \
   --group-name EmergencyAccessGroup
```

為無法使用 IAM 角色的工作負載建立 IAM 使用者

Important

最佳實務是，我們建議您要求人類使用者在存取時使用臨時登入資料 AWS。或者，您可以使用 AWS IAM Identity Center 來管理使用者身分，包括管理使用者。建議您使用 IAM Identity Center 來管理帳戶的存取權和這些帳戶內的許可。如果使用的是外部身分提供者，也可以在 IAM Identity Center 中設定使用者身分的存取許可。

如果您的使用案例需要具有程式化存取權和長期憑證的 IAM 使用者，則建議您建立程序，以在需要時更新存取金鑰。如需詳細資訊，請參閱更新存取金鑰。

若要執行一些帳戶和服務管理任務，必須使用根使用者憑證來登入。若要檢視您必須以根使用者身分登入的任務，請參閱需要根使用者憑證的任務。

若要為無法使用 IAM 角色的工作負載建立 IAM 使用者

最低許可

若要執行下列步驟，您至少必須擁有下列 IAM 許可：

- `iam:AddUserToGroup`
- `iam:AttachGroupPolicy`
- `iam:CreateAccessKey`
- `iam:CreateGroup`
- `iam:CreateServiceSpecificCredential`

- iam:CreateUser
- iam:GetAccessKeyLastUsed
- iam:GetAccountPasswordPolicy
- iam:GetAccountSummary
- iam:GetGroup
- iam:GetLoginProfile
- iam:GetPolicy
- iam:GetRole
- iam:GetUser
- iam:ListAccessKeys
- iam:ListAttachedGroupPolicies
- iam:ListAttachedUserPolicies
- iam:ListGroupPolicies
- iam:ListGroups
- iam:ListGroupsForUser
- iam:ListInstanceProfilesForRole
- iam:ListMFADevices
- iam:ListPolicies
- iam:ListRoles
- iam:ListRoleTags
- iam:ListSSHPublicKeys
- iam:ListServiceSpecificCredentials
- iam:ListSigningCertificates
- iam:ListUserPolicies
- iam:ListUserTags
- iam:ListUsers
- iam:UploadSSHPublicKey
- iam:UploadSigningCertificate

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者，然後選擇建立使用者。
4. 在指定詳細資訊頁面上，執行下列動作：
   a. 對於使用者名稱，輸入 ***WorkloadName***。將 ***WorkloadName*** 替換為將使用帳戶的工作負載的名稱。
   b. 選擇下一步。
5. 在設定許可頁面上，執行下列動作：
   a. 選擇將使用者新增至群組。
   b. 選擇建立群組。
   c. 在建立使用者群組對話方塊的使用者群組名稱中，輸入代表工作負載在群組中的用途的名稱。在此範例中，使用名稱 **Automation**。
   d. 在許可政策下，選取 PowerUserAccess 受管政策的核取方塊。

      > **Tip**
      > 在許可政策搜尋方塊中輸入 Power，以快速尋找受管政策。

   e. 選擇建立使用者群組。
   f. 回到包含 IAM 群組清單的頁面，選取新使用者群組的核取方塊。如果沒有在清單中看到新使用者群組，選擇重新整理。
   g. 選擇下一步。
6. (選用) 在標籤區段，透過將標籤做為鍵值對連接，將中繼資料新增到使用者。如需詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
7. 驗證新使用者的使用者群組成員資格。準備好繼續時，選擇建立使用者。
8. 此時將顯示一則狀態通知，告知您使用者已成功建立。選取檢視使用者以前往使用者詳細資訊頁面
9. 選取安全憑證索引標籤。然後建立工作負載所需的憑證。
   - 存取金鑰：選取建立存取金鑰可產生並為使用者下載存取金鑰。

> **Important**
>
> 這是您檢視或下載私密存取金鑰的唯一機會，您必須先將此資訊提供給使用者，然後他們才能使用 AWS API。請將使用者的新存取金鑰 ID 和私密存取金鑰存放在安全處。在此步驟之後，您將無法再存取該私密金鑰。

- 的 SSH 公有金鑰 AWS CodeCommit – 選取上傳 SSH 公有金鑰以上傳 SSH 公有金鑰，讓使用者可以透過 SSH 與 CodeCommit 儲存庫通訊。
- 的 HTTPS Git 登入資料 AWS CodeCommit – 選取產生登入資料以產生一組唯一的使用者登入資料，以便與 Git 儲存庫搭配使用。選取下載憑證可將使用者名稱和密碼儲存至 .csv 檔案。這是唯一可取得資訊的時機。如果忘記或遺失密碼，則需要重設。
- Amazon Keyspaces (適用於 Apache Cassandra) 的憑證：選取產生憑證產生服務特定的使用者憑證，以搭配 Amazon Keyspaces 使用。選取下載憑證可將使用者名稱和密碼儲存至 .csv 檔案。這是唯一可取得資訊的時機。如果忘記或遺失密碼，則需要重設。

> **Important**
>
> 服務特定憑證是與特定 IAM 使用者關聯的長期憑證，只能用於為其建立的服務。若要授予 IAM 角色或聯合身分使用暫時登入資料存取所有 AWS 資源的許可，請使用身分 AWS 驗證搭配 Amazon Keyspaces 的 SigV4 身分驗證外掛程式。如需詳細資訊，請參閱《Amazon Keyspaces (適用於 Apache Cassandra) 開發人員指南》中的透過 IAM 角色和 SigV4 外掛程式使用臨時憑證連線到 Amazon Keyspaces (適用於 Apache Cassandra)。

- X.509 簽署憑證 – 如果您需要提出安全的 SOAP 通訊協定請求，且位於 不支援的區域，請選取建立 X.509 憑證 AWS Certificate Manager。ACM 為佈建、管理和部署伺服器憑證的首選工具。如需有關使用 ACM 的詳細資訊，請參閱《AWS Certificate Manager 使用者指南》。

您已建立具有程式化存取權的使用者，並使用 PowerUserAccess 任務函數對其進行設定。此使用者的許可政策授予每個服務的完整存取權，IAM 和 除外 AWS Organizations。

如果工作負載無法擔任 IAM 角色，您可以使用這個相同的程序，讓其他工作負載以程式設計方式存取您的 AWS 帳戶 資源。此程序使用 PowerUserAccess 受管政策指派許可。若要遵循最低權限的最佳實務，請考慮使用更嚴格的政策或建立自訂政策，限制僅存取程式所需的資源。若要了解如何使用將使用者許可限制為特定 AWS 資源的政策，請參閱 AWS 資源的存取管理和 以身分為基礎的

IAM 政策範例。若要在建立使用者群組之後向其中新增其他使用者，請參閱編輯 IAM 群組中的使用者。

AWS CLI

1. 建立名為 **Automation** 的使用者。

   - aws iam create-user

```
aws iam create-user \
    --user-name Automation
```

2. 建立名為 的 IAM 使用者群組**AutomationGroup**，將 AWS 受管政策連接至PowerUserAccess群組，然後將**Automation**使用者新增至群組。

> **Note**
>
> 「AWS 受管政策」為獨立的政策，由 AWS建立並管理。每個政策都有自己的 Amazon Resource Name (ARN)，其中包含政策名稱。例如，arn:aws:iam::aws:policy/IAMReadOnlyAccess 是 AWS 受管政策。如需有關 ARN 的詳細資訊，請參閱 IAM ARN。如需 受 AWS 管政策的清單 AWS 服務，請參閱 AWS 受管政策。

   - aws iam create-group

```
aws iam create-group \
    --group-name AutomationGroup
```

   - aws iam attach-group-policy

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/PowerUserAccess \
    --group-name AutomationGroup
```

   - aws iam add-user-to-group

```
aws iam add-user-to-group \
    --user-name Automation \
    --group-name AutomationGroup
```

- 執行 aws iam get-group 命令會列出 **AutomationGroup** 及其成員。

```
aws iam get-group \
    --group-name AutomationGroup
```

3. 建立工作負載所需的安全憑證。

- 建立用於測試的存取金鑰 – aws iam create-access-key

```
aws iam create-access-key \
    --user-name Automation
```

此命令的輸出中包含私密存取金鑰和存取金鑰 ID。記錄此資訊並存放在安全的位置。這些憑證一旦遺失，就無法復原，您必須建立新的存取金鑰。

> **Important**
>
> 這些 IAM 使用者存取金鑰是長期憑證，會對您的帳戶造成安全風險。完成測試後，建議刪除這些存取金鑰。如果您遇到正在考慮使用存取金鑰的情況，請調查是否可以為工作負載 IAM 使用者啟用 MFA，並使用 aws sts get-session-token 取得工作階段的臨時憑證，而不要使用 IAM 存取金鑰。

- 上傳 SSH 公有金鑰給 AWS CodeCommit –aws iam upload-ssh-public-key

下列範例假設您將 SSH 公有金鑰存放在 `sshkey.pub` 檔案中。

```
aws upload-ssh-public-key \
    --user-name Automation \
    --ssh-public-key-body file://sshkey.pub
```

- 上傳 X.509 簽署憑證 – aws iam upload-signing-certificate

  如果您需要提出安全的 SOAP 通訊協定請求，且位於 不支援的區域，請上傳 X.509 憑證 AWS Certificate Manager。ACM 為佈建、管理和部署伺服器憑證的首選工具。如需有關使用 ACM 的詳細資訊，請參閱《AWS Certificate Manager 使用者指南》。

  下列範例假設您將 X.509 簽署憑證存放在 `certificate.pem` 檔案中。

  ```
  aws iam upload-signing-certificate \
  --user-name Automation \
  --certificate-body file://certificate.pem
  ```

如果工作負載無法擔任 IAM 角色，您可以使用這個相同的程序，讓其他工作負載以程式設計方式存取您的 AWS 帳戶 資源。此程序使用 PowerUserAccess 受管政策指派許可。若要遵循最低權限的最佳實務，請考慮使用更嚴格的政策或建立自訂政策，限制僅存取程式所需的資源。若要了解如何使用將使用者許可限制為特定 AWS 資源的政策，請參閱 AWS 資源的存取管理和 以身分為基礎的 IAM 政策範例。若要在建立使用者群組之後向其中新增其他使用者，請參閱編輯 IAM 群組中的使用者。

對身分使用多重要素驗證

對身分使用多重要素驗證 (MFA) 是另一個 IAM 最佳實務。MFA 是額外的安全層，要求使用者在提供使用者名稱和密碼後，提供額外的驗證因素以驗證其身分。即使使用者的密碼外洩，它也能讓攻擊者很難取得未經授權的存取權，因此大幅提升安全性。MFA 被廣泛採用為保護存取線上帳戶、雲端服務和其他敏感資源的最佳實務。 AWS 支援根使用者、IAM 使用者、IAM Identity Center 中的使用者、建置器 ID 和聯合身分使用者的 MFA。為了提高安全性，您可以在允許使用者存取資源或採取特定動作，並將這些政策連接到 IAM 角色之前，建立需要設定 MFA 的政策。依預設，IAM Identity Center 已預先設定啟用了 MFA，因此 IAM Identity Center 中的所有使用者在登入時除了使用其使用者名稱和密碼之外，還必須使用 MFA。

> **Note**
>
> 所有 AWS 帳戶 類型 （ 獨立、管理和成員帳戶 ） 都需要為其根使用者設定 MFA。 AWS 管理主控台 如果尚未啟用 MFA，使用者必須在第一次登入嘗試存取 的 35 天內註冊 MFA。

如需詳細資訊，請參閱 Configure MFA in IAM Identity Center 和 AWS IAM 中的多重要素驗證。

為最低權限許可做好準備

使用最低權限許可是 IAM 的最佳實務建議。最低權限許可的概念是指僅授與使用者執行任務所需的許可，而且無需額外許可。設定完成後，請思考支援最低權限許可的方式。根使用者、管理使用者和緊急存取 IAM 使用者擁有日常工作不需要的強大許可。當您了解 AWS 並測試不同的服務時，我們建議您在 IAM Identity Center 中建立至少一個額外的使用者，並具有較少的許可，您可以在不同的案例中使用。您可以使用 IAM 政策定義特定條件下可對特定資源採取的動作，然後透過權限較低的帳戶連結至這些資源。

如果您使用的是 IAM Identity Center，請考慮使用 IAM Identity Center 許可集來展開行動。若要了解詳情，請參閱《IAM Identity Center 使用者指南》中的建立許可集。

如果您使用的不是 IAM Identity Center，請使用 IAM 角色為不同的 IAM 實體定義許可。如需詳細資訊，請參閱 IAM 角色建立。

IAM 角色和 IAM Identity Center 許可集都可以根據任務函數使用 AWS 受管政策。如需這些政策所授與權限的詳細資訊，請參閱 AWS 任務函數的 受管政策。

> **Important**
>
> 請記住， AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為它們可供所有 AWS 客戶使用。完成設定後，建議您使用 IAM Access Analyzer 來根據記錄在 AWS CloudTrail的存取活動產生最低權限政策。如需政策產生的詳細資訊，請參閱 IAM Access Analyzer 政策產生。

當您開始使用時，我們建議您使用 AWS 受管政策來授予許可。在過了預先定義的閒置期間 (例如 90 天) 後，您可以檢閱人員和工作負載存取過的服務。然後，您可以建立新的客戶受管政策，並降低取代 AWS 受管政策的許可。新的政策應僅包含範例期間內存取的服務。更新您的許可以移除 AWS 受管政策，並連接您建立的新客戶受管政策。

如需精簡許可的說明，請參閱 檢閱您 AWS 帳戶的上次存取資訊。識別未使用的服務之後，請參閱 根據存取活動產生政策。

檢閱您 AWS 帳戶的上次存取資訊

您可以使用 IAM 主控台 AWS CLI或 AWS API 來檢視 IAM 的服務上次存取資訊。如需關於資料、必要許可、疑難排解及支援區域的重要資訊，請參閱AWS 使用上次存取的資訊在 中精簡許可。

您可以在 IAM 中檢視下列資源類型的資訊。在每個案例中，資訊涵蓋指定報告期間允許的服務：

- IAM 使用者：檢視使用者上一次嘗試存取每個允許服務的時間。
- IAM 群組：檢視有關上一次 IAM 群組成員嘗試存取每個允許服務的資訊。此報告也包含曾嘗試存取的成員總數。
- IAM 角色：檢視上一次有人使用該角色嘗試存取每個允許服務的時間。
- 政策 – 檢視有關上一次使用者或角色嘗試存取每個允許服務的資訊。此報告也包含曾嘗試存取的實體總數。

> **Note**
>
> 在檢視 IAM 中的資源的存取資訊之前，請確定您了解報告期間、報告的實體，以及您資訊的評估政策類型。如需詳細資訊，請參閱the section called “關於上次存取資訊的注意事項”。

如需上次存取資訊的詳細資訊，請參閱AWS 使用上次存取的資訊在 中精簡許可。

若要檢閱 的上次存取資訊 AWS 帳戶

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者群組、使用者、角色或政策。
4. 選擇任意使用者、使用者群組、角色或政策名稱以開啟其摘要頁面，然後選擇上次存取索引標籤。根據您選擇的資源，檢視下列資訊：
   - 使用者群組 – 檢視使用者群組成員可存取的服務清單。您也可以檢視成員上次存取服務的時間、他們使用的使用者群組政策，以及提出要求的使用者群組成員。選擇政策的名稱以了解其為受管政策或內嵌使用者群組政策。選擇使用者群組成員的名稱以檢視該使用者群組的所有成員，以及他們上次存取該服務的時間。
   - 使用者 – 檢視使用者可存取的服務清單。您也可以檢視他們上次存取服務的時間，以及目前與該名使用者關聯的政策有哪些。選擇政策的名稱，以了解該政策是受管政策、內嵌使用者政策還是使用者群組的內嵌政策。
   - 角色 – 檢視角色可存取的服務清單、角色上次存取該服務的時間，以及他們使用哪些政策。選擇政策的名稱以了解其為受管政策還是內嵌角色政策。

- 政策 – 檢視政策中包含允許動作的服務清單。您也可以檢視上次使用政策來存取服務的時間，以及使用政策的實體 (使用者或角色)。上次存取日期也包含透過其他政策授予此政策存取權的時間。選擇實體的名稱以了解哪些實體有連接政策，以及它們上次存取服務的時間。

5. 在表格的服務欄中，選擇其中一項包含動作上次存取資訊的服務之名稱，以檢視 IAM 實體嘗試存取的管理動作清單。您可以檢視 AWS 區域 和時間戳記，顯示上次有人嘗試執行動作的時間。
6. 針對包含動作上次存取資訊的服務，將為其服務和管理動作顯示上次存取欄。檢閱此欄中傳回的下列可能結果。這些結果會根據是否允許、存取服務或動作，以及 AWS 是否追蹤上次存取的資訊而有所不同。

   <number of> 天前

   自追蹤期間內使用服務或動作之後的天數。服務的追蹤期為過去 400 天。Amazon S3 動作的追蹤期由 2020 年 4 月 12 日開始。Amazon EC2、IAM 和 Lambda 行動追蹤期由 2021 年 4 月 7 日開始。所有其他服務的追蹤期從 2023 年 5 月 23 日開始。若要進一步了解每個追蹤開始日期 AWS 區域，請參閱 其中 會 AWS 追蹤上次存取的資訊。

   在追蹤期內未存取

   追蹤的服務或動作尚未由實體在追蹤期間內使用。

   您可以擁有清單中未出現的動作許可。如果 AWS目前不包含動作的追蹤資訊，就可能發生這種情況。您不應該只根據有沒有追蹤資訊來決定許可。相反地，建議您使用此資訊來告知並支援授予最低權限的整體策略。檢查您的政策以確認存取層級是否適當。

AWS CLI

您可以使用 AWS CLI 來擷取上次 AWS 帳戶 使用 中的 IAM 資源來嘗試存取 AWS 服務和 Amazon S3、Amazon EC2、IAM 和 Lambda 動作的相關資訊。IAM 資源可以是使用者、使用者群組、角色或政策。

- 在 AWS 帳戶中產生 IAM 資源的報告。此請求必須包含您需要報告的 IAM 資源 (使用者、使用者群組、角色或政策) 的 ARN。您可以在報告中指定要產生的資料粒度層級，以檢視任一服務或同時檢視服務和動作的存取詳細資訊。它會傳回 `job-id`，然後您可將它用於 `get-service-last-accessed-details` 和 `get-service-last-accessed-details-with-entities` 操作以監控 `job-status`，直到任務完成。

  - aws iam generate-service-last-accessed-details

a. 使用上個步驟的 `job-id` 參數來擷取報告的詳細資訊。

   - aws iam get-service-last-accessed-details

   根據您在 `generate-service-last-accessed-details` 操作中請求的資源類型與精細程度，此操作會傳回以下資訊：

   - 使用者 – 傳回指定的使用者可存取的服務清單。對於每個服務，此操作會傳回使用者最後一次嘗試的日期與時間，以及該使用者的 ARN。
   - 使用者群組 – 傳回指定使用者群組的成員可使用連接至該使用者群組的政策進行存取的服務清單。對於每個服務，此操作會傳回任何使用者群組成員最後一次嘗試的日期與時間。它也會傳回該使用者的 ARN 以及曾嘗試存取服務的使用者群組成員總數。使用 GetServiceLastAccessedDetailsWithEntities 操作來擷取所有成員的清單。
   - 角色 – 傳回指定的角色可存取的服務清單。對於每個服務，此操作會傳回角色最後一次嘗試的日期與時間，以及該角色的 ARN。
   - 政策 – 傳回指定的政策允許存取的服務清單。對於每個服務，此操作會傳回實體 (使用者或角色) 上次使用政策來嘗試存取服務的日期和時間。它也會傳回實體的 ARN 以及嘗試存取的實體總數。

b. 進一步了解在嘗試存取特定服務時使用使用者群組或政策許可的實體。此操作會傳回實體清單，包括各實體的 ARN、ID、名稱、路徑、類型 (使用者或角色)，以及它們最後一次嘗試存取服務的時間。您也可以針對使用者和角色使用此操作，但只會傳回有關該實體的資訊。

   - aws iam get-service-last-accessed-details-with-entities

c. 進一步了解有關身分 (使用者、使用者群組或角色) 在嘗試存取特定服務時使用之以身分為基礎的政策。當您指定身分和服務時，此操作會傳回實體可用於存取指定服務的許可政策清單。此操作可提供政策的目前狀態，而且不倚賴產生的報告。它也不會傳回其他政策類型，例如以資源為基礎的政策、存取控制清單、 AWS Organizations 政策、IAM 許可界限或工作階段政策。如需詳細資訊，請參閱政策類型或單一帳戶中請求的政策評估。

   - aws iam list-policies-granting-service-access

API

您可以使用 AWS API 擷取上次使用 IAM 資源嘗試存取 AWS 服務和 Amazon S3、Amazon EC2、IAM 和 Lambda 動作的相關資訊。IAM 資源可以是使用者、使用者群組、角色或政策。您可以在報告中指定要產生的資料細微層級，以檢視任一服務或同時檢視服務和動作的詳細資訊。

1. 產生報告。此請求必須包含您需要報告的 IAM 資源 (使用者、使用者群組、角色或政策) 的 ARN。它會傳回 `JobId`，然後您可將它用於 `GetServiceLastAccessedDetails` 和 `GetServiceLastAccessedDetailsWithEntities` 操作以監控 `JobStatus`，直到任務完成。

   - GenerateServiceLastAccessedDetails

2. 使用上個步驟的 `JobId` 參數來擷取報告的詳細資訊。

   - GetServiceLastAccessedDetails

   根據您在 `GenerateServiceLastAccessedDetails` 操作中請求的資源類型與精細程度，此操作會傳回以下資訊：

   - 使用者 – 傳回指定的使用者可存取的服務清單。對於每個服務，此操作會傳回使用者最後一次嘗試的日期與時間，以及該使用者的 ARN。
   - 使用者群組 – 傳回指定使用者群組的成員可使用連接至該使用者群組的政策進行存取的服務清單。對於每個服務，此操作會傳回任何使用者群組成員最後一次嘗試的日期與時間。它也會傳回該使用者的 ARN 以及曾嘗試存取服務的使用者群組成員總數。使用 GetServiceLastAccessedDetailsWithEntities 操作來擷取所有成員的清單。
   - 角色 – 傳回指定的角色可存取的服務清單。對於每個服務，此操作會傳回角色最後一次嘗試的日期與時間，以及該角色的 ARN。
   - 政策 – 傳回指定的政策允許存取的服務清單。對於每個服務，此操作會傳回實體 (使用者或角色) 上次使用政策來嘗試存取服務的日期和時間。它也會傳回實體的 ARN 以及嘗試存取的實體總數。

3. 進一步了解在嘗試存取特定服務時使用使用者群組或政策許可的實體。此操作會傳回實體清單，包括各實體的 ARN、ID、名稱、路徑、類型 (使用者或角色)，以及它們最後一次嘗試存取服務的時間。您也可以針對使用者和角色使用此操作，但只會傳回有關該實體的資訊。

   - GetServiceLastAccessedDetailsWithEntities

4. 進一步了解有關身分 (使用者、使用者群組或角色) 在嘗試存取特定服務時使用之以身分為基礎的政策。當您指定身分和服務時，此操作會傳回實體可用於存取指定服務的許可政策清單。此

操作可提供政策的目前狀態，而且不倚賴產生的報告。它也不會傳回其他政策類型，例如以資源為基礎的政策、存取控制清單、 AWS Organizations 政策、IAM 許可界限或工作階段政策。如需詳細資訊，請參閱政策類型或單一帳戶中請求的政策評估。

- ListPoliciesGrantingServiceAccess

根據存取活動產生政策

您可以使用 中針對 IAM 使用者或 IAM 角色記錄 AWS CloudTrail 的存取活動，讓 IAM Access Analyzer 產生客戶受管政策，僅允許存取特定使用者和角色所需的服務。

IAM Access Analyzer 產生 IAM 政策時，系統會傳回資訊以協助您進一步自訂政策。產生政策時，可以傳回兩種類別的資訊：

- 具有動作層級資訊的政策 – 對於某些 AWS 服務，例如 Amazon EC2，IAM Access Analyzer 可以識別 CloudTrail 事件中找到的動作，並列出它產生的政策中使用的動作。如需支援服務的清單，請參閱 IAM Access Analyzer 政策產生服務。對於某些服務，IAM Access Analyzer 會提示您將服務的動作新增至產生的政策。
- 具有服務層級資訊 – 的政策 IAM Access Analyzer 會使用最近存取的資訊來建立包含最近使用之所有服務的政策範本。使用 時 AWS 管理主控台，我們會提示您檢閱服務並新增動作以完成政策。

若要根據存取活動產生政策

在下列程序中，我們將根據使用者的使用量減少授予角色的許可。當您選擇使用者時，請選擇使用方式可體現角色的使用者。許多客戶使用 PowerUser 許可設定測試使用者帳戶，然後讓他們在短時間內執行一組特定的任務，以判斷執行這些任務所需的存取權。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，先選擇使用者，然後選擇使用者名稱以前往使用者詳細資訊頁面。
4. 在許可索引標籤的「根據 CloudTrail 事件產生政策」下，選擇產生政策。
5. 在產生政策頁面上，設定下列項目：
   - 對於選取期間，選擇過去 7 天。
   - 對於要分析的 CloudTrail 追蹤，選取記錄此使用者活動的區域和追蹤。

- 選擇建立並使用新的服務角色。

6. 選擇產生政策，然後等待角色建立。在出現正在產生政策通知訊息之前，請勿重新整理或離開主控台頁面。
7. 產生政策後，必須視需要使用資源的帳戶 ID 和 ARN 檢閱和自訂政策。此外，自動產生的政策可能不會包含完成政策所需的動作層級資訊。如需詳細資訊，請參閱產生 IAM Access Analyzer 政策。

   例如，您可能會編輯包含 `Allow` 效果和 `NotAction` 元素的第一個陳述式，以僅允許 Amazon EC2 和 Amazon S3 動作。為此，請使用含 `FullAccessToSomeServices` ID 的陳述式取代它。您的新政策可能如以下範例政策所示。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Sid": "FullAccessToSomeServices",
          "Effect": "Allow",
          "Action": [
              "ec2:*",
              "s3:*"
          ],
          "Resource": "*"
      },
      {
          "Effect": "Allow",
          "Action": [
              "iam:CreateServiceLinkedRole",
              "iam:DeleteServiceLinkedRole",
              "iam:ListRoles",
              "organizations:DescribeOrganization"
          ],
          "Resource": "*"
      }
  ]
}
```

8. 若要支援授予最低權限的最佳實務，請檢閱和修正政策驗證期間返回的任何錯誤、警告或建議。

9. 若要進一步降低您政策對特定動作和資源的許可，請在 CloudTrail Event history (事件歷程記錄) 查看您的事件。您可以在這裡查看您的使用者曾存取之特定動作和資源的詳細資訊。如需詳細資訊，請參閱 AWS CloudTrail 使用者指南中的在 CloudTrail 主控台中檢閱 CloudTrail 事件。
10. 檢閱並驗證政策後，使用描述性名稱儲存它。
11. 導覽至角色頁面，選擇當人員執行新政策允許的任務時，將擔任的角色。
12. 在許可索引標籤上依序選擇新增許可、連接政策。
13. 在連接許可政策頁面的其他許可政策清單中，選取您建立的政策，然後選擇連接政策。
14. 您會返回角色詳細資訊頁面。有兩個政策連接到角色，您先前的 AWS 受管政策，例如 PowerUserAccess，以及您的新政策。選取 AWS 受管政策的核取方塊，然後選擇移除。當系統要求確認移除時，選擇移除。

擔任此角色的 IAM 使用者、SAML 和 OIDC 聯合身分主體及工作負載的存取權現已根據您建立的新政策縮減。

AWS CLI

可以使用以下列命令藉由 AWS CLI產生政策。

產生政策

- aws accessanalyzer start-policy-generation

檢視產生的政策

- aws accessanalyzer get-generated-policy

取消政策產生請求

- aws accessanalyzer cancel-policy-generation

檢視政策產生請求的清單

- aws accessanalyzer list-policy-generations

API

您可以使用下列操作來使用 AWS API 產生政策。

產生政策

- StartPolicyGeneration

檢視產生的政策

- GetGeneratedPolicy

取消政策產生請求

- CancelPolicyGeneration

檢視政策產生請求的清單

- ListPolicyGenerations

使用搜尋尋找 IAM 資源

當您處理存取權調查結果時，可以使用 IAM 主控台搜尋頁面做為更快的選項來尋找 IAM 資源。您可以使用部分資源名稱或 ARN 來搜尋資源。

Console

IAM 主控台搜尋功能可以尋找下列任何內容：

- 符合您的搜尋關鍵字 IAM 實體名稱 (適用於使用者、群組、角色、身分提供者和政策)
- 符合您搜尋關鍵字的任務

IAM 主控台搜尋功能不會傳回 IAM Access Analyzer 的相關資訊。

在搜尋結果的每一列都是作用中的連結。例如，您可以在搜尋結果中選擇使用者名稱，該使用者名稱將帶您前往該使用者的詳細資訊頁面。或者，您可以選擇動作連結 (例如建立使用者) 前往 Create User (建立使用者) 頁面。

> **Note**
>
> 存取金鑰搜尋要求您在搜尋方塊中輸入完整的存取金鑰 ID。搜尋結果顯示與該金鑰關聯的使用者。然後您可以直接導覽至該使用者的頁面，您可以在其中管理存取金鑰。

使用 IAM 主控台中的 Search (搜尋) 頁面找到與該帳戶相關的項目。

在 IAM 主控台中搜尋項目

1. 按照《AWS 登入使用者指南》 如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇 Search (搜尋)。
4. 在 Search (搜尋) 方塊中，輸入您的搜尋關鍵字。
5. 在搜尋結果清單中選擇連結來導覽至主控台的相應部分。

以下圖示識別搜尋發現的項目類型：

圖示	描述
	IAM 使用者
	IAM 群組
	IAM 角色
	IAM 政策
	任務，例如「建立使用者」或「連接政策」
	關鍵字 **`delete`** 的結果

搜尋字詞範例

您可以在 IAM 搜尋中使用以下字詞。用您想要尋找的實際 IAM 使用者、群組、角色、存取金鑰、政策或身分提供者的名稱取代斜體中的術語。

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

中的安全最佳實務和使用案例 AWS Identity and Access Management

AWS Identity and Access Management (IAM) 提供數種安全功能，供您在開發和實作自己的安全政策時考慮。以下最佳實務為一般準則，並不代表完整的安全解決方案。

這些最佳實務可能不適合或不足以滿足您的特定環境需求。因此，請將它們視為有用的考量，而非要求。

若要取得 IAM 的最大好處，請花些時間了解所建議的最佳實務。其中一個有效方法是，了解 IAM 搭配其他 AWS 服務用於真實世界的案例。

主題

- IAM 中的安全最佳實務
- 您 的根使用者最佳實務 AWS 帳戶
- 適用於 IAM 的商業使用案例

IAM 中的安全最佳實務

Follow us on Twitter

為了協助保護您的 AWS 資源，請遵循 (IAM) 的 AWS Identity and Access Management 這些最佳實務。

實作這些最佳實務後，請參閱 IAM 和 中的安全性 AWS STS 以取得其他安全主題，包括合規驗證、彈性和基礎設施安全。

主題

- 要求人類使用者使用聯合身分提供者，以 AWS 使用臨時憑證存取
- 要求工作負載使用具有 IAM 角色的臨時憑證來存取 AWS
- 需要多重要素驗證 (MFA)
- 對於需要長期憑證的使用案例，請視需要更新存取金鑰
- 遵循最佳實務以保護您的根使用者憑證
- 套用最低權限許可
- 開始使用 AWS 受管政策，並朝最低權限的許可邁進

- 使用 IAM Access Analyzer 根據存取活動產生最低權限政策
- 定期檢閱並移除未使用的使用者、角色、許可、政策和憑證
- 使用 IAM 政策中的條件進一步限制存取權
- 使用 IAM Access Analyzer 驗證對資源的公開與跨帳戶存取權
- 使用 IAM Access Analyzer 驗證您的 IAM 政策是否有安全且功能正常的許可
- 建立跨多個帳戶的許可防護機制
- 使用許可界限委派帳戶內的許可管理

要求人類使用者使用聯合身分提供者，以 AWS 使用臨時憑證存取

人類使用者具有人類身分，是應用程式的相關人員、管理員、開發人員、操作員和消費者。它們必須具有身分才能存取您的 AWS 環境和應用程式。擁有您組織成員身分的人類使用者，也具有人力身分。人類使用者也可以是與您合作的外部使用者，以及與您的 AWS 資源互動的使用者。他們可以透過 Web 瀏覽器、用戶端應用程式、行動應用程式或互動式命令列工具來執行此操作。

要求您的人類使用者在存取時使用臨時登入資料 AWS。您可以為您的人類使用者使用身分提供者，AWS 帳戶 透過擔任提供臨時登入資料的 角色來提供 的聯合存取。如果是集中式存取管理，我們建議您使用 AWS IAM Identity Center (IAM Identity Center) 管理您帳戶的存取權和這些帳戶內的許可。您可以使用 IAM Identity Center 管理使用者身分，或從外部身分提供者管理 中使用者身分的存取許可。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 AWS IAM Identity Center。

您可以使用帳戶存取管理員 — IAM 功能，可讓您將組織帳戶中現有的 IAM 角色指派給 IAM Identity Center 使用者和群組。帳戶存取管理員可讓您存取完整的 IAM 角色功能集，包括自訂信任政策、ABAC 的角色標籤，以及可設定的角色路徑。您可以搭配許可集或單獨使用。

如需角色的詳細資訊，請參閱角色術語和概念。

要求工作負載使用具有 IAM 角色的臨時憑證來存取 AWS

工作負載是可提供商業價值的資源和程式碼的集合，例如應用程式或後端程序。您的工作負載可能有需要憑證才能向 AWS 服務發出請求 (例如讀取 Amazon S3 中資料的請求) 的應用程式、操作工具和元件。

當您在 Amazon EC2 或 Lambda 等 AWS 運算服務上建置 時， 會將 IAM 角色的臨時登入資料 AWS 交付至該運算資源。使用 AWS SDK 撰寫的應用程式將探索和使用這些臨時登入資料來存取 AWS 資源，而且不需要將 IAM 使用者的長時間存留登入資料分發到執行 的工作負載 AWS。

在 外部執行的工作負載 AWS，例如您的內部部署伺服器、來自其他雲端供應商的伺服器，或受管持續整合和持續交付 (CI/CD) 平台，仍然可以使用臨時憑證。不過，您需要將這些臨時憑證提供給工作負載。以下是將臨時憑證提供給工作負載時可採用的方式：

- 您可以使用 IAM Roles Anywhere，從公有金鑰基礎設施 (PKI) 使用 X.509 憑證請求工作負載的臨時 AWS 憑證。
- 您可以使用外部身分提供者 AWS STS `AssumeRoleWithSAML`(IdP) 的 SAML 聲明，呼叫 API 來請求工作負載的臨時 AWS 憑證 AWS 帳戶。
- 您可以使用 IdP AWS STS `AssumeRoleWithWebIdentity`中的 JSON Web 權杖 (JWT)，呼叫 API 來請求工作負載的臨時 AWS 憑證 AWS 帳戶。
- 您可以使用 的 Mutual Transport Layer Security (MTLS) 身分驗證，從您的 IoT 裝置請求臨時 AWS 憑證 AWS IoT Core。

有些 AWS 服務 也支援整合，將臨時登入資料交付至在下列位置之外執行的工作負載 AWS：

- Amazon Elastic Container Service (Amazon ECS) Anywhere 可讓您在自己的運算資源上執行 Amazon ECS 任務，並將臨時 AWS 憑證提供給在這些運算資源上執行的 Amazon ECS 任務。
- Amazon Elastic Kubernetes Service Hybrid Nodes 可讓您將在 外部作為 AWS 節點執行的運算資源加入 Amazon EKS 叢集。Amazon EKS 可以將臨時憑證提供給在運算資源上執行的 Amazon EKS Pod。
- AWS Systems ManagerHybrid Activations 可讓您 AWS 使用 SSM 管理在 之外執行的運算資源，並將臨時 AWS 登入資料交付給在運算資源上執行的 SSM 代理程式。

需要多重要素驗證 (MFA)

我們建議將 IAM 角色用於會存取您 AWS 資源的人類使用者和工作負載，以便他們使用暫時性憑證。但是，對於帳戶中需要 IAM 使用者或根使用者的情況，則需要 MFA 提供額外的安全性。使用 MFA，使用者便可擁有一個裝置，針對身分驗證查問產生回應。擁有每位使用者的憑證及裝置產生的回應，才能完成登入程序。如需詳細資訊，請參閱AWS IAM 中的多重要素驗證。我們建議您盡可能使用網路釣魚防護 MFA，例如通行金鑰和安全金鑰。如需詳細資訊，請參閱在 中指派通行金鑰或安全金鑰 AWS 管理主控台。

如果將 AM Identity Center 用於人類使用者的集中式存取管理，則可在使用 IAM Identity Center 身分存放區、 AWS Managed Microsoft AD 或 AD Connector 設定身分來源時使用 IAM Identity Center MFA 功能。如需有關 IAM Identity Center 中的 MFA 的詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的多重要素驗證。

對於需要長期憑證的使用案例，請視需要更新存取金鑰

我們建議您盡可能依賴暫時性憑證，而不要建立長期憑證，例如存取金鑰。但是，對於需要具有程式化存取和長期憑證的 IAM 使用者的情況，我們建議您視需要更新存取金鑰，如在員工離職時。我們建議您使用 IAM 存取上次使用的資訊，以便安全地更新和移除存取金鑰。如需詳細資訊，請參閱更新存取金鑰。

有特定的使用案例需要具有 IAM 使用者的長期登入資料 AWS。以下是部分使用案例：

- 無法使用 IAM 角色的工作負載 - 您可以從需要存取 AWS的位置執行工作負載。在某些情況下，您無法使用 IAM 角色提供暫時性憑證，例如針對 WordPress 外掛程式。在這些情況下，請將 IAM 使用者長期存取金鑰用於該工作負載，對 AWS進行身分驗證。
- 第三方 AWS 用戶端 – 如果您使用不支援透過 IAM Identity Center 存取的工具，例如非託管的第三方 AWS 用戶端或廠商 AWS，請使用 IAM 使用者長期存取金鑰。
- AWS CodeCommit 存取 – 如果您使用 CodeCommit 存放程式碼，您可以使用 IAM 使用者搭配 SSH 金鑰或服務特定的登入資料，讓 CodeCommit 驗證您的儲存庫。除了將 IAM Identity Center 中的使用者用於一般身分驗證之外，我們也建議您這樣做。IAM Identity Center 中的使用者是您人力中需要存取您的 AWS 帳戶 或雲端應用程式的人員。若要將您 CodeCommit 儲存庫的存取權授予使用者，而不設定 IAM 使用者，您可以設定 git-remote-codecommit 公用程式。如需 IAM 和 CodeCommit 的詳細資訊，請參閱 CodeCommit 的 IAM 登入資料：Git 登入資料、SSH 金鑰和 AWS 存取金鑰。如需設定git-remote-codecommit公用程式的詳細資訊，請參閱AWS CodeCommit 《 使用者指南》中的使用輪換憑證連線至 AWS CodeCommit 儲存庫。
- Amazon Keyspaces (適用於 Apache Cassandra) 存取 – 在無法使用 IAM Identity Center 中的使用者的情況下，例如為了測試 Cassandra 相容性，您可以將 IAM 使用者搭配服務特定憑證使用，以便使用 Amazon Keyspaces 進行身分驗證。IAM Identity Center 中的使用者是您人力中需要存取您的 AWS 帳戶 或雲端應用程式的人員。您也可以使用暫時性憑證連線到 Amazon Keyspaces。如需詳細資訊，請參閱 Amazon Keyspaces (for Apache Cassandra) Developer Guide 中的 Using temporary credentials to connect to Amazon Keyspaces using an IAM role and the SigV4 plugin。

遵循最佳實務以保護您的根使用者憑證

建立 時 AWS 帳戶，您可以建立根使用者登入資料以登入 AWS 管理主控台。如同保護其他敏感的個人資訊那樣，保護您的根使用者憑證。若要更好地了解如何保護和擴展根使用者程序，請參閱 您 的根使用者最佳實務 AWS 帳戶。

套用最低權限許可

設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。探索工作負載或使用案例所需的許可時，您可能會從廣泛許可開始。隨著使用案例的成熟，您可以設法減少授予的許可，以便朝向最低權限的目標邁進。如需使用 IAM 套用許可的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

開始使用 AWS 受管政策，並朝最低權限的許可邁進

若要開始授予許可給使用者和工作負載，請使用 AWS 受管政策，為許多常見使用案例授予許可。它們可在您的 中使用 AWS 帳戶。請記住， AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為它們可供所有 AWS 客戶使用。因此，我們建議您定義使用案例的客戶管理政策，以便進一步減少許可。如需詳細資訊，請參閱AWS 受管政策。如需專為特定任務函數設計的 AWS 受管政策的詳細資訊，請參閱 AWS 任務函數的 受管政策。

使用 IAM Access Analyzer 根據存取活動產生最低權限政策

若只授予執行任務所需的許可，您可以根據在 AWS CloudTrail中記錄的存取活動產生政策。IAM Access Analyzer 會分析您 IAM 角色使用的服務和動作，然後產生您可以使用的精細政策。測試產生的每個政策後，您可以將政策部署到生產環境。這可確保您僅授予所需的許可給工作負載。如需政策產生的詳細資訊，請參閱 IAM Access Analyzer 政策產生。

定期檢閱並移除未使用的使用者、角色、許可、政策和憑證

您的 AWS 帳戶中可能存在不再需要的 IAM 使用者、角色、許可、政策或憑證。IAM 會提供上次存取的資訊，協助您識別不再需要的使用者、角色、許可、政策和憑證，以便您移除這些資料。這可協助您減少必須監控的使用者、角色、許可、政策和憑證數量。您也可以使用此資訊來精簡 IAM 政策，以便更完善地遵循最低權限許可。如需詳細資訊，請參閱AWS 使用上次存取的資訊在 中精簡許可。

使用 IAM 政策中的條件進一步限制存取權

您可以根據生效的政策陳述式指定條件。如此，您便可以授予對動作和資源的存取權，前提是存取請求符合特定條件。例如，您可以撰寫政策條件，指定必須使用 TLS 傳送所有請求。您也可以使用條件來授予對服務動作的存取權，但僅限透過特定 使用時 AWS 服務，例如 CloudFormation。如需詳細資訊，請參閱IAM JSON 政策元素：Condition。

使用 IAM Access Analyzer 驗證對資源的公開與跨帳戶存取權

在授予公有或跨帳戶存取的許可之前 AWS，建議您驗證是否需要此類存取。您可以使用 IAM Access Analyzer，協助自己預覽和分析所支援資源類型的公開和跨帳戶存取權。為此，您可以檢閱 IAM Access Analyzer 產生的問題清單。這些問題清單有助於您驗證資源存取控制是否授予您預期的存取權。此外，隨著您更新公開和跨帳戶許可，您可以在將新的存取控制部署到資源前驗證變更的影響。IAM Access Analyzer 也會持續監控支援的資源類型，並為允許公開或跨帳戶存取權的資源產生問題清單。如需詳細資訊，請參閱使用 IAM Access Analyzer API 預覽存取。

使用 IAM Access Analyzer 驗證您的 IAM 政策是否有安全且功能正常的許可

驗證您建立的政策，以確保其遵守 IAM 政策語言 (JSON) 和 IAM 最佳實務。您可以使用 IAM Access Analyzer 政策驗證來驗證自己的政策。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您編寫安全且實用的政策。您在主控台中編寫新政策或編輯現有政策時，IAM Access Analyzer 會提供建議，協助您在儲存政策前完善和驗證政策。此外，我們建議您檢閱並驗證所有現有政策。如需詳細資訊，請參閱 IAM Access Analyzer 政策驗證。如需 IAM Access Analyzer 所提供政策檢查的詳細資訊，請參閱《IAM Access Analyzer 政策檢查參考》。

建立跨多個帳戶的許可防護機制

當您擴展工作負載時，請使用 管理的多個帳戶來分隔它們 AWS Organizations。我們建議您使用 AWS Organizations 服務控制政策 (SCPs) 來建立許可防護機制，以控制您帳戶間所有主體 (IAM 角色和使用者) 的存取。建議使用 AWS Organizations 資源控制政策 (RCP) 來建立許可防護機制，以控制整個組織中的 AWS 資源存取。SCPs和 RCPs 是組織政策的類型，可用於在 AWS 組織、組織單位 (OU) 或帳戶層級管理組織中的許可。

不過，僅有 SCP 和 RCP 並不足以授予許可給您組織中的主體和資源。SCP 和 RCP 沒有授予任何許可。要授予許可，您必須將身分型或資源型政策連接到 IAM 使用者、IAM 角色或您帳戶中的資源。如需詳細資訊，請參閱 SRA 建置區塊 AWS Organizations-、帳戶和護欄。

使用許可界限委派帳戶內的許可管理

在某些情況下，您可能想要將帳戶內的許可管理委派給其他人。例如，您可以允許開發人員為其工作負載建立和管理角色。委派許可給其他人時，請使用許可界限來設定您委派許可的上限。許可界限是一種進階功能，可供您使用受管政策來設定身分型政策可以授予 IAM 角色的許可上限。許可界限不會自行授予許可。如需詳細資訊，請參閱IAM 實體的許可界限。

您 的根使用者最佳實務 AWS 帳戶

第一次建立 時 AWS 帳戶，您會從一組預設登入資料開始，完整存取您帳戶中的所有 AWS 資源。此身分稱為 AWS 帳戶 根使用者。強烈建議您不要存取 AWS 帳戶 根使用者，除非您的任務需要根使用者憑證。您需要保護根使用者憑證和帳戶復原機制，以協助確保您不會公開高權限憑證以供未經授權的使用。

對於透過 AWS 帳戶 管理的多個 AWS Organizations，我們建議您從成員帳戶移除根使用者憑證，以協助防止未經授權的使用。您可以移除根使用者密碼、存取金鑰、簽署憑證，以及停用和刪除多重要素驗證 (MFA)。成員帳戶無法登入其根使用者，也無法為其根使用者執行密碼復原。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。

並非存取根使用者，而是建立日常任務的系統管理使用者。

- 如果您有新的 AWS 帳戶，請參閱 設定您的 AWS 帳戶。
- 對於透過 AWS 帳戶 管理的多個 AWS Organizations，請參閱設定 IAM Identity Center 管理使用者的 AWS 帳戶 存取權。

然後，您可以使用管理使用者為需要存取 AWS 帳戶中資源的使用者建立其他身分。我們強烈建議您在存取 時，要求使用者使用臨時憑證進行身分驗證 AWS。

- 對於單一獨立 AWS 帳戶，請使用 在您的帳戶中IAM 角色建立具有特定許可的身分。角色旨在可由任何需要它的使用者擔任。此外，角色沒有與之關聯的標準長期憑證，例如密碼或存取金鑰。反之，當您擔任角色時，其會為您的角色工作階段提供臨時安全性憑證。與 IAM 角色不同，IAM 使用者 擁有長期憑證，例如密碼和存取金鑰。最佳實務建議盡可能依賴暫時憑證，而不是建立擁有長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。
- 對於透過 AWS 帳戶 管理的多個 AWS Organizations，請使用 IAM Identity Center 人力資源使用者。透過 IAM Identity Center，您可以集中管理這些帳戶中跨 AWS 帳戶 和 許可的使用者。可使用 IAM Identity Center 或外部身分供應商來管理使用者身分。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 AWS IAM Identity Center。

主題

- 保護您的根使用者憑證，以防止未經授權使用
- 使用強式根使用者密碼來協助保護存取
- 使用多重要素驗證 (MFA) 保護您的根使用者登入
- 不要為根使用者建立存取金鑰

- 盡可能為根使用者登入使用多人核准
- 使用群組電子郵件地址作為根使用者憑證
- 限制存取帳戶復原機制
- 保護 AWS Organizations 您的帳戶根使用者登入資料
- 監控存取和用量

保護您的根使用者憑證，以防止未經授權使用

保護您的根使用者憑證，僅將其用於需要它們的任務。為了防止未經授權的使用，請勿與任何人共用您的根使用者密碼、MFA、存取金鑰、CloudFront 金鑰配對或簽署憑證，除非此人具有存取根使用者憑證的嚴格業務需求。

請勿將根使用者密碼與依存 AWS 服務 於使用相同密碼存取之帳戶中的工具一起存放。如果您遺失或忘記根使用者密碼，將無法存取這些工具。我們建議您優先考慮彈性，並考慮要求兩個或更多人授權存取儲存位置。應記錄並監控任何對密碼的存取或其儲存位置。

使用強式根使用者密碼來協助保護存取

我們建議您使用高強度且唯一的密碼。具有強密碼產生演算法的密碼管理器之類的工具可協助您實現這些目標。 AWS 要求您的密碼必須符合下列條件：

- 它必須至少有 8 個字元，最多 128 個字元。
- 它至少混用 3 種下列類型字元：大寫、小寫、數字和 ! @ # $ % ^ & * () <> [] {} | _+-= 符號。
- 它不能與 AWS 帳戶 您的姓名或電子郵件地址相同。

如需詳細資訊，請參閱變更 的密碼 AWS 帳戶根使用者。

使用多重要素驗證 (MFA) 保護您的根使用者登入

因為根使用者可以執行特權動作，因此除了將電子郵件地址和密碼作為登入憑證之外，還要將根使用者的 MFA 新增為第二個身分驗證因素。您可以向 AWS 帳戶 根使用者註冊最多八個目前支援 MFA 類型之任何組合的 MFA 裝置。

強烈建議為根使用者憑證啟用多個 MFA 裝置，以在安全策略中提供額外的靈活性和彈性。所有 AWS 帳戶 類型 (獨立帳戶、管理帳戶和成員帳戶) 都要求為其根使用者設定 MFA。如果尚未啟用 MFA，使用者必須在首次嘗試登入以存取 AWS 管理主控台 的 35 天內註冊 MFA。

- 由第三方供應商提供經 FIDO 認證的硬體安全金鑰。如需詳細資訊，請參閱啟用 AWS 帳戶 根使用者的 FIDO 安全金鑰。
- 一種在以時間為基礎的一次性密碼 (TOTP) 演算法的基礎上產生六位數字程式碼的硬體裝置。如需詳細資訊，請參閱為 AWS 帳戶 根使用者啟用硬體 TOTP 字符。
- 在手機或其他裝置上執行並模擬實體裝置的虛擬驗證器應用程式。如需詳細資訊，請參閱為您的 AWS 帳戶 根使用者啟用虛擬 MFA 裝置。

不要為根使用者建立存取金鑰

存取金鑰可讓您在命令列界面 (AWS CLI) 中執行 AWS 命令，或從其中一個 AWS SDKs 使用 API 操作。我們強烈建議您不要為根使用者建立存取金鑰，因為根使用者擁有帳戶中所有 AWS 服務 和資源的完整存取權，包括帳單資訊。

由於只有少數任務需要根使用者，而且您通常不常執行這些任務，因此我們建議您登入 AWS 管理主控台 以執行根使用者任務。對於程式設計存取，而不是為根使用者建立存取金鑰，請使用 `aws login`命令搭配您的根憑證來驗證 AWS CLI 和 SDKs。此方法提供暫時、自動輪換的登入資料，可增強您的安全狀態，同時無需管理長期存取金鑰。在建立存取金鑰之前，請檢閱 長期存取金鑰的替代方案。

盡可能為根使用者登入使用多人核准

考慮使用多人核准，以確保沒有人可以同時存取根使用者的 MFA 和密碼。有些公司會設定可存取密碼的管理員群組，以及另一個可存取 MFA 的管理員群組，藉此增加額外的安全性層級。每個群組必須各有一位成員共同作為根使用者登入。

使用群組電子郵件地址作為根使用者憑證

使用由企業管理的電子郵件地址，直接將收到的訊息轉寄給使用者群組。如果 AWS 必須聯絡帳戶的擁有者，此方法可以降低延遲回應的風險，即使個人正在休假、生病或已離開公司。用於根使用者的電子郵件地址不得用於其他目的。

限制存取帳戶復原機制

請務必制定程序來管理根使用者憑證復原機制，以便在緊急情況下需要存取該憑證，例如接管您的管理帳戶。

- 請確定您能夠存取根使用者電子郵件收件匣，以便重設遺失或忘記的根使用者密碼。

- 如果 AWS 帳戶 根使用者的 MFA 遺失、損壞或無法運作，您可以使用向相同根使用者憑證註冊的另一個 MFA 登入。如果您無法存取所有 MFA，則需要用於註冊賬戶的最新且可存取的電話號碼和電子郵件，以便復原您的 MFA。如需詳細資訊，請參閱復原根使用者 MFA 裝置。
- 如果您選擇不儲存根使用者密碼和 MFA，則可以使用帳戶中註冊的電話號碼作為復原根使用者憑證的替代方法。請確定您能夠存取聯絡電話號碼、讓電話號碼保持最新狀態，以及限制誰有權管理電話號碼。

應沒有任何人有權同時存取電子郵件收件匣和電話號碼，因為兩者都是復原根使用者密碼的驗證渠道。務必安排兩組人員管理這些渠道。一個群組可以存取您的主要電子郵件地址，另一個群組可以存取主要電話號碼，從而復原以根使用者身分存取您帳戶的許可。

保護 AWS Organizations 您的帳戶根使用者登入資料

當您使用 移至多帳戶策略時 AWS Organizations，每個 AWS 帳戶 都有自己的根使用者憑證，您需要保護這些憑證。您用來建立組織的帳戶是管理帳戶，組織中的其餘帳戶則為成員帳戶。

保護管理帳戶的根使用者憑證

AWS 要求您為組織的管理帳戶的根使用者註冊 MFA。MFA 註冊必須在首次登入嘗試期間或 35 天寬限期內完成。如果未在此時間內啟用 MFA，您需要先完成註冊才能存取 AWS 管理主控台。如需詳細資訊，請參閱的多重要素驗證 AWS 帳戶根使用者。

保護成員帳戶的根使用者憑證

如果您使用 AWS Organizations 管理多個帳戶，您可以採取兩種策略來保護 中的根使用者存取 AWS Organizations。

- 集中化根存取權並從成員帳戶中移除根使用者憑證。移除根使用者憑證、存取金鑰、簽署憑證，以及停用和刪除多重要素驗證 (MFA)。使用此策略時，成員帳戶無法登入其根使用者，也無法為其根使用者執行密碼復原。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。
- 使用 MFA 保護 AWS Organizations 您帳戶的根使用者憑證，以增強帳戶安全性。如需詳細資訊，請參閱的多重要素驗證 AWS 帳戶根使用者。

如需詳細資訊，請參閱 AWS Organizations User Guide 中的 Accessing member accounts in your organization。

AWS Organizations 使用服務控制政策 (SCP) 在 中設定預防性安全控制

如果組織中的成員帳戶已啟用根使用者憑證，您可以套用 SCP 來限制對成員帳戶根使用者的存取。拒絕成員帳戶中的所有根使用者動作 (除了某些僅限根的動作)，這樣有助於防止未經授權的存取。有關詳細資訊，請參閱使用 SCP 來限制成員帳戶中根使用者可執行的動作。

監控存取和用量

我們建議您使用目前的追蹤機制來監控、警示和報告根使用者憑證的登入和使用情況，包括宣告根使用者登入和用量的警示。下列 服務可協助確保追蹤根使用者憑證用量，並執行安全檢查，以協助防止未經授權的使用。

> **Note**
>
> CloudTrail 會記錄根使用者和特權根使用者工作階段的不同登入事件。這些特權工作階段允許在您組織的成員帳戶中執行需要根使用者憑證的任務。您可以使用登入事件來識別管理帳戶或使用 `sts:AssumeRoot` 的委派管理員所採取的動作。如需詳細資訊，請參閱在 CloudTrail 中追蹤特權任務。

- 如果您希望收到有關帳戶中根使用者登入活動的通知，可以利用 Amazon CloudWatch 建立事件規則，以偵測何時使用根使用者憑證，並向安全管理員觸發通知。如需詳細資訊，請參閱監控和通知 AWS 帳戶 根使用者活動。
- 如果您想要設定通知以提醒您核准的根使用者動作，可以利用 Amazon EventBridge 和 Amazon SNS 撰寫 EventBridge 規則，以追蹤根使用者對特定動作的使用情況，並使用 Amazon SNS 主題通知您。如需範例，請參閱建立 Amazon S3 物件時傳送通知。
- 如果您已使用 GuardDuty 做為威脅偵測服務，可以擴展其功能，以便在帳戶中使用根使用者憑證時通知您。

此警示應包含但不得限於根使用者的電子郵件地址。制定如何回應警示的適當程序，從而收到根使用者存取警示的人員可了解如何驗證預期的根使用者存取權，以及如果他們認為安全事件正在進行中該如何上報。如需如何設定警示的範例，請參閱監控和通知 AWS 帳戶 根使用者活動。

評估根使用者 MFA 合規

下列服務可協助評估根使用者憑證的 MFA 合規性。

> 如果遵循移除根使用者憑證的最佳實務，則 MFA 相關規則會傳回不合規。
>
> 建議從組織的成員帳戶中移除根使用者憑證，以協助防止未經授權的使用。在移除根使用者憑證 (包括 MFA) 之後，這些成員帳戶將會評估為不適用。

- AWS Config 提供規則來監控根使用者最佳實務的合規性。您可以使用 AWS Config 受管規則來協助您強制執行根使用者憑證的 MFA。 AWS Config 也可以識別根使用者的存取金鑰。
- Security Hub CSPM 為您提供 中安全狀態的全方位檢視， AWS 並協助您根據安全產業標準和最佳實務評估您的 AWS 環境，例如在根使用者上擁有 MFA，而沒有根使用者存取金鑰。如需可用規則的詳細資訊，請參閱 Security Hub CSPM 使用者指南中的AWS Identity and Access Management 控制項。
- Trusted Advisor 提供安全檢查，讓您知道根使用者帳戶上是否未啟用 MFA。如需詳細資訊，請參閱 AWS 支援使用者指南中的根帳戶上的 MFA。

如果您需要回報帳戶的安全性問題，請參閱回報可疑電子郵件或弱點回報。或者，您可以聯絡 AWS 以取得協助和其他指導。

適用於 IAM 的商業使用案例

IAM 的簡單商業使用案例可協助您了解實作服務的基本方式，以控制使用者擁有的 AWS 存取。使用案例是在一般條款中說明，並不包含您如何使用 IAM API 來達到您想要的結果的機制。

此使用案例示範稱為 Example Corp 的虛構公司可以兩種典型的方式使用 IAM。第一個案例會考慮 Amazon Elastic Compute Cloud (Amazon EC2)。第二個則考慮 Amazon Simple Storage Service (Amazon S3)。

如需搭配其他 服務使用 IAM 的詳細資訊 AWS，請參閱 AWS 使用 IAM 的 服務。

主題

- 範例企業的初始設定
- 搭配 Amazon EC2 的 IAM 使用案例
- 搭配 Amazon S3 的 IAM 使用案例

範例企業的初始設定

Nikki Wolf 和 Mateo Jackson 是 Example Corp. 的創辦人。 啟動公司時，他們會建立 AWS 帳戶 並設定 AWS IAM Identity Center (IAM Identity Center) 來建立管理帳戶，以搭配其 AWS 資源使用。當您為管理使用者設定帳戶存取權時，IAM Identity Center 會建立對應的 IAM 角色。此角色由 IAM Identity Center 控制，在相關 中建立 AWS 帳戶，而 AdministratorAccess 許可集中指定的政策會連接到角色。

因為他們現在有管理員帳戶，Nikki 和 Mateo 不再需要使用其根使用者來存取他們的 AWS 帳戶。他們計劃僅將根使用者用於完成只能由根使用者執行的任務。在檢閱安全最佳實務之後，他們要為根使用者憑證設定多重要素驗證 (MFA)，並決定如何保護其根使用者憑證的安全。

隨著公司的發展，他們聘請員工擔任開發人員、管理員、測試人員、經理和系統管理員。Nikki 負責營運，而 Mateo 管理工程團隊。他們要設定 Active Directory Domain Server 來管理員工帳戶並管理對內部公司資源的存取權。

為了提供員工存取 AWS 資源的權限，他們使用 IAM Identity Center 將公司的 Active Directory 連接到他們的 AWS 帳戶。

因為他們將 Active Directory 連接到 IAM Identity Center，使用者、群組和群組成員資格將會被同步並且得到定義。他們必須將許可集和角色指派給不同的群組，讓使用者能夠正確存取 AWS 資源。他們在 AWS 任務函數的 受管政策中使用 AWS 管理主控台 來建立這些許可集：

- 管理員
- 帳單
- 開發人員
- 網路管理員
- 資料庫管理員
- 系統管理員
- 支援使用者

然後，他們要指派這些許可集合到被指派給其 Active Directory 群組的角色。

如需了解描述 IAM Identity Center 初始組態的逐步指南，請參閱《AWS IAM Identity Center 使用者指南》中的入門。如需有關佈建 IAM Identity Center 使用者存取權的詳細資訊，請參閱《AWS IAM Identity Center 使用者指南》中的單一登入存取 AWS 帳戶。

搭配 Amazon EC2 的 IAM 使用案例

與 Example Corp 類似的公司通常會使用 IAM 與像是 Amazon EC2 的服務互動。若要了解此部分的使用案例，您需要對 Amazon EC2 有基本的理解。如需 Amazon EC2 的詳細資訊，請參閱《Amazon EC2 使用者指南》。

使用者群組的 Amazon EC2 權限

為了提供「周邊」控制，Nikki 將政策連接到 AllUsers 使用者群組。如果來源 IP 地址位於 Example Corp 的公司網路之外，則此政策會拒絕使用者提出的任何 AWS 請求。

在 Example Corp，不同 IAM 群組需要不同的許可：

- System administrators (系統管理員) – 需要許可才能建立及管理受管 AMI、執行個體、快照、磁碟區、安全群組等。Nikki 會將`AmazonEC2FullAccess` AWS 受管政策連接至 SysAdmins 使用者群組，讓群組成員有權使用所有 Amazon EC2 動作。
- Developers (開發人員) – 僅需要使用執行個體的能力。Nikki 因此建立政策並將其連接到 Developers 使用者群組，讓開發人員能夠呼叫 `DescribeInstances`、`RunInstances`、`StopInstances`、`StartInstances` 和 `TerminateInstances`。

  > **Note**
  >
  > Amazon EC2 使用 SSH 金鑰、Windows 密碼及安全群組控制擁有特定 Amazon EC2 執行個體作業系統存取權限的對象。IAM 系統中沒有方法，可允許或拒絕存取特定執行個體的作業系統。

- Support 使用者 – 除了列出目前可用的 Amazon EC2 資源之外，不應擁有執行任何 Amazon EC2 動作的許可。因此，Nikki 建立政策並將其連接到 Support 使用者群組，只讓他們呼叫 Amazon EC2「描述」API 操作。

如需這些政策的外觀範例，請參閱《Amazon EC2 使用者指南》中的 以身分為基礎的 IAM 政策範例和使用 AWS Identity and Access Management。

使用者的任務職能變更

在某個時間點，其中一名開發人員 Paulo Santos 變更了工作職能成為經理。身為經理，Paulo 成為 Support 使用者群組的一員，因此他可以為其開發人員開啟支援案例。Mateo 將 Paulo 從 Developers

使用者群組移到 Support 使用者群組。其結果是，他與 Amazon EC2 執行個體互動的能力受到限制。他無法啟動或開始執行個體。他也無法停止或終止現有的執行個體，即使他是啟動或開始執行個體的使用者。他只能列出 Example Corp 使用者啟動的執行個體。

搭配 Amazon S3 的 IAM 使用案例

與 Example Corp 相似的公司通常也會搭配 Amazon S3 使用 IAM。John has created an Amazon S3 bucket for the company called amzn-s3-demo-bucket.

建立其他使用者和使用者群組

員工 Zhang Wei 和 Mary Major 每個都需要能夠在公司的儲存貯體中建立自己的資料。他們也需要讀取和寫入所有開發人員共用的資料。為啟用此功能，Mateo 依據邏輯使用 Amazon S3 金鑰字首 (key prefix) 機制在 amzn-s3-demo-bucket 中安排資料，如下圖所示。

```
/amzn-s3-demo-bucket
    /home
        /zhang
        /major
    /share
        /developers
        /managers
```

Mateo 為每個員工將 `/amzn-s3-demo-bucket` 區分為一組主目錄，以及為開發人員和管理員的群組一個共用的區域。

現在 Mateo 建立一組政策以指派許可給使用者和使用者群組：

- Zhang 的主目錄存取權 – Mateo 將政策連接到 Wei，讓他可以讀取、寫入及列出具有 Amazon S3 金鑰字首 (key prefix) `/amzn-s3-demo-bucket/home/zhang/` 的任何物件
- Major 的主目錄存取權 – Mateo 將政策連接到 Mary，讓她可以讀取、寫入及列出具有 Amazon S3 金鑰字首 (key prefix) `/amzn-s3-demo-bucket/home/major/` 的任何物件
- 開發人員使用者群組的共享目錄存取權 – Mateo 將政策連接到使用者群組，讓開發人員可以讀取、寫入及列出 `/amzn-s3-demo-bucket/share/developers/` 中的任何物件
- 管理員群組的共享目錄存取權 – Mateo 將政策連接到使用者群組，可讓管理員讀取、寫入及列出 `/amzn-s3-demo-bucket/share/managers/` 中的物件

> **Note**
>
> Amazon S3 不會自動授予建立儲存貯體或物件的使用者，在該儲存貯體或物件上執行其他動作的許可。因此，在您的 IAM 政策中，您必須明確地提供使用者許可，以使用他們建立的 Amazon S3 資源。

如需這些政策的範例，請參閱 Amazon Simple Storage Service 使用者指南中的存取控制。如需執行時間期間評估政策方式的資訊，請參閱政策評估邏輯。

使用者的任務職能變更

在某個時間點，其中一名開發人員 Zhang Wei 變更了工作職能成為經理。我們假設他不再需要存取 `share/developers` 目錄中的文件。Mateo 身為管理員，將 Wei 移到 `Managers` 使用者群組並移出 `Developers` 使用者群組。只需簡單的重新指派，Wei 就會自動取得授予到 `Managers` 使用者群組的所有許可，但不再存取 `share/developers` 目錄中的資料。

與第三方企業整合

組織通常會與合作夥伴公司、顧問和承包商合作。Example Corp 有合作夥伴稱為 Widget Company 的公司，而 Widget Company 雇用名為 Shirley Rodriguez 的員工，其需要將資料放入儲存貯體中供 Example Corp 使用。Nikki 建立一個名為 WidgetCo 的使用者群組和名為 `Shirley` 的使用者，然後將 Shirley 新增到 WidgetCo 使用者群組。Nikki 也建立名為 amzn-s3-demo-bucket1 的特殊儲存貯體，供 Shirley 使用。

Nikki 更新現有政策或新增新的政策，以包含合作夥伴公司 Widget Company。例如，Nikki 可以建立新的政策，拒絕 WidgetCo 使用者群組的成員能夠使用寫入以外的任何動作。此政策只在有廣泛的政策可讓所有使用者存取多種 Amazon S3 動作時才必要。

IAM 教學課程

下列教學課程會針對 AWS Identity and Access Management (IAM) 的一般工作，提供完整的端對端程序。它們適用於實驗室類型的環境，包括虛構公司名稱、使用者名稱等。他們的目的是提供一般指導。他們不能直接用於您的生產環境，需要仔細審查更動以符合組織環境的獨特需求。

教學課程

- IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權
- IAM 教程：建立並連接您的第一個客戶受管政策
- IAM 教學課程：定義根據標籤存取 AWS 資源的許可
- IAM 教學課程：允許使用者管理其憑證和 MFA 設定
- IAM 教學課程：使用 CloudFormation 範本建立 SAML Identity Provider (IdP)
- IAM 教學課程：使用 CloudFormation 範本建立 SAML 聯合 IAM 角色
- IAM 教學課程：使用 CloudFormation 範本建立 SAML 身分提供者 (IdP) 和 SAML 聯合 IAM 角色

IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權

> **Important**
>
> IAM 最佳實務建議您要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 ，而不是使用具有長期憑證的 IAM 使用者。建議您僅將 IAM 使用者用於聯合身分使用者不支援的特定使用案例。

此教學課程教導您如何使用角色將存取權委派給不同 AWS 帳戶 (稱為目的地和來源) 中的資源。您與不同帳戶中的使用者分享一個帳戶中的資源。透過以這種方式設定跨帳戶存取，您不需要在每個帳戶中建立個別的 IAM 使用者。此外，使用者不需要為了存取不同 AWS 帳戶中的資源，登出一個帳戶然後登入另一個帳戶。設定角色之後，您會看到如何從 AWS 管理主控台 AWS CLI、 和 API 使用角色。

在本教學課程中，目的地帳戶會管理不同應用程式和團隊存取的應用程式資料。在每個帳戶中，應用程式資訊存放在 Amazon S3 儲存貯體中。可以在來源帳戶中管理 IAM 使用者，您在其中擁有兩個 IAM 使用者角色：開發人員和分析師。開發人員和分析師使用來源帳戶來產生多個微服務共用的資料。兩個角色都有權在來源帳戶中工作並存取資源。開發人員必須經常更新目的地帳戶中的共用資料。開發人員將此資料存放在稱為 `amzn-s3-demo-bucket-shared-container` 的 Amazon S3 儲存貯體中。

在本教學的結尾，您會擁有：

- 允許來源帳戶 (受信任帳戶) 中的使用者在目的地帳戶中擔任特定角色。
- 允許目的地帳戶 (信任帳戶) 中的角色存取特定 Amazon S3 儲存貯體。
- 目的地帳戶中的 amzn-s3-demo-bucket-shared-container 儲存貯體。

開發人員可以使用 中的 角色 AWS 管理主控台 來存取目的地帳戶中的 amzn-s3-demo-bucket-shared-container儲存貯體。他們也可以透過使用 API 呼叫來存取儲存貯體，而該呼叫是由角色提供的暫時性憑證進行身分驗證。分析師進行類似嘗試來使用角色會失敗。

此工作流程有三個基本步驟：

在目的地帳戶中建立角色

首先，您可以使用 AWS 管理主控台 建立目的地帳戶 (ID 編號 999999999999) 與原始帳戶 (ID 編號 111111111111) 之間的信任。透過建立一個名為 UpdateData 的 IAM 角色開始。建立角色後，您可以將來源帳戶定義為受信任的實體並指定許可政策，該政策允許受信任的使用者更新 amzn-s3-demo-bucket-shared-container 儲存貯體。

授予角色存取權

在本節中，您修改角色政策，以拒絕分析師存取 UpdateData 角色。由於分析師在此案例中具有 PowerUser 存取權，您必須明確拒絕使用角色的能力。

透過切換角色測試存取

最後，作為開發人員，使用 UpdateData 角色來更新目的地帳戶中的 amzn-s3-demo-bucket-shared-container 儲存貯體。您會了解如何透過 AWS 主控台 AWS CLI、 和 API 存取角色。

考量事項

在您使用 IAM 角色委派資源存取權之前 AWS 帳戶，請務必考慮下列事項：

- 以 AWS 帳戶根使用者身分登入時無法切換到特定角色。
- IAM 角色和資源型政策只會在單一分割內跨帳戶委派存取許可。例如，假設您在標準 aws 分割區的美國西部 (加利佛尼亞北部) 中有一個帳戶。您在 aws-cn 分割區的中國 (北京) 中也有一個帳戶。您不能使用中國 (北京) 中帳戶的 Amazon S3 資源型政策，對標準 aws 帳戶中的使用者允許存取許可。

- 您可以使用 AWS IAM Identity Center 來使用安全性聲明標記語言 (SAML) 促進外部 AWS 帳戶 （您外部的帳戶 AWS Organizations) 的單一登入 (SSO)。如需詳細資訊，請參閱使用 SAML 2.0 將外部整合 AWS 帳戶 到具有獨立帳單 AWS IAM Identity Center 的中央存取管理
- 您可以將角色與 Amazon EC2 執行個體或 AWS Lambda 函數等 AWS 資源建立關聯。如需詳細資訊，請參閱建立角色以將許可委派給 AWS 服務。
- 如果您想要讓應用程式在另一個 中擔任角色 AWS 帳戶，您可以使用 AWS SDK 來擔任跨帳戶角色。如需詳細資訊，請參閱 AWS SDKs and Tools Reference Guide 中的 Authentication and access。
- 使用 切換角色 AWS 管理主控台 僅適用於不需要 的帳戶`ExternalId`。例如，假設您將您帳戶的存取權授與第三方，並在許可政策的 `Condition` 元素中需要 `ExternalId`。在這種情況下，第三方只能使用 AWS API 或命令列工具存取您的帳戶。第三方無法使用主控台，因為它必須提供 `ExternalId` 的值。如需此案例的詳細資訊，請參閱 存取第三方 AWS 帳戶 擁有的和 AWS 安全部落格中的如何啟用跨帳戶存取 AWS 管理主控台 。

先決條件

此教學課程假設您已備妥下列項目：

- 您可以使用兩個不同的 AWS 帳戶 ，一個代表原始帳戶，另一個代表目的地帳戶。
- 來源帳戶中使用者和角色的建立和設定，如下所示：

職稱	使用者	許可
開發人員	David	兩個使用者可以登入並使用原始帳戶中 AWS 管理主控台的 。
分析師	Jane	

- 不需要在目的地帳戶中建立任何使用者。
- 在目的地帳戶中建立的 Amazon S3 儲存貯體。您可在此教學課程中將其稱為 `amzn-s3-demo-bucket-shared-container`，但因為 S3 儲存貯體名稱必須是全域唯一，所以您必須使用不同名稱的儲存貯體。

在目的地帳戶中建立角色

您可以允許某個使用者 AWS 帳戶 存取另一個資源 AWS 帳戶。在本教學課程中，我們將透過建立一個角色來完成操作，該角色定義誰可以存取它，以及它會將哪些許可授予給切換到它的使用者。

在教學課程的此步驟中，您在目的地帳戶中建立角色，並指定來源帳戶作為受信任的實體。您也限制角色的許可為僅擁有讀取和寫入 amzn-s3-demo-bucket-shared-container 儲存貯體的存取許可。被授予許可以使用角色的任何人都可以讀取和寫入 shared-container 儲存貯體。

在建立角色之前，您需要原始 AWS 帳戶伺服器的帳戶 ID。每個 AWS 帳戶 都有指派的唯一帳戶 ID 識別符。

若要取得來源 AWS 帳戶 ID

1. 以原始帳戶的管理員 AWS 管理主控台 身分登入 ，然後開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在 IAM 主控台中，在右上角的導覽列中選擇您的使用者名稱。它通常如下：*username*@*account_ID_number_or_alias*。

   在此案例中，您可以使用帳戶 ID 111111111111 表示來源帳戶。不過，如果您在測試環境中使用此案例，則應該使用有效的帳戶 ID。

若要在目的地帳戶中建立來源帳戶可以使用的角色

1. 以目的地帳戶的管理員 AWS 管理主控台 身分登入 ，然後開啟 IAM 主控台。
2. 建立角色之前，請準備定義角色所需許可的受管政策。您可以在稍後的步驟將此政策連接至角色。

   您想要設定 amzn-s3-demo-bucket-shared-container 儲存貯體的讀取和寫入存取許可。雖然 AWS 提供一些 Amazon S3 受管政策，但沒有提供單一 Amazon S3 儲存貯體的讀取和寫入存取權。您可以改為建立自己的政策。

   在導覽窗格中，選擇 Policies (政策)，然後選擇 Create policy (建立政策)。
3. 選擇 JSON 標籤並從下列 JSON 政策文件複製文字。將此文字貼到 JSON 文字方塊中，以您 Amazon S3 儲存貯體的真實 ARN 替換資源 ARN (arn:aws:s3:::shared-container)。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container/*"
    }
  ]
}
```

`ListAllMyBuckets` 動作授予許可，可列出已經過身分驗證的請求發送者擁有的所有儲存貯體 `ListBucket` 許可允許使用者檢視 `amzn-s3-demo-bucket-shared-container` 儲存貯體中的物件。`GetObject`、`PutObject`、`DeleteObject` 許可允許使用者檢視、更新和刪除 `amzn-s3-demo-bucket-shared-container` 儲存貯體中的內容。

> **Note**
>
> 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

4. 在檢視與建立頁面上，針對政策名稱輸入 **`read-write-app-bucket`**。檢視政策授與的許可，然後選擇建立政策來儲存您的工作。

   新的政策會出現在受管政策清單中。

5. 在導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
6. 選擇 AWS 帳戶 角色類型。
7. 針對帳戶 ID，輸入來源帳戶 ID。

此教學課程將範例帳戶 ID **111111111111** 用於來源帳戶。您應該使用有效的帳戶 ID。如果您使用無效的帳戶 ID (例如 **111111111111**)，IAM 不會讓您建立新的角色。

現在您不需要要求外部 ID，或要求使用者要有多重要素驗證 (MFA) 才能擔任角色。請保持不選取這些選項。如需更多詳細資訊，請參閱 AWS IAM 中的多重要素驗證。

8. 選擇 Next: Permissions (下一步：許可)，以設定與角色建立關聯的許可。
9. 選擇您之前建立的政策旁的核取方塊。

> **秘訣**
>
> 針對 Filter (篩選條件)，選擇 Customer managed (客戶受管) 以篩選清單，使其僅包含建立的政策。這會隱藏 AWS 建立的政策並讓找出所需項目更容易。

然後選擇下一步。

10. (選用) 藉由連接標籤作為鍵值對，將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
11. (選用) 在 Description (說明) 中，輸入新角色的說明。
12. 檢閱角色之後，選擇 Create role (建立角色)。

`UpdateData` 角色會顯示在角色清單中。

現在，您必須取得角色的 Amazon Resource Name (ARN)，這是該角色的唯一識別碼。當您在來源帳戶中修改開發人員的角色時，可以從目的地帳戶指定角色 ARN，以授予或拒絕許可。

若要取得 UpdateData 的 ARN

1. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)。
2. 在角色清單中，選擇 `UpdateData` 角色。
3. 在詳細資訊窗格的 Summary (摘要) 區段中，複製 Role ARN (角色 ARN) 值。

目的地帳戶的帳戶 ID 為 999999999999，所以角色 ARN 為 `arn:aws:iam::999999999999:role/UpdateData`。請確定您提供目的地帳戶的真實 AWS 帳戶 ID。

此時，您已在目的地和來源帳戶之間建立信任。完成上述作業的方法是在目的地帳戶中建立角色，它可將來源帳戶識別為受信任主體。您也定義切換到 UpdateData 角色的使用者可以執行什麼動作。

接下來，修改開發人員角色的許可。

授予角色存取權

此時，分析師和開發人員都擁有許可，允許他們管理來源帳戶中的資料。使用以下所需的步驟以新增各種許可允許切換到該角色。

若要修改開發人員角色，以允許他們切換到 UpdateData 角色

1. 在來源帳戶中以管理員身分登入，並開啟 IAM 主控台。
2. 選擇角色，然後選擇開發人員。
3. 選擇 Permissions (許可) 標籤，選擇 Add permissions (新增許可)，然後選擇 Attach policy (連接政策)。
4. 請選擇 JSON 標籤。
5. 新增以下政策陳述式，以允許針對目的地帳戶中的 UpdateData 角色進行 AssumeRole 動作。請務必將 Resource元素中的 *DESTINATION-ACCOUNT-ID* 變更為目的地帳戶的實際 AWS 帳戶 ID。

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Action": "sts:AssumeRole",
           "Resource": "arn:aws:iam::111122223333:role/UpdateData"
       }
   }
   ```

   Allow 效果明確地允許開發人員群組存取目的地帳戶中的 UpdateData 角色。任何嘗試存取角色的開發人員都會成功。
6. 選擇 Review policy (檢閱政策)。
7. 輸入名稱，例如 **allow-assume-S3-role-in-destination**。
8. 選擇建立政策。

在大多數環境中，可能不需要以下程序。不過，如果您使用 PowerUserAccess 許可，則一些群組可能已經能夠切換角色。下列程序顯示如何將 "Deny" 許可新增到分析師群組，以確保他們無法擔任角色。如果在您的環境中並不需要此程序，我們建議您不要新增。"Deny" 許可會讓整體許可情況變得更複雜，不易管理和了解。只在沒有更好的選項時使用 "Deny" 許可。

若要修改分析師角色以拒絕擔任 **UpdateData** 角色的許可

1. 選擇角色，然後選擇分析師。
2. 選擇 Permissions (許可) 標籤，選擇 Add permissions (新增許可)，然後選擇 Attach policy (連接政策)。
3. 請選擇 JSON 標籤。
4. 新增以下政策陳述式以拒絕 UpdateData 角色的 AssumeRole 動作。請務必將 Resource元素中的 *DESTINATION-ACCOUNT-ID* 變更為目的地帳戶的實際 AWS 帳戶 ID。

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Deny",
           "Action": "sts:AssumeRole",
           "Resource": "arn:aws:iam::111122223333:role/UpdateData"
       }
   }
   ```

   Deny 效果明確拒絕分析師群組存取目的地帳戶中的 UpdateData 角色。任何嘗試存取角色的分析師都會收到存取遭拒的訊息。
5. 選擇檢閱政策。
6. 輸入名稱，例如 **deny-assume-S3-role-in-destination**。
7. 選擇建立政策。

開發人員角色現在擁有許可，可使用目的地帳戶中的 UpdateData 角色。會防止分析師角色使用 UpdateData 角色。

接下來，您可以了解開發人員 David 如何存取目的地帳戶中的 amzn-s3-demo-bucket-shared-container 儲存貯體。David 可以從 AWS 管理主控台 AWS CLI、 或 AWS API 存取儲存貯體。

透過切換角色測試存取

完成此教學課程的前兩個步驟後，您會有一個角色可授予存取目的地帳戶中的資源。您在來源帳戶中也有一個角色，允許使用者使用該角色。此步驟討論如何測試從 AWS 管理主控台、 AWS CLI和 AWS API 切換到該角色。

若要取得您在使用 IAM 角色時可能遇到的常見問題說明，請參閱 IAM 角色疑難排解。

切換角色 (主控台)

如果 David 需要更新 中目的地帳戶中的資料 AWS 管理主控台，他可以使用切換角色來執行此操作。他會指定帳戶 ID 或別名和角色名稱，而其許可會立即切換到角色所允許的項目。然後他可以利用主控台來使用 `amzn-s3-demo-bucket-shared-container` 儲存貯體，但無法使用目的地中的任何其他資源。當 David 使用角色時，他也無法運用他在來源帳戶中的進階使用者權限。那是因為一次只有一組許可有效。

David 可以使用 IAM 提供的兩種方式來進入 Switch Role (切換角色) 頁面：

- David 從他們的管理員那裡收到指向預先定義的 Switch Role (切換角色) 組態的連結。該連結是在 Create Role (建立角色) 精靈的最終頁面上，或在跨帳戶角色的 Role Summary (角色摘要) 頁面上，提供給管理員。選擇此連結會帶領 David 前往已填寫 Account ID (帳戶 ID) 和 Role name (角色名稱) 欄位的 Switch Role (切換角色) 頁面。David 需要做的只有選擇切換角色。
- 管理員不會在電子郵件中傳送連結，但會改為傳送 Account ID (帳戶 ID) 號碼和 Role Name (角色名稱) 的值。若要切換角色，大衛必須手動輸入該值。下列程序中會以圖表說明：

擔任角色

1. David 在原始帳戶中 AWS 管理主控台 使用他的一般使用者登入 。
2. 他們選擇管理員電郵給他們的連結。這會將 David 導航到 Switch Role (切換角色) 頁面，其中帳戶 ID 或別名以及角色名稱資訊已填寫。

   —或—

   David 在導覽列上選擇他們的名稱 (Identity (身分) 選單)，然後選擇 Switch Roles (切換角色)。

   如果這是 David 第一次以這種方式嘗試存取 Switch Role (切換角色) 頁面，他首先停留在初次執行的 Switch Role (切換角色) 頁面。此頁面提供有關切換角色如何讓使用者跨 AWS 帳戶管理資源的額外資訊。David 必須在此頁面上選擇 Switch Role (切換角色)，以完成此程序的剩餘部分。

3. 接下來，為了存取角色，David 必須手動輸入目的地帳戶 ID 號碼 (999999999999) 和角色名稱 (UpdateData)。

   此外，David 想要監控 IAM 中目前作用中的角色 (以及相關聯的許可)。為了追蹤此資訊，他在 Display Name (顯示名稱) 文字方塊中輸入 Destination、選擇紅色選項，然後選擇 Switch Role (切換角色)。

4. David 現在可以使用 Amazon S3 主控台來使用 UpdateData 角色所具有許可的 Amazon S3 儲存貯體或任何其他資源。

5. 完成後，David 可以回到他們的原始許可。為了執行此作業，他們在導覽列上選擇目的地角色顯示名稱，然後選擇返回到 David @ 111111111111。

6. 下次 David 想要切換角色和在導覽列中選擇身分選單時，他會看到上次的目的地項目仍在那裡。他可以只要選擇該項目即可立即切換角色，而無需重新輸入帳戶 ID 和角色名稱。

切換角色 (AWS CLI)

如果 David 需要在命令列的目的地環境中工作，則可以使用 AWS CLI。他執行 aws sts assume-role 命令，並傳遞角色 ARN 以取得該角色的暫時性安全憑證。然後，他會在環境變數中設定這些登入資料，以便後續 AWS CLI 命令使用角色的許可運作。雖然 David 使用該角色，但他無法在來源帳戶中使用他的進階使用者許可，因為一次只能有一組許可有效。

請注意，所有存取金鑰和權杖僅為範例，不能如下所示般使用。以您實際環境中的適當值取代。

擔任角色

1. David 會開啟命令提示視窗，並透過執行 命令來確認 AWS CLI 用戶端是否正常運作：

```
aws help
```

> **Note**
>
> David 的預設環境使用其預設設定檔的 David 使用者憑證，這是使用 aws configure 命令建立的。如需詳細資訊，請參閱 AWS Command Line Interface 使用者指南中的設定 AWS Command Line Interface。

2. 他透過執行以下命令開始切換角色程序，以切換到目的地帳戶中的 UpdateData 角色。他從建立角色的管理員角色收到角色 ARN。該命令也會要求您提供工作階段名稱，您可以選擇您想要的任何文字。

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

David 然後會在輸出中看到以下：

```
{
    "Credentials": {
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
CvSRyh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDy
EXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
        "Expiration": "2014-12-11T23:08:07Z",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

3. David 在輸出的 Credentials (憑證) 區段看到他們需要的三個部分。

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

   David 需要將 AWS CLI 環境設定為在後續呼叫中使用這些參數。如需設定憑證各種方式的資訊，請參閱設定 AWS Command Line Interface。您不能使用 `aws configure` 命令，因為它不支援擷取工作階段權杖。不過，您可以將資訊手動輸入到組態檔案中。由於這些是暫時性憑證，其過期時間相對較短，最簡單的方式是將它們新增至您目前命令列工作階段的環境。

4. 為新增三個值到環境，David 剪下並貼上前一個步驟的輸出到以下命令中。建議您剪下並貼上到簡單的文字編輯器，來處理工作階段權杖輸出中的換行問題。它必須新增為單一長字串，即使在這裡為清楚起見以換行方式顯示。

   以下範例顯示 Windows 環境中指定命令，其中 "set" 是用來建立環境變數的命令。在 Linux 或 macOS 電腦上，請改為使用命令 "export"。範例的所有其他部分在所有三個環境中均為有效。

如需使用 Windows Powershell 工具的詳細資訊，請參閱切換至 IAM 角色 (Tools for Windows PowerShell)

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

此時，任何下列命令在這些憑證所識別角色的許可下執行。在 David 的案例中，為 UpdateData 角色。

> **Important**
>
> 您可以將常用的組態設定和憑證儲存在 AWS CLI維護的檔案中。如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的使用現有組態與憑證檔案。

5. 執行命令來存取目的地帳戶中的資源。在這個範例中，David 只需要使用下列命令列出他們的 S3 儲存貯體的內容。

```
aws s3 ls s3://shared-container
```

由於 Amazon S3 儲存貯體名稱是全域唯一，所以不需要指定擁有該儲存貯體的帳戶 ID。若要存取其他 AWS 服務的資源，請參閱該服務 AWS CLI 的文件，以取得參考其資源所需的命令和語法。

使用 AssumeRole (AWS API)

當 David 需要從程式碼進行目的地帳戶的更新時，他可進行 AssumeRole 呼叫以擔任 UpdateData 角色。此呼叫會傳回他可用來存取目的地帳戶中 amzn-s3-demo-bucket-shared-container 儲存貯體的臨時憑證。David 可以使用這些憑證，來進行 API 呼叫以更新 amzn-s3-demo-bucket-shared-container 儲存貯體。不過，他無法進行 API 呼叫以存取目的地帳戶中任何其他資源，即使他在來源帳戶中具有進階使用者許可。

擔任角色

1. David 呼叫 AssumeRole 作為應用程式的一部分。他們必須指定 UpdateData ARN：arn:aws:iam::999999999999:role/UpdateData。

   來自 AssumeRole 呼叫的回應包含具有 AccessKeyId 和 SecretAccessKey 的暫時性憑證。也包含 Expiration 時間，指出憑證何時過期以及必須請求新憑證的時間。當您使用 AWS SDK 設定角色鏈結時，許多登入資料提供者會在登入資料過期之前自動重新整理登入資料。
2. David 使用暫時性憑證，進行 s3:PutObject 呼叫以更新 amzn-s3-demo-bucket-shared-container 儲存貯體。他們會將憑證作為 AuthParams 參數傳遞到 API 呼叫。由於臨時角色憑證只有 amzn-s3-demo-bucket-shared-container 儲存貯體的讀取和寫入存取權，因此目的地帳戶中的任何其他動作會被拒絕。

如需程式碼範例 (使用 Python)，請參閱 切換到 IAM 角色 (AWS API)。

其他資源

下列資源可協助您進一步了解本教學課程中的主題：

- 如需有關 IAM 使用者的詳細資訊，請參閱 IAM 身分。
- 如需使用 Amazon S3 儲存貯體的詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的建立儲存貯體。
- 若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？。

摘要

您已完成跨帳戶 API 存取教學課程。您建立了一個角色，以建立與另一個帳戶的信任，並定義了受信任實體可採取的動作。然後，修改角色政策，以控制哪些 IAM 使用者可存取角色。因此，來源帳戶的開發人員可以透過使用臨時憑證，來更新目的地帳戶中的 amzn-s3-demo-bucket-shared-container 儲存貯體。

IAM 教程：建立並連接您的第一個客戶受管政策

在本教學課程中，您會使用 AWS 管理主控台 來建立客戶受管政策，然後將該政策連接至您 中的 IAM 使用者 AWS 帳戶。您建立的政策允許 IAM 測試使用者 AWS 管理主控台 使用唯讀許可直接登入 。

此工作流程有三個基本步驟：

步驟 1：建立政策

根據預設，IAM 使用者沒有執行任何動作的許可。它們無法存取 AWS 管理主控台或管理其中的資料，除非您允許。在此步驟中，您建立客戶受管政策，其允許任何連接的使用者登入主控台。

步驟 2：連接政策

當您將政策連接到使用者時，使用者會繼承與該政策相關聯的所有存取許可。在此步驟中，您會將新的政策連接到測試使用者。

步驟 3：測試使用者存取許可

一旦連接政策，便可以使用者身分登入並測試政策。

先決條件

若要執行此教學課程中的步驟，您需具備以下內容：

- 您可以具有管理許可的 IAM 使用者身分 AWS 帳戶 登入 。
- 未擁有指派許可或群組成員資格的測試 IAM 使用者，如下所示：

使用者名稱	群組	許可
PolicyUser	<無>	<無>

步驟 1：建立政策

在此步驟中，您會建立客戶受管政策，允許任何連接的使用者以唯讀存取 IAM 資料 AWS 管理主控台的方式登入 。

為您的測試使用者建立政策

1. 以具有管理員許可的使用者身分登入 IAM 主控台 (https://console.aws.amazon.com/iam/)。
2. 在導覽窗格上選擇 Policies (政策)。
3. 在內容窗格中，選擇 Create policy (建立政策)。
4. 選擇 JSON 選項，並從下列 JSON 政策文件中複製文字。將此文字貼上至 JSON 文字方框中。

JSON

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

5. 解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。

> **Note**
>
> 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行更改或在 Visual editor (視覺編輯工具) 索引標籤中選擇 Review policy (檢閱政策)，IAM 可能會調整您的政策結構以針對視覺編輯工具進行最佳化。如需詳細資訊，請參閱政策結構調整。

6. 在檢視與建立頁面上，針對政策名稱輸入 **UsersReadOnlyAccessToIAMConsole**。檢視政策授與的許可，然後選擇建立政策來儲存您的工作。

   新的政策會出現在受管政策清單中，並且已準備好連接。

步驟 2：連接政策

接下來，將您剛建立的政策連接到測試 IAM 使用者。

連接政策到您的測試使用者

1. 在 IAM 主控台的導覽窗格中，選擇 Policies (政策)。
2. 在政策清單頂端的搜尋方塊中，開始輸入 **UsersReadOnlyAccessToIAMConsole** 直到您可以看到您的政策。然後選擇清單中 UsersReadOnlyAccessToIAMConsole 旁的選項按鈕。
3. 選擇 Actions (動作) 按鈕，然後選擇 Attach (連接)。

4. 在 IAM 實體中，選擇篩選使用者的選項。
5. 在搜尋方塊中，開始輸入 **PolicyUser** 直到該使用者顯示在清單上。然後勾選清單中該使用者旁的方塊。
6. 選擇連接政策。

您可以連接政策到 IAM 測試使用者，這表示使用者現在擁有唯讀存取 IAM 主控台的許可。

步驟 3：測試使用者存取許可

對於本教學，建議您以測試使用者身分登入來測試存取許可，如此才能了解使用者可能經歷的情況。

以測試使用者登入來測試存取許可

1. 以 PolicyUser 測試使用者身分登入位於 https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 瀏覽主控台頁面並嘗試建立新的使用者或群組。請注意，PolicyUser 可以顯示資料，但無法建立或修改現有 IAM 資料。

相關資源

如需相關資訊，請參閱下列資源：

- 受管政策與內嵌政策
- 控制 IAM 使用者對 的存取 AWS 管理主控台

摘要

現在您已成功完成所有建立和連接客戶受管政策的必要步驟。因此，您可以利用您的測試帳戶登入 IAM 主控台，以查看使用者的體驗。

IAM 教學課程：定義根據標籤存取 AWS 資源的許可

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在 中 AWS，這些屬性稱為標籤。您可以將標籤連接至 IAM 資源，包括 IAM 實體（使用者或角色）和 AWS 資源。您可以定義使用標籤條件金鑰的政策，以根據主體的標籤來授與許可。當您使用標籤來控制對 AWS 資源的存取時，您可以允許您的團隊和資源以較少的政策變更 AWS 來成長。ABAC 政策比傳統 AWS 政策更靈活，需要您

列出每個個別資源。如需 ABAC 及其優於傳統政策的詳細資訊，請參閱 使用 ABAC 授權根據屬性定義許可。

> **Note**
>
> 您必須傳遞每個工作階段標籤的單一值。 AWS Security Token Service 不支援多值工作階段標籤。

主題

- 教學課程概觀
- 先決條件
- 步驟 1：建立測試使用者
- 步驟 2：建立 ABAC 政策
- 步驟 3：建立角色
- 步驟 4：測試建立秘密
- 步驟 5：測試檢視秘密
- 步驟 6：測試可擴展性
- 步驟 7：測試更新和刪除秘密
- 摘要
- 相關資源
- IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記

教學課程概觀

本教學說明如何建立和測試政策，以允許具有主體標籤的 IAM 角色存取具有相符標籤的資源。當主體向 AWS發出請求時，會根據主體和資源標籤是否相符來授與其許可。此策略允許個人僅檢視或編輯其任務所需的 AWS 資源。

案例

假設您是 Example Corporation 這間大型公司的首席開發人員，也是經驗豐富的 IAM 管理員。您很熟悉如何建立及管理 IAM 使用者、角色和政策。您想要確保您的開發工程師和品質保證團隊成員可以存取所需資源。您也需要可隨著公司成長而擴展的策略。

您可以選擇使用 AWS 資源標籤和 IAM 角色主體標籤，為支援它的服務實作 ABAC 策略，以 開頭 AWS Secrets Manager。若要了解哪些服務支援以標籤為基礎的授權，請參閱 AWS 使用 IAM 的 服務。若要了解您可以在政策中搭配每個服務的動作和資源使用哪些標記條件金鑰，請參閱 AWS 服務的動作、資源和條件金鑰。您可以設定您的 SAML 類型或 Web 身分提供者，將工作階段標籤傳遞至 AWS。當您的員工聯合到 時 AWS，其屬性會套用至其產生的委託人 AWS。您接著可以使用 ABAC 來根據這些屬性允許或拒絕許可。若要了解搭配 SAML 聯合身分使用工作階段標籤與本教學的不同，請參閱 IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記。

您的工程和品質保證團隊成員負責 Pegasus 或 Unicorn 專案。您可以選擇下列 3 個字元的專案和團隊標籤值：

- `access-project` = `peg` 適用於 Pegasus 專案
- `access-project` = `uni` 適用於 Unicorn 專案
- `access-team` = `eng` 適用於工程團隊
- `access-team` = `qas` 適用於品質保證團隊

此外，您選擇要求`cost-center`成本分配標籤來啟用自訂 AWS 帳單報告。如需詳細資訊，請參閱 AWS 帳單與成本管理 使用者指南中的使用成本分配標籤。

重要決策摘要

- 員工使用 IAM 使用者憑證登入，然後擔任其團隊和專案的 IAM 角色。如果您的公司有專屬身分系統，您可以設定聯合以允許員工在沒有 IAM 使用者的情況下擔任角色。如需更多詳細資訊，請參閱 IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記。
- 相同的政策會連接至所有角色。動作會根據標籤而受允許或拒絕。
- 員工可以建立新的資源，但僅限於將相同的標籤連接至套用到其角色的資源時。這可確保員工在建立資源之後能夠檢視資源。管理員不再需要使用新資源的 ARN 來更新政策。
- 員工可以讀取其團隊所擁有的資源，不論專案為何。
- 員工可以更新及刪除自己團隊和專案所擁有的資源。
- IAM 管理員可以新增新專案的角色。他們可以建立並標記新的 IAM 使用者，以允許存取適當的角色。管理員不需要編輯政策來支援新的專案或團隊成員。

在此教學中，您將為每個資源加上標籤、為您的專案角色加上標籤並將政策新增至角色，以允許前述的行為。產生的政策允許角色 `Create`、`Read`、`Update` 和 `Delete` 存取以相同專案和團隊標籤所標記的資源。政策也允許跨專案 `Read` 存取以相同團隊標記的資源。

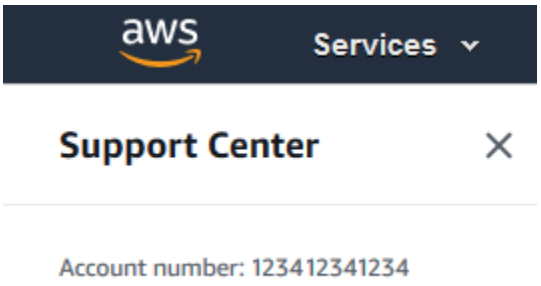
圖表顯示兩個專案，其中角色僅限於在其專案外部唯讀存取，同時擁有在其自己的專案中建立、讀取、更新和刪除資源的許可。

先決條件

若要執行此教學課程中的步驟，您必須具備以下內容：

- 您可以具有管理許可的使用者身分 AWS 帳戶 登入 。
- 您的 12 位數帳戶 ID，用於在步驟 3 中建立角色。

  若要使用 尋找 AWS 您的帳戶 ID 號碼 AWS 管理主控台，請選擇右上角導覽列上的支援，然後選擇支援中心。帳戶號碼 (ID) 會顯示在左側導覽窗格。



- 體驗在 AWS 管理主控台中建立並編輯 IAM 使用者、角色和政策。不過，如果您需要記住 IAM 管理程序的協助，此教學提供的連結可讓您檢視逐步說明。

步驟 1：建立測試使用者

針對測試，請建立四個 IAM 使用者，並授與其許可擔任具有相同標籤的角色。這可讓您更輕鬆地新增更多使用者到您的團隊。當您標記使用者時，使用者會自動取得擔任正確角色的存取權。如果使用者只在一個專案和團隊中工作，則您不必將使用者新增至角色的信任政策。

1. 建立下列名為 `access-assume-role` 的客戶受管政策。如需如何建立 JSON 政策的詳細資訊，請參閱 建立 IAM 政策。

   ABAC 政策：擔任任何 ABAC 角色，但僅限使用者和角色標籤相符時

   下列政策允許使用者擔任您帳戶中名稱字首為 `access-` 的任何角色。角色也必須以與使用者相同的專案、團隊和成本中心標籤來標記。

   若要使用此政策，請將斜體預留位置文字取代為您的帳戶資訊。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::111122223333:role/access-*",
            "Condition": {
                "StringEquals": {
                    "iam:ResourceTag/access-project": "${aws:PrincipalTag/
access-project}",
                    "iam:ResourceTag/access-team": "${aws:PrincipalTag/
access-team}",
                    "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
center}"
                }
            }
        }
    ]
}
```

若要將本教學擴展為大量使用者，您可以將政策連接至群組，並將每個使用者新增至群組。如需詳細資訊，請參閱建立 IAM 群組及編輯 IAM 群組中的使用者。

2. 建立下列 IAM 使用者、連接 `access-assume-role` 許可政策。務必選取為使用者提供 AWS 管理主控台的存取權限，然後新增下列標籤。

使用者名稱	使用者標籤索引鍵	使用者標籤值
access-Arnav-peg-eng	`access-project`	`peg`
	`access-team`	`eng`
	`cost-center`	`987654`
access-Mary-peg-qas	`access-project`	`peg`

使用者名稱	使用者標籤索引鍵	使用者標籤值
	access-team	qas
	cost-center	987654
access-Saanvi-uni-eng	access-project	uni
	access-team	eng
	cost-center	123456
access-Carlos-uni-qas	access-project	uni
	access-team	qas
	cost-center	123456

步驟 2：建立 ABAC 政策

建立下列名為 **access-same-project-team** 的政策。您會在後續步驟將此政策新增至角色。如需如何建立 JSON 政策的詳細資訊，請參閱 建立 IAM 政策。

如需您可以針對本教學進行調整的其他政策，請參閱下列頁面：

- 控制 IAM 主體的存取權限
- Amazon EC2：允許以程式設計方式和在主控台中開始或停止使用者已標記的 EC2 執行個體
- EC2：依據比對主體和資源的標籤啟動或停止執行個體
- EC2：依據標籤啟動或停止執行個體
- IAM：擔任具有特定標籤的角色

ABAC 政策：僅在主體與資源標籤相符時，才存取 Secrets Manager 資源

下列政策允許主體建立、讀取、編輯及刪除資源，但僅限資源受標記為與主體相同的鍵值對時。當主體建立資源時，必須新增 access-project、access-team、和 cost-center 標籤，其值符合主體的標籤。此政策也允許新增選用的 Name 或 OwnedBy 標籤。

JSON

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

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

此政策的功能為何？

- AllActionsSecretsManagerSameProjectSameTeam 陳述式允許在所有相關資源上執行此服務的所有動作，但僅限於資源標籤符合主體標籤時。透過將 "Action": "secretsmanager:*" 新增至政策，政策會隨著 Secrets Manager 成長而成長。如果 Secrets Manager 新增新的 API 操

作，您不需要將該動作新增至陳述式。陳述式使用三個條件區塊來實作 ABAC。只有在三個區塊全部傳回 true 時，才會允許此請求。

- 如果資源上有指定的標籤鍵，而且其值符合主體的標籤，則此陳述式的第一個條件區塊會傳回 true。此區塊會為不相符的標籤或不支援資源標記的動作傳回 false。若要了解此區塊不允許哪些動作，請參閱 動作、資源和條件金鑰 AWS Secrets Manager。該頁面顯示對 Secret (秘密) 資源類型執行的動作支援 `secretsmanager:ResourceTag/tag-key` 條件索引鍵。部分 Secrets Manager 動作不支援該資源類型，包括 `GetRandomPassword` 和 `ListSecrets`。您必須建立額外的陳述式以允許這些動作。
- 如果請求中傳遞的每個標籤鍵都包含在指定清單中，則第二個條件區塊會傳回 true。此動作是使用 `ForAllValues` 搭配 `StringEquals` 條件運算子來完成。如果未傳遞金鑰或金鑰集的子集，則條件會傳回 true。這允許了不允許在請求中傳遞標籤的 `Get*` 操作。如果申請者包含了不在清單中的標籤鍵，則條件會傳回 false。在請求中傳遞的每個標籤鍵，都必須符合此清單的成員。如需詳細資訊，請參閱用於多值內容索引鍵的集運算子。
- 如果請求支援傳遞標籤 (如果這三個標籤都存在)，且符合主體標籤值，則第三個條件區塊會傳回 true。如果請求不支援傳遞標籤，此區塊也會傳回 true。這要歸功於條件運算子中的 `...IfExists`。如果在支援該標籤的動作期間沒有傳遞任何標籤，或標籤鍵和值不相符，則區塊會傳回 false。

- `AllResourcesSecretsManagerNoTags` 陳述式允許第一個陳述式不允許的 `GetRandomPassword` 和 `ListSecrets` 動作。
- 如果主體所標記的存取團隊標籤與資源相同，則 `ReadSecretsManagerSameTeam` 陳述式允許唯讀操作。無論專案或成本中心標籤為何，都允許此操作。
- `DenyUntagSecretsManagerReservedTags` 陳述式拒絕從 Secrets Manager 移除鍵開頭為 "access-" 之標籤的請求。這些標籤用於控制對資源的存取，因此移除標籤可能會移除許可。
- `DenyPermissionsManagement` 陳述式拒絕建立、編輯或刪除 Secrets Manager 以資源為基礎的政策的存取。這些政策可用來變更秘密的許可。

> **Important**
>
> 此政策使用策略來允許服務的所有動作，但會明確拒絕更改許可的動作。拒絕動作會覆寫任何其他允許主體執行該動作的政策。這可能會產生意外的結果。根據最佳實務，只有在該動作無論何種情況都不應允許的情況下，才使用明確拒絕。否則，請允許包含個別動作的清單，且將不需要的動作預設拒絕。

步驟 3：建立角色

建立下列 IAM 角色，並連接您在先前步驟中建立的 **access-same-project-team** 政策。如需建立 IAM 角色的詳細資訊，請參閱 建立角色以將許可授予 IAM 使用者。如果您選擇使用聯合，而非 IAM 使用者和角色，請參閱 IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記。

任務職能	角色名稱	角色標籤	角色描述
Pegasus 工程專案	access-peg-engineering	access-project = peg access-team = eng cost-center = 987654	允許工程師讀取所有工程資源，並建立和管理 Pegasus 工程資源。
Pegasus 品質保證專案	access-peg-quality-assurance	access-project = peg access-team = qas cost-center = 987654	允許 QA 團隊讀取所有 QA 資源，並建立和管理所有 Pegasus QA 資源。
Unicorn 工程專案	access-uni-engineering	access-project= uni access-team = eng cost-center = 123456	允許工程師讀取所有工程資源，並建立和管理 Unicorn 工程資源。
Unicorn Quality Assurance 專案	access-uni-quality-assurance	access-project = uni access-team = qas cost-center = 123456	允許 QA 團隊讀取所有 QA 資源，並建立和管理所有 Unicorn QA 資源。

步驟 4：測試建立秘密

連接至角色的許可政策允許員工建立秘密。只有在秘密以其專案、團隊和成本中心來標記時，才允許此操作。以使用者身分登入來確認您的許可如預期般運作且擔任正確角色，並測試 Secrets Manager 中的活動。

測試使用和不使用必要的標籤來建立秘密

1. 在您的主要瀏覽器視窗中，保持以管理員使用者的身分登入，讓您可以檢閱 IAM 中的使用者、角色和政策。使用瀏覽器無痕式視窗或個別的瀏覽器進行測試。前往以下位置以 `access-Arnav-peg-eng` IAM 使用者身分登入 Secrets Manager 主控台：https://console.aws.amazon.com/secretsmanager/。
2. 嘗試切換到 `access-uni-engineering` 角色。

   此操作會失敗，因為 `access-project` 和 `cost-center` 標籤值與 `access-Arnav-peg-eng` 使用者和 `access-uni-engineering` 角色不相符。

   如需在 中切換角色的詳細資訊 AWS 管理主控台，請參閱 從使用者切換至 IAM 角色 (主控台)
3. 切換到 `access-peg-engineering` 角色。
4. 使用下列資訊存放新的秘密。若要了解如何存放秘密，請參閱 AWS Secrets Manager 使用者指南中的建立基本秘密。

   > **Important**
   >
   > Secrets Manager 會顯示警示，指出您沒有與 Secrets Manager 搭配使用之其他 AWS 服務的許可。例如，若要建立 Amazon RDS 資料庫的憑證，您必須擁有描述 RDS 執行個體、RDS 叢集和 Amazon Redshift 叢集的許可。您可以忽略這些提醒，因為您未在本教學課程中使用這些特定 AWS 服務。

   1. 在 Select secret type (選取秘密類型) 區段中，選擇 Other type of secrets (其他類型的秘密)。在兩個文字方塊中，輸入 `test-access-key` 和 `test-access-secret`。
   2. 針對 Secret name (秘密名稱) 欄位輸入 `test-access-peg-eng`。
   3. 從下表新增不同的標籤組合，並檢視預期的行為。
   4. 選擇 Store (存放) 以嘗試建立秘密。在儲存失敗時，返回先前的 Secrets Manager 主控台頁面，並使用下表中的下一個標籤集。最後一個標籤集會受到允許，且將成功建立秘密。

   下表顯示 `test-access-peg-eng` 角色的 ABAC 標籤組合。

access-project 標籤值	**access-team** 標籤值	**cost-center** 標籤值	其他標籤	預期行為
(無)	(無)	(無)	(無)	已拒絕，因為 access-project 標籤值不符合角色的值 peg。
uni	eng	987654	(無)	已拒絕，因為 access-project 標籤值不符合角色的值 peg。
peg	qas	987654	(無)	已拒絕，因為 access-team 標籤值不符合角色的值 eng。
peg	eng	123456	(無)	已拒絕，因為 cost-center 標籤值不符合角色的值 987654。
peg	eng	987654	Owner = Jane	已拒絕，因為政策不允許額外的標籤 owner，即使所有三個必要標籤都存在且其值符合角色的值。
peg	eng	987654	Name = Jane	已允許，因為三個必要標籤都存在，且其值符合角色的值。您也可以包含選用的 Name 標籤。

5. 登出並針對每個下列角色和標籤值來重複此程序的前三個步驟。在此程序的第四個步驟中，測試任何遺漏的標籤、選用標籤、不允許的標籤，和您選擇的無效標籤值。然後，使用必要標籤來建立具有下列標籤和名稱的秘密。

使用者名稱	角色名稱	秘密名稱	秘密標籤
access-Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = peg access-team = qas cost-center = 987654

使用者名稱	角色名稱	秘密名稱	秘密標籤
access-Saanvi-uni-eng	access-uni-engineering	test-access-uni-eng	access-project = uni access-team = eng cost-center = 123456
access-Carlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	access-project = uni access-team = qas cost-center = 123456

步驟 5：測試檢視秘密

您連接到每個角色的政策允許員工檢視以其團隊名稱標記的任何秘密，不論他們的專案為何。在 Secrets Manager 中測試您的角色，確認您的許可如預期般運作。

測試檢視包含和不含必要標籤的秘密

1. 使用下列其中一個 IAM 使用者身分登入:
   - access-Arnav-peg-eng
   - access-Mary-peg-qas
   - access-Saanvi-uni-eng
   - access-Carlos-uni-qas
2. 切換至相符的角色：
   - access-peg-engineering
   - access-peg-quality-assurance
   - access-uni-engineering
   - access-uni-quality-assurance

   如需在 中切換角色的詳細資訊 AWS 管理主控台，請參閱 從使用者切換至 IAM 角色 (主控台)。

3. 在左側的導覽窗格中，選擇選單圖示以展開選單，然後選擇 Secrets (秘密)。
4. 無論您目前的角色為何，您應該都會看到資料表中的全部四個秘密。這是預期的結果，因為名為 `access-same-project-team` 的政策允許對所有資源執行 `secretsmanager:ListSecrets` 動作。
5. 選擇其中一個秘密的名稱。
6. 在秘密的詳細資訊頁面上，角色的標籤將決定您是否可以檢視頁面內容。將您的角色名稱與秘密名稱進行比較。如果這兩個名稱具有相同團隊名稱，則 `access-team` 標籤會相符。如果不相符，則會拒絕存取。

   下表顯示每個角色的 ABAC 秘密檢視行為。

角色名稱	秘密名稱	預期行為
`access-peg-engineering`	`test-access-peg-eng`	允許
	test-access-peg-qas	已拒絕
	test-access-uni-eng	允許
	test-access-uni-qas	已拒絕
access-peg-quality-assurance	test-access-peg-eng	已拒絕
	test-access-peg-qas	允許
	test-access-uni-eng	已拒絕
	test-access-uni-qas	允許
access-uni-engineering	test-access-peg-eng	允許
	test-access-peg-qas	已拒絕
	test-access-uni-eng	允許
	test-access-uni-qas	已拒絕
access-uni-quality-assurance	test-access-peg-eng	已拒絕
	test-access-peg-qas	允許

角色名稱	秘密名稱	預期行為
	test-access-uni-eng	已拒絕
	test-access-uni-qas	允許

7. 從頁面頂端的頁面導覽路徑中，選擇 Secrets (秘密) 以返回秘密清單。使用不同角色重複此程序中的步驟，以測試您是否可以檢視每個秘密。

步驟 6：測試可擴展性

偏好以屬性為基礎的存取控制 (ABAC) 多過以角色為基礎的存取控制 (RBAC) 的重要原因是可擴展性。當您的公司新增專案、團隊或人員時 AWS，您不需要更新 ABAC 驅動的政策。例如，假設 Example Company 為名為 Centaur 的新專案提供資金。一位名為 Saanvi Sarkar 的工程師將擔任 Centaur 的主要工程師，同時繼續進行 Unicorn 專案。Sanvi 還將檢閱 Peg 專案的工作。另外還有幾位新聘的工程師，包括 Nikhil Jayashankar，只負責 Centaur 專案。

將新專案新增至 AWS

1. 以 IAM 管理原使用者身分登入，並開啟位於 https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Roles (角色)，然後新增名為 `access-cen-engineering` 的 IAM 角色 將 **`access-same-project-team`** 許可政策連接至角色，並新增下列角色標籤：
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
3. 在左側導覽窗格中，選擇 Users (使用者)。
4. 新增名為 `access-Nikhil-cen-eng` 的新使用者、連接名為 `access-assume-role` 的政策，並新增以下使用者標籤。
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
5. 使用 步驟 4：測試建立秘密 和 步驟 5：測試檢視秘密 中的程序。在另一個瀏覽器視窗中測試 Nikhil 是否只能建立 Centaur 工程秘密，並且可以檢視所有工程秘密。

6. 在您以管理員身分登入的主要瀏覽器視窗中，選擇使用者 `access-Saanvi-uni-eng`。
7. 在 Permissions (許可) 標籤上，移除 access-assume-role 許可政策。
8. 新增下列名為 `access-assume-specific-roles` 的內嵌政策 。如需將內嵌政策新增至使用者的詳細資訊，請參閱 為使用者或角色嵌入內嵌政策 (主控台)。

   ABAC 政策：僅擔任特定角色

   此政策允許 Saanvi 擔任 Pegasus 和 Centaur 專案的工程角色。您必須建立此自訂政策，因為 IAM 不支援多值標籤。您無法以 `access-project = peg` 和 `access-project = cen` 來標記 Saanvi 的使用者。此外， AWS 授權模型不能同時符合這兩個值。如需詳細資訊，請參閱在 IAM 和 中標記的規則 AWS STS。您必須改為手動指定她可以擔任的兩個角色。

   若要使用此政策，請將斜體預留位置文字取代為您的帳戶資訊。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::111122223333:role/access-peg-engineering",
                "arn:aws:iam::111122223333:role/access-cen-engineering"
            ]
        }
    ]
}
```

9. 使用 步驟 4：測試建立秘密 和 步驟 5：測試檢視秘密 中的程序。在另一個瀏覽器視窗中，確認 Saanvi 可以擔任這兩個角色。根據角色的標籤，檢查她是否只能為她的專案、團隊和成本中心建立秘密。同時確認她可以檢視工程團隊所擁有之任何秘密的詳細資訊，包括她剛建立的秘密。

步驟 7：測試更新和刪除秘密

連接至角色的 `access-same-project-team` 政策允許員工更新和刪除任何以其專案、團隊和成本中心加上標籤的秘密。在 Secrets Manager 中測試您的角色，確認您的許可如預期般運作。

測試更新和刪除含有與不含必要標籤的秘密

1. 使用下列其中一個 IAM 使用者身分登入:
   - access-Arnav-peg-eng
   - access-Mary-peg-qas
   - access-Saanvi-uni-eng
   - access-Carlos-uni-qas
   - access-Nikhil-cen-eng
2. 切換至相符的角色：
   - access-peg-engineering
   - access-peg-quality-assurance
   - access-uni-engineering
   - access-peg-quality-assurance
   - access-cen-engineering

   如需在 中切換角色的詳細資訊 AWS 管理主控台，請參閱 從使用者切換至 IAM 角色 (主控台)。
3. 針對每個角色嘗試更新秘密描述，然後嘗試刪除下列秘密。如需詳細資訊，請參閱AWS Secrets Manager 使用者指南中的修改秘密以及刪除與還原秘密。

   下表顯示每個角色的 ABAC 秘密更新和刪除行為。

角色名稱	秘密名稱	預期行為
access-peg-engineering	test-access-peg-eng	允許
	test-access-uni-eng	已拒絕
	test-access-uni-qas	已拒絕
access-peg-quality-assurance	test-access-peg-qas	允許
	test-access-uni-eng	已拒絕
access-uni-engineering	test-access-uni-eng	允許

角色名稱	秘密名稱	預期行為
	test-access-uni-qas	已拒絕
access-peg-quality-assurance	test-access-uni-qas	允許

摘要

您現在已成功完成使用標籤進行以屬性為基礎的存取控制 (ABAC) 所需的所有步驟。您已學會如何定義標記策略。您已將該策略套用到主體和資源。您已建立並套用了實施 Secrets Manager 策略的政策。您也已了解當您新增專案和團隊成員時，ABAC 可以輕鬆擴展。因此，您可以使用測試角色登入 IAM 主控台，並體驗如何在 AWS中使用 ABAC 標籤。

> **Note**
>
> 您已新增僅在特定條件下允許執行動作的政策。如果您將不同的政策套用至具有更廣泛許可的使用者或角色，則動作可能不會受限為需要標記。例如，如果您使用 `AdministratorAccess` AWS 受管政策為使用者提供完整的管理許可，則這些政策不會限制該存取。更多涉及多個政策時如何決定許可的詳細資訊，請參閱 AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求。

相關資源

如需相關資訊，請參閱下列資源：

- 使用 ABAC 授權根據屬性定義許可
- AWS 全域條件內容索引鍵
- 建立角色以將許可授予 IAM 使用者
- AWS Identity and Access Management 資源的標籤
- 使用標籤控制對 AWS 資源的存取
- 從使用者切換至 IAM 角色 (主控台)
- IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記

若要了解如何監控帳戶中的標籤，請參閱使用無伺服器工作流程和 Amazon CloudWatch Events 監控 AWS 資源上的標籤變更。

IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記

屬性為基礎的存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在 中 AWS，這些屬性稱為標籤。您可以將標籤連接至 IAM 資源，包括 IAM 實體（使用者或角色）和 AWS 資源。當實體用於向 提出請求時 AWS，它們會成為委託人，而這些委託人包含標籤。

您也可以在擔任角色或與使用者聯合身分時傳遞工作階段標籤。然後您可以定義使用標籤條件索引鍵的政策，以根據主體的標籤來授與許可。當您使用標籤來控制對 AWS 資源的存取 ，您就是允許團隊和資源能夠用更少的 AWS 政策變更來增長。ABAC 政策比傳統 AWS 政策更靈活，需要您列出每個個別資源。如需 ABAC 及其優於傳統政策的詳細資訊，請參閱 使用 ABAC 授權根據屬性定義許可。

如果您的公司使用 SAML 類型的身分提供者 (IdP) 來管理企業使用者身分，您可以將 SAML 屬性用於AWS中精細的存取控制。屬性可包含成本中心識別符、使用者電子郵件地址、部門分類和專案指派。當您將這些屬性做為工作階段標籤傳遞時，您接著便可以根據這些工作階段標籤來控制對 AWS 的存取。

如要將 SAML 屬性傳遞到您的工作階段主體來完成 ABAC 教學，請使用本主題中包含的變更，完成 IAM 教學課程：定義根據標籤存取 AWS 資源的許可 中的任務。

先決條件

如要執行步驟以使用 ABAC 的 SAML 工作階段標籤，您必須已具備以下項目：

- 對 SAML 類型 IdP 的存取，可讓您使用特定屬性建立測試使用者。
- 以具有管理許可的使用者身分登入的能力。
- 體驗在 AWS 管理主控台中建立並編輯 IAM 使用者、角色和政策。不過，如果您需要記住 IAM 管理程序的協助，ABAC 教學課程提供的連結可讓您檢視逐步說明。
- 在 IAM 中體驗設定 SAML 類型的 IdP。如要檢視更多詳細資訊及更詳細 IAM 文件的連結，請參閱 使用 AssumeRoleWithSAML 傳遞工作階段標籤。

步驟 1：建立測試使用者

請跳過 步驟 1：建立測試使用者 中的說明。由於您的身分是定義在您的供應商中，您不需要為您的員工新增 IAM 使用者。

步驟 2：建立 ABAC 政策

遵循 步驟 2：建立 ABAC 政策 中的說明，在 IAM 中建立指定受管政策。

步驟 3：建立和設定 SAML 角色

當您使用 SAML 的 ABAC 教學課程時，您必須執行其他步驟來建立角色、設定 SAML IdP，以及啟用 AWS 管理主控台 存取。如需詳細資訊，請參閱步驟 3：建立角色。

步驟 3A：建立 SAML 角色

建立單一角色，該角色信任您的 SAML 身分提供者，以及您在步驟 1 中建立的 `test-session-tags` 使用者。ABAC 教學會使用具備不同角色標籤的不同角色。因為您是從您的 SAML IdP 傳遞工作階段標籤，您只需要一個角色。若要了解如何建立 SAML 類型的角色，請參閱 為 SAML 2.0 聯合身分建立角色 (主控台)。

將角色命名為 `access-session-tags`。將 `access-same-project-team` 許可政策連接到角色。編輯角色信任政策，以使用以下政策。如需如何編輯角色信任關係的詳細說明，請參閱 更新角色信任政策。

以下角色信任政策會允許您的 SAML 身分提供者和 `test-session-tags` 使用者擔任角色。當 SAML 身分提供者和該使用者擔任角色時，必須傳遞三個指定工作階段標籤。`sts:TagSession` 動作是允許傳遞工作階段標籤的必要項目。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSamlIdentityAssumeRole",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:TagSession"
            ],
            "Principal": {"Federated":"arn:aws:iam::123456789012:saml-
provider/ExampleCorpProvider"},
            "Condition": {
                "StringLike": {
```

```
                    "aws:RequestTag/cost-center": "*",
                    "aws:RequestTag/access-project": "*",
                    "aws:RequestTag/access-team": [
                        "eng",
                        "qas"
                    ]
                },
                "StringEquals": {"SAML:aud": "https://region-
code.signin.aws.amazon.com/saml"}
            }
        }
    ]
}
```

`AllowSamlIdentityAssumeRole` 陳述式允許工程和品質保證團隊的成員在 AWS 從 Example Corporation IdP 聯合到 時擔任此角色。`ExampleCorpProvider` SAML 供應商是在 IAM 中定義。管理員已設定 SAML 聲明，傳遞三個必要的工作階段標籤。聲明可以傳遞其他標籤，但必須要有這三個標籤。身分的屬性可針對 `cost-center` 和 `access-project` 標籤具備任何值。但是，`access-team` 屬性值必須與 `eng` 或 `qas` 相符，才能指出身分是位於 Engineering 或 Quality Assurance 團隊。

步驟 3B：設定 SAML IdP

設定您的 SAML IdP，將 `cost-center`、`access-project` 和 `access-team` 屬性做為工作階段標籤傳遞。如需更多詳細資訊，請參閱 使用 AssumeRoleWithSAML 傳遞工作階段標籤。

若要將這些屬性做為工作階段標籤傳遞，請在您的 SAML 聲明中包含下列元素。

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

步驟 3C：啟用主控台存取

為您的聯合 SAML 使用者啟用主控台存取。如需更多詳細資訊，請參閱 啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。

步驟 4：測試建立秘密

AWS 管理主控台 使用 `access-session-tags`角色聯合到 。如需詳細資訊，請參閱啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。然後遵循 步驟 4：測試建立秘密 中的說明來建立秘密。搭配屬性使用不同的 SAML 身分，來匹配 ABAC 教學中指出的標籤。如需更多詳細資訊，請參閱 步驟 4：測試建立秘密。

步驟 5：測試檢視秘密

遵循 步驟 5：測試檢視秘密 中的說明來檢視您在上一個步驟中建立的秘密。搭配屬性使用不同的 SAML 身分，來匹配 ABAC 教學中指出的標籤。

步驟 6：測試可擴展性

遵循 步驟 6：測試可擴展性 中的說明來測試可擴展性。請使用下列屬性在您的 SAML 類型 IdP 中新增新的身分，來執行此作業：

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

步驟 7：測試更新和刪除秘密

遵循 步驟 7：測試更新和刪除秘密 中的說明來更新和刪除秘密。搭配屬性使用不同的 SAML 身分，來匹配 ABAC 教學中指出的標籤。

> **Important**
>
> 刪除您建立的所有秘密來避免產生費用。如需關於 Secrets Manager 定價的詳細資訊，請參閱 AWS Secrets Manager 定價。

摘要

您現在已成功完成使用 SAML 工作階段標籤和資源標籤進行許可管理必要的所有步驟。

> **Note**
>
> 您已新增僅在特定條件下允許執行動作的政策。如果您將不同的政策套用至具有更廣泛許可的使用者或角色，則動作可能不會受限為需要標記。例如，如果您使用

`AdministratorAccess` AWS 受管政策為使用者提供完整的管理許可，則這些政策不會限制該存取。更多涉及多個政策時如何決定許可的詳細資訊，請參閱 AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求。

IAM 教學課程：允許使用者管理其憑證和 MFA 設定

您可以允許使用者在安全憑證頁面上自行管理其多重要素驗證 (MFA) 裝置和憑證。您可以使用 AWS 管理主控台 來設定登入資料 （ 存取金鑰、密碼、簽署憑證和 SSH 公有金鑰 ）、刪除或停用不需要的登入資料，並為使用者啟用 MFA 裝置。當使用者人數較少時，它十分好用。但隨著使用者人數增加，這個任務很快會變得非常耗時。本教程是為您介紹如何實現這些最佳實務，而不給您的管理員帶來負擔。

本教學課程示範如何允許使用者存取 AWS 服務，但僅限於使用者使用 MFA 登入時。如果未使用 MFA 裝置登入，則使用者無法存取其他服務。

此工作流程有三個基本步驟。

步驟 1：建立政策以實施 MFA 登入

建立一個客戶受管政策，其禁止除少數 IAM 動作之外的所有動作。這些例外狀況允許使用者變更自己的憑證，並在安全憑證頁面上管理 MFA 裝置。如需存取該頁面的詳細資訊，請參閱 IAM 使用者如何變更他們自己的密碼 (主控台)。

步驟 2：將政策連接到您的測試使用者群組

建立一個使用者群組，其成員在使用 MFA 登入時擁有對所有 Amazon EC2 動作的完全存取權。若要建立這類使用者群組，請同時連接名為 的 AWS 受管政策，`AmazonEC2FullAccess`以及您在第一個步驟中建立的客戶受管政策。

步驟 3：測試您的使用者存取權限

以測試使用者身分登入，驗證在使用者建立 MFA 裝置之前對 Amazon EC2 的存取被封鎖。之後使用者可以使用該裝置登入。

先決條件

若要執行此教學課程中的步驟，您必須具備以下內容：

- 您可以具有管理許可的 IAM 使用者身分 AWS 帳戶 登入 。

- 您在步驟 1 中輸入政策的帳戶 ID 號碼。

  若要尋找您的帳戶 ID 編號，請在頁面頂部的導覽列上，選擇 Support (支援)，然後選擇 Support Center (支援中心)。您可以在此頁面的 Support (支援) 選單下尋找您的帳戶 ID。
- 虛擬 (軟體式) MFA 裝置、FIDO 安全性金鑰，或硬體式 MFA 裝置。
- 一個做為群組成員的測試 IAM 使用者，如下所示：

使用者名稱	使用者名稱說明	使用者群組名稱	新增使用者為成員	使用者群組說明
MFAUser	僅選擇 啟用主控台存取 (選用) 選項，並指定一組密碼。	EC2MFA	MFAUser	請勿連接任何政策或者授與許可給此使用者群組。

步驟 1：建立政策以實施 MFA 登入

首先建立一個 IAM 客戶受管政策，除 IAM 使用者對管理其自有憑證和 MFA 裝置所需的許可權外，該政策拒絕其他所有許可。

1. 以具有管理員登入資料的使用者身分登入 AWS 管理主控台。為了遵循 IAM 最佳實務，請勿使用您的 AWS 帳戶根使用者 登入資料登入。

   > **Important**
   >
   > IAM 最佳實務建議您要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 ，而不是使用具有長期憑證的 IAM 使用者。建議您僅將 IAM 使用者用於聯合身分使用者不支援的特定使用案例。

2. 在 https://console.aws.amazon.com/iam/ 中開啟 IAM 主控台。
3. 在導覽窗格中，選擇 Policies (政策)，然後選擇 Create policy (建立政策)。
4. 選擇 JSON 標籤並從下列 JSON 政策文件複製文字：AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。。
5. 將此政策文字貼上至 JSON 文字方框中。解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇下一步。

> **Note**
>
> 您可以隨時切換視覺化編輯器與 JSON 選項。但是，上方的政策會包含 `NotAction` 元素，其不受視覺化編輯器支援。針對此政策，您會在 Visual editor (視覺化編輯器) 標籤上看到一個通知。返回 JSON 以繼續處理此政策。
> 本範例政策不允許使用者在首次登入 AWS 管理主控台 時重設密碼。在新使用者登入並重設密碼之前，建議您不要將許可授予給他們。

6. 在檢視與建立頁面上，針對政策名稱輸入 **`Force_MFA`**。針對政策說明，在標籤區域中輸入 **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`**，您可以選擇性地新增標籤鍵值對至客戶管理政策。檢視政策授與的許可，然後選擇建立政策來儲存您的工作。

   新的政策會出現在受管政策清單中，並且已準備好連接。

步驟 2：將政策連接到您的測試使用者群組

接下來，您將兩個政策連接到 IAM 使用者 群組，將使用該群組授與以 MFA 提供保護的許可。

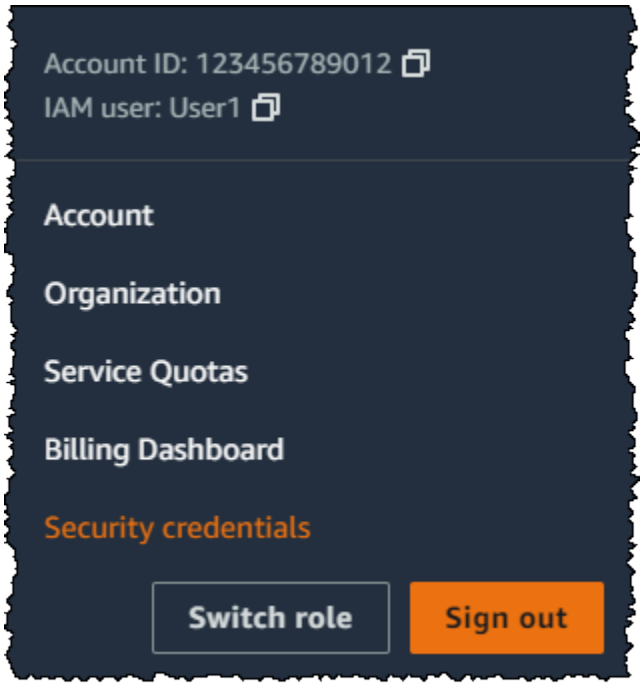
1. 在導覽窗格中，選擇 User groups (使用者群組) 。
2. 在搜尋方塊中，輸入 **`EC2MFA`**，然後在清單中選擇群組名稱 (而非核取方塊)。
3. 選擇 許可 標籤、選擇 新增許可，然後選擇 連接政策。
4. 在 Attach permission policies to EC2MFA group (將許可政策連接到 EC2MFA 群組) 頁面上，在搜尋方塊中輸入 **`EC2Full`**。然後在清單中，選取 AmazonEC2FullAccess 旁的核取方塊。目前先不要儲存您的變更。
5. 在搜尋方塊中，輸入 **`Force`**，然後在清單中選擇 Force_MFA 旁的核取方塊。
6. 選擇連接政策。

步驟 3：測試您的使用者存取權限

在本教程的這一部分中，您以測試使用者身分登入並確認政策是否正常運作。

1. **`MFAUser`** 使用您在上一節中指派的密碼，以 AWS 帳戶 身分登入您的 。使用 URL：`https://<alias or account ID number>.signin.aws.amazon.com/console`
2. 選擇 EC2 來開啟 Amazon EC2 主控台，並確認使用者沒有許可執行任何操作。

3. 在右上方的導覽列中，選擇您的 MFAUser 使用者名稱，然後選擇 安全憑證 。



4. 現在新增一個 MFA 裝置。在 Multi-Factor Authentication (MFA) (多重要素驗證 (MFA)) 區段，選擇 Assign MFA device (指派 MFA 裝置)。

> **Note**
>
> 您可能會收到錯誤，告知您未被授權執行 iam:DeleteVirtualMFADevice。如果有人先前開始將虛擬 MFA 裝置指派給這個使用者但取消程序，就可能發生此錯誤。若要繼續，您或其他管理員必須刪除該使用者未經指派的現有虛擬 MFA 裝置。如需詳細資訊，請參閱我未獲得授權，不得執行：iam:DeleteVirtualMFADevice。

5. 對於本教程，我們使用一個虛擬 (軟體式) MFA 裝置，例如，手機上的 Google Authenticator 應用程式。選擇 Authenticator app (驗證器應用程式)，然後按 Next (下一步)。

   IAM 將產生並顯示虛擬 MFA 裝置的配置資訊，包括 QR 碼圖形。此圖形是私密組態金鑰的表示形式，適用於不支援 QR 碼的裝置上的手動輸入。

6. 開啟您的虛擬 MFA 應用程式。(如需可以作為託管虛擬 MFA 裝置的應用程式清單，請查看虛擬 MFA 應用程式。) 如果虛擬 MFA 應用程式支援多個帳戶 (多個虛擬 MFA 裝置)，請選擇對應的選項以建立新帳戶 (新的虛擬 MFA 裝置)。

7. 判定 MFA 應用程式是否支援 QR 碼，然後執行以下操作之一：

- 從精靈中，選擇 Show QR code (顯示 QR 碼)。然後使用應用程式來掃描 QR 碼。例如，您可選擇相機圖示或選擇與 Scan code (掃描碼) 類似的選項，然後使用裝置的相機掃描碼。
- 在 Set up device (設定裝置) 精靈中，選擇 Show secret key (顯示私密金鑰)，然後在您的 MFA 應用程式中輸入私密金鑰。

完成操作後，虛擬 MFA 裝置會開始產生一次性密碼。

8. 在 Set up device (設定裝置) 精靈中的 Enter the code from your authenticator app. (輸入來自您的驗證器應用程式的代碼。) 方塊中，輸入虛擬 MFA 裝置上目前顯示的一次性密碼。選擇 Register MFA (註冊 MFA)。

> **Important**
>
> 產生代碼之後立即提交您的請求。如果在產生代碼後等待太久才提交請求，MFA 裝置將成功與使用者建立關聯。但是，MFA 裝置並不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。這種情況下，您可以重新同步裝置。

虛擬 MFA 裝置現在可以與 搭配使用 AWS。

9. 登出主控台，然後再次以 **`MFAUser`** 身分登入。這次會 AWS 提示您從手機輸入 MFA 代碼。收到該代碼後，請將代碼輸入方塊中，然後選擇 Submit (提交)。
10. 選擇 EC2 來再次開啟 Amazon EC2 主控台。請注意，這次您可以看到所有資訊，並且可以執行所需的任何動作。如果您以此使用者身分前往任何其他主控台，您會看到存取遭拒訊息。原因在於此教學中的政策僅會將存取權授與 Amazon EC2。

相關資源

如需其他資訊，請參閱以下主題：

- AWS IAM 中的多重要素驗證
- 啟用 MFA 的登入

IAM 教學課程：使用 CloudFormation 範本建立 SAML Identity Provider (IdP)

若要為 AWS 您的帳戶設定 SAML 聯合，您需要建立 SAML 身分提供者 (IdP)。本教學課程說明如何使用 CloudFormation 範本來建立 SAML IdP，以在 AWS 和外部 IdP 之間建立信任。

該範本會建立使用 IdP 中繼資料文件設定的 SAML IdP。然後，聯合 IAM 角色可以參考此 IdP，以允許來自外部 IdP 的已驗證使用者存取 AWS 資源。

部署的資源包含使用 IdP 中繼資料文件和選用加密設定來設定的 SAML IdP。

先決條件

此教學課程假設您已備妥下列項目：

- 在本機電腦上安裝的 Python 3.6 或更新版本，以便執行本教學課程中用於格式化 IdP SAML 中繼資料 XML 檔案的 Python 命令。
- 外部 IdP 中儲存為 XML 檔案的 SAML 中繼資料文件。

使用 建立 SAML IdP CloudFormation

若要建立 SAML IdP，請建立 CloudFormation 範本，並使用它來建立包含 IdP 資源的堆疊。

建立範本

首先，建立 CloudFormation 範本。

1. 在 範本區段中，選擇 JSON 或 YAML 標籤上的複製圖示，以複製範本內容。
2. 在新檔案中貼上範本內容。
3. 在本機儲存檔案。

建立 堆疊。

接下來，使用剛儲存的範本來佈建 CloudFormation 堆疊。

1. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 CloudFormation 主控台。
2. 在堆疊頁面上，從建立堆疊功能表中選擇使用新資源 (標準)。

3. 指定範本：

   a. 在必要條件下，選擇選擇現有範本。

   b. 在指定範本下，選擇上傳範本檔案。

   c. 選擇選擇檔案，導覽至範本檔案並選取它。

   d. 選擇下一步。

4. 指定下列堆疊詳細資訊：

   a. 輸入堆疊名稱。

   b. 對於 IdentityProviderName，您可以將其留空，以根據堆疊名稱自動產生名稱，或輸入 SAML IdP 的自訂名稱。自訂名稱只能包含字母數字字元、句號、底線和連字號。

   c. 對於 IdentityProviderSAMLMetadataDocument，您需要將 SAML 中繼資料 XML 檔案格式化為單行，然後才能將其貼入此欄位。這是必要的，因為 CloudFormation 主控台要求透過主控台參數傳遞的 XML 內容必須格式化為單行。

      使用下列 Python 命令來重新格式化 XML 檔案：

      ```
      python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>
      \s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-
      metadata.xml
      ```

      > **Note**
      >
      > IdP 的 SAML 中繼資料文件必須格式化為單行，才能用於主控台參數輸入。Python 命令會移除換行符號和多餘的空格，以建立所需的格式，同時保留所有原始內容和結構。

      從 Python 命令複製輸出，並將其貼至 IdentityProviderSAMLMetadataDocument 欄位。

      格式化的 SAML 中繼資料文件範例 (縮寫)：

      ```
      <?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
       xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
       entityID="https://portal.sso.example.com/saml/assertion/
      CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
       protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
       use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
      ```

```
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. 對於其他參數，請接受預設值或根據需求輸入自己的值：

- IdentityProviderAddPrivateKey：用於解密 SAML 聲明的選用私有金鑰
- IdentityProviderAssertionEncryptionMode – 選用，為 SAML 聲明設定加密模式 (「已允許」、「必要」或留空)

e. 選擇下一步。

5. 設定堆疊選項：

a. 在堆疊失敗選項下，選擇刪除所有新建立的資源。

> **Note**
>
> 選擇此選項可避免堆疊建立失敗時，因資源刪除政策指定為保留而可能產生的資源費用。

b. 接受所有其他預設值。

c. 在功能下，勾選方塊以確認 CloudFormation 可能會在帳戶中建立 IAM 資源。

d. 選擇下一步。

6. 檢閱堆疊詳細資訊，然後選擇提交。

CloudFormation 會建立堆疊。堆疊建立完成後，堆疊資源即可使用。您可以使用堆疊詳細資訊頁面上的資源索引標籤，來檢視帳戶中佈建的資源。

堆疊將輸出下列值，您可以在輸出索引標籤上檢視：

- ProviderARN：已建立 SAML IdP 的 ARN (例如 `arn:aws:iam::123456789012:saml-provider/CompanyIdP`)。建立信任此提供者的角色時，您將需要此 ARN。

- ProviderName：已建立 SAML IdP 的名稱 (例如，如果指定了自訂名稱，則為 `CompanyIdP`；或者如果使用了預設命名，則為 `my-saml-stack-saml-provider`)。

這些輸出也會匯出，並允許其他 CloudFormation 堆疊使用 `Fn::ImportValue` 函數匯入。

驗證 SAML IdP

建立 SAML IdP 之後，您可以驗證其組態，並記下其 ARN 以搭配聯合角色使用。

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，請選擇 Identity providers (身分提供者)。

   您應該會在清單中看到新建立的 SAML IdP。
3. 選擇 IdP 名稱以檢視其詳細資訊。

   在 IdP 詳細資訊頁面上，可以查看 SAML 中繼資料文件及其他組態詳細資訊。
4. 記下詳細資訊頁面上顯示的提供者 ARN。

   建立信任此 IdP 的聯合身分 IAM 角色時，您將需要此 ARN。
5. 檢閱中繼資料文件，以確保其符合您從外部 IdP 提供的內容。

SAML IdP 現已準備就緒，可供聯合身分 IAM 角色使用。您可以建立信任此 IdP 的角色，以允許來自外部 IdP 的已驗證使用者擔任這些角色和存取 AWS 資源。

清理：刪除資源

最後一個步驟是，刪除堆疊及其包含的資源。

1. 開啟 CloudFormation 主控台。
2. 在堆疊頁面上，選擇從範本建立的堆疊，然後選擇刪除，接著確認刪除。

   CloudFormation 會啟動刪除堆疊及其包含的所有資源。

CloudFormation 範本詳細資訊

Resources

本教學課程的 CloudFormation 範本會在您的帳戶中建立下列資源：

AWS::IAM::SAMLProvider：在 AWS 和外部 IdP 之間建立信任的 SAML IdP。

組態

範本包含下列可設定的參數：

- IdentityProviderName – SAML IdP 的名稱 (留空以自動產生名稱)

  範例：CompanyIdP 或 EnterpriseSSO
- IdentityProviderSAMLMetadataDocument – 來自外部 IdP 的 SAML 中繼資料文件 (格式化為單行)
- IdentityProviderAddPrivateKey：用於解密 SAML 聲明的選用私有金鑰
- IdentityProviderAssertionEncryptionMode – 選用，為 SAML 聲明設定加密模式

CloudFormation 範本

將下列 JSON 或 YAML 程式碼儲存為個別檔案，以用作本教學課程的 CloudFormation 範本。

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-idp",
  "Parameters": {
    "IdentityProviderName": {
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be empty or contain only alphanumeric
 characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
 private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
 to decrypt SAML assertions.",
```

```
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    }
  },
  "Conditions": {
    "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
    "HasEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]},
    "HasCustomName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "IdentityProviderName"},
""]}]}
  },
  "Resources": {
    "SAMLProvider": {
      "Type": "AWS::IAM::SAMLProvider",
      "Properties": {
        "Name": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
{"Ref": "AWS::NoValue"}]},
        "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
        "Tags": [
          {
            "Key": "Name",
            "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
{"Fn::Sub": "${AWS::StackName}-saml-provider"}]}
          }
        ],
        "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
"IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
        "AssertionEncryptionMode": {"Fn::If": ["HasEncryptionMode", {"Ref":
"IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
      }
    }
  },
  "Outputs": {
    "ProviderARN": {
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderARN"}
```

```
      }
    },
    "ProviderName": {
      "Description": "Name of the SAML Identity Provider",
      "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
 {"Fn::Sub": "${AWS::StackName}-saml-provider"}]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderName"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
    ConstraintDescription: 'Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens'

  IdentityProviderSAMLMetadataDocument:
    Type: String
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
    Description: Optional private key for decrypting SAML assertions. The private
 key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
 decrypt SAML assertions.
    Default: ""

  IdentityProviderAssertionEncryptionMode:
    Type: String
    Description: Optional, sets encryption mode for SAML assertions
    Default: ""
```

```
    AllowedValues:
      - ""
      - "Allowed"
      - "Required"

Conditions:
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasEncryptionMode: !Not [!Equals [!Ref IdentityProviderAssertionEncryptionMode,
 ""]]
  HasCustomName: !Not [!Equals [!Ref IdentityProviderName, ""]]

Resources:
  SAMLProvider:
    Type: 'AWS::IAM::SAMLProvider'
    Properties:
      Name: !If
        - HasCustomName
        - !Ref IdentityProviderName
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      Tags:
        - Key: Name
          Value: !If
            - HasCustomName
            - !Ref IdentityProviderName
            - !Sub '${AWS::StackName}-saml-provider'
      AddPrivateKey: !If
        - HasPrivateKey
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
        - HasEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
        - !Ref AWS::NoValue

Outputs:
  ProviderARN:
    Description: 'ARN of the created SAML Identity Provider'
    Value: !Ref SAMLProvider
    Export:
      Name: !Sub '${AWS::StackName}-ProviderARN'

  ProviderName:
    Description: 'Name of the SAML Identity Provider'
```

```
    Value: !If
      - HasCustomName
      - !Ref IdentityProviderName
      - !Sub '${AWS::StackName}-saml-provider'
    Export:
      Name: !Sub '${AWS::StackName}-ProviderName'
```

IAM 教學課程：使用 CloudFormation 範本建立 SAML 聯合 IAM 角色

當您在 AWS 帳戶中設定現有的 SAML Identity Provider (IdP) 時，您可以建立信任該 IdP 的聯合 IAM 角色。本教學課程說明如何使用 CloudFormation 範本來建立 SAML 聯合 IAM 角色，該角色可由透過外部 IdP 驗證的使用者擔任。

該範本會建立具有信任政策的聯合身分 IAM 角色，允許 SAML IdP 擔任該角色。透過外部 IdP 進行身分驗證的使用者可以擔任此角色，以便根據連接至該角色的許可存取 AWS 資源。

部署的資源包含下列項目：

- 信任現有 SAML IdP 的聯合身分 IAM 角色。
- 可連接至角色以授予特定許可的可設定受管政策。
- 許可界限和工作階段持續時間的選用設定。

先決條件

此教學課程假設您已備妥下列項目：

- 您 AWS 帳戶中設定的現有 SAML IdP。如果沒有，可以使用 IAM 教學課程：使用 CloudFormation 範本建立 SAML Identity Provider (IdP) 教學課程建立一個。
- SAML IdP 的 ARN，您需要在建立堆疊時將其指定為參數。
- 在本機電腦上安裝的 Python 3.6 或更新版本，以便執行本教學課程中用於格式化 IdP SAML 中繼資料 XML 檔案的 Python 命令。

使用 建立 SAML 聯合角色 CloudFormation

若要建立 SAML 聯合角色，您將建立 CloudFormation 範本，並使用它來建立包含該角色的堆疊。

建立範本

首先，建立 CloudFormation 範本。

1. 在 範本 區段中，按一下 JSON 或 YAML 索引標籤上的複製圖示，以複製範本內容。
2. 在新檔案中貼上範本內容。
3. 在本機儲存檔案。

建立 堆疊。

接下來，使用剛儲存的範本來佈建 CloudFormation 堆疊。

1. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 CloudFormation 主控台。
2. 在堆疊頁面上，從建立堆疊功能表中選擇使用新資源 (標準)。
3. 指定範本：
   a. 在必要條件下，選擇選擇現有範本。
   b. 在指定範本下，選擇上傳範本檔案。
   c. 選擇選擇檔案，導覽至範本檔案並選取它。
   d. 選擇下一步。
4. 指定下列堆疊詳細資訊：
   a. 輸入堆疊名稱。
   b. 在 SAMLProviderARN 中，輸入現有 SAML IdP 的 ARN。其格式應為 `arn:aws:iam::123456789012:saml-provider/YourProviderName`。

      範例：`arn:aws:iam::123456789012:saml-provider/CompanyIdP`

      > **Note**
      > 如果使用 IAM 教學課程：使用 CloudFormation 範本建立 SAML Identity Provider (IdP) 教學課程建立 SAML IdP，則可在該 CloudFormation 堆疊的「輸出」索引標籤中找到提供者 ARN。

   c. 對於 RoleName，可以將其留空，以根據堆疊名稱自動產生名稱，或輸入 IAM 角色的自訂名稱。

範例：SAML-Developer-Access 或 SAML-ReadOnly-Role

d. 對於其他參數，請接受預設值或根據需求輸入自己的值：

- RoleSessionDuration – 工作階段持續時間上限 (以秒為單位，3600-43200，預設值 7200)

  範例：14400 (4 小時)

- RolePermissionsBoundary – 許可界限政策的選用 ARN

  範例：arn:aws:iam::123456789012:policy/DeveloperBoundary

- RolePath – IAM 角色的路徑 (預設為 /)

  範例：/saml-roles/

- ManagedPolicy1-5 – 最多可連接 5 個受管政策的選用 ARN

  ManagedPolicy1 的範例：arn:aws:iam::aws:policy/ReadOnlyAccess

  ManagedPolicy2 的範例：arn:aws:iam::123456789012:policy/CustomPolicy

e. 選擇下一步。

5. 設定堆疊選項：

a. 在堆疊失敗選項下，選擇刪除所有新建立的資源。

> **Note**
>
> 選擇此選項可避免堆疊建立失敗時，因資源刪除政策指定為保留而可能產生的資源費用。

b. 接受所有其他預設值。

c. 在功能下，勾選方塊以確認 CloudFormation 可能會在帳戶中建立 IAM 資源。

d. 選擇下一步。

6. 檢閱堆疊詳細資訊，然後選擇提交。

CloudFormation 會建立堆疊。堆疊建立完成後，堆疊資源即可使用。您可以使用堆疊詳細資訊頁面上的資源索引標籤，來檢視帳戶中佈建的資源。

堆疊將輸出下列值，您可以在輸出索引標籤上檢視該值：

- RoleARN：已建立 IAM 角色的 ARN (例如 `arn:aws:iam::123456789012:role/SAML-Developer-Access`，或者如果使用自動產生的名稱，則為 `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4`)。

在設定 IdP 以傳送用於擔任角色的相應 SAML 屬性時，您將需要此角色 ARN。

測試 SAML 聯合身分角色

SAML 聯合身分角色建立後，您可以驗證其組態並測試聯合身分設定。

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色。
3. 找到並選擇新建立的聯合身分角色。

   如果您提供了自訂角色名稱，請尋找該名稱。如果將 RoleName 參數留空，則該角色將根據堆疊名稱和唯一識別碼自動產生名稱。
4. 選擇信任關係索引標籤以檢閱信任政策。

   信任政策應該會顯示 SAML IdP 受信任可擔任此角色，條件是 SAML 對象 (`SAML:aud`) 與 `https://signin.aws.amazon.com/saml` 相符。
5. 選擇許可索引標籤以檢閱連接的政策。

   您可以看到在建立期間連接至角色的所有受管政策。
6. 記下角色摘要頁面上顯示的角色 ARN。

   您需要此 ARN 才能設定外部 IdP，以允許使用者擔任此角色。

SAML 聯合身分角色現已準備就緒，可供使用。將外部 IdP 設定為在 SAML 聲明中包含此角色的 ARN，經過驗證的使用者將能夠擔任此角色來存取 AWS 資源。

清理：刪除資源

最後一個步驟是，刪除堆疊及其包含的資源。

1. 開啟 CloudFormation 主控台。
2. 在堆疊頁面上，選擇從範本建立的堆疊，然後選擇刪除，接著確認刪除。

   CloudFormation 會啟動刪除堆疊及其包含的所有資源。

CloudFormation 範本詳細資訊

Resources

本教學課程的 CloudFormation 範本會在您的帳戶中建立下列資源：

- `AWS::IAM::Role`：可由透過 SAML IdP 進行身分驗證之使用者擔任的聯合身分 IAM 角色。

組態

範本包含下列可設定的參數：

- RoleName – IAM 角色的名稱 (留空以自動產生名稱)
- SAMLProviderARN – SAML IdP 的 ARN (必要)
- RoleSessionDuration – 工作階段持續時間上限 (以秒為單位，3600-43200，預設值 7200)
- RolePermissionsBoundary – 許可界限政策的選用 ARN
- RolePath – IAM 角色的路徑 (預設值 /)
- ManagedPolicy1-5 – 最多可連接 5 個受管政策的選用 ARN

CloudFormation 範本

將下列 JSON 或 YAML 程式碼儲存為個別檔案，以用作本教學課程的 CloudFormation 範本。

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-federated-role",
  "Parameters": {
    "RoleName": {
      "Type": "String",
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
 '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-"
    },
    "SAMLProviderARN": {
```

```
      "Type": "String",
      "Description": "ARN of the SAML Identity Provider",
      "AllowedPattern": "^arn:aws:iam::\\d{12}:saml-provider/[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be a valid SAML provider ARN"
    },
    "RoleSessionDuration": {
      "Type": "Number",
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RolePermissionsBoundary": {
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM role (must start and end with /)",
      "Default": "/",
      "AllowedPattern": "^\/.*\/$|^\/$",
      "ConstraintDescription": "Role path must start and end with forward slash (/)"
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
```

```
      "Default": ""
    },
    "RoleManagedPolicy5": {
      "Type": "String",
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
    "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
    "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
    "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
    "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
    "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
    "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
    "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"}, ""]}]}
  },
  "Resources": {
    "SAMLFederatedRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
        "Description": "IAM role with SAML provider trust",
        "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
        "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
        "Path": {"Ref": "RolePath"},
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Federated": {"Ref": "SAMLProviderARN"}
              },
              "Action": "sts:AssumeRoleWithSAML",
              "Condition": {
                "StringEquals": {
```

```
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
 "AWS::NoValue"}]}
                ]
              ]
            }
          ]
        }
      }
    }
  },
  "Outputs": {
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-federated-role'

Parameters:
  RoleName:
    Type: String
    Description: 'Name of the IAM Role (leave empty for auto-generated name like
 ''{StackName}-{UniqueId}'')'
    Default: ""
    AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
    ConstraintDescription: 'Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-'

  SAMLProviderARN:
    Type: String
    Description: 'ARN of the SAML Identity Provider'
    AllowedPattern: '^arn:aws:iam::\d{12}:saml-provider/[a-zA-Z0-9._-]+$'
    ConstraintDescription: 'Must be a valid SAML provider ARN'

  RoleSessionDuration:
    Type: Number
    Description: 'The maximum session duration (in seconds) that you want to set for
 the specified role (3600-43200)'
    MinValue: 3600
    MaxValue: 43200
    Default: 7200

  RolePermissionsBoundary:
    Type: String
    Description: Optional ARN of the permissions boundary policy (leave empty for
 none)
    Default: ""

  RolePath:
    Type: String
    Description: 'Path for the IAM role (must start and end with /)'
    Default: "/"
    AllowedPattern: '^\/.*\/$|^\/$'
    ConstraintDescription: 'Role path must start and end with forward slash (/)'

  RoleManagedPolicy1:
    Type: String
```

```
    Description: Optional managed policy ARN 1
    Default: ""
  RoleManagedPolicy2:
    Type: String
    Description: Optional managed policy ARN 2
    Default: ""
  RoleManagedPolicy3:
    Type: String
    Description: Optional managed policy ARN 3
    Default: ""
  RoleManagedPolicy4:
    Type: String
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]

Resources:
  SAMLFederatedRole:
    Type: 'AWS::IAM::Role'
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
        - !Ref AWS::NoValue
      Description: 'IAM role with SAML provider trust'
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      AssumeRolePolicyDocument:
```

```
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProviderARN
            Action: 'sts:AssumeRoleWithSAML'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns:
        !Split
          - ','
          - !Join
            - ','
            - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref 'AWS::NoValue']
              - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref 'AWS::NoValue']
              - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref 'AWS::NoValue']
              - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref 'AWS::NoValue']
              - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref 'AWS::NoValue']

Outputs:
  RoleARN:
    Description: 'ARN of the created IAM Role'
    Value: !GetAtt SAMLFederatedRole.Arn
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'
```

IAM 教學課程：使用 CloudFormation 範本建立 SAML 身分提供者 (IdP) 和 SAML 聯合 IAM 角色

若要熟悉 SAML 聯合及其功能，您將使用 CloudFormation 範本來設定 SAML 身分提供者 (IdP) 和相關聯的聯合 IAM 角色。本教學課程說明如何在單一堆疊中同時建立這兩個資源。

範本會建立 SAML IdP，可用於聯合存取 AWS 資源，以及信任 SAML 供應商的 IAM 角色。外部 IdP 驗證的使用者可以擔任此角色來存取 AWS 資源。

部署的資源包含下列項目：

- 使用 IdP 中繼資料文件設定的 SAML IdP。
- 信任 SAML IdP 且可由經過身分驗證之使用者擔任的聯合身分 IAM 角色。

- 可連接至角色以授予特定許可的可設定受管政策。

先決條件

此教學課程假設您已備妥下列項目：

- 在本機電腦上安裝的 Python 3.6 或更新版本，以便執行本教學課程中用於格式化 IdP SAML 中繼資料 XML 檔案的 Python 命令。
- 外部 IdP 中儲存為 XML 檔案的 SAML 中繼資料文件。

使用 建立 SAML IdP 和角色 CloudFormation

若要建立 SAML IdP 和聯合角色，您將建立 CloudFormation 範本，並使用它來建立包含這兩個資源的堆疊。

建立範本

首先，建立 CloudFormation 範本。

1. 在 範本 區段中，按一下 JSON 或 YAML 索引標籤上的複製圖示，以複製範本內容。
2. 在新檔案中貼上範本內容。
3. 在本機儲存檔案。

建立 堆疊。

接下來，使用剛儲存的範本來佈建 CloudFormation 堆疊。

1. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 CloudFormation 主控台。
2. 在堆疊頁面上，從建立堆疊功能表中選擇使用新資源 (標準)。
3. 指定範本：
    a. 在必要條件下，選擇選擇現有範本。
    b. 在指定範本下，選擇上傳範本檔案。
    c. 選擇選擇檔案，導覽至範本檔案並選取它。
    d. 選擇下一步。
4. 指定下列堆疊詳細資訊：

a. 輸入堆疊名稱。

b. 對於 IdentityProviderName，您可以將其留空，以根據堆疊名稱自動產生名稱，或輸入 SAML IdP 的自訂名稱。

   範例：`CompanyIdP` 或 `EnterpriseSSO`

c. 對於 IdentityProviderSAMLMetadataDocument，您需要將 SAML 中繼資料 XML 檔案格式化為單行，然後才能將其貼入此欄位。這是必要的，因為 CloudFormation 主控台要求透過主控台參數傳遞的 XML 內容必須格式化為單行。

   使用下列 Python 命令來重新格式化 XML 檔案：

```
python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>
\s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-
metadata.xml
```

> **Note**
>
> IdP 的 SAML 中繼資料文件必須格式化為單行，才能用於主控台參數輸入。Python 命令會移除換行符號和多餘的空格，以建立所需的格式，同時保留所有原始內容和結構。

   從 Python 命令複製輸出，並將其貼至 IdentityProviderSAMLMetadataDocument 欄位。

   格式化的 SAML 中繼資料文件範例 (縮寫)：

```
<?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 entityID="https://portal.sso.example.com/saml/assertion/
CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
 use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
```

```
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. 對於 RoleName，可以將其留空，以根據堆疊名稱自動產生名稱，或輸入聯合身分 IAM 角色的自訂名稱。

   範例：`SAML-Developer-Access` 或 `SAML-ReadOnly-Role`

e. 對於其他參數，請接受預設值或根據需求輸入自己的值：

   - IdentityProviderAddPrivateKey：用於解密 SAML 聲明的選用私有金鑰
   - IdentityProviderAssertionEncryptionMode – SAML 聲明的加密模式

     範例值：`Allowed`、`Required` 或留空表示不加密
   - RoleSessionDuration – 工作階段持續時間上限 (以秒為單位，3600-43200，預設值 7200)

     範例：`14400` (4 小時)
   - RolePermissionsBoundary – 許可界限政策的選用 ARN

     範例：`arn:aws:iam::123456789012:policy/DeveloperBoundary`
   - RolePath – IAM 角色的路徑 (預設為 /)

     範例：`/saml-roles/`
   - RoleManagedPolicy1-5 – 最多可連接 5 個受管政策的選用 ARN

     RoleManagedPolicy1 的範例：`arn:aws:iam::aws:policy/ReadOnlyAccess`

     RoleManagedPolicy2 的範例：`arn:aws:iam::123456789012:policy/CustomPolicy`

f. 選擇下一步。

5. 設定堆疊選項：

   a. 在堆疊失敗選項下，選擇刪除所有新建立的資源。

      > **Note**
      >
      > 選擇此選項可避免堆疊建立失敗時，因資源刪除政策指定為保留而可能產生的資源費用。

b. 接受所有其他預設值。

c. 在功能下，勾選方塊以確認 CloudFormation 可能會在帳戶中建立 IAM 資源。

d. 選擇下一步。

6. 檢閱堆疊詳細資訊，然後選擇提交。

CloudFormation 會建立堆疊。堆疊建立完成後，堆疊資源即可使用。您可以使用堆疊詳細資訊頁面上的資源索引標籤，來檢視帳戶中佈建的資源。

堆疊將輸出下列值，您可以在輸出索引標籤上檢視：

- RoleARN：已建立 IAM 角色的 ARN (例如 `arn:aws:iam::123456789012:role/SAML-Developer-Access`，或者如果使用自動產生的名稱，則為 `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4`)。
- IdentityProviderARN：已建立 SAML IdP 的 ARN (例如 `arn:aws:iam::123456789012:saml-provider/CompanyIdP`)。

在設定 IdP 以傳送用於擔任角色的相應 SAML 屬性時，您將需要這兩個 ARN。

測試 SAML 聯合身分

SAML IdP 和聯合身分角色建立後，您可以測試聯合身分設定。

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，請選擇 Identity providers (身分提供者)。

   您應該會在清單中看到新建立的 SAML IdP。
3. 選擇 IdP 名稱以檢視其詳細資訊。

   在 IdP 詳細資訊頁面上，可以查看 SAML 中繼資料文件及其他組態詳細資訊。
4. 在導覽窗格中，選擇角色。
5. 找到並選擇新建立的聯合身分角色。

   在角色詳細資訊頁面上，可以查看允許 SAML IdP 擔任此角色的信任政策。
6. 選擇信任關係索引標籤以檢閱信任政策。

   信任政策應該會顯示 SAML IdP 受信任可擔任此角色，條件是 SAML 對象 (`SAML:aud`) 與 `https://signin.aws.amazon.com/saml` 相符。

清理：刪除資源

最後一個步驟是，刪除堆疊及其包含的資源。

1. 開啟 CloudFormation 主控台。
2. 在堆疊頁面上，選擇從範本建立的堆疊，然後選擇刪除，接著確認刪除。

   CloudFormation 會啟動刪除堆疊及其包含的所有資源。

CloudFormation 範本詳細資訊

Resources

本教學課程的 CloudFormation 範本會在您的帳戶中建立下列資源：

- `AWS::IAM::SAMLProvider`：在 AWS 和外部 IdP 之間建立信任的 SAML IdP。
- `AWS::IAM::Role`：可由透過 SAML IdP 進行身分驗證之使用者擔任的聯合身分 IAM 角色。

組態

範本包含下列可設定的參數：

- IdentityProviderName – SAML IdP 的名稱 (留空以自動產生名稱)
- IdentityProviderSAMLMetadataDocument – 來自 IdP 的 SAML 中繼資料文件 (必要)
- IdentityProviderAddPrivateKey：用於解密 SAML 聲明的選用私有金鑰
- IdentityProviderAssertionEncryptionMode – SAML 聲明的加密模式
- RoleName – IAM 角色的名稱 (留空以自動產生名稱)
- RolePath – IAM 角色的路徑 (預設值 /)
- RolePermissionsBoundary – 許可界限政策的選用 ARN
- RoleSessionDuration – 工作階段持續時間上限 (以秒為單位，3600-43200，預設值 7200)
- RoleManagedPolicy1-5 – 最多可連接 5 個受管政策的選用 ARN

CloudFormation 範本

將下列 JSON 或 YAML 程式碼儲存為個別檔案，以用作本教學課程的 CloudFormation 範本。

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-idp-and-federated-role",
  "Parameters": {
    "IdentityProviderName": {
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be empty or contain only alphanumeric
 characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
 private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
 to decrypt SAML assertions.",
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    },
    "RoleName": {
      "Type": "String",
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
 '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-"
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM Role",
```

```
      "AllowedPattern": "(^\\/$)|(^\\/.*\\/$)",
      "Default": "/"
    },
    "RolePermissionsBoundary": {
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RoleSessionDuration": {
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
      "Type": "Number",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
      "Default": ""
    },
    "RoleManagedPolicy5": {
      "Type": "String",
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
```

```
   "HasCustomProviderName": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderName"}, ""]}]},
   "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
   "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
   "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
   "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
   "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
   "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
   "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"},
""]}]},
   "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
   "HasAssertionEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]}
 },
 "Resources": {
   "SAMLProvider": {
     "Type": "AWS::IAM::SAMLProvider",
     "Properties": {
       "Name": {"Fn::If": ["HasCustomProviderName", {"Ref":
"IdentityProviderName"}, {"Ref": "AWS::NoValue"}]},
       "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
       "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
"IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
       "AssertionEncryptionMode": {"Fn::If": ["HasAssertionEncryptionMode", {"Ref":
"IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
     }
   },
   "SAMLFederatedRole": {
     "Type": "AWS::IAM::Role",
     "Properties": {
       "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
       "Path": {"Ref": "RolePath"},
       "Description": "SAML federated IAM role for SSO access with specified
permissions",
       "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
       "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
```

```
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Federated": {"Ref": "SAMLProvider"}
              },
              "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity",
                "sts:TagSession"
              ],
              "Condition": {
                "StringEquals": {
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
"AWS::NoValue"}]}
                ]
              ]
            }
          ]
        }
      }
```

```
    }
  },
  "Outputs": {
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    },
    "IdentityProviderARN": {
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-IdentityProviderARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp-and-federated-role'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
    ConstraintDescription: Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens

  IdentityProviderSAMLMetadataDocument:
    Type: String
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
```

```
    Description: Optional private key for decrypting SAML assertions. The private
key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
decrypt SAML assertions.
    Default: ""

  IdentityProviderAssertionEncryptionMode:
    Type: String
    Description: Optional, sets encryption mode for SAML assertions
    Default: ""
    AllowedValues:
      - ""
      - "Allowed"
      - "Required"

  RoleName:
    Type: String
    Description: Name of the IAM Role (leave empty for auto-generated name like
'{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
    ConstraintDescription: "Must be empty or 1-64 characters and can contain
alphanumeric characters and +=,.@-"

  RolePath:
    Type: String
    Description: Path for the IAM Role
    AllowedPattern: (^\/$)|(^\/.*\/$)
    Default: "/"

  RolePermissionsBoundary:
    Type: String
    Description: Optional ARN of the permissions boundary policy (leave empty for
none)
    Default: ""

  RoleSessionDuration:
    Description: The maximum session duration (in seconds) that you want to set for
the specified role (3600-43200)
    Type: Number
    MinValue: 3600
    MaxValue: 43200
    Default: 7200

  RoleManagedPolicy1:
```

```
    Type: String
    Description: Optional managed policy ARN 1
    Default: ""
  RoleManagedPolicy2:
    Type: String
    Description: Optional managed policy ARN 2
    Default: ""
  RoleManagedPolicy3:
    Type: String
    Description: Optional managed policy ARN 3
    Default: ""
  RoleManagedPolicy4:
    Type: String
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomProviderName: !Not [!Equals [!Ref IdentityProviderName, ""]]
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasAssertionEncryptionMode: !Not [!Equals [!Ref
 IdentityProviderAssertionEncryptionMode, ""]]

Resources:
  SAMLProvider:
    Type: AWS::IAM::SAMLProvider
    Properties:
      Name: !If
        - HasCustomProviderName
        - !Ref IdentityProviderName
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      AddPrivateKey: !If
        - HasPrivateKey
```

```
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
        - HasAssertionEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
        - !Ref AWS::NoValue

  SAMLFederatedRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      Description: "SAML federated IAM role for SSO access with specified
permissions"
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProvider
            Action:
              - 'sts:AssumeRole'
              - 'sts:SetSourceIdentity'
              - 'sts:TagSession'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns: !Split
        - ','
        - !Join
          - ','
          - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref "AWS::NoValue"]
            - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref "AWS::NoValue"]
            - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref "AWS::NoValue"]
            - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref "AWS::NoValue"]
            - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref "AWS::NoValue"]
```

```
Outputs:
  RoleARN:
    Description: ARN of the created IAM Role
    Value: !GetAtt SAMLFederatedRole.Arn
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'

  IdentityProviderARN:
    Description: ARN of the created SAML Identity Provider
    Value: !Ref SAMLProvider
    Export:
      Name: !Sub '${AWS::StackName}-IdentityProviderARN'
```

IAM 身分

IAM 身分可以與一或多個政策建立關聯，這些政策會決定身分獲授權執行的動作、 AWS 資源以及條件。IAM 身分包括 IAM 使用者、IAM 群組及 IAM 角色。IAM 實體是一種身分類型，代表可以進行身分驗證並授權在 AWS 帳戶中執行動作的人類使用者或程式設計式工作負載。IAM 實體包括 IAM 使用者和 IAM 角色。如需常用詞彙的定義，請參閱條款。

您可以從外部身分提供者聯合現有的身分。這些身分將擔任 IAM 角色來存取 AWS 資源。如需詳細資訊，請參閱the section called “身分提供者和 的聯合身分 AWS”。

您也可以使用 AWS IAM Identity Center 來建立和管理身分和對 AWS 資源的存取。IAM Identity Center 許可集會自動建立提供對資源的存取權所需的 IAM 角色。如需詳細資訊，請參閱什麼是 IAM Identity Center？

您可以使用帳戶存取管理員 — IAM 功能，可讓您將具有完整 IAM 角色功能集的現有 IAM 角色指派給 IAM Identity Center 使用者和群組。您可以將它與許可集一起使用或單獨使用。

AWS 帳戶根使用者 是在 AWS 帳戶 建立 時建立的 AWS 帳戶 委託人。根使用者可存取帳戶中的所有 AWS 服務和資源。如需詳細資訊，請參閱the section called “IAM 根使用者”。

> **Note**
>
> - 使用 IAM 身分時，遵循 IAM 中的安全最佳實務。
> - 使用根使用者時，遵循 AWS 帳戶的根使用者最佳實務。
> - 如果您登入時遇到問題，請參閱登入 AWS 管理主控台。

IAM 根使用者

當您第一次建立 時 AWS 帳戶，您會從一個登入身分開始，該身分可完整存取帳戶中的所有 AWS 服務和資源。此身分稱為 AWS 帳戶 Theroot 使用者。如需詳細資訊，請參閱 AWS 帳戶根使用者。

IAM 使用者

IAM 使用者是 中的身分 AWS 帳戶 ，具有單一人員或應用程式的特定許可。如需詳細資訊，請參閱 IAM 使用者。

IAM 使用者群組

IAM 使用者群組是一種指定 IAM 使用者集合的身分。如需詳細資訊，請參閱使用者群組。

IAM 角色

IAM 角色是 中具有特定許可 AWS 帳戶 的身分。它類似 IAM 使用者，但未與特定的人員建立關聯。如需詳細資訊，請參閱 IAM 角色。

AWS 帳戶根使用者

當您首次建立 Amazon Web Services (AWS) 帳戶時，您會從單一登入身分開始，該身分可完整存取帳戶中的所有 AWS 服務和資源。此身分稱為 AWS 帳戶根使用者。您用來建立 的電子郵件地址和密碼 AWS 帳戶 是您用來以根使用者身分登入的登入資料。

- 僅使用根使用者來執行需要根層級許可的任務。如需檢視需要您以根使用者身分登入的任務完整清單，請參閱 需要根使用者憑證的任務。
- 遵循適用於 AWS 帳戶的根使用者最佳實務。
- 如果您登入時遇到問題，請參閱登入 AWS 管理主控台。

> **Important**
>
> 強烈建議您不使用根使用者處理日常任務，並且遵循 AWS 帳戶的根使用者最佳實務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需檢視需要您以根使用者身分登入的任務完整清單，請參閱 需要根使用者憑證的任務。

儘管依預設，根使用者會強制啟用 MFA，但這需要使用者在初始帳戶建立期間或登入期間依提示手動新增 MFA。如需有關使用 MFA 保護根使用者的詳細資訊，請參閱的多重要素驗證 AWS 帳戶根使用者。

除了主控台存取之外，您還可以管理根使用者的程式設計存取。您可以為根使用者建立存取金鑰和 刪除根使用者的存取金鑰 何時不再需要它們。

集中管理成員帳戶的根存取權

若要協助您大規模管理憑證，您可以集中安全地存取 AWS Organizations中成員帳戶的根使用者憑證。啟用 時 AWS Organizations，您可以將所有 AWS 帳戶合併為組織以進行集中管理。集中根存取權可讓您移除根使用者憑證，並對成員帳戶執行下列特權任務。

移除成員帳戶根使用者憑證

集中根存取成員帳戶後，您可以選擇從 Organizations 中的成員帳戶刪除根使用者憑證。您可以移除根使用者密碼、存取金鑰、簽署憑證，以及停用多重要素驗證 (MFA)。您在 Organizations 中建立的新帳戶預設沒有根使用者憑證。成員帳戶無法登入其根使用者，也無法為其根使用者執行密碼復原 (除非啟用帳戶復原)。

執行需要根使用者憑證的特權任務

某些任務只有在您以帳戶的根使用者身分登入時才能執行。其中一些 需要根使用者憑證的任務 可以由 IAM 的管理帳戶或委派管理員執行。若要進一步了解如何對成員帳戶採取特權動作，請參閱執行特權任務。

啟用根使用者的帳戶復原

如果您需要復原成員帳戶的根使用者憑證，Organizations 管理帳戶或委派管理員可以執行允許密碼復原特權任務。有權存取成員帳戶根使用者電子郵件收件匣的人員可以重設根使用者密碼，以復原根使用者憑證。我們建議您在完成需要存取根使用者的任務後刪除根使用者憑證。

集中成員帳戶的根存取權

根使用者登入資料是指派給每個 AWS 帳戶 的初始登入資料，可完整存取帳戶中的所有 AWS 服務和資源。啟用 時 AWS Organizations，您可以將所有 AWS 帳戶合併為組織以進行集中管理。每個成員帳戶都有自己的根使用者，具有在成員帳戶中執行任何動作的預設許可。我們建議您集中保護使用 管理的 AWS 帳戶 根使用者憑證 AWS Organizations ，以防止根使用者憑證大規模復原和存取。

集中根存取權後，您可以選擇從組織的成員帳戶中刪除根使用者憑證。您可以移除根使用者密碼、存取金鑰、簽署憑證，以及停用多重要素驗證 (MFA)。依預設，您在 AWS Organizations 中建立的新帳戶沒有根使用者憑證。成員帳戶無法登入其根使用者，也無法為其根使用者執行密碼復原。

> **Note**
>
> 雖然某些需要根使用者憑證的任務可由管理帳戶或 IAM 的委派管理員執行，但某些任務只能以帳戶的根使用者身分登入才能執行。

如果您需要復原成員帳戶的根使用者憑證才能執行這些任務之一，請遵循執行特權任務中的步驟，然後選取允許密碼復原。之後，有權存取成員帳戶根使用者電子郵件收件匣的人員可以遵循步驟重設根使用者密碼，並登入成員帳戶根使用者。
我們建議您在完成需要存取根使用者的任務後，刪除根使用者憑證。

先決條件

在集中根存取權之前，您必須使用下列設定設定帳戶：

- 您必須具備下列 IAM 許可：
  - `iam:GetAccessKeyLastUsed`
  - `iam:GetAccountSummary`
  - `iam:GetLoginProfile`
  - `iam:GetUser`
  - `iam:ListAccessKeys`
  - `iam:ListMFADevices`
  - `iam:ListSigningCertificates`
  - `sts:AssumeRoot`

  > **Note**
  >
  > 若要稽核成員帳戶的根使用者登入資料狀態，您可以在 AWS Organizations 成員帳戶上執行特殊權限任務時，使用 IAMAuditRootUserCredentials AWS 受管政策來縮小許可範圍，或使用可存取 的任何政策`iam:GetAccountSummary`。
  > 若要產生根使用者憑證資訊報告，其他政策只需 `iam:GetAccountSummary` 動作即可產生相同的輸出。您也可以列出或取得個別根使用者憑證資訊，包括：
  >
  > - 根使用者密碼是否存在
  > - 根使用者存取金鑰是否存在以及上次使用時間
  > - 根使用者是否具有相關聯的簽署憑證
  > - 根使用者關聯的 MFA 裝置
  > - 合併的根使用者憑證狀態清單

- 您必須在 AWS 帳戶 中管理 AWS Organizations。
- 您必須具有下列許可，以在組織中啟用此功能：

  - iam:EnableOrganizationsRootCredentialsManagement
  - iam:EnableOrganizationsRootSessions
  - iam:ListOrganizationsFeatures
  - organizations:EnableAwsServiceAccess
  - organizations:ListAccountsForParent
  - organizations:RegisterDelegatedAdministrator
- 為確保最佳的主控台功能，建議啟用下列其他許可：
  - organizations:DescribeAccount
  - organizations:DescribeOrganization
  - organizations:ListAWSServiceAccessForOrganization
  - organizations:ListDelegatedAdministrators
  - organizations:ListOrganizationalUnitsForParent
  - organizations:ListParents
  - organizations:ListTagsForResource

啟用集中式根存取權 (主控台)

若要在 中為成員帳戶啟用此功能 AWS 管理主控台

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在主控台的導覽窗格中，選擇根存取權管理，然後選取啟用。

> **Note**
>
> 如果您看到根存取管理已停用，請啟用 AWS Identity and Access Management 中的受信任存取 AWS Organizations。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的 AWS IAM 和 AWS Organizations。

3. 在「要啟用的功能」區段中，選擇要啟用的功能。
   - 選取根憑證管理，以允許管理帳戶和 IAM 的委派管理員刪除成員帳戶的根使用者憑證。您必須啟用「成員帳戶中的特權根動作」，以允許成員帳戶在根使用者憑證刪除後復原其根使用者憑證。

- 選取成員帳戶中的特權根動作，以允許管理帳戶和 IAM 的委派管理員執行需要根使用者憑證的特定任務。

4. (選用) 輸入授權管理根使用者存取權並對成員帳戶採取特權動作的委派管理員的帳戶 ID。我們建議使用用於安全或管理目的的帳戶。
5. 選擇啟用。

啟用集中式根存取權 (AWS CLI)

從 AWS Command Line Interface (AWS CLI) 啟用集中式根存取

1. 如果您尚未在 AWS Identity and Access Management 中啟用 的信任存取 AWS Organizations，請使用下列命令：aws organizations enable-aws-service-access。
2. 使用下列命令，允許管理帳戶和委派管理員刪除成員帳戶的根使用者憑證：aws iam enable-organizations-root-credentials-management。
3. 使用下列命令，允許管理帳戶和委派管理員執行需要根使用者憑證的特定任務：aws iam enable-organizations-root-sessions。
4. (選用) 使用下列命令來註冊委派管理員：aws organizations register-delegated-administrator。

   下列範例會將帳戶 111111111111 指派為 IAM 服務的委派管理員。

   ```
   aws organizations register-delegated-administrator
   --service-principal iam.amazonaws.com
   --account-id 111111111111
   ```

啟用集中式根存取 (AWS API)

從 AWS API 啟用集中式根存取

1. 如果您尚未在 AWS Identity and Access Management 中啟用 的受信任存取 AWS Organizations，請使用下列命令：EnableAWSServiceAccess。
2. 使用下列命令，允許管理帳戶和委派管理員刪除成員帳戶的根使用者憑證：EnableOrganizationsRootCredentialsManagement。
3. 使用下列命令，允許管理帳戶和委派管理員執行需要根使用者憑證的特定任務：EnableOrganizationsRootSessions。
4. (選用) 使用下列命令來註冊委派管理員：RegisterDelegatedAdministrator。

後續步驟

集中保護組織中成員帳戶的特權憑證後，請參閱執行特權任務以對成員帳戶採取特權動作。

在 AWS Organizations 成員帳戶上執行特權任務

IAM 的 AWS Organizations 管理帳戶或委派管理員帳戶可以對成員帳戶執行一些特殊權限任務，否則需要根使用者憑證。透過集中式根存取，這些任務會透過短期特權工作階段執行。這些工作階段提供範圍限定於特定特殊權限動作的臨時登入資料，而不需要成員帳戶的根使用者登入。

啟動特權工作階段後，您可以刪除設定錯誤的 Amazon S3 儲存貯體政策、刪除設定錯誤的 Amazon SQS 佇列政策、刪除成員帳戶的根使用者憑證，以及為成員帳戶重新啟用根使用者憑證。

> **Note**
>
> 若要使用集中式根存取，您必須透過管理帳戶或委派管理員帳戶以明確授予`sts:AssumeRoot`許可的 IAM 使用者或角色身分登入。您無法使用根使用者登入資料來呼叫 `sts:AssumeRoot`。

先決條件

您必須先具有下列設定，然後才能啟動特權工作階段：

- 您已在組織中啟用集中式根存取權。如需啟用此功能的步驟，請參閱集中成員帳戶的根存取權。
- 您的管理帳戶或委派管理員帳戶具有下列許可：`sts:AssumeRoot`

對成員帳戶執行特權動作 (主控台)

在 中為成員帳戶中的特權動作啟動工作階段 AWS 管理主控台

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在主控台的導覽窗格中，選擇根存取權管理。
3. 從成員帳戶清單中選取名稱，然後選擇執行特權動作。
4. 選擇您要在成員帳戶中執行的特權動作。
   - 選取刪除 Amazon S3 儲存貯體政策，以移除設定錯誤的儲存貯體政策，此政策拒絕所有主體存取 Amazon S3 儲存貯體。

a. 選擇瀏覽 S3 以從成員帳戶擁有的儲存貯體中選取名稱，然後選取選擇。

b. 選擇刪除儲存貯體政策。

c. 刪除設定錯誤的政策後，使用 Amazon S3 主控台來更正儲存貯體政策。如需詳細資訊，請參閱 Amazon S3 User Guide 中的 Adding a bucket policy by using the Amazon S3 console。

- 選取刪除 Amazon SQS 政策，以刪除拒絕所有主體存取 Amazon SQS 佇列的 Amazon Simple Queue Service 資源型政策。

a. 在 SQS 佇列名稱中輸入佇列名稱，然後選取刪除 SQS 政策。

b. 刪除設定錯誤的政策後，使用 Amazon SQS 主控台來更正佇列政策。如需詳細資訊，請參閱《Amazon SQS 開發人員指南》中的在 Amazon SQS 中設定存取政策。

- 選取刪除根憑證，以從成員帳戶移除根存取權。刪除根使用者憑證會移除成員帳戶的根使用者密碼、存取金鑰、簽署憑證和停用多重要素驗證 (MFA)。

  - 選擇刪除根憑證。

- 選取允許密碼復原，以復原成員帳戶的根使用者憑證。

  只有在成員帳戶沒有根使用者憑證時，才能使用此選項。

a. 選擇允許密碼復原。

b. 執行此特權動作後，有權存取成員帳戶根使用者電子郵件收件匣的人員可以重設根使用者密碼，並登入成員帳戶根使用者。

對成員帳戶執行特權動作 (AWS CLI)

從 為成員帳戶中的特權動作啟動工作階段 AWS Command Line Interface

1. 使用下列命令來擔任根使用者工作階段：aws sts assume-root。

> **Note**
>
> `sts:AssumeRoot` 不支援全域端點。您必須將此請求傳送至區域 AWS STS 端點。如需詳細資訊，請參閱在 AWS STS 中管理 AWS 區域。

為成員帳戶啟動特權根使用者工作階段時，您必須定義 `task-policy-arn`，以將工作階段範圍限制為工作階段期間要執行的特權動作。您可以使用下列其中一個 AWS 受管政策，來限制特權工作階段動作的範圍。

- IAMAuditRootUserCredentials
- IAMCreateRootUserPassword
- IAMDeleteRootUserCredentials
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

若要限制管理帳戶或委派管理員在特權根使用者工作階段期間可執行的動作，您可以使用 AWS STS 條件金鑰 sts：TaskPolicyArn。

在下列範例中，委派管理員假設以根使用者身分刪除成員帳戶 ID *111122223333* 的根使用者憑證。

```
aws sts assume-root \
  --target-principal 111122223333 \
  --task-policy-arn arn=arn:aws:iam::aws:policy/root-task/
IAMDeleteRootUserCredentials \
  --duration-seconds 900
```

2. 使用回應中的 `AccessKeyId`、`SessionToken` 和 `SecretAccessKey`，在成員帳戶中執行特權動作。您可以省略請求中的使用者名稱和密碼，以預設為成員帳戶。
   - 檢查根使用者憑證的狀態。使用下列命令，來檢查成員帳戶的根使用者憑證狀態。
     - get-user
     - get-login-profile
     - list-access-keys
     - list-signing-certificates
     - list-mfa-devices
     - get-access-key-last-used
   - 刪除根使用者憑證。使用下列命令刪除根存取權。您可以移除根使用者密碼、存取金鑰、簽署憑證，以及停用多重要素驗證 (MFA)，以移除對根使用者的所有存取和復原。

    - delete-login-profile
    - delete-access-key
    - delete-signing-certificate
    - deactivate-mfa-device
- 刪除 Amazon S3 儲存貯體政策。使用下列命令來讀取、編輯和刪除設定錯誤的儲存貯體政策，此政策拒絕所有主體存取 Amazon S3 儲存貯體。
    - list-buckets
    - get-bucket-policy
    - put-bucket-policy
    - delete-bucket-policy
- 刪除 Amazon SQS 政策。使用下列命令，來檢視和刪除拒絕所有主體存取 Amazon SQS 佇列的 Amazon Simple Queue Service 資源型政策。
    - list-queues
    - get-queue-url
    - get-queue-attributes
    - set-queue-attributes
- 允許密碼復原。使用下列命令來檢視使用者名稱，並復原成員帳戶的根使用者憑證。
    - get-login-profile
    - create-login-profile

對成員帳戶 (AWS API) 採取特權動作

從 AWS API 為成員帳戶中的特權動作啟動工作階段

1. 使用下列命令來擔任根使用者工作階段：AssumeRoot。

> **Note**
>
> AssumeRoot 不支援全域端點。您必須將此請求傳送至區域 AWS STS 端點。如需詳細資訊，請參閱在 AWS STS 中管理 AWS 區域。

為成員帳戶啟動特權根使用者工作階段時，您必須定義 TaskPolicyArn，以將工作階段範圍限制為工作階段期間要執行的特權動作。您可以使用下列其中一個 AWS 受管政策來限制特權工作階段動作的範圍。

- IAMAuditRootUserCredentials
- IAMCreateRootUserPassword
- IAMDeleteRootUserCredentials
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

若要限制管理帳戶或委派管理員在特權根使用者工作階段期間可執行的動作，您可以使用 AWS STS 條件金鑰 sts：TaskPolicyArn。

在下列範例中，委派管理員擔任根使用者身分，以讀取、編輯和刪除成員帳號 ID *111122223333* 的 Amazon S3 儲存貯體之設定錯誤的資源型政策。

```
https://sts.us-east-2.amazonaws.com/
  ?Version=2011-06-15
  &Action=AssumeRoot
  &TargetPrincipal=111122223333
  &PolicyArns.arn=arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy
  &DurationSeconds=900
```

2. 使用回應中的 AccessKeyId、SessionToken 和 SecretAccessKey，在成員帳戶中執行特權動作。您可以省略請求中的使用者名稱和密碼，以預設為成員帳戶。
   - 檢查根使用者憑證的狀態。使用下列命令，來檢查成員帳戶的根使用者憑證狀態。
     - GetUser
     - GetLoginProfile
     - ListAccessKeys
     - ListSigningCertificates
     - ListMFADevices
     - GetAccessKeyLastUsed
   - 刪除根使用者憑證。使用下列命令刪除根存取權。您可以移除根使用者密碼、存取金鑰、簽署憑證，以及停用多重要素驗證 (MFA)，以移除對根使用者的所有存取和復原。

    - DeleteLoginProfile
    - DeleteAccessKey
    - DeleteSigningCertificate
    - DeactivateMfaDevice
- 刪除 Amazon S3 儲存貯體政策。使用下列命令來讀取、編輯和刪除設定錯誤的儲存貯體政策，此政策拒絕所有主體存取 Amazon S3 儲存貯體。
    - ListBuckets
    - GetBucketPolicy
    - PutBucketPolicy
    - DeleteBucketPolicy
- 刪除 Amazon SQS 政策。使用下列命令，來檢視和刪除拒絕所有主體存取 Amazon SQS 佇列的 Amazon Simple Queue Service 資源型政策。
    - ListQueues
    - GetQueueUrl
    - GetQueueAttributes
    - SetQueueAttributes
- 允許密碼復原。使用下列命令來檢視使用者名稱，並復原成員帳戶的根使用者憑證。
    - GetLoginProfile
    - CreateLoginProfile

的多重要素驗證 AWS 帳戶根使用者

> **Important**
>
> AWS 建議您盡可能使用 MFA 的通行金鑰或安全金鑰 AWS，因為它們更能抵禦網路釣魚等攻擊。如需詳細資訊，請參閱通行密鑰和安全金鑰。

多重要素驗證 (MFA) 是一種簡單且有效的機制，可增強您的安全性。第一個因素：您的密碼，是您記住的秘密，也稱為知識因素。其他因素可以是擁有因素 (您擁有的事物，例如安全金鑰) 或繼承因素 (您

自身的事物，例如生物特徵掃描)。為了提高安全性，強烈建議您設定多重要素驗證 (MFA)，以協助保護您的 AWS 資源。

> **Note**
>
> 所有 AWS 帳戶 類型 （ 獨立、管理和成員帳戶 ） 都需要為其根使用者設定 MFA。 AWS 管理主控台 如果尚未啟用 MFA，使用者必須在第一次登入嘗試存取 的 35 天內註冊 MFA。

您可以為 AWS 帳戶根使用者 和 IAM 使用者啟用 MFA。當您為根使用者啟用 MFA 時，它僅影響根使用者憑證。如需如何針對 IAM 使用者啟用 MFA 的詳細資訊，請參閱 AWS IAM 中的多重要素驗證。

> **Note**
>
> AWS 帳戶 使用 受管 AWS Organizations 可能可以選擇集中管理成員帳戶的根存取權，以防止憑證復原和大規模存取。如果啟用了此選項，您可以從成員帳戶中刪除根使用者憑證，包括密碼和 MFA，從而有效地防止以根使用者身分登入、密碼復原或設定 MFA。或者，如果您偏好使用基於密碼的登入方式，請註冊 MFA 以增強帳戶保護來確保帳戶安全。

啟用根使用者的 MFA 之前，請檢閱並更新您的帳戶設定和聯絡資訊，以確保您有權存取電子郵件和電話號碼。如果您的 MFA 裝置遺失、遭竊或無法運作，您仍然可以使用該電子郵件和電話號碼驗證身分，以根使用者身分登入。如需了解如何使用其他身分驗證方法登入的詳細資訊，請參閱在 IAM 中復原受 MFA 保護的身分。若要停用這項功能，請聯絡 AWS 支援。

AWS 支援根使用者的下列 MFA 類型：

- 通行密鑰和安全金鑰
- 虛擬驗證器應用程式
- 硬體 TOTP 權杖

通行密鑰和安全金鑰

AWS Identity and Access Management 支援 MFA 的通行金鑰和安全金鑰。根據 FIDO 標準，通行金鑰使用公有金鑰密碼編譯來提供比密碼更安全的強大、網路釣魚防護身分驗證。 AWS 支援兩種類型的通行金鑰：裝置繫結的通行金鑰 （ 安全金鑰 ） 和同步的通行金鑰。

- 安全金鑰：這些是用作身分驗證的第二個因素的實體裝置，例如 YubiKey。單一安全金鑰可以支援多個根使用者帳戶和 IAM 使用者。
- 同步通行密鑰：這些使用來自提供者 (例如 Google、Apple、Microsoft 帳戶等) 和第三方服務 (例如 1Password、Dashlane 和 Bitwarden 等) 的憑證管理工具作為第二個因素。

您可以使用內建的生物識別驗證器，例如 Apple MacBooks 上的 Touch ID，解鎖您的憑證管理工具並登入 AWS。通行密鑰是透過您選擇的提供者，使用指紋、面部或裝置 PIN 碼建立的。您也可以使用一台裝置 (例如行動裝置或硬體安全金鑰) 中的跨帳戶身分驗證 (CDA) 通行密鑰，在筆記型電腦等其他裝置上登入。如需詳細資訊，請參閱 cross-device authentication (CDA)。

您可以跨裝置同步通行金鑰，以便使用 登入 AWS，增強可用性和可復原性。如需啟用通行密鑰和安全金鑰的詳細資訊，請參閱為根使用者啟用通行密鑰或安全金鑰 (主控台)。

FIDO Alliance 維護與 FIDO 規範相容的所有經 FIDO 認證的產品的清單。

虛擬驗證器應用程式

在手機或其他裝置上執行並模擬實體裝置的虛擬驗證器應用程式。虛擬驗證器應用程式實作以時間為基礎的一次性密碼 (TOTP) 算法，並且支援在單台裝置上使用多個權杖。使用者必須在登入期間出現提示時輸入裝置中的有效代碼。每個指派給使用者的權杖都必須是唯一的。使用者無法輸入另一個使用者的權杖來進行身分驗證。

我們強烈建議您在等待硬體的購買核准或等待硬體就定位時，使用虛擬 MFA 裝置。如需可以用來做為虛擬 MFA 裝置的支援應用程式清單，請參閱多重要素驗證 (MFA)。如需使用 設定虛擬 MFA 裝置的指示 AWS，請參閱 針對根使用者啟用虛擬 MFA 裝置 (主控台)。

硬體 TOTP 權杖

一種在以時間為基礎的一次性密碼 (TOTP) 演算法的基礎上產生六位數字程式碼的硬體裝置。使用者必須在登入期間在第二個網頁上輸入裝置中的有效程式碼。每個指派給使用者的 MFA 裝置都必須是唯一的。使用者無法輸入另一個使用者裝置的代碼來進行身分驗證。如需支援的硬體 MFA 裝置資訊，請參閱多重要素驗證 (MFA)。如需使用 AWS設定硬體 TOTP 權杖的說明，請參閱 針對根使用者啟用硬體 TOTP 權杖 (主控台)。

如果想要使用實體 MFA 裝置，我們建議您使用 FIDO 安全金鑰作為硬體 TOTP 裝置的替代方案。FIDO 安全金鑰具有無需電池和防釣魚的優勢，而且可在單一裝置上支援多個根使用者和 IAM 使用者，以增強安全性。

主題

- 為根使用者啟用通行密鑰或安全金鑰 (主控台)
- 針對根使用者啟用虛擬 MFA 裝置 (主控台)
- 針對根使用者啟用硬體 TOTP 權杖 (主控台)

為根使用者啟用通行密鑰或安全金鑰 (主控台)

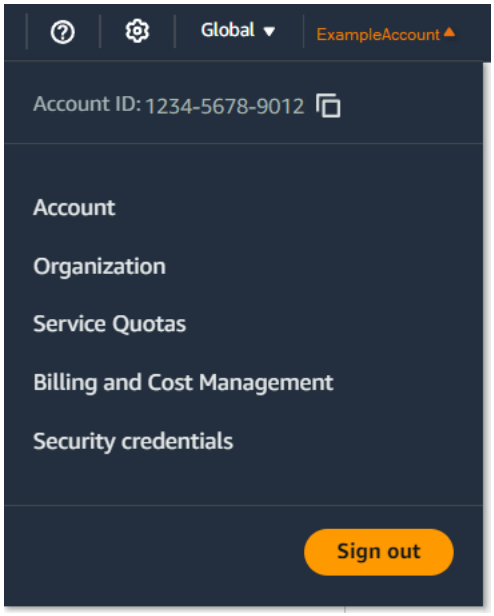
您只能從 為根使用者設定和啟用通行金鑰 AWS 管理主控台 ，而不是從 AWS CLI 或 AWS API。

若要為您的根使用者啟用通行密鑰或安全金鑰 (主控台)

1. 開啟 AWS 管理主控台，並使用您的根使用者憑證登入。

   如需說明，請參閱AWS 登入 《 使用者指南》中的以根使用者的 AWS 管理主控台 身分登入 。

2. 在導覽列右側選擇您的帳戶名稱，然後選擇 安全憑證 。



3. 在根使用者我的安全憑證頁面的多重要素驗證 (MFA) 下，選擇指派 MFA 裝置。
4. 在 MFA 裝置名稱頁面上，輸入裝置名稱，選擇通行密鑰或安全金鑰，然後選擇下一步。
5. 在設定裝置上，設定您的通行密鑰。使用臉部或指紋等生物識別資料、裝置 PIN，或將 FIDO 安全金鑰插入電腦的 USB 連接埠並點選，由此建立通行密鑰。
6. 遵循瀏覽器上的指示，選擇通行密鑰提供者或您要存放通行密鑰的位置，以便跨裝置使用。
7. 選擇繼續。

您現在已註冊要與 搭配使用的通行密鑰 AWS。下一次當您使用根使用者憑證登入時，您必須使用通行密鑰進行驗證，才能完成登入程序。

如需針對 FIDO 安全金鑰問題進行疑難排解的協助，請參閱對通行密鑰和 FIDO 安全金鑰進行疑難排解。

針對根使用者啟用虛擬 MFA 裝置 (主控台)

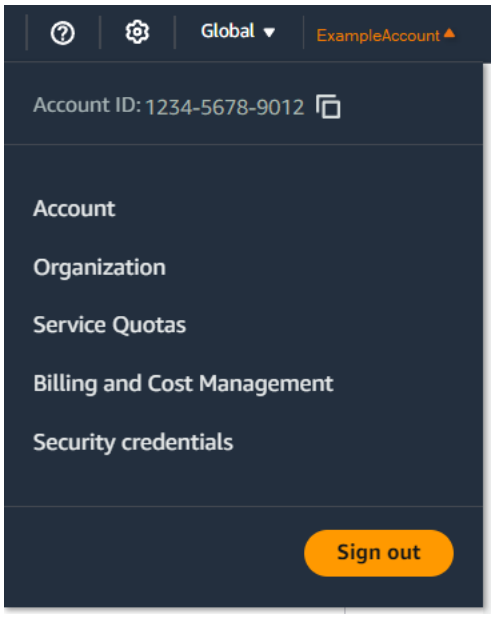
您可以使用 AWS 管理主控台 為您的根使用者設定和啟用虛擬 MFA 裝置。若要為 啟用 MFA 裝置 AWS 帳戶，您必須 AWS 使用根使用者憑證登入 。

設定和啟用虛擬 MFA 裝置以與根使用者一起使用 (主控台)

1. 開啟 AWS 管理主控台，並使用您的根使用者憑證登入。

   如需說明，請參閱AWS 登入 《 使用者指南》中的以根使用者的 AWS 管理主控台 身分登入 。

2. 在導覽列右側，選擇您的帳戶名稱，然後選擇 安全憑證 。



3. 在 Multi-Factor Authentication (MFA) (多重要素驗證 (MFA)) 區段中，選擇 Assign MFA device (指派 MFA 裝置)。
4. 在精靈中輸入 Device name，然後選擇 驗證器應用程式，再選擇 下一步。

   IAM 將產生並顯示虛擬 MFA 裝置的配置資訊，包括 QR 碼圖形。此圖形是私密組態金鑰的表示形式，適用於不支援 QR 代碼的裝置上的手動輸入。

5. 在裝置上開啟虛擬 MFA 應用程式。

如果虛擬 MFA 應用程式支援多個虛擬 MFA 裝置或帳戶，請選擇對應的選項以建立新的虛擬 MFA 裝置或帳戶。

6. 設定應用程式的最簡單方法是使用該應用程式掃描 QR 條碼。如果您無法掃描代碼，則可以手動輸入組態資訊。IAM 產生的 QR 程式碼和私密組態金鑰會繫結至您的 AWS 帳戶 ，無法與不同的帳戶搭配使用。但是，如果您無法存取原始 MFA 裝置，則他們可以重複使用，以便為您的帳戶設定新的 MFA 裝置。

   - 若要使用 QR 碼設定虛擬 MFA 裝置，請從精靈中選擇 Show QR code (顯示 QR 碼)。然後，遵循應用程式說明來掃描代碼。例如，您可能需要選擇相機圖示，或選擇與 Scan account barcode (掃描帳戶代碼) 相似的命令，然後使用裝置的相機掃描 QR 碼。
   - 在 Set up device (設定裝置) 精靈中，選擇 Show secret key (顯示私密金鑰)，然後在您的 MFA 應用程式中輸入私密金鑰。

   > **Important**
   >
   > 對 QR 條碼或私密組態金鑰進行安全備份，或確保為您的帳戶啟用多台 MFA 裝置。您可以向 AWS 帳戶根使用者 和 IAM 使用者註冊最多八個目前支援 MFA 類型之任何組合的 MFA 裝置。虛擬 MFA 裝置可能無法使用，例如，如果您遺失託管在虛擬 MFA 裝置的智慧型手機。如果發生這種情況且您沒有連接到使用者的其他 MFA 裝置，甚至無法透過 復原根使用者 MFA 裝置 登入您的帳戶，您將無法登入您的帳戶，您必須聯絡客戶服務以移除該帳戶的 MFA 保護。

   裝置開始產生六位數的號碼。

7. 在精靈中的 MFA code 1 (MFA 代碼 1) 方塊內，輸入虛擬 MFA 裝置上目前顯示的一次性密碼。請等待 30 秒，裝置將產生新的一次性密碼。然後將第二個一次性密碼輸入 MFA code 2 (MFA 代碼 2) 方塊中。選擇 Add MFA (新增 MFA)。

   > **Important**
   >
   > 產生代碼之後立即提交您的請求。如果在產生代碼後等待很長時間才提交請求，MFA 裝置會成功地與使用者建立關聯，但 MFA 裝置不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。這種情況下，您可以重新同步裝置。

裝置已準備好與 搭配使用 AWS。如需與 AWS 管理主控台一起使用 MFA 的詳細資訊，請參閱 啟用 MFA 的登入。

針對根使用者啟用硬體 TOTP 權杖 (主控台)

您只能從 AWS 管理主控台 為根使用者設定和啟用實體 MFA 裝置，而不是從 AWS CLI 或 AWS API。

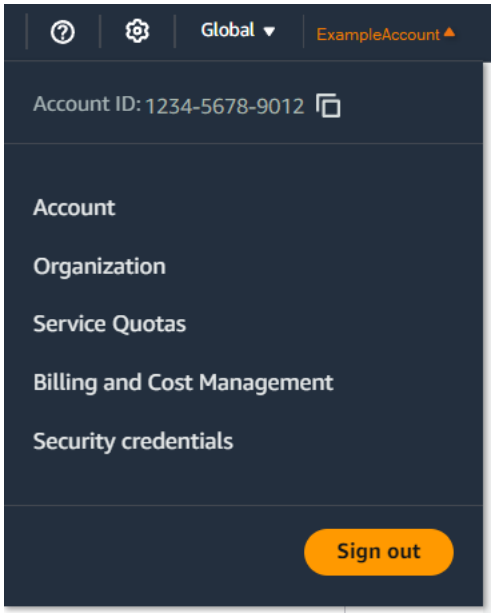
> **Note**
>
> 您可能會看到不同的文字，例如使用 MFA 登入和對您的身分驗證裝置進行疑難排解。不過，其功能是相同的。在任一情況下，若無法使用其他身分驗證要素來驗證您的帳戶電子郵件地址和電話號碼，請聯絡 AWS 支援刪除您的 MFA 設定。

若要針對您的根使用者啟用硬體 TOTP 權杖 (主控台)

1. 開啟 AWS 管理主控台，並使用您的根使用者憑證登入。

   如需說明，請參閱AWS 登入 《 使用者指南》中的以根使用者的 AWS 管理主控台 身分登入 。

2. 在導覽列右側選擇您的帳戶名稱，然後選擇 安全憑證 。



3. 展開 Multi-factor authentication (MFA) (多重要素驗證 (MFA)) 區段。
4. 選擇 Assign MFA device (指派 MFA 裝置)。
5. 在精靈中，輸入一個裝置名稱，然後選取 Hardware TOTP token (硬體 TOTP 權杖)，再選擇 Next (下一步)。

6. 在 Serial number (序號) 方塊中，輸入在 MFA 裝置背面找到的序號。
7. 在 MFA code 1 (MFA 代碼 1) 方塊中，輸入 MFA 裝置顯示的六位數字。您可能需要按下裝置正面的按鈕以顯示數字。



8. 當裝置在重新整理代碼時，等候 30 秒時間後，在 MFA code 2 (MFA 代碼 2) 方塊中輸入六位數字。您可能需要再次按下裝置正面的按鈕以顯示第二個數字。
9. 選擇 Add MFA (新增 MFA)。現在，MFA 裝置已與 AWS 帳戶建立關聯。

> **Important**
>
> 產生驗證代碼之後立即提交您的請求。如果在產生代碼後等待很長時間才提交請求，MFA 裝置會成功地與使用者建立關聯，但 MFA 裝置卻變成不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。這種情況下，您可以重新同步裝置。

下次使用根使用者憑證登入時，您必須輸入 MFA 裝置的代碼。

變更 的密碼 AWS 帳戶根使用者

您可以在安全憑證或帳戶頁面上變更電子郵件地址和密碼。您也可以選擇忘記密碼？ 在 AWS 登入頁面上重設您的密碼。

若要變更根使用者的密碼，您必須以 身分登入 AWS 帳戶根使用者 ，而不是以 IAM 使用者身分登入。若要了解如何重設忘記的根使用者密碼，請參閱 重設遺失或忘記的根使用者密碼。

為了保護您的密碼，請務必遵循這些最佳實務：

- 定期變更您的密碼。
- 請保持您密碼的私密性，因為任何獲知您的密碼的人員都能存取您的帳戶。
- 在 上使用 AWS 不同於您在其他網站上使用的密碼。
- 避免使用容易猜中的密碼。這些密碼包含，例如 `secret`、`password`、`amazon` 或 `123456`。另外還要避免使用字典字詞、您的名字、電子郵件地址或其他可以輕鬆取得的個人資訊。

> **Important**
>
> AWS 帳戶 使用 受管 AWS Organizations 可能已啟用成員帳戶的集中式根存取。這些成員帳戶沒有根使用者憑證，無法以根使用者身分登入，而且無法復原根使用者密碼。如果需要執行要求根使用者憑證的任務，請聯絡您的管理員。

AWS 管理主控台

變更根使用者的密碼

> **最低許可**
>
> 若要執行下列步驟，您至少必須擁有下列 IAM 許可：
>
> - 您必須以 AWS 帳戶 根使用者身分登入，這不需要額外的 AWS Identity and Access Management (IAM) 許可。您無法以 IAM 使用者或角色的身分執行這些步驟。

1. 開啟 AWS 管理主控台，並使用您的根使用者憑證登入。

   如需說明，請參閱AWS 登入 《 使用者指南》中的以根使用者的 AWS 管理主控台 身分登入 。
2. 在主控台的右上角，選擇帳戶名稱或號碼，然後選取安全憑證。
3. 在帳戶頁面的帳戶設定旁，選擇編輯。出於安全目的，系統會提示您再次進行身分驗證。

   > **Note**
   >
   > 如果您看不到編輯選項，則可能是您未以帳戶根使用者的身分登入。在以 IAM 使用者或角色的身分登入時，您無法修改帳戶設定。
4. 在更新帳戶設定頁面上，選擇密碼下方的編輯。
5. 在更新您的密碼頁面上，填寫目前的密碼、新密碼和確認新密碼欄位。

   > **Important**
   >
   > 確保選擇強式密碼。雖然您可以設定 IAM 使用者的帳戶密碼政策，但該政策不適用於根使用者。

AWS 需要您的密碼符合下列條件：

- 它必須至少有 8 個字元，最多 128 個字元。
- 它至少混用 3 種下列類型字元：大寫、小寫、數字和 ! @ # $ % ^ & * () <> [] {} | _+-= 符號。
- 它不能與 AWS 帳戶 您的姓名或電子郵件地址相同。

6. 選擇儲存變更。

AWS CLI or AWS SDK

中 AWS CLI 或其中一個 AWS SDKs 的 API 操作不支援此任務。您只能使用 來執行此任務 AWS 管理主控台。

重設遺失或忘記的根使用者密碼

當您第一次建立 時 AWS 帳戶，您提供了電子郵件地址和密碼。這些是您的 AWS 帳戶根使用者 登入資料。若您忘記根使用者密碼，可從 AWS 管理主控台重設密碼。

AWS 帳戶 使用 受管 AWS Organizations 可能已啟用成員帳戶的集中式根存取。這些成員帳戶沒有根使用者憑證，無法以根使用者身分登入，而且無法復原根使用者密碼。如果需要執行要求根使用者憑證的任務，請聯絡您的管理員。

> **Important**
>
> 無法登入 AWS？ 請確定您位在使用者類型的正確 AWS 登入頁面。如果您是 AWS 帳戶根使用者（帳戶擁有者），您可以使用您在建立 時設定的登入資料登入 AWS AWS 帳戶。如果您是 IAM 使用者，您的帳戶管理員可以為您提供可用來登入 AWS的憑證。如果您需要請求支援，請勿使用此頁面上的意見回饋連結，因為表單是由 AWS 文件團隊接收，而不是 支援。相反地，在聯絡我們頁面上，選擇仍然無法登入 AWS 您的帳戶，然後選擇其中一個可用的支援選項。

若要重設您的根使用者密碼

1. 開啟 AWS 管理主控台，並使用您的根使用者憑證登入。

   如需說明，請參閱AWS 登入 《 使用者指南》中的以根使用者的 AWS 管理主控台 身分登入 。

> **Note**
>
> 如果您是使用 IAM 使用者憑證登入 AWS 管理主控台，則必須先登出再重設根使用者密碼。如果您看到帳戶專屬的 IAM 使用者登入頁面，請選擇頁面底部旁的 Sign-in using root account credentials (使用根帳戶憑證來登入)。如有需要，請提供您的帳戶電子郵件地址，然後選擇 Next (下一步)，以存取 Root user sign in (根使用者登入) 頁面。

2. 選擇 Forgot your password? (忘記您的密碼？)。

> **Note**
>
> 如果您是 IAM 使用者，則此選項無法使用。忘記您的密碼？選項僅適用於根使用者帳戶。IAM 使用者必須要求管理員重設已忘記的密碼。如需詳細資訊，請參閱我忘記 AWS 帳戶的 IAM 使用者密碼。如果您透過 AWS 存取入口網站登入，請參閱重設 IAM Identity Center 使用者密碼。

3. 提供與帳戶相關聯的電子郵件地址。然後提供 CAPTCHA 文字並選擇 Continue (繼續)。
4. 檢查與您的 相關聯的電子郵件 AWS 帳戶 是否有來自 Amazon Web Services 的訊息。電子郵件來自於結尾為 `@verify.signin.aws` 的地址。請遵循電子郵件中的指示進行。若您在帳戶中沒有看到電子郵件，請檢查您的垃圾郵件資料夾。如果您不再能夠存取電子郵件，請參閱AWS 登入《使用者指南》中的我無法存取我 AWS 帳戶的電子郵件。

為根使用者建立存取金鑰

> **Warning**
>
> 我們強烈建議您不要為您的根使用者建立存取金鑰對。由於只有少數任務需要根使用者，而且您通常不常執行這些任務，因此我們建議您登入 AWS 管理主控台 以執行根使用者任務。在建立存取金鑰前，請檢閱長期存取金鑰的替代方案。

雖然我們不建議這麼做，但您可以為您的根使用者建立存取金鑰，以便您可以在 AWS Command Line Interface (AWS CLI) 中執行命令，或使用根使用者憑證從其中一個 AWS SDKs 使用 API 操作。當您建立存取金鑰時，可以將存取金鑰 ID 與私密存取金鑰建立為一組。在建立存取金鑰期間， AWS 可讓您有機會檢視和下載存取金鑰的私密存取金鑰部分。如果您沒有下載或者遺失私密存取金鑰，可以先刪除再建立新的存取金鑰。您可以使用 主控台 AWS CLI或 AWS API 建立根使用者存取金鑰。

新建立存取金鑰的狀態為「作用中」，這表示您可以使用存取金鑰進行 CLI 和 API 呼叫。您可以將最多兩個存取金鑰指派給根使用者。

未使用的存取金鑰應該停用。存取金鑰處於非作用中狀態後，您就無法將其用於 API 呼叫。非作用中金鑰仍會計入您的限制。您可以隨時建立或刪除存取金鑰。不過，刪除存取金鑰之後，即永久消失且無法擷取。

AWS 管理主控台

建立 的存取金鑰 AWS 帳戶根使用者

> **最低許可**
>
> 若要執行下列步驟，您至少必須擁有下列 IAM 許可：
>
> - 您必須以 AWS 帳戶 根使用者身分登入，這不需要額外的 AWS Identity and Access Management (IAM) 許可。您無法以 IAM 使用者或角色的身分執行這些步驟。

1. 開啟 AWS 管理主控台，並使用您的根使用者憑證登入。

   如需說明，請參閱AWS 登入 《 使用者指南》中的以根使用者的 AWS 管理主控台 身分登入 。
2. 在主控台的右上角，選擇您的帳戶名稱或號碼，然後選擇 安全憑證 。
3. 在存取金鑰區段中，選擇建立存取金鑰。如果此選項不可用，表示您所擁有的存取金鑰已達最大數目。您必須先刪除其中一個現有的存取金鑰，才能建立新的金鑰。如需詳細資訊，請參閱 IAM 物件配額。
4. 在根使用者存取金鑰的替代方案頁面上，檢閱安全建議。若要繼續，請選取核取方塊，然後選擇建立存取金鑰。
5. 在擷取存取金鑰頁面上，會顯示您的存取金鑰 ID。
6. 在私密存取金鑰下方，選擇顯示，然後從您的瀏覽器視窗複製存取金鑰 ID 和私密金鑰，將其貼到安全的地方。或者，您可以選擇下載 .csv 檔案，這樣做會下載一個名為 `rootkey.csv` 的檔案，其中包含存取金鑰 ID 和私密金鑰。將檔案儲存到安全的位置。
7. 選擇完成。當您不再需要存取金鑰時，我們建議您將其刪除，或者至少考慮停用該金鑰，以防止其他人濫用。

AWS CLI & SDKs

若要為根使用者建立存取金鑰

> **Note**
>
> 若要以根使用者身分執行下列命令或 API 操作，您必須已有一個作用中的存取金鑰對。如果您沒有任何存取金鑰，可使用 AWS 管理主控台建立第一個存取金鑰。然後，您可以將該第一個存取金鑰中的登入資料與 搭配使用 AWS CLI ，以建立第二個存取金鑰，或刪除存取金鑰。

- AWS CLI：aws iam create-access-key

  Example

```
$ aws iam create-access-key
{
    "AccessKey": {
        "UserName": "MyUserName",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2021-04-08T19:30:16+00:00"
    }
}
```

- AWS API：IAM API 參考中的 CreateAccessKey。

刪除根使用者的存取金鑰

您可以使用 AWS 管理主控台、 AWS CLI 或 AWS API 來刪除根使用者存取金鑰。

AWS 管理主控台

若要刪除根使用者的存取金鑰

> **最低許可**
>
> 若要執行下列步驟，您至少必須擁有下列 IAM 許可：

- 您必須以 AWS 帳戶 根使用者身分登入，這不需要額外的 AWS Identity and Access Management (IAM) 許可。您無法以 IAM 使用者或角色的身分執行這些步驟。

1. 開啟 AWS 管理主控台，並使用您的根使用者憑證登入。

   如需說明，請參閱AWS 登入 《 使用者指南》中的以根使用者的 AWS 管理主控台 身分登入 。
2. 在主控台的右上角，選擇您的帳戶名稱或號碼，然後選擇 安全憑證 。
3. 在存取金鑰區段中，選擇您想要刪除的存取金鑰，然後在動作下方選擇刪除。

   > **Note**
   >
   > 或者，您可以停用存取金鑰，而不是永久刪除它。這可讓您在未來繼續使用它，而無需變更金鑰 ID 或私密金鑰。當金鑰處於非作用中狀態時，任何在 AWS API 請求中使用金鑰的嘗試都會失敗，錯誤存取遭拒。
4. 在刪除 <access key ID> 對話方塊中，選擇停用，輸入存取金鑰 ID 以確認您想要刪除它，然後選擇刪除。

AWS CLI & SDKs

若要刪除根使用者的存取金鑰

> **最低許可**
>
> 若要執行下列步驟，您至少必須擁有下列 IAM 許可：
>
> - 您必須以 AWS 帳戶 根使用者身分登入，這不需要額外的 AWS Identity and Access Management (IAM) 許可。您無法以 IAM 使用者或角色的身分執行這些步驟。

- AWS CLI：aws iam delete-access-key

  Example

```
$ aws iam delete-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE
```

此命令成功後就不會產生輸出。

- AWS API：DeleteAccessKey

需要根使用者憑證的任務

建議您在 中設定管理使用者 AWS IAM Identity Center，以執行每日任務和存取 AWS 資源。不過，您可以只以帳戶根使用者身分登入來執行任務下面所列的任務。

為了簡化在 中跨成員帳戶管理特權根使用者憑證 AWS Organizations，您可以啟用集中式根存取，以協助您集中保護對 的高度特權存取 AWS 帳戶。 集中管理成員帳戶的根存取權可讓您集中移除和防止長期根使用者憑證復原，改善組織中的帳戶安全性。啟用此功能後，您可以對成員帳戶執行下列特權任務。

- 移除成員帳戶根使用者憑證，以防止根使用者的帳戶復原。您也可以允許密碼復原，來復原成員帳戶的根使用者憑證。
- 移除設定錯誤的儲存貯體政策，此政策拒絕所有主體存取 Amazon S3 儲存貯體。
- 刪除拒絕所有主體存取 Amazon SQS 佇列的 Amazon Simple Queue Service 資源型政策。

帳戶管理任務

- 變更 AWS 帳戶 設定。不屬於 AWS Organizations 的獨立 AWS 帳戶 需要根登入資料，才能更新電子郵件地址、根使用者密碼和根使用者存取金鑰。其他帳戶設定，例如帳戶名稱、聯絡資訊、替代聯絡人、付款貨幣偏好設定 AWS 區域，以及不需要根使用者憑證。

  > **Note**
  >
  > AWS Organizations啟用所有功能之後，可用於從管理帳戶和委派管理員帳戶集中管理成員帳戶設定。管理帳戶和委派管理員帳戶中的授權 IAM 使用者或 IAM 角色可以關閉成員帳戶，並更新成員帳戶的根電子郵件地址、帳戶名稱、聯絡資訊、替代聯絡人和 AWS 區域 。

- 關閉您的 AWS 帳戶。不屬於 的獨立 AWS 帳戶 AWS Organizations 需要根登入資料才能關閉帳戶。使用 AWS Organizations，您可以從管理帳戶和委派的管理員帳戶集中關閉成員帳戶。
- 重新開啟您的 AWS 帳戶。如果您自行關閉獨立帳戶，但沒有未結餘額，您可以在關閉後期間重新開啟該帳戶。您必須以根使用者身分登入才能執行此操作。
- 還原 IAM 使用者許可。如果唯一的 IAM 管理員不小心撤銷自己的許可，您可以根使用者的身分登入，即可編輯政策和還原這些許可。

帳單任務

- 啟動對帳單與成本管理主控台的 IAM 存取。
- 一些帳單任務僅限於根使用者。如需詳細資訊，請參閱 AWS Billing 《使用者指南》中的管理 AWS 帳戶 。
- 檢視特定稅務發票。具有 aws-portal：ViewBilling 許可的 IAM 使用者可以從 AWS 歐洲檢視和下載增值稅發票，但不能從 AWS Inc. 或 Amazon Internet Services Private Limited(AISPL) 下載增值稅發票。

AWS GovCloud (US) 任務

- 註冊 AWS GovCloud (US)。
- 從 請求 AWS GovCloud (US) 帳戶根使用者存取金鑰 AWS 支援。

Amazon EC2 任務

- 在預留執行個體市場註冊為賣方。

AWS KMS 任務

- 如果 AWS Key Management Service 金鑰變得無法管理，管理員可以透過聯絡 來復原金鑰 支援；不過， 會透過確認票證 OTP 來 支援 回應根使用者的主要電話號碼以進行授權。

Amazon Simple Storage Service 任務

- 設定 Amazon S3 儲存貯體，以啟用 MFA (多重因素認證)。
- 編輯或刪除拒絕所有主體的 Amazon S3 儲存貯體政策。

  您可以使用特權動作，來解除鎖定儲存貯體政策設定錯誤的 Amazon S3 儲存貯體。如需詳細資訊，請參閱在 AWS Organizations 成員帳戶上執行特權任務。

Amazon Simple Queue Service 任務

- 編輯或刪除拒絕所有主體的 Amazon SQS 資源型政策。

  您可以使用特權動作，來解除鎖定資源型政策設定錯誤的 Amazon SQS 佇列。如需詳細資訊，請參閱在 AWS Organizations 成員帳戶上執行特權任務。

其他資源

如需 AWS 根使用者的詳細資訊，請參閱下列資源：

- 如需根使用者問題的協助，請參閱 使用根使用者來疑難排解問題。
- 若要集中管理 中的根使用者電子郵件地址 AWS Organizations，請參閱AWS Organizations 《 使用者指南》中的更新成員帳戶的根使用者電子郵件地址。

下列文章將提供有關如何使用根使用者的更多資訊。

- 保護我 AWS 帳戶 及其資源的最佳實務有哪些？
- 如何建立 EventBridge 事件規則，以便在有人使用我的根使用者時通知我？
- 監控並通知 AWS 帳戶根使用者 活動
- 監控 IAM 根使用者活動

IAM 使用者

> **Important**
>
> IAM 最佳實務建議您要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 ，而不是使用具有長期憑證的 IAM 使用者。建議您僅將 IAM 使用者用於聯合身分使用者不支援的特定使用案例。
> 如果您正在使用我們的新 AWS 體驗，並希望讓人類使用者存取您的專案或 AWS 設定，則必須將他們新增至您的團隊。在我們的新體驗中，您無法建立具有主控台存取權的 IAM 使用者。如需詳細資訊，請參閱邀請團隊成員。

IAM 使用者是您在 AWS 帳戶中建立的實體。IAM 使用者表示使用 IAM 使用者與 AWS 資源互動的人類使用者或工作負載。IAM 使用者包含名稱和憑證。

具有管理員許可的 IAM 使用者與 AWS 帳戶根使用者並不相同。如需有關根使用者的詳細資訊，請參閱 AWS 帳戶根使用者。

如何 AWS 識別 IAM 使用者

當建立 IAM 使用者時，IAM 建立這些方法來識別該使用者：

- IAM 使用者的「易用名稱」，即您在建立 IAM 使用者時指定的名稱，例如 `Richard` 或 `Anaya`。這些是您在 AWS 管理主控台中看到的名稱。由於 IAM 使用者名稱會顯示在 Amazon Resource Name (ARN) 中，因此不建議在 IAM 名稱中包含個人識別資訊。如需 IAM 名稱的需求和限制，請參閱 IAM 名稱需求。
- IAM 使用者的 Amazon Resource Name (ARN)。當您需要唯一識別所有 的 IAM 使用者時，您可以使用 ARN AWS。例如，在 Amazon S3 儲存貯體的 IAM 政策中，您可以使用 ARN 將 IAM 使用者指定為 `Principal`。IAM 使用者的 ARN 可能如下所示：

  `arn:aws:iam::account-ID-without-hyphens:user/Richard`
- IAM 使用者的唯一識別碼。只有在您使用 API、Tools for Windows PowerShell 或 AWS CLI 建立 IAM 使用者時，才會傳回此 ID；您在 主控台中看不到此 ID。

如需有關這些識別碼的詳細資訊，請參閱 IAM 識別碼。

IAM 使用者和憑證

您可以根據 IAM 使用者登入資料 AWS ，以不同的方式存取 ：

- 主控台密碼：IAM 使用者可以輸入的密碼，以登入互動式工作階段 (如 AWS 管理主控台)。停用 IAM 使用者的密碼 （主控台存取） 可防止他們 AWS 管理主控台 使用登入憑證登入 。它不會變更其許可，也不會阻止他們使用擔任的角色來存取主控台。已啟用主控台存取的 IAM 使用者也可以使用這些相同的登入資料，使用 `aws login` AWS CLI 命令來驗證 AWS CLI 和 SDK 存取。這些使用者將需要具有 SignInLocalDevelopmentAccess 許可。如需詳細資訊，請參閱AWS Command Line Interface 《 使用者指南》中的 的身分驗證和存取憑證 AWS CLI。
- 存取金鑰：用來向 AWS發出程式化呼叫。但在為 IAM 使用者建立存取金鑰之前，您可以考慮其他更安全的替代方案。如需詳細資訊，請參閱 AWS 一般參考 中的長期存取金鑰的考量事項和替代方案。如果 IAM 使用者具有作用中的存取金鑰，他們會繼續運作並允許透過 AWS CLI、Tools for Windows PowerShell、 AWS API 或 AWS 主控台行動應用程式進行存取。
- 與 CodeCommit 一起使用的 SSH 金鑰：OpenSSH 格式的 SSH 公有金鑰，可用於透過 CodeCommit 驗證身分。
- 伺服器憑證：您可以使用 SSL/TLS 憑證來驗證某些 AWS 服務。建議您使用 AWS Certificate Manager (ACM) 來佈建、管理和部署伺服器憑證。只有當您必須在 ACM 不支援的區域中支援 HTTPS 連接時，才應使用 IAM。如需了解哪些區域支援 ACM，請參閱 AWS 一般參考 中的 AWS Certificate Manager 端點和配額。

您可以選擇最適合您 IAM 使用者的憑證。當您使用 AWS 管理主控台 來建立 IAM 使用者，您必須選擇至少包含一個主控台密碼或存取金鑰。根據預設，使用 AWS CLI 或 AWS API 建立的全新 IAM 使用者沒有任何類型的登入資料。您必須根據使用案例，建立 IAM 使用者的憑證類型。

您可以使用以下選項來管理密碼、存取金鑰和多重要素驗證 (MFA) 裝置：

- 管理 IAM 使用者的密碼。建立和變更允許存取 AWS 管理主控台的密碼。設定密碼政策，以強制執行最低密碼複雜性。允許 IAM 使用者變更自己的密碼。
- 管理 IAM 使用者的存取金鑰。建立和更新存取金鑰，以便以程式設計方式存取您帳戶中的資源。
- 為 IAM 使用者啟用多重要素驗證 (MFA)。根據最佳實務，我們建議您要求帳戶中的所有 IAM 使用者進行多重要素驗證。使用 MFA 時，IAM 使用者需要提供兩種身分驗證形式：首先，提供憑證，那是屬於他們的使用者身分 (密碼或存取金鑰) 的一部分。此外，他們提供硬體裝置或智慧型手機或平板電腦上的應用程式所產生的臨時數字代碼。
- 尋找未使用的密碼和存取金鑰。擁有您帳戶或帳戶中 IAM 使用者的密碼或存取金鑰的任何人都可以存取您的 AWS 資源。安全最佳實務是在 IAM 使用者不再需要密碼和存取金鑰時將其移除。
- 下載您帳戶的憑證報告。您可以產生並下載「憑證報告」，其中會列出帳戶中的所有 IAM 使用者，及其各種憑證的狀態，包括密碼、存取金鑰和 MFA 裝置。對於密碼和存取金鑰、憑證報告會顯示密碼或存取金鑰最近使用的狀況。

IAM 使用者和許可

在預設情況下，新的 IAM 使用者沒有許可採取任何行動。他們無權執行任何 AWS 操作或存取任何 AWS 資源。擁有個別 IAM 使用者的好處是您可以為每個使用者個別指派許可。您可以指派管理許可給幾個使用者，這些使用者接著可以管理您的 AWS 資源，甚至可以建立和管理其他 IAM 使用者。不過，在大多數情況下，您想要將使用者的許可限制為只有任務所需的任務 (AWS 動作或操作） 和資源。

試想一名叫做 Diego 的使用者。當您建立 IAM 使用者 `Diego` 時，您為該使用者建立一個密碼並連接許可，讓該使用者可以啟動特定 Amazon EC2 執行個體並從 Amazon RDS 資料庫的表格中讀取 (GET) 資訊。有關如何建立 IAM 使用者並授予其初始憑證和許可的程序的更多資訊，請參閱 在 中建立 IAM 使用者 AWS 帳戶。有關如何變更現有使用者的程序的詳細資訊，請參閱 變更 IAM 使用者的許可。有關如何變更使用者的密碼或存取金鑰的程序的詳細資訊，請參閱 中的使用者密碼 AWS 和 管理 IAM 使用者的存取金鑰。

您也可以新增許可界限到您的 IAM 使用者。許可界限是一項進階功能，可讓您使用 AWS 受管政策來限制身分型政策可授予 IAM 使用者或角色的最大許可。如需有關政策類型及其使用的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

IAM 使用者和帳戶

每位 IAM 使用者只能與一個 AWS 帳戶建立關聯。由於 IAM 使用者是在您的 中定義 AWS 帳戶，因此他們不需要在 的檔案中有付款方式 AWS。您帳戶中 IAM 使用者執行的任何 AWS 活動都會向您的帳戶收費。

AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

IAM 使用者做為服務帳戶

IAM 使用者是擁有關聯憑證和許可之 IAM 中的資源。IAM 使用者可以代表人員或應用程式，其會使用它的憑證來做出 AWS 請求。這通常稱為「服務帳戶」。如果您選擇在您的應用程式使用 IAM 使用者的長期憑證，請勿在應用程式程式碼中直接內嵌存取金鑰。 AWS SDKs和 AWS Command Line Interface 可讓您將存取金鑰放在已知位置，讓您不必將其保留在程式碼中。如需詳細資訊，請參閱 AWS 一般參考 中的適當管理 IAM 使用者存取金鑰。或者，最佳實務的作法是可以使用臨時安全憑證 (IAM 角色)，而不是長期存取金鑰。

建立 IAM 使用者之後，您可以重新命名 IAM 使用者視需要檢視 IAM 使用者在帳戶中和 。

IAM 使用者如何登入 AWS

若要以 IAM 使用者 AWS 管理主控台 身分登入 ，除了使用者名稱和密碼之外，您還必須提供您的帳戶 ID 或帳戶別名。當您的管理員在主控台中建立 IAM 使用者時，他們應已將您的登入憑證傳送給您，包括您的使用者名稱，以及包含您的帳戶 ID 或帳戶別名的帳戶登入頁面 URL。

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

> **秘訣**
>
> 要在網頁瀏覽器為您的帳戶登入頁面建立書籤，您應該在書籤項目手動輸入帳戶的登入 URL。請勿使用 web 瀏覽器的書籤功能，因為重新引導會模糊登入 URL。

您也可以在以下通用登入端點登入，並手動輸入您的帳戶 ID 或帳戶別名：

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

為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住 IAM 使用者名稱和帳戶資訊。下次使用者前往 中的任何頁面時 AWS 管理主控台，主控台會使用 Cookie 將使用者重新導向至帳戶登入頁面。

您只能存取管理員在連接到 IAM 使用者身分的政策中指定的 AWS 資源。若要在主控台中運作，您必須具有執行主控台所執行動作的許可，例如列出和建立 AWS 資源。如需詳細資訊，請參閱AWS 資源的存取管理及以身分為基礎的 IAM 政策範例。

> **Note**
>
> 如果您的組織已有現有的身分系統，您可能會想建立單一登入 (SSO) 選項。SSO 可讓使用者存取 AWS 管理主控台 您帳戶的 ，而無需擁有 IAM 使用者身分。SSO 也消除了使用者登入組織網站和 AWS 單獨登入 的需求。如需詳細資訊，請參閱啟用 AWS 主控台的自訂身分代理程式存取。

在 CloudTrail 中記錄登入詳細資訊

若您啟用 CloudTrail 將登入事件記錄到您的日誌，您需要注意 CloudTrail 如何選擇記錄事件日誌的地方。

- 如果您的使用者直接登入到主控台，則會根據所選服務主控台是否支援區域，將它們重新引導到全域登入或區域登入端點。例如，主要主控台首頁支援區域，所以您若登入以下 URL：

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

  您重新引導到區域登入端點，例如 `https://us-east-2.signin.aws.amazon.com`，從而導致在使用者的區域日誌中產生區域 CloudTrail 日誌項目：

  另一方面，Amazon S3 主控台不支援區域，所以您若登入以下 URL

  ```
  https://alias.signin.aws.amazon.com/console/s3
  ```

  AWS 會將您重新導向至 的全域登入端點`https://signin.aws.amazon.com`，導致全域 CloudTrail 日誌項目。
- 您可以手動請求特定區域登入端點，只要使用如下所示的 URL 語法，登入啟用區域的主要主控台首頁：

  ```
  https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
  ```

  AWS 會將您重新導向至`ap-southeast-1`區域登入端點，並導致區域 CloudTrail 日誌事件。

如需有關 CloudTrail 和 IAM 的詳細資訊，請參閱使用 CloudTrail 記錄 IAM 事件日誌。

如果使用者需要以程式設計存取的方式使用您的帳戶，則您可以為每位使用者建立存取金鑰對 (存取金鑰 ID 和私密存取金鑰)。但在為使用者建立存取金鑰之前，您可以考慮其他更安全的替代方案。如需詳細資訊，請參閱 AWS 一般參考 中的長期存取金鑰的考量事項和替代方案。

其他資源

下列資源可協助您進一步了解 AWS 登入。

- AWS Sign-In User Guide 可協助您了解根據使用者類型登入 Amazon Web Services (AWS) 的不同方式。
- 您可以在 AWS 管理主控台中的單一 Web 瀏覽器中同時登入最多五個不同的身分。如需詳細資訊，請參閱 AWS 管理主控台 Getting Started Guide 中的 Signing in to multiple accounts。
- 若要尋找登入所需的帳戶 ID 或別名，請參閱 檢視您的 AWS 帳戶 ID和 為您的 AWS 帳戶 ID 使用別名。

啟用 MFA 的登入

設定使用者的多重要素驗證 (MFA) 裝置必須使用自己的 MFA 裝置登入 AWS 管理主控台。使用者輸入登入憑證後， 會 AWS 檢查使用者帳戶，以查看該使用者是否需要 MFA。

> **Important**
>
> 如果您使用存取金鑰和私密金鑰登入資料透過 `GetFederationToken` API AWS STS 呼叫直接 AWS 管理主控台 存取，則不需要 MFA。如需詳細資訊，請參閱使用存取金鑰和秘密金鑰憑證存取主控台。

以下各主題提供使用者在需要 MFA 時完成登入方式的資訊。

主題

- 已啟用多個 MFA 裝置
- FIDO 安全性金鑰
- 虛擬 MFA 裝置
- 硬體 TOTP 權杖

已啟用多個 MFA 裝置

如果使用者以 AWS 帳戶 根使用者或 IAM 使用者 AWS 管理主控台 身分登入 ，且該帳戶已啟用多個 MFA 裝置，他們只需要使用一個 MFA 裝置登入。使用者使用其密碼進行身分驗證以後，他們要選取希望使用哪種 MFA 裝置類型來完成身分驗證。然後，使用者會收到使用其所選裝置類型進行身分驗證的提示。

FIDO 安全性金鑰

若使用者需要 MFA，則會出現第二個登入頁面。使用者需要點選 FIDO 安全性金鑰。

> **Note**
>
> Google Chrome 瀏覽器使用者不應在要求 Verify your identity with amazon.com (使用 amazon.com 驗證您身分) 的彈出畫面中選擇任何可用選項。您只需要點選安全性金鑰。

與其他 MFA 裝置不同，FIDO 安全性金鑰不會失去同步。管理員可以在其遺失或損壞時停用 FIDO 安全性金鑰。如需詳細資訊，請參閱停用 MFA 裝置 (主控台)。

如需支援 WebAuthn 的瀏覽器和 AWS 支援 的 FIDO 相容裝置的相關資訊，請參閱 使用通行密鑰和安全金鑰的支援組態。

虛擬 MFA 裝置

若使用者需要 MFA，則會出現第二個登入頁面。在 MFA code (MFA 代碼) 方塊中，使用者必須輸入 MFA 應用程式提供的數字代碼。

如果 MFA 代碼正確，則使用者可以存取 AWS 管理主控台。如果程式碼不正確，使用者可以重試其他代碼。

虛擬 MFA 裝置可能會失去同步。如果使用者在多次嘗試 AWS 管理主控台 後無法登入 ，系統會提示使用者同步虛擬 MFA 裝置。使用者可以根據螢幕上的提示同步虛擬 MFA 裝置。如需有關如何在 中代表使用者同步裝置的資訊 AWS 帳戶 ，請參閱 重新同步虛擬和硬體 MFA 裝置。

硬體 TOTP 權杖

若使用者需要 MFA，則會出現第二個登入頁面。在 MFA code (MFA 代碼) 方塊中，使用者必須輸入硬體 TOTP 權杖提供的數字代碼。

如果 MFA 代碼正確，則使用者可以存取 AWS 管理主控台。如果程式碼不正確，使用者可以重試其他代碼。

硬體 TOTP 權杖可能不同步。如果使用者在多次嘗試 AWS 管理主控台 後無法登入 ，系統會提示使用者同步 MFA 字符裝置。使用者可以根據螢幕上的提示同步 MFA 權杖裝置。如需有關如何在 中代表使用者同步裝置的資訊 AWS 帳戶，請參閱 重新同步虛擬和硬體 MFA 裝置。

在 中建立 IAM 使用者 AWS 帳戶

> **Important**
>
> IAM 最佳實務建議您要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 ，而不是使用具有長期憑證的 IAM 使用者。建議您僅將 IAM 使用者用於聯合身分使用者不支援的特定使用案例。
> 如果您正在使用我們的新 AWS 體驗，並希望讓人類使用者存取您的專案或 AWS 設定，則必須將他們新增至您的團隊。在我們的新體驗中，您無法建立具有主控台存取權的 IAM 使用者。如需詳細資訊，請參閱邀請團隊成員。

建立 IAM 使用者並讓該使用者執行任務的程序，包含以下步驟：

1. 在 AWS 管理主控台、、Tools for Windows PowerShell 或使用 API 操作建立使用者 AWS CLI。PowerShell AWS 如果您在 中建立使用者 AWS 管理主控台，則會根據您的選擇自動處理步驟 1-4。如果您透過程式設計的方式建立 IAM 使用者，則必須單獨執行每一個步驟。
2. 根據使用者需要的存取類型，建立使用者的憑證：
   - 啟用主控台存取 – 選用：如果使用者需要存取 AWS 管理主控台，請為使用者建立密碼。針對使用者停用主控台存取可以防止他們使用其使用者名稱和密碼登入 AWS 管理主控台 。它不會變更其許可，也不會阻止他們使用擔任的角色來存取主控台。

   > **Tip**
   >
   > 僅建立使用者需要的憑證。例如，對於只需要透過 存取的使用者 AWS 管理主控台，請勿建立存取金鑰。

3. 為使用者提供許可，以執行必要的任務。建議您將 IAM 使用者放入群組中，透過連接到這些群組的政策來管理許可。不過，也可以將許可政策直接連接到使用者來授予許可。如果使用主控台來新增使用者，可以將許可從現有使用者複製到新使用者。

   也可以新增許可界限，透過指定可定義使用者可擁有之最大許可的政策來限制使用者的許可。許可界限不會授予任何許可。

如需建立自訂許可政策以用於授予許可或設定許可界限的說明，請參閱使用客戶管理政策定義自訂 IAM 許可。

4. (選用) 藉由連接標籤將中繼資料新增至使用者。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
5. 提供使用者必要的登入資訊。這包括使用者提供這些憑證所在的帳戶登入頁面的密碼和主控台 URL。如需詳細資訊，請參閱 IAM 使用者如何登入 AWS。
6. (選用) 設定使用者的多重要素驗證 (MFA)。MFA 要求使用者每次登入 AWS 管理主控台時提供一次性使用的代碼。
7. (選用) 為 IAM 使用者提供許可，以管理他們自己的安全憑證。(根據預設，IAM 使用者沒有管理他們自己的憑證的許可)。如需詳細資訊，請參閱允許 IAM 使用者變更自己的密碼。

> **Note**
>
> 如果使用主控台建立使用者，並選取使用者必須在下次登入時建立新密碼 (建議)，則使用者擁有必要的許可。

如需有關建立使用者所需的許可資訊，請參閱 存取 IAM 資源所需的許可。

如需為特定使用案例建立 IAM 使用者的說明，請參閱下列主題：

- 建立 IAM 使用者以進行緊急存取
- 為無法使用 IAM 角色的工作負載建立 IAM 使用者

檢視 IAM 使用者

您可以列出 AWS 帳戶 或特定 IAM 群組中的 IAM 使用者，並列出使用者所在的所有 IAM 群組。有關列出使用者所需的許可資訊，請參閱 存取 IAM 資源所需的許可。

若要列出帳戶中的所有 IAM 使用者

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。

主控台會在您的 中顯示 IAM 使用者 AWS 帳戶。

AWS CLI

執行以下命令：

- `aws iam list-users`

API

呼叫以下操作：

- `ListUsers`

若要列出 IAM 群組中的 IAM 使用者

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇 User groups (使用者群組)。
4. 選擇使用者群組的名稱。

屬於該群組的 IAM 使用者會列在使用者索引標籤中。

AWS CLI

執行以下命令：

- `aws iam get-group`

API

呼叫以下操作：

- `GetGroup`

若要列出使用者所屬的 IAM 群組

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中選擇 IAM 使用者名稱。
5. 選取群組索引標籤，以顯示包含目前使用者的群組清單。

AWS CLI

執行以下命令：

- `aws iam list-groups-for-user`

API

呼叫以下操作：

- `ListGroupsForUser`

後續步驟

取得 IAM 使用者清單後，可以使用下列程序重新命名、刪除或停用 IAM 使用者。

- 重新命名 IAM 使用者
- 移除或停用 IAM 使用者

重新命名 IAM 使用者

> **Note**
>
> 最佳實務是，建議您要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 。如果遵循最佳實務，則您不用管理 IAM 使用者和群組。反之，您的使用者和群組是在 外部管理 AWS ，並且能夠以聯合身分的形式存取 AWS 資源。聯合身分是來自您的企業使用者目錄、Web 身分提供者、 AWS Directory Service、Identity Center 目錄或任何使用透過身分來

源提供的登入資料存取 AWS 服務的使用者。聯合身分使用由其身分提供者定義的群組。如果您使用的是 AWS IAM Identity Center，請參閱AWS IAM Identity Center 《使用者指南》中的在 IAM Identity Center 中管理身分，以取得在 IAM Identity Center 中建立使用者和群組的相關資訊。

Amazon Web Services 提供在 AWS 帳戶中管理 IAM 使用者的多項工具。您可以列出帳戶或使用者群組中的 IAM 使用者，也可以列出使用者所屬的所有 IAM 群組。您可以重新命名或變更 IAM 使用者的路徑。如果您要改用聯合身分而非 IAM 使用者，您可以刪除 AWS 帳戶的 IAM 使用者，或停用該使用者。

如需有關新增、變更或移除 IAM 使用者之受管政策的詳細資訊，請參閱 變更 IAM 使用者的許可。如需管理 IAM 使用者之內嵌政策的資訊，請參閱新增和移除 IAM 身分許可、編輯 IAM 政策 和 刪除 IAM 政策。最佳實務的做法是，使用受管政策而不是內嵌政策。AWS 受管政策為很多常見使用案例授予許可。請記住， AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為它們可供所有 AWS 客戶使用。因此，我們建議您定義使用案例的客戶管理政策，以便進一步減少許可。如需詳細資訊，請參閱AWS 受管政策。如需專為特定任務函數設計的 AWS 受管政策的詳細資訊，請參閱 AWS 任務函數的 受管政策。

若要了解驗證 IAM 政策的資訊，請參閱 IAM 政策驗證。

> **Tip**
>
> IAM Access Analyzer 可以分析您 IAM 角色使用的服務和動作，然後產生您可以使用的精細政策。測試產生的每個政策後，您可以將政策部署到生產環境。這可確保您僅授予所需的許可給工作負載。如需政策產生的詳細資訊，請參閱 IAM Access Analyzer 政策產生。

如需有關管理 IAM 使用者密碼的資訊，請參閱 管理 IAM 使用者的密碼。

重新命名 IAM 使用者

若要變更使用者名稱或路徑，您必須使用 AWS CLI、Tools for Windows PowerShell 或 AWS API。主控台中沒有用於重新命名使用者的選項。有關重新命名使用者所需的許可資訊，請參閱 存取 IAM 資源所需的許可。

當您更改使用者名稱或路徑時，會發生以下情況：

- 連接到使用者的所有政策繼續採用新使用名稱。

- 使用者以新名稱保留在相同的 IAM 群組中。
- 該使用者的唯一 ID 保持不變。如需有關唯一 ID 的詳細資訊，請參閱 唯一識別碼。
- 任何將該使用者視為「主體」(向該使用者授予存取權) 的資源或角色政策都會自動更新，以使用新名稱或路徑。例如，Amazon SQS 中的任何佇列類型政策或 Amazon S3 中的任何資源類型政策都會自動更新，以使用新名稱和路徑。

IAM 不會自動更新將該使用者視為資源以使用新使用者名稱或路徑的政策；您只能手動更新。例如，該使用者 Richard 連接了一個政策，而該政策讓使用者可管理其自己的安全憑證。如果管理員將 Richard 重新命名為 Rich，則管理員還需要更新該政策以將資源從：

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

變更為此：

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

如果管理員更改路徑，也會發生這種情況；管理員需要更新政策以反映該使用者使用新路徑。

重新命名使用者

- AWS CLI：aws iam update-user
- AWS API：UpdateUser

移除或停用 IAM 使用者

最佳實務建議您從 中移除未使用的 IAM 使用者 AWS 帳戶。如果想要保留 IAM 使用者憑證以供日後使用，而不是從帳戶中刪除它們，可以停用使用者的存取權。如需詳細資訊，請參閱停用 IAM 使用者。

> **Warning**
>
> IAM 使用者及其存取金鑰一經刪除，將無法還原或復原。

先決條件 – 檢視 IAM 使用者存取權

在移除使用者之前，請檢閱其最近的服務層級活動。這有助於防止從正在使用它的主體 (人員或應用程式) 中移除存取權。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

移除 IAM 使用者 (主控台)

當您使用 AWS 管理主控台 移除 IAM 使用者時，IAM 會自動刪除下列相關資訊：

- IAM 使用者識別符
- 任何群組成員關係，即從該 IAM 使用者所屬的任何群組中移除該 IAM 使用者
- 與 IAM 使用者有關的任何密碼
- 嵌入於 IAM 使用者的所有內嵌政策 (透過使用者群組許可套用至 IAM 使用者的政策不受影響)

> **Note**
>
> 當您刪除使用者時，IAM 會移除任何連接至 IAM 使用者的受管政策，但不會刪除受管政策。

- 任何相關的 MFA 裝置

若要移除 IAM 使用者 (主控台)

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者，然後選取您要刪除之 IAM 使用者名稱旁的核取方塊。
4. 在頁面頂端，選擇 Delete (刪除)。

   > **Note**
   >
   > 如果任何使用者擁有作用中的存取金鑰，則必須先停用這些存取金鑰，然後再刪除使用者。如需詳細資訊，請參閱若要停用 IAM 使用者的存取金鑰。

5. 確認對話方塊中，在文字輸入欄位中輸入使用者名稱以確認刪除使用者。選擇 刪除。

主控台會顯示已刪除 IAM 使用者的狀態通知。

刪除 IAM 使用者 (AWS CLI)

與 不同 AWS 管理主控台，當您使用 刪除 IAM 使用者時 AWS CLI，您必須手動刪除連接到 IAM 使用者的項目。此程序說明步驟。

從 AWS 帳戶 (AWS CLI) 刪除 IAM 使用者

1. 如果使用者有密碼，刪除該使用者的密碼。

   aws iam delete-login-profile
2. 如果使用者有存取金鑰，則將其刪除。

   aws iam list-access-keys (列出使用者的存取金鑰) 和 aws iam delete-access-key
3. 刪除使用者的簽署憑證。注意，當您刪除安全憑證時，將會永遠消失、無法還原。

   aws iam list-signing-certificates (列出使用者的簽署的憑證) 和 aws iam delete-signing-certificate
4. 如果使用者有 SSH 公有金鑰，則將其刪除。

   aws iam list-ssh-public-keys (列出使用者的 SSH 公有金鑰) 和 aws iam delete-ssh-public-key
5. 刪除使用者的 Git 憑證。

   aws iam list-service-specific-credentials (列出使用者的 Git 憑證) 和 aws iam delete-service-specific-credential
6. 如果使用者有多重要素驗證 (MFA) 裝置，請將其停用。

   aws iam list-mfa-devices (列出使用者的 MFA 裝置)、aws iam deactivate-mfa-device (停用裝置) 和 aws iam delete-virtual-mfa-device (永久刪除虛擬 MFA 裝置)
7. 刪除使用者的內嵌政策。

   aws iam list-user-policies (列出使用者的內嵌政策) 和 aws iam delete-user-policy (刪除政策)
8. 分開連接到該使用者的任何受管政策。

   aws iam list-attached-user-policies (列出連接到使用者的受管政策) 和 aws iam detach-user-policy (分開政策)
9. 將該使用者從任何 IAM 群組中移除。

   aws iam list-groups-for-user (列出使用者所屬的 IAM 群組) 和 aws iam remove-user-from-group
10. 刪除使用者。

- aws iam delete-user

停用 IAM 使用者

當 IAM 使用者暫時離開公司時，您可能需要停用他們。您可以將他們的 IAM 使用者憑證保留在適當位置，但仍然封鎖他們的 AWS 存取權。

若要停用使用者，請建立並連接政策以拒絕使用者存取 AWS。您可以稍後恢復使用者的存取權限。

以下是您可以連接到使用者以拒絕其存取的兩個拒絕政策範例。

下列政策不包含時間限制。您必須移除政策才能恢復使用者的存取權。

JSON

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

下列政策包括在 2024 年 12 月 24 日晚上 11:59 (UTC) 啟動政策的條件，並於 2025 年 2 月 28 日晚上 11:59 (UTC) 結束該政策。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
```

```
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
}
```

控制 IAM 使用者對 的存取 AWS 管理主控台

具有 AWS 帳戶 透過 登入 許可的 IAM AWS 管理主控台 使用者可以存取您的 AWS 資源。以下清單顯示您可以透過 授予 IAM 使用者存取 AWS 帳戶 資源的方式 AWS 管理主控台。它還顯示 IAM 使用者如何透過 AWS 網站存取其他 AWS 帳戶功能。

> **Note**
>
> 使用 IAM 免費。

的 AWS 管理主控台

您為需要存取 AWS 管理主控台的每個 IAM 使用者建立密碼。使用者透過啟用 IAM 的 AWS 帳戶登入頁面存取主控台。如需有關存取登入頁面的資訊，請參閱《AWS 登入 使用者指南》 中的 如何登入 AWS。如需有關建立密碼的資訊，請參閱中的使用者密碼 AWS。

您可以 AWS 管理主控台 移除 IAM 使用者的密碼，以防止其存取 。這可防止他們 AWS 管理主控台 使用其登入憑證登入 。它不會變更其許可，也不會阻止他們使用擔任的角色來存取主控台。如果使用者有作用中的存取金鑰，他們會繼續運作並允許透過 AWS CLI、Tools for Windows PowerShell、 AWS API 或 AWS 主控台行動應用程式進行存取。

您的 AWS 資源，例如 Amazon EC2 執行個體、Amazon S3 儲存貯體等

即使 IAM 使用者有密碼，還是需要許可才能存取您的 AWS 資源。在建立 IAM 使用者時，該使用者預設是沒有許可的。若要為您的 IAM 使用者提供所需的許可，請將政策連接到他們。如果您有多個 IAM 使用者使用相同的資源執行相同的任務，則可以將這些 IAM 使用者指派給一個群組。然後，將許可指派給該群組。如需有關建立 IAM 使用者與群組的詳細資訊，請參閱 IAM 身分。如需使用政策來設定許可的詳細資訊，請參閱 AWS 資源的存取管理。

AWS 開發論壇

任何人都可以讀取 AWS 開發論壇上的文章。想要將問題或評論發佈到 AWS 開發論壇的使用者可以使用其使用者名稱這樣做。使用者第一次張貼到 AWS 開發論壇時，系統會提示使用者輸入暱稱和電子郵件地址。只有該使用者可以在 AWS 開發論壇中使用該暱稱。

您的 AWS 帳戶 帳單和用量資訊

您可以授予使用者存取您的 AWS 帳戶 帳單和用量資訊。如需詳細資訊，請參閱 AWS Billing 使用者指南中的控制帳單資訊的存取。

您的 AWS 帳戶 設定檔資訊

使用者無法存取您的 AWS 帳戶 設定檔資訊。

您的 AWS 帳戶 安全登入資料

使用者無法存取您的 AWS 帳戶 安全登入資料。

> **Note**
>
> 無論介面為何，IAM 政策控制存取權。例如，您可以提供使用者存取 AWS 管理主控台所需的密碼。該使用者 (或使用者所屬的任何群組) 適用的政策會控制使用者可以在 AWS 管理主控台中做哪些事。或者，您可以為使用者提供 AWS 存取金鑰，以便對 進行 API 呼叫 AWS。這些政策會控制使用者可以透過使用這些存取金鑰進行身分驗證的文件庫或用戶端，呼叫哪些動作。

變更 IAM 使用者的許可

您可以變更 IAM 使用者的許可， AWS 帳戶 方法是變更其群組成員資格、從現有使用者複製許可、將政策直接連接到使用者，或設定許可界限。許可界限控制使用者可以擁有的許可上限。許可界限是一項進階 AWS 功能。

如需有關修改使用者許可所需的許可資訊，請參閱 存取 IAM 資源所需的許可。

主題

- 檢視使用者存取
- 根據使用者的存取活動產生政策
- 新增許可到使用者 (主控台)

- 變更使用者的許可 (主控台)
- 若要從使用者中移除許可政策 (主控台)
- 若要從使用者中移除許可界限 (主控台)
- 新增和移除使用者的許可 (AWS CLI 或 AWS API)

檢視使用者存取

變更使用者的許可之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

根據使用者的存取活動產生政策

您有時可能會對 IAM 實體 (使用者或角色) 授予超出其要求的許可。為了協助您精簡所授予的許可，您可以根據實體的存取活動產生 IAM 政策。IAM Access Analyzer 會檢閱您的 AWS CloudTrail 日誌，並產生政策範本，其中包含實體在指定日期範圍內已使用的許可。您可以使用範本建立具有精細許可的受管政策，然後將其連接至 IAM 實體。如此一來，您只會授予使用者或角色與特定使用案例 AWS 的資源互動所需的許可。若要進一步了解，請參閱 IAM Access Analyzer 政策產生。

新增許可到使用者 (主控台)

IAM 提供三種方法將許可政策新增到使用者：

- 新增 IAM 使用者至 IAM 群組：讓使用者成為群組的成員。群組的政策連接到使用者。
- 從現有的 IAM 使用者複製許可：複製所有群組成員資格、連接的受管政策、內嵌政策，以及來源使用者的任何現有許可界限。
- 直接連接政策到 IAM 使用者：將受管政策直接連接到使用者。為方便管理許可，請將您的政策連接到群組，然後讓 IAM 使用者成為相應群組的成員。

> **Important**
>
> 如果使用者具有許可界限，則您無法新增比許可界限允許的還要多的許可至使用者。

若要透過將 IAM 使用者新增到群組來新增許可

將 IAM 使用者新增至 IAM 群組，會立即使用為群組定義的許可來更新使用者的許可。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中選擇 IAM 使用者名稱。
5. 選取群組索引標籤，以顯示包含目前使用者的群組清單。
6. 選擇將使用者新增至群組。
7. 選取您要使用者加入的各個群組的核取方塊。如果成為群組的成員，清單會顯示每個群組的名稱和使用者所接收的政策。
8. (選用) 可以選擇建立群組來定義新群組。如果想要將使用者新增至連接政策與現有群組不同的群組，這很有用：

   a. 在新標籤中的 使用者群組名稱 中輸入新群組的名稱。

   > **Note**
   >
   > AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。群組名稱可以是長達 128 個字母、數字以及這些字元的組合：加號 (+)、等號 (=)、逗號 (,)、句號 (.)、@ 符號以及連字號 (-)。名稱在帳戶中必須是唯一的。它們無法透過大小寫進行區分。例如，您無法建立兩個名為 TESTGROUP 和 testgroup 的群組。

   b. 選擇一個或多個您要連接到群組的受管政策的核取方塊。您也可以選擇 建立政策 來建立新的受管政策。如果您這麼做時，在新政策完成時會返回此瀏覽器標籤或視窗；選擇 重新整理，然後選擇新的政策，以將它連接至您的群組。如需詳細資訊，請參閱使用客戶管理政策定義自訂 IAM 許可。

   c. 選擇 建立使用者群組。

   d. 返回原始標籤，重新整理您的群組清單。然後，選取新群組的核取方塊。

9. 選擇將使用者新增至群組。

主控台會顯示狀態訊息，通知您使用者已新增至指定的群組。

若要透過從其他 IAM 使用者中複製來新增許可

如果選擇透過複製許可來為 IAM 使用者新增許可，IAM 會從指定的使用者中複製所有群組成員資格、連接的受管政策、內嵌政策以及任何現有的許可界限，並立即將它們套用到目前選取的使用者。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中選擇 IAM 使用者名稱。
5. 在許可索引標籤中，選取新增許可。
6. 在新增許可頁面中，選擇複製許可。清單會顯示可用的 IAM 使用者及其群組成員資格和連接的政策。
7. 選擇您要複製許可的使用者旁的選項按鈕。
8. 選擇 下一步，以查看要對使用者進行的變更清單。然後選擇 新增許可。

主控台會顯示狀態訊息，通知您已從指定的 IAM 使用者中複製許可。

若要透過直接連接政策到 IAM 使用者來新增許可

可以將受管政策直接連接至 IAM 使用者。會立即套用更新的許可。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中選擇 IAM 使用者名稱。
5. 在許可索引標籤中，選取新增許可。
6. 在新增許可頁面中，選擇直接連接政策。許可政策清單會顯示可用政策及其政策類型和連接的實體。
7. 選取要連接的政策名稱旁的選項按鈕。
8. 選擇 下一步，以查看要對使用者進行的變更清單。然後選擇 新增許可。

主控台會顯示狀態訊息，通知您政策已新增至指定的 IAM 使用者。

若要設定 IAM 使用者的許可界限

許可界限是用於管理 中許可的進階功能 AWS ，用於設定 IAM 使用者可擁有的最大許可。設定許可界限會立即將 IAM 使用者許可限制在界限內，無論授予的其他許可為何。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中，選擇您要變更其許可界限的 IAM 使用者名稱。
5. 選擇 許可 標籤。如有必要，請開啟 許可界限 區段，然後選擇 設定許可界限。
6. 在設定許可界限頁面的許可政策下，選取要用於許可界限的政策。
7. 選擇 設定界限。

主控台會顯示狀態訊息，通知您已新增許可界限。

變更使用者的許可 (主控台)

IAM 可讓您以下列方式變更與使用者建立關聯的許可：

- 編輯許可政策 – 編輯使用者的內嵌政策、使用者群組的內嵌政策，或編輯直接連接到使用者或從群組連接的受管政策。如果使用者具有許可界限，則您無法提供比當成使用者的許可界限來人用的政策所允許的還要更多的許可。
- 變更許可界限 – 變更用作使用者的許可界限的政策。這會擴展或限制使用者可以擁有的許可上限。

編輯連接到使用者的許可政策

變更許可會立即更新使用者的存取權。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。

3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中，選擇您要變更其許可界限的 IAM 使用者名稱。
5. 選擇許可索引標籤。如有必要，請開啟許可界限區段。
6. 選擇您要編輯以檢視詳細資訊的政策名稱。選擇連接的實體索引標籤，檢視編輯政策時可能受影響的其他實體 (IAM 使用者、群組和角色）。
7. 選擇 許可 標籤，然後檢閱政策授予的許可。若要對許可進行變更，請選擇編輯。
8. 編輯該政策並解決任何政策驗證建議。如需詳細資訊，請參閱編輯 IAM 政策。
9. 選擇下一步，檢閱政策摘要，然後選擇儲存變更。

主控台會顯示狀態訊息，通知您已更新政策。

若要變更使用者的許可界限

變更許可界限會立即更新使用者的存取權。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中，選擇您要變更其許可界限的 IAM 使用者名稱。
5. 選擇 Permissions (許可) 標籤。如有必要，開啟 許可界限 區段，然後選擇 變更界限。
6. 選擇要用於許可界限的政策。
7. 選擇 設定界限。

主控台會顯示狀態訊息，通知您已變更許可界限。

若要從使用者中移除許可政策 (主控台)

移除許可政策會立即更新使用者的存取權。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。

2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 選擇您想要移除其許可政策的使用者名稱。
5. 選擇許可索引標籤。
6. 如果想要透過移除現有政策來移除許可，在選擇移除來移除該政策之前，請檢視連接方式資料欄來了解使用者如何取得該政策：

   - 如果因群組成員資格而套用政策，則選擇 移除 會從群組移除使用者。請記住，您可能會有多個政策連接到單一群組。如果您移除群組中的使用者，使用者會失去透過群組成員資格收到的「所有」政策的存取權。
   - 如果政策是直接連接到使用者的受管政策，則選擇 移除 會將政策與使用者分開。這不會影響政策本身或政策可能連接的任何其他目標實體。
   - 如果政策是內嵌政策，則選擇移除會從 IAM 移除政策。直接連接到使用者的內嵌政策只存在於該使用者。

如果政策是透過群組成員資格授予給使用者，主控台會顯示狀態訊息，通知您已從 IAM 群組中移除 IAM 使用者。如果政策為直接連接或內嵌，則狀態訊息會通知您已移除政策。

若要從使用者中移除許可界限 (主控台)

移除許可界限會立即更新使用者的存取權。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇使用者 。
4. 在使用者清單中，選擇您要移除其許可界限的 IAM 使用者名稱。
5. 選擇許可索引標籤。如有必要，請開啟許可界限區段。
6. 選擇 Change boundary (變更界限)。若要確認您要移除許可界限，在確認對話方塊中，選擇移除界限。

主控台會顯示狀態訊息，通知您已移除許可界限。

新增和移除使用者的許可 (AWS CLI 或 AWS API)

若要以程式設計方式新增或移除許可，您必須新增或移除群組成員資格、連接或分開受管政策，或者新增或刪除內嵌政策。如需詳細資訊，請參閱下列主題：

- 編輯 IAM 群組中的使用者
- 新增和移除 IAM 身分許可

中的使用者密碼 AWS

您可以您帳戶中的 IAM 使用者管理密碼。IAM 使用者需要密碼才能存取 AWS 管理主控台。使用者不需要密碼，即可透過使用 Tools for Windows PowerShell AWS CLI、 AWS SDKs或 APIs 以程式設計方式存取 AWS 資源。對於這些環境，您可以選擇指派 IAM 使用者存取金鑰。不過，建議您先考慮其他更安全的方案來替代存取金鑰。如需詳細資訊，請參閱AWS 安全登入資料。

> **Note**
>
> 如果您的其中一個 IAM 使用者遺失或忘記其密碼，您無法從 IAM 擷取它。根據您的設定，使用者或管理員必須建立新密碼。

目錄

- 設定 IAM 使用者的帳戶密碼政策
- 管理 IAM 使用者的密碼
- 允許 IAM 使用者變更自己的密碼
- IAM 使用者如何變更自己的密碼

設定 IAM 使用者的帳戶密碼政策

您可以在 上設定自訂密碼政策 AWS 帳戶 ，以指定 IAM 使用者密碼的複雜性要求和強制輪換期間。如果您未設定自訂密碼政策，IAM 使用者密碼必須符合預設 AWS 密碼政策。如需詳細資訊，請參閱自訂密碼政策選項。

主題

- 設定密碼政策的規則
- 設定密碼政策所需的許可

- 預設密碼政策
- 自訂密碼政策選項
- 若要設定密碼政策 (主控台)
- 若要變更密碼政策 (主控台)
- 要刪除自訂密碼政策 (主控台)
- 設定密碼政策 (AWS CLI)
- 設定密碼政策 (AWS API)

設定密碼政策的規則

IAM 密碼政策不適用於 AWS 帳戶根使用者 密碼或 IAM 使用者存取金鑰。如果密碼過期，IAM 使用者無法登入 ， AWS 管理主控台 但可以繼續使用其存取金鑰。

當您建立或變更密碼政策時，大多數密碼政策設定將在使用者下次變更密碼時強制執行。但是，某些設定會立即強制執行。例如：

- 當最短長度和字元類型要求變更時，將在使用者下次變更密碼時強制執行這些設定。即使現有的密碼不遵守更新的密碼政策，也不會強制使用者變更現有的密碼。
- 當您設定密碼過期期間時，會立即強制執行過期期間。例如，假設您將密碼過期期間設定為 90 天。在這種情況下，現有密碼超過 90 天的所有 IAM 使用者的密碼會到期。這些使用者必須在下次登入時變更其密碼。

在指定次數的登入嘗試失敗之後，您無法建立「鎖定政策」來鎖定使用者帳戶。為了增強安全性，我們建議您將強式密碼政策與多重要素驗證 (MFA) 結合。如需有關 MFA 的詳細資訊，請參閱 AWS IAM 中的多重要素驗證。

設定密碼政策所需的許可

您必須設定許可，以允許 IAM 實體 (使用者或角色) 檢視或編輯其帳戶密碼政策。您可以在 IAM 政策中包含下列密碼政策動作：

- `iam:GetAccountPasswordPolicy` – 允許實體檢視其帳戶的密碼政策
- `iam:DeleteAccountPasswordPolicy` – 允許實體刪除其帳戶的自訂密碼政策，並還原為預設密碼政策
- `iam:UpdateAccountPasswordPolicy` – 允許實體建立或變更其帳戶的自訂密碼政策

下列政策允許完整存取權以檢視和編輯帳戶密碼政策。若要了解如何使用此範例 JSON 政策文件來建立 IAM 政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

如需 IAM 使用者變更其自己密碼所需許可的資訊，請參閱 允許 IAM 使用者變更自己的密碼。

預設密碼政策

如果管理員未設定自訂密碼政策，IAM 使用者密碼必須符合預設 AWS 密碼政策。

預設密碼政策會強制執行下列條件：

- 密碼長度最短為 8 個字元，最長為 128 個字元。
- 至少混用 3 種下列類型字元：大寫、小寫、數字和非英數字元 (! @ # $ % ^ & * ( ) _ + - = [ ] { } | ')
- 與 AWS 帳戶 您的姓名或電子郵件地址不同
- 保證密碼不會過期

自訂密碼政策選項

當您設定帳戶的自訂密碼政策時，您可以指定下列條件：

- 密碼最小長度 – 您可以指定至少 6 個字元，最多可指定 128 個字元。

- 密碼強度：您可以選取下列任一核取方塊來定義 IAM 使用者密碼的強度：
  - 至少需要一個拉丁字母 (A–Z) 的大寫字母
  - 至少需要一個拉丁字母 (a–z) 的小寫字母
  - 至少需要有一個數字
  - 至少需要一個非英數字元的字元 ! @ # $ % ^ & * ( ) _ + - = [ ] { } | '
- Turn on password expiration (啟用密碼過期) – 您可以選取並指定至少 1 天，且最長為 1,095 天的 IAM 使用者密碼有效期，設定後生效。例如，如果您指定 90 天的到期日，它會立即影響所有使用者。對於密碼超過 90 天的使用者，當他們在變更後登入主控台時，必須設定新密碼。密碼為 75-89 天的使用者會收到密碼過期的 AWS 管理主控台 警告。IAM 使用者可隨時變更密碼 (如果他們有許可)。當他們設定新的密碼時，該密碼的到期期間將重新開始。IAM 使用者一次只能有一個有效的密碼。
- 密碼過期需要管理員重設 – 選取此選項，以防止 IAM 使用者在密碼過期後使用 AWS 管理主控台 更新自己的密碼。在選取此選項之前，請確認您的 AWS 帳戶 有多個具有管理許可的使用者，以重設 IAM 使用者密碼。具有 `iam:UpdateLoginProfile` 許可的管理員可以重設 IAM 使用者密碼。具有 `iam:ChangePassword` 許可和作用中存取金鑰的 IAM 使用者可以程式設計方式重設自己的 IAM 使用者主控台密碼。如果您清除此核取方塊，密碼到期的 IAM 使用者仍必須先設定新密碼，然後才能存取 AWS 管理主控台。
- Allow users to change their own password (允許使用者變更自己的密碼) – 您可以允許帳戶中的所有 IAM 使用者變更自己的密碼。這可讓使用者僅針對其使用者存取 `iam:ChangePassword` 動作，並存取 `iam:GetAccountPasswordPolicy` 動作。此選項不會將許可政策連接到每個使用者。反之，IAM 會在帳戶層級套用許可到所有使用者。或者，您只允許一些使用者管理自己的密碼。若要執行這項操作，請清除此核取方塊。如需有關使用政策以限制誰可以管理密碼的詳細資訊，請參閱 允許 IAM 使用者變更自己的密碼。
- 防止密碼重複使用 – 您可以防止 IAM 使用者重複使用舊密碼的指定數字。您可以指定長度下限為 1，以及長度上限為 24 的舊密碼，無法重複使用。

若要設定密碼政策 (主控台)

您可以使用 AWS 管理主控台 來建立、變更或刪除自訂密碼政策。對密碼政策的變更適用於此政策變更後建立的新 IAM 使用者，以及變更密碼時的現有 IAM 使用者。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。

3. 在導覽窗格中，選擇帳戶設定。
4. 在 Password policy (密碼政策) 區段中，選擇 Edit (編輯)。
5. 選擇 Custom (自訂) 以使用自訂密碼政策。
6. 選取要套用至密碼政策的選項，然後選擇 Save changes (儲存變更)。
7. 選擇 Set custom (設定自訂)，確認您要設定自訂密碼政策。

主控台會顯示狀態訊息，通知您 IAM 使用者的密碼需求已更新。

若要變更密碼政策 (主控台)

您可以使用 AWS 管理主控台 來建立、變更或刪除自訂密碼政策。對密碼政策的變更適用於此政策變更後建立的新 IAM 使用者，以及變更密碼時的現有 IAM 使用者。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇帳戶設定。
4. 在 Password policy (密碼政策) 區段中，選擇 Edit (編輯)。
5. 選取要套用至密碼政策的選項，然後選擇 Save changes (儲存變更)。
6. 選擇 Set custom (設定自訂)，確認您要設定自訂密碼政策。

主控台會顯示狀態訊息，通知您 IAM 使用者的密碼需求已更新。

要刪除自訂密碼政策 (主控台)

您可以使用 AWS 管理主控台 來建立、變更或刪除自訂密碼政策。對密碼政策的變更適用於此政策變更後建立的新 IAM 使用者，以及變更密碼時的現有 IAM 使用者。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇帳戶設定。

4. 在 Password policy (密碼政策) 區段中，選擇 Edit (編輯)。
5. 選擇 IAM default (IAM 預設值) 以刪除自訂密碼政策，然後選擇 Save changes (儲存變更)。
6. 選擇 Set default (設定預設值)，確認您要設定 IAM 預設密碼政策。

主控台會顯示狀態訊息，通知您密碼政策已設定為 IAM 預設。

設定密碼政策 (AWS CLI)

您可以使用 AWS Command Line Interface 來設定密碼政策。

從 管理自訂帳戶密碼政策 AWS CLI

執行下列命令：

- 若要建立或變更自訂密碼政策：`aws iam update-account-password-policy`
- 若要檢視密碼政策：`aws iam get-account-password-policy`
- 若要刪除自訂密碼政策：`aws iam delete-account-password-policy`

設定密碼政策 (AWS API)

您可以使用 AWS API 操作來設定密碼政策。

從 AWS API 管理自訂帳戶密碼政策

呼叫以下操作：

- 若要建立或變更自訂密碼政策：UpdateAccountPasswordPolicy
- 若要檢視密碼政策：GetAccountPasswordPolicy
- 若要刪除自訂密碼政策：DeleteAccountPasswordPolicy

管理 IAM 使用者的密碼

使用 處理 AWS 資源 AWS 管理主控台 的 IAM 使用者必須擁有密碼才能登入。您可以建立、更改或刪除您的 AWS 帳戶中 IAM 使用者的密碼。

將密碼指派給使用者之後，使用者可以 AWS 管理主控台 使用帳戶的登入 URL 登入 ，如下所示：

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

如需 IAM 使用者如何登入 的詳細資訊 AWS 管理主控台，請參閱AWS 登入 《 使用者指南》中的如何登入 AWS。

即使使用者有自己的密碼，還是需要許可才能存取您的 AWS 資源。使用者預設沒有任何許可。為授予使用者所需的許可，您可向使用者或使用者所屬的群組分配政策。如需有關建立使用者與群組的詳細資訊，請參閱 IAM 身分。如需使用政策來設定許可的詳細資訊，請參閱 變更 IAM 使用者的許可。

您可以授予使用者變更自己密碼的許可。如需詳細資訊，請參閱允許 IAM 使用者變更自己的密碼。如需有關使用者如何存取您帳戶登入頁面的資訊，請參閱 《AWS 登入 使用者指南》 中的 如何登入 AWS。

主題

- 建立、變更或刪除 IAM 使用者密碼 (主控台)

建立、變更或刪除 IAM 使用者密碼 (主控台)

您可以使用 AWS 管理主控台 來管理 IAM 使用者的密碼。

您的使用者存取需求可能會隨著時間而變更。您可能需要讓打算讓 CLI 存取的使用者擁有主控台存取權、變更使用者的密碼，因為他們會收到含有其登入資料的電子郵件，或在使用者離開組織或不再需要 AWS 存取權時將其刪除。

若要建立 IAM 使用者密碼 (主控台)

使用此程序建立與使用者名稱關聯的密碼，以授予使用者主控台存取權。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇 Users (使用者)。
4. 選擇您想要為之建立密碼的使用者名稱。
5. 選擇 安全憑證 標籤，然後在 主控台登入 下方選擇 啟用主控台存取。
6. 在啟用主控台存取對話方塊中，選取重設密碼，然後選擇是否要讓 IAM 產生密碼或建立自訂密碼：
   - 若要讓 IAM 產生密碼，請選擇 Autogenerated password (自動產生的密碼)。

- 若要建立自訂密碼，請選擇 Custom password (自訂密碼) 並輸入密碼。

> **Note**
>
> 您建立的密碼必須符合帳戶的密碼政策。

7. 若要要求使用者登入時建立新的密碼，請選擇下次登入時需要變更密碼。
8. 若要要求使用者立即使用新密碼，請選取撤銷作用中主控台工作階段。這會將內嵌政策連接至 IAM 使用者，如果使用者的憑證比政策指定的時間更舊，則會拒絕該使用者存取資源。
9. 選擇重設密碼
10. 主控台密碼對話方塊會通知您已啟用使用者的新密碼。若要檢視密碼以便與使用者共用，請選擇主控台密碼對話方塊中的顯示。選取下載 .csv 檔案，以下載具有使用者憑證的檔案。

> **Important**
>
> 基於安全原因，在完成此步驟後您無法存取該密碼，但您可以隨時建立新密碼。

主控台會顯示狀態訊息，通知您已啟用主控台存取。

為 IAM 使用者變更密碼 (主控台)

使用此程序來更新與使用者名稱關聯的密碼。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇 Users (使用者)。
4. 選擇您想要為之變更密碼的使用者名稱。
5. 選擇 安全憑證 標籤，然後在 主控台登入 下方選擇 管理主控台存取。
6. 在管理主控台存取對話方塊中，選取重設密碼，然後選擇是否要讓 IAM 產生密碼或建立自訂密碼：

- 若要讓 IAM 產生密碼，請選擇 Autogenerated password (自動產生的密碼)。
- 若要建立自訂密碼，請選擇 Custom password (自訂密碼) 並輸入密碼。

> **Note**
>
> 您建立的密碼必須符合帳戶的密碼政策。

7. 若要要求使用者登入時建立新的密碼，請選擇下次登入時需要變更密碼。
8. 若要要求使用者立即使用新密碼，請選取撤銷作用中主控台工作階段。這會將內嵌政策連接至 IAM 使用者，如果使用者的憑證比政策指定的時間更舊，則會拒絕該使用者存取資源。
9. 選擇重設密碼
10. 主控台密碼對話方塊會通知您已啟用使用者的新密碼。若要檢視密碼以便與使用者共用，請選擇主控台密碼對話方塊中的顯示。選取下載 .csv 檔案，以下載具有使用者憑證的檔案。

> **Important**
>
> 基於安全原因，在完成此步驟後您無法存取該密碼，但您可以隨時建立新密碼。

主控台會顯示狀態訊息，通知您已更新主控台存取。

刪除 (停用) IAM 使用者密碼 (主控台)

使用此程序刪除與使用者名稱關聯的密碼，從而移除使用者的主控台存取權。

> **Important**
>
> 您可以 AWS 管理主控台 移除 IAM 使用者的密碼，以防止其存取 。這可防止他們 AWS 管理主控台 使用其登入憑證登入 。它不會變更其許可，也不會阻止他們使用擔任的角色來存取主控台。如果使用者有作用中的存取金鑰，他們會繼續運作並允許透過 AWS CLI、Tools for Windows PowerShell、 AWS API 或 AWS 主控台行動應用程式進行存取。

Console

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中適合您使用者類型的登入程序操作。
2. 在 IAM 主控台首頁的左側導覽窗格中，在搜尋 IAM 文字方塊中輸入查詢。
3. 在導覽窗格中，選擇 Users (使用者)。
4. 選擇您想要為之刪除密碼的使用者名稱。

5. 選擇 安全憑證 標籤，然後在 主控台登入 下方選擇 管理主控台存取。
6. 若要要求使用者立即停止使用主控台，請選取撤銷作用中主控台工作階段。這會將內嵌政策連接至 IAM 使用者，如果使用者的憑證比政策指定的時間更舊，則會拒絕該使用者存取資源。
7. 選擇停用存取

主控台會顯示狀態訊息，通知您已停用主控台存取。

建立、變更或刪除 IAM 使用者密碼 (AWS CLI)

您可以使用 AWS CLI 來管理 IAM 使用者的密碼。

建立密碼 (AWS CLI)

1. (選用) 若要判斷使用者是否有密碼，請執行此命令：aws iam get-login-profile
2. 若要建立密碼，請執行此命令：aws iam create-login-profile

若要變更使用者的密碼 (AWS CLI)

1. (選用) 若要判斷使用者是否有密碼，請執行此命令：aws iam get-login-profile
2. 若要變更密碼，請執行此命令：aws iam update-login-profile

若要刪除 (停用) 使用者的密碼 (AWS CLI)

1. (選用) 若要判斷使用者是否有密碼，請執行此命令：aws iam get-login-profile
2. (選用) 若要判斷密碼前次使用的時間，請執行此命令：aws iam get-user
3. 若要刪除密碼，請執行此命令：aws iam delete-login-profile

> **Important**
>
> 當您刪除使用者密碼時，使用者將無法繼續登入 AWS 管理主控台。如果使用者有作用中的存取金鑰，他們會繼續運作並允許透過 AWS CLI、Tools for Windows PowerShell 或 AWS API 函數呼叫進行存取。當您使用 AWS CLI、Tools for Windows PowerShell 或 AWS API 從您的中刪除使用者時 AWS 帳戶，您必須先使用此操作刪除密碼。如需詳細資訊，請參閱刪除 IAM 使用者 (AWS CLI)。

在指定時間之前撤銷使用者的作用中主控台工作階段 (AWS CLI)

1. 若要嵌入內嵌政策，以在指定時間之前撤銷 IAM 使用者的作用中主控台工作階段，請使用下列內嵌政策並執行以下命令：aws iam put-user-policy

   此內嵌政策會拒絕所有許可，並包含 aws:TokenIssueTime 條件索引鍵。它會在內嵌政策的 Condition 元素中指定的時間之前，撤銷使用者的作用中主控台工作階段。將 aws:TokenIssueTime 條件索引鍵值取代為您自己的值。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (選用) 若要列出嵌入在 IAM 使用者中的內嵌政策名稱，請執行以下命令：aws iam list-user-policies
3. (選用) 若要檢視嵌入在 IAM 使用者中的具名內嵌政策，請執行以下命令：aws iam get-user-policy

建立、變更或刪除 IAM 使用者密碼 (AWS API)

您可以使用 AWS API 來管理 IAM 使用者的密碼。

建立密碼 (AWS API)

1. (選用) 若要判斷使用者是否有密碼，請呼叫此操作：GetLoginProfile
2. 若要建立密碼，請呼叫此操作：CreateLoginProfile

變更使用者的密碼 (AWS API)

1. (選用) 若要判斷使用者是否有密碼，請呼叫此操作：GetLoginProfile
2. 若要變更密碼，請呼叫此操作：UpdateLoginProfile

刪除（停用）使用者的密碼 (AWS API)

1. (選用) 若要判斷使用者是否有密碼，請執行此命令：GetLoginProfile
2. (選用) 若要判斷密碼前次使用的時間，請執行此命令：GetUser
3. 若要刪除密碼，請執行此命令：DeleteLoginProfile

> **Important**
>
> 當您刪除使用者密碼時，使用者將無法繼續登入 AWS 管理主控台。如果使用者有作用中的存取金鑰，他們會繼續運作並允許透過 AWS CLI、Tools for Windows PowerShell 或 AWS API 函數呼叫進行存取。當您使用 AWS CLI、Tools for Windows PowerShell 或 AWS API 從您的中刪除使用者時 AWS 帳戶，您必須先使用此操作刪除密碼。如需詳細資訊，請參閱刪除 IAM 使用者 (AWS CLI)。

在指定時間 (AWS API) 之前撤銷使用者的作用中主控台工作階段

1. 若要內嵌內嵌政策，以在指定時間之前撤銷 IAM 使用者的作用中主控台工作階段，請使用下列內嵌政策並執行以下命令：PutUserPolicy

   此內嵌政策會拒絕所有許可，並包含 `aws:TokenIssueTime` 條件索引鍵。它會在內嵌政策的 `Condition` 元素中指定的時間之前，撤銷使用者的作用中主控台工作階段。將 `aws:TokenIssueTime` 條件索引鍵值取代為您自己的值。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
```

```
    "Condition": {
      "DateLessThan": {
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (選用) 若要列出嵌入在 IAM 使用者中的內嵌政策名稱，請執行以下命令：ListUserPolicies
3. (選用) 若要檢視嵌入在 IAM 使用者中的具名內嵌政策，請執行以下命令：GetUserPolicy

允許 IAM 使用者變更自己的密碼

> **Note**
>
> 具有聯合身分的使用者將使用其身分提供者定義的程序來變更他們的密碼。最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 使用臨時憑證存取 。

您可以授予 IAM 使用者許可，讓他們能夠登入 AWS 管理主控台變更自己的密碼。您可以使用兩種方式的其中一種來執行此動作：

- 允許帳戶中的所有 IAM 使用者變更自己的密碼。
- 僅允許選定的 IAM 使用者變更自己的密碼。在這種情況下您停用選項，讓所有使用者變更自己的密碼，並使用 IAM 政策僅為一些使用者授予許可。這種方法允許這些使用者變更自己的密碼以及選擇性的其他憑證，如他們自己的存取金鑰。

> **Important**
>
> 建議您設定自訂密碼政策，以要求 IAM 使用者建立強式密碼。

允許所有 IAM 使用者變更自己的密碼

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。

2. 在導覽窗格中，選擇帳戶設定。
3. 在 Password policy (密碼政策) 區段中，選擇 Edit (編輯)。
4. 選擇 Custom (自訂) 以使用自訂密碼政策。
5. 選取 Allow users to change their own password (允許使用者變更自己的密碼)，然後選擇 Save changes (儲存變更)。這允許帳戶中的所有使用者僅針對其使用者存取 `iam:ChangePassword` 動作，並可存取 `iam:GetAccountPasswordPolicy` 動作。
6. 為使用者提供下列指示，以便變更其密碼：IAM 使用者如何變更自己的密碼。

AWS CLI

執行以下命令：

- `aws iam update-account-password-policy`

API

若要更新帳戶密碼政策，請呼叫下列操作：

- `UpdateAccountPasswordPolicy`

允許選取的 IAM 使用者變更自己的密碼

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇帳戶設定。
3. 在 Password policy (密碼政策) 區段中，請確認未選取 Allow users to change their own password (允許使用者變更自己的密碼)。如果已選取此核取方塊，所有使用者都可以變更自己的密碼。(請參閱之前的程序)。
4. 如果沒有使用者，可建立使用者，讓他們可以變更自己的密碼。如需詳細資訊，請參閱在 中建立 IAM 使用者 AWS 帳戶。
5. (選用) 為被允許變更密碼的使用者建立 IAM 群組，然後從之前的步驟將使用者加入到群組。如需詳細資訊，請參閱IAM 使用者群組。
6. 指派以下政策到群組。如需詳細資訊，請參閱管理 IAM 政策。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

此政策會授予 ChangePassword 動作的存取權，讓使用者只能從主控台、、 AWS CLI Tools for Windows PowerShell 或 API 變更自己的密碼。它還授予存取權至 GetAccountPasswordPolicy 動作，其可讓使用者檢視目前的密碼政策；此許可是必需的，以便使用者可在 Change password (變更密碼) 頁面檢視帳戶密碼政策。必須允許使用者讀取目前的密碼政策，以確保變更的密碼符合政策的要求。

7. 為使用者提供下列指示，以便變更其密碼：IAM 使用者如何變更自己的密碼。

如需詳細資訊

如需有關管理憑證的詳細資訊，請參閱下列主題：

- 允許 IAM 使用者變更自己的密碼
- 中的使用者密碼 AWS
- 設定 IAM 使用者的帳戶密碼政策
- 管理 IAM 政策
- IAM 使用者如何變更自己的密碼

IAM 使用者如何變更自己的密碼

如果您已獲授予變更自己的 IAM 使用者密碼的許可，您可以使用 中的特殊頁面 AWS 管理主控台 來執行此操作。您也可以使用 AWS CLI 或 AWS API。

主題

- 必要許可
- IAM 使用者如何變更他們自己的密碼 (主控台)
- IAM 使用者如何變更自己的密碼 (AWS CLI 或 AWS API)

必要許可

若要變更您 IAM 使用者的密碼，您必須擁有以下政策的許可：AWS：允許 IAM 使用者在「安全憑證」頁面中變更其主控台密碼。

IAM 使用者如何變更他們自己的密碼 (主控台)

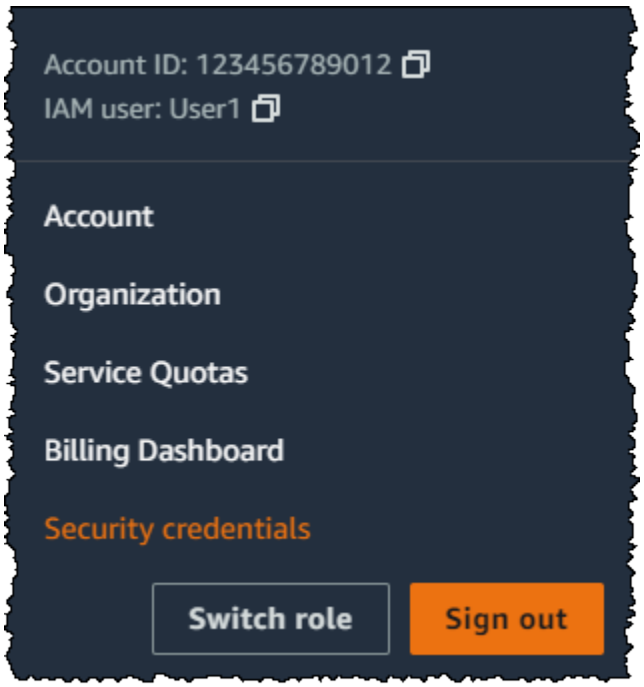
下列程序說明 IAM 使用者如何使用 AWS 管理主控台 來變更自己的密碼。

變更您 IAM 使用者的密碼 (主控台)

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

   > **Note**
   >
   > 為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入要重新導向至 AWS 您帳戶的 IAM 使用者登入頁面的帳戶 ID 或帳戶別名。

   若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。
2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在 AWS IAM 憑證索引標籤上，選擇更新密碼。
4. 針對 Current password (目前的密碼)，請輸入目前的密碼。輸入 New password (新密碼) 及 Confirm new password (確認新密碼) 的新密碼。接著選擇更新密碼。

> **Note**
>
> 新密碼必須符合帳戶密碼政策的要求。如需詳細資訊，請參閱設定 IAM 使用者的帳戶密碼政策。

IAM 使用者如何變更自己的密碼 (AWS CLI 或 AWS API)

下列程序說明 IAM 使用者如何使用 AWS CLI 或 AWS API 來變更自己的密碼。

若要變更您自己的 IAM 密碼，請使用下列：

- AWS CLI: `aws iam change-password`
- AWS API： `ChangePassword`

管理 IAM 使用者的存取金鑰

> **Important**
>
> 最佳實務的作法是使用臨時性安全憑證 (如 IAM 角色)，而不是建立長期憑證 (如存取金鑰)。在建立存取金鑰前，請檢閱長期存取金鑰的替代方案。

存取金鑰是 IAM 使用者或 AWS 帳戶根使用者的長期憑證。您可以使用存取金鑰簽署對 AWS CLI 或 AWS API 的程式設計請求 （ 直接使用 AWS SDK)。如需詳細資訊，請參閱使用 AWS 安全登入資料進行程式設計存取。

存取金鑰包含兩個部分：存取金鑰 ID (例如 `AKIAIOSFODNN7EXAMPLE`) 和私密存取金鑰 (例如 `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`)。您必須一起使用存取金鑰 ID 和私密存取金鑰來驗證您的請求。

建立存取金鑰對時，將存取金鑰 ID 和私密存取金鑰儲存在安全位置。私密存取金鑰只能在您建立時擷取。如果您遺失了私密存取金鑰，則必須刪除該存取金鑰並新建一個。如需更多說明，請參閱 更新存取金鑰。

每位使用者最多可以擁有兩個存取金鑰。

> **Important**
>
> 具有存取金鑰的 IAM 使用者可能給帳戶帶來安全風險。安全地管理存取金鑰。請勿將您的存取金鑰提供給未經授權的當事方，即便是協助尋找您的帳戶識別符也不妥。如果這麼做，就可能會讓他人能夠永久存取您的帳戶。
>
> 使用存取金鑰時，請注意下列事項：
>
> - 請勿使用帳戶的根憑證來建立存取金鑰。
> - 請勿將存取金鑰或憑證資訊放置在應用程式檔案中。
> - 請勿將含有存取金鑰或憑證資訊的檔案包含在專案區域中。
> - 存放在共用登入資料檔案中的存取金鑰或 AWS 登入資料資訊會以純文字儲存。

監控建議

建立存取金鑰之後：

- Use AWS CloudTrail 可監控存取金鑰使用情況，並偵測任何未經授權的存取嘗試。如需詳細資訊，請參閱使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail。
- 設定 CloudWatch 警示，以便通知管理員被拒絕的存取嘗試，協助偵測惡意活動。如需更多資訊，請參閱 Amazon CloudWatch 使用者指南。
- 視需要定期檢閱、更新和刪除存取金鑰。

下列主題將詳細說明與存取金鑰相關的管理任務。

主題

- 透過將內嵌政策連接至 IAM 使用者來控制存取金鑰的使用
- 管理存取金鑰所需的許可
- IAM 使用者如何管理自己的存取金鑰
- IAM 管理員如何管理 IAM 使用者存取金鑰
- 更新存取金鑰
- 保護存取金鑰的安全

透過將內嵌政策連接至 IAM 使用者來控制存取金鑰的使用

建議的最佳實務是，工作負載使用具有 IAM 角色的臨時憑證來存取 AWS。具有存取金鑰的 IAM 使用者應獲指派最低權限存取權，並啟用多重要素驗證 (MFA)。如需有關擔任 IAM 角色的詳細資訊，請參閱 擔任角色的方法。

不過，如果您要建立服務自動化或其他短期使用案例的概念驗證測試，並選擇使用具有存取金鑰的 IAM 使用者執行工作負載，建議您使用政策條件來進一步限制其 IAM 使用者憑證的存取。

在這種情況下，您可以建立有時限的政策，使憑證在指定的時間後過期；或者，如果您是從安全網路執行工作負載，則可以使用 IP 限制政策。

在這兩種使用案例中，您都可以使用連接至具有存取金鑰之 IAM 使用者的內嵌政策。

為 IAM 使用者設定有時限的政策

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇使用者，然後選取短期使用案例的使用者。如果您尚未建立使用者，現在可以建立使用者。
3. 在使用者詳細資訊頁面上，選擇許可索引標籤。

4. 選擇新增許可，然後選取建立內嵌政策。
5. 在政策編輯器區段中，選取 JSON 以顯示 JSON 編輯器。
6. 在 JSON 編輯器中，輸入下列政策，將 `aws:CurrentTime` 時間戳記的值取代為所需的過期日期和時間：

   JSON

   ```
   {
   "Version":"2012-10-17",
     "Statement": [
       {
         "Effect": "Deny",
         "Action": "*",
         "Resource": "*",
         "Condition": {
         "DateGreaterThan": {
         "aws:CurrentTime": "2025-03-01T00:12:00Z"
           }
         }
       }
     ]
   }
   ```

   此政策使用 Deny 效果來限制在指定日期後對所有資源執行的所有動作。`DateGreaterThan` 條件會將目前時間與您設定的時間戳記進行比較。
7. 選取 下一步 以繼續前往 檢閱和建立 頁面。在政策詳細資訊中，在政策名稱下輸入政策的名稱，然後選擇建立政策。

建立政策後，其會顯示在使用者的許可索引標籤上。當目前時間大於或等於政策中指定的時間時，使用者將無法再存取 AWS 資源。請務必通知工作負載開發人員您為這些存取金鑰指定的過期日期。

為 IAM 使用者設定 IP 限制政策

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇使用者，然後選取將從安全網路執行工作負載的使用者。如果您尚未建立使用者，現在可以建立使用者。

3. 在使用者詳細資訊頁面上，選擇許可索引標籤。
4. 選擇新增許可，然後選取建立內嵌政策。
5. 在政策編輯器區段中，選取 JSON 以顯示 JSON 編輯器。
6. 將下列 IAM 政策複製到 JSON 編輯器，並根據您的需求更改公有 IPv4 或 IPv6 位址或範圍。您可以使用 https://checkip.amazonaws.com 來確定目前的公有 IP 位址。可以使用斜線表示法指定個別 IP 位址或 IP 位址範圍。如需詳細資訊，請參閱aws:SourceIp。

> **Note**
>
> 不得透過 VPN 或代理伺服器來混淆 IP 位址。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"IpRestrictionIAMPolicyForIAMUser",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

在此政策範例中，除非請求源自於網路 (以 CIDR 表示法指定) "203.0.113.0/24"、"2001:DB8:1234:5678::/64" 或特定 IP 位址 "203.0.114.1"，否則此政策會拒絕使用套用了此政策的 IAM 使用者的存取金鑰

7. 選取 下一步 以繼續前往 檢閱和建立 頁面。在政策詳細資訊中，在政策名稱下輸入政策的名稱，然後選擇建立政策。

建立政策後，其會顯示在使用者的許可索引標籤上。

您也可以將此政策做為服務控制政策 (SCP) 套用至 中的多個 AWS 帳戶 AWS Organizations，我們建議您使用其他條件，`aws:PrincipalArn`讓此政策陳述式僅適用於受此 SCP 規範 AWS 的帳戶內的 IAM 使用者。下列政策包含該更新：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IpRestrictionServiceControlPolicyForIAMUsers",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:user/*"
        }
      }
    }
  ]
```

```
}
```

管理存取金鑰所需的許可

> **Note**
>
> `iam:TagUser` 是新增和編輯存取金鑰描述的選用許可。如需詳細資訊，請參閱標記 IAM 使用者

為 IAM 使用者建立存取金鑰，您必須擁有以下政策的許可：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

為您自己的 IAM 使用者更新存取金鑰，您必須擁有以下政策的許可：

JSON

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

```
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

IAM 使用者如何管理自己的存取金鑰

IAM 管理員可以透過連接管理存取金鑰所需的許可中所述的政策，授予 IAM 使用者自行管理其存取金鑰的許可。

透過這些許可，IAM 使用者可以使用下列程序來建立、啟用、停用和刪除與其使用者名稱關聯的存取金鑰。

主題

- 為自己建立存取金鑰 (主控台)
- 停用您的存取金鑰 (主控台)
- 啟用您的存取金鑰 (主控台)
- 刪除您的存取金鑰 (主控台)

為自己建立存取金鑰 (主控台)

如果您已獲授予適當的許可，您可以使用 為自己 AWS 管理主控台 建立存取金鑰。

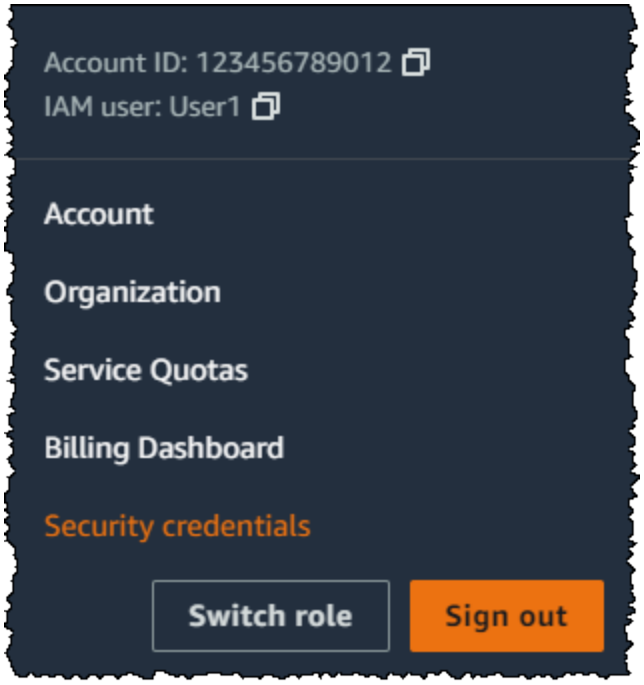
若要建立您自己的存取金鑰 (主控台)

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

> **Note**
>
> 為了方便起見，AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入要重新導向至 AWS 您帳戶的 IAM 使用者登入頁面的帳戶 ID 或帳戶別名。

若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。

2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在存取金鑰區段中，選擇建立存取金鑰。如果您已經有兩個存取金鑰，則此按鈕將被停用，您必須先刪除一個存取金鑰才能建立新的存取金鑰。
4. 在 Access key best practices & alternatives (存取金鑰最佳實務與替代方案) 頁面上，選擇您的使用案例以了解可協助您避免建立長期存取金鑰的其他選項。若您判定您的使用案例仍需要存取金鑰，請選取 Other (其他)，然後再選擇 Next (下一步)。
5. (選用) 為存取金鑰設定描述標籤值。這將新增標籤鍵值對到您的 IAM 使用者。這可協助您在未來辨別與更新存取金鑰。標籤索引鍵被設定為存取金鑰 id。標籤值被設定為您指定的存取金鑰描述。完成時，選擇 Create access key (建立存取金鑰)。

6. 在 Retrieve access keys (擷取存取金鑰) 頁面上，選擇 Show (顯示) 以顯示您的使用者的私密存取金鑰的值，或選擇 Download .csv file (下載 .csv 檔案)。這是您儲存您的私密存取金鑰的唯一機會。在將您的私密存取金鑰儲存到安全位置以後，選取 Done (完成)。

停用您的存取金鑰 (主控台)

如果您已獲授予適當的許可，您可以使用 AWS 管理主控台 來停用您的存取金鑰。

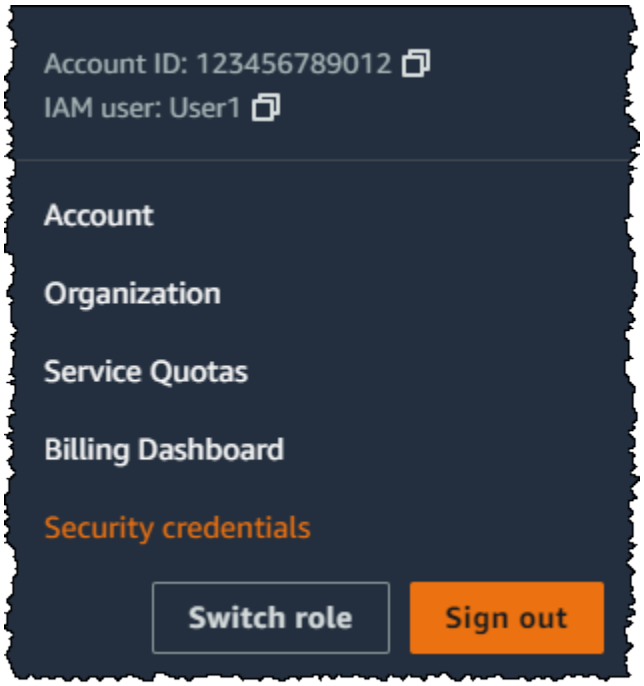
要停用存取金鑰

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

> **Note**
>
> 為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入要重新導向至 AWS 您帳戶的 IAM 使用者登入頁面的帳戶 ID 或帳戶別名。

   若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。

2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在 Access keys (存取金鑰) 區段中，找到您想要停用的金鑰，然後選取 Actions (動作)，再選擇 Deactivate (停用)。出現確認提示時，請選取 Deactivate (停用)。停用的存取金鑰仍會計入兩個存取金鑰的限制。

啟用您的存取金鑰 (主控台)

如果您已獲授予適當的許可，您可以使用 AWS 管理主控台 來啟用您的存取金鑰。

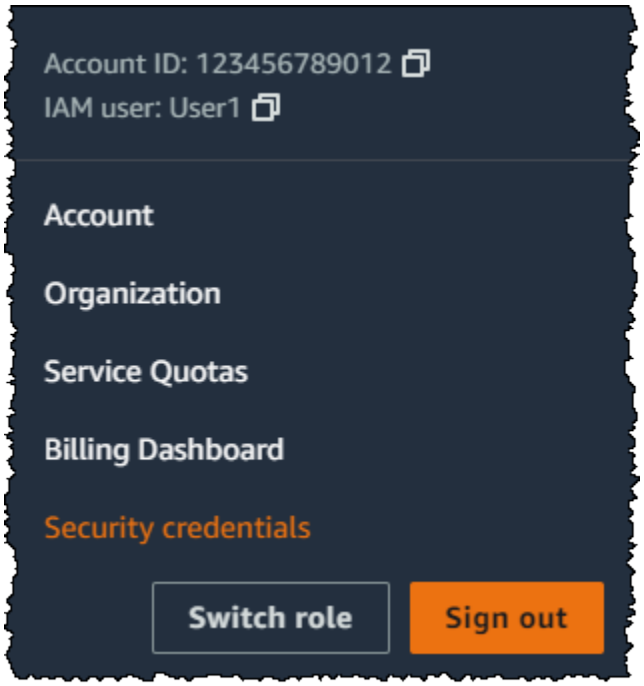
要啟用存取金鑰

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

> **Note**
>
> 為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入要重新導向至 AWS 您帳戶的 IAM 使用者登入頁面的帳戶 ID 或帳戶別名。

   若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。

2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在 Access keys (存取金鑰) 區段中，找到要啟用的金鑰，然後選取 Actions (動作)，再選擇 Activate (啟用)。

刪除您的存取金鑰 (主控台)

如果您已獲授予適當的許可，您可以使用 AWS 管理主控台 刪除您的存取金鑰。

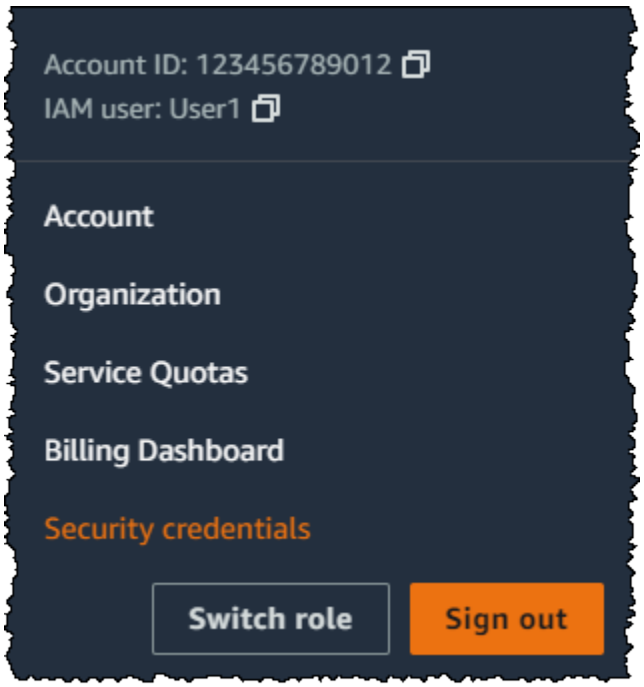
要刪除不再需要的存取金鑰

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

> **Note**
>
> 為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入要重新導向至 AWS 您帳戶的 IAM 使用者登入頁面的帳戶 ID 或帳戶別名。

若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。

2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在 Access keys (存取金鑰) 區段中，找到您想要刪除的金鑰，然後選取 Actions (動作)，再選擇 Delete (刪除)。依照對話中的指示先 Deactivate (停用)，然後再確認刪除。建議您在永久刪除存取金鑰前驗證不再使用該存取金鑰。

IAM 管理員如何管理 IAM 使用者存取金鑰

IAM 管理員可以建立、啟用、停用和刪除與個別 IAM 使用者關聯的存取金鑰。他們也可以列出帳戶中具有存取金鑰的 IAM 使用者，並找出哪些 IAM 使用者具有特定的存取金鑰。

主題

- 為 IAM 使用者建立存取金鑰
- 若要停用 IAM 使用者的存取金鑰
- 若要啟用 IAM 使用者的存取金鑰
- 刪除 IAM 使用者的存取金鑰
- 若要列出 IAM 使用者的存取金鑰
- 若要顯示您帳戶中的使用者的所有存取金鑰 ID
- 若要使用存取金鑰 ID 尋找使用者
- 若要尋找存取金鑰 ID 的最近使用

為 IAM 使用者建立存取金鑰

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 選擇使用者名稱以前往使用者詳細資訊頁面。
4. 在安全性憑證索引標籤的存取金鑰區段中，選擇建立存取金鑰。

   如果按鈕被停用，則您必須先刪除其中一個現有的金鑰，才能建立新的金鑰。
5. 在 Access key best practices & alternatives (存取金鑰最佳實務與替代方案) 頁面上，檢閱最佳實務和替代方案。選擇您的使用案例以了解可協助您避免建立長期存取金鑰的其他選項。
6. 若您判定您的使用案例仍需要存取金鑰，請選取 Other (其他)，然後再選擇 Next (下一步)。
7. (選用) 可以在設定描述標籤頁面上將描述標籤新增至存取金鑰，協助追蹤存取金鑰。選取建立存取金鑰。

8. 在 Retrieve access keys (擷取存取金鑰) 頁面上，選擇 Show (顯示) 以顯示您的使用者的私密存取金鑰的值。
9. 如要將存取金鑰 ID 和私密存取金鑰儲存到 `.csv` 檔案並在您電腦中的安全位置保管，請選擇 Download .csv file (下載 .csv 檔案) 按鈕。

> **Important**
>
> 這是您檢視或下載新建立之存取金鑰的唯一機會，之後您無法再將其復原。請務必安全地保管存取金鑰。

當您為您的使用者建立存取金鑰時，在預設情況下，該金鑰對是作用中的，且您的使用者可以立即使用該金鑰對。

AWS CLI

執行以下命令：

- `aws iam create-access-key`

API

呼叫以下操作：

- `CreateAccessKey`

若要停用 IAM 使用者的存取金鑰

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 選擇使用者名稱以前往使用者詳細資訊頁面。
4. 在安全憑證索引標籤的存取金鑰區段中，選擇動作下拉式功能表，然後選擇停用。
5. 在停用對話方塊中，選擇停用，確認您想要停用存取金鑰

停用存取金鑰後，API 呼叫便無法再使用它。如有需要，可以再次啟用。

AWS CLI

執行以下命令：

- aws iam update-access-key

API

呼叫以下操作：

- UpdateAccessKey

若要啟用 IAM 使用者的存取金鑰

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 選擇使用者名稱以前往使用者詳細資訊頁面。
4. 在安全憑證索引標籤的存取金鑰區段中，選擇動作下拉式功能表，然後選擇啟用。

啟用存取金鑰後，API 呼叫便可以使用。如有需要，可以再次停用。

AWS CLI

執行以下命令：

- aws iam update-access-key

API

呼叫以下操作：

- UpdateAccessKey

刪除 IAM 使用者的存取金鑰

停用存取金鑰後，如果不再需要，請將其刪除。

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 選擇使用者名稱以前往使用者詳細資訊頁面。
4. 在安全憑證索引標籤的存取金鑰區段中，選擇非活動存取金鑰的動作下拉式功能表，然後選擇刪除。
5. 在刪除對話方塊中，在文字輸入欄位中輸入存取金鑰 ID，然後選擇刪除，確認您想要刪除存取金鑰。

刪除存取金鑰後，便無法復原。

AWS CLI

執行以下命令：

- `aws iam delete-access-key`

API

呼叫以下操作：

- `DeleteAccessKey`

若要列出 IAM 使用者的存取金鑰

您可以檢視與 IAM 使用者相關聯的存取金鑰 ID 清單。

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 選擇使用者名稱以前往使用者詳細資訊頁面。
4. 在安全憑證索引標籤上，存取金鑰區段會列出使用者的存取金鑰 ID，包括所顯示的每個金鑰的狀態。

> **Note**
>
> 只有使用者的存取金鑰 ID 是可見的。只有在建立金鑰時才能擷取私密存取金鑰。

每個 IAM 使用者可以有兩個存取金鑰。

AWS CLI

執行以下命令：

- `aws iam list-access-keys`

API

呼叫以下操作：

- `ListAccessKeys`

若要顯示您帳戶中的使用者的所有存取金鑰 ID

您可以在 AWS 帳戶中檢視使用者的存取金鑰 ID 清單。

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 選擇使用者名稱以前往使用者詳細資訊頁面。
4. 如有必要，請透過完成以下步驟將存取金鑰 ID 資料欄新增到使用者表格：
   a. 在資料表最右側的上方，選擇偏好設定圖示
   ()。
   b. 在偏好設定對話方塊的選取可見資料欄下，開啟存取金鑰 ID。
   c. 選擇 確認 返回使用者清單。清單更新為包含存取金鑰 ID。
5. 存取金鑰 ID 欄會顯示每個存取金鑰的狀態，其後跟隨其 ID，例如，**`Active - AKIAIOSFODNN7EXAMPLE`** 或**`Inactive - AKIAI44QH8DHBEXAMPLE`**。

您可以使用此資訊來檢視和複製具有一個或兩個存取金鑰的使用者的存取金鑰 ID。該欄會向沒有存取金鑰的使用者顯示 -。

> **Note**
>
> 只有在建立金鑰時才能擷取私密存取金鑰。

每個 IAM 使用者可以有兩個存取金鑰。

若要使用存取金鑰 ID 尋找使用者

您可以使用存取金鑰 ID 在 AWS 帳戶中尋找使用者。

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格的搜尋方塊中輸入存取金鑰 ID，例如 AKIAI44QH8DHBEXAMPLE。
3. 與存取金鑰 ID 關聯的 IAM 使用者出現在導覽窗格中。選擇使用者名稱以前往使用者詳細資訊頁面。

若要尋找存取金鑰 ID 的最近使用

存取金鑰的最近使用會顯示在 IAM 使用者頁面的使用者清單中、使用者詳細資訊頁面上，並且是憑證報告的一部分。

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在使用者清單中，檢視上次使用的存取金鑰資料欄。

   如果未顯示該資料欄，請選擇偏好設定圖示
   ( )，
   然後在選取可見資料欄下方開啟上次使用的存取金鑰以顯示該資料欄。

3. (選用) 在導覽窗格的存取報告下方，選取憑證報告以下載包含您帳戶中所有 IAM 使用者之存取金鑰上次使用資訊的報告。
4. (選用) 選取 IAM 使用者以檢視使用者詳細資訊。摘要區段包含存取金鑰 ID、其狀態，以及上次使用的時間。

AWS CLI

執行以下命令：

- `aws iam get-access-key-last-used`

API

呼叫以下操作：

- `GetAccessKeyLastUsed`

更新存取金鑰

作為安全最佳實務，我們建議在需要時更新 IAM 使用者存取金鑰，如當員工離職時。如果 IAM 使用者已獲得所需許可，他們可以更新自己的存取金鑰。

如需有關授予 IAM 使用者許可，讓他們可更新自己的存取金鑰的詳細資訊，請參閱 AWS：允許 IAM 使用者在「安全憑證」頁面中管理其密碼、存取金鑰和 SSH 公有金鑰。您還可以套用密碼政策到您的帳戶，以要求所有 IAM 使用者定期更新其密碼並規定必須多久更新一次。如需詳細資訊，請參閱設定 IAM 使用者的帳戶密碼政策。

> **Note**
>
> 如果您遺失了私密存取金鑰，則必須刪除該存取金鑰並新建一個。私密存取金鑰只能在您建立時擷取。使用此步驟進行停用，然後使用新的憑證取代任何遺失的存取金鑰。

主題

- 更新 IAM 使用者存取金鑰 (主控台)
- 更新存取金鑰 (AWS CLI)
- 更新存取金鑰 (AWS API)

更新 IAM 使用者存取金鑰 (主控台)

您可以從 AWS 管理主控台中更新存取金鑰。

在不會中斷您的應用程式下更新 IAM 使用者的存取金鑰 (主控台)

1. 當第一個存取金鑰仍然有效時，建立第二個存取金鑰。

   a. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。

   b. 在導覽窗格中，選擇 Users (使用者)。

   c. 選擇目標使用者的名稱，然後選擇 Security credentials (安全憑證) 索引標籤。

   d. 在存取金鑰區段中，選擇建立存取金鑰。在 Access key best practices & alternatives (存取金鑰最佳實務與替代方案) 頁面上，選取 Other (其他)，然後再選擇 Next (下一步)。

   e. (選用) 為存取金鑰設定描述標籤值，以新增標籤鍵值對到此 IAM 使用者。這可協助您在未來辨別與更新存取金鑰。標籤索引鍵被設定為存取金鑰 id。標籤值被設定為您指定的存取金鑰描述。完成時，選擇 Create access key (建立存取金鑰)。

   f. 在 Retrieve access keys (擷取存取金鑰) 頁面上，選擇 Show (顯示) 以顯示您的使用者的私密存取金鑰的值，或選擇 Download .csv file (下載 .csv 檔案)。這是您儲存您的私密存取金鑰的唯一機會。在將您的私密存取金鑰儲存到安全位置以後，選取 Done (完成)。

      當您為您的使用者建立存取金鑰時，在預設情況下，該金鑰對是作用中的，且您的使用者可以立即使用該金鑰對。此時，使用者有兩個作用中的存取金鑰。

2. 更新所有應用程式和工具以使用新的存取金鑰。

3. 透過查看 Last used (上次使用) 資訊中最舊的存取金鑰，判斷第一個存取金鑰是否仍在使用中。其中一個方法是等待幾天，然後在繼續之前檢查舊的存取金鑰以供使用。

4. 即使 Last used (上次使用) 資訊指示從未使用舊金鑰，我們建議您不要立即刪除第一個存取金鑰。反之，選擇 Actions (動作)，然後選擇 Deactivate (停用) 來停用第一個存取金鑰。

5. 僅使用新的存取金鑰來確認您的應用程式正在工作。仍然使用原始存取金鑰的任何應用程式和工具此時都會停止運作，因為他們不再能夠存取 AWS 資源。如果您找到此類應用程式或工具，則可以重新啟用第一個存取金鑰。然後，返回 Step 3 並更新此應用程式以使用新的金鑰。

6. 在您等待一段時間以確保所有應用程式和工具都已更新後，您可以刪除第一個存取金鑰：

   a. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。

   b. 在導覽窗格中，選擇 Users (使用者)。

c. 選擇目標使用者的名稱，然後選擇 Security credentials (安全憑證) 索引標籤。

d. 在 Access keys (存取金鑰) 區段中，找到您想要刪除的存取金鑰，然後選取 Actions (動作)，再選擇 Delete (刪除)。依照對話中的指示先 Deactivate (停用)，然後再確認刪除。

判斷需要更新或刪除哪些存取金鑰 （主控台）

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)。
3. 如有必要，請透過完成以下步驟將 Access key age (存取金鑰使用期限) 欄新增到使用者表格：

   a. 在最右側的表格上方，選擇設定圖示 ( )。

   b. 在 Manage columns (管理欄) 中，選取 Access key age (存取金鑰使用期限)。

   c. 選擇 Close (關閉) 返回使用者清單。

4. Access key age (存取金鑰使用期限) 列顯示自建立最早的作用中存取金鑰以來的天數。您可以使用此資訊來尋找可能需要更新或刪除存取金鑰的使用者。對於沒有存取金鑰的使用者，該欄會顯示 None (無)。

更新存取金鑰 (AWS CLI)

您可以從 AWS Command Line Interface中更新存取金鑰。

在不會中斷您的應用程式下更新存取金鑰 (AWS CLI)

1. 當第一個存取金鑰仍然有效時，建立第二個存取金鑰，該索引鍵在預設情況下處於作用中。執行以下命令：

   - `aws iam create-access-key`

   此時，使用者有兩個作用中的存取金鑰。
2. 更新所有應用程式和工具以使用新的存取金鑰。
3. 使用此命令判斷第一個存取金鑰是否仍在使用中：

   - `aws iam get-access-key-last-used`

   其中一個方法是等待幾天，然後在繼續之前檢查舊的存取金鑰以供使用。

4. 即使步驟 Step 3 表示不使用舊金鑰，我們也建議您不要立即刪除第一個存取金鑰。反之，使用此命令將第一個存取金鑰的狀態變更為 Inactive：

   - aws iam update-access-key

5. 僅使用新的存取金鑰來確認您的應用程式正在工作。仍然使用原始存取金鑰的任何應用程式和工具此時都會停止運作，因為他們不再能夠存取 AWS 資源。如果找到此類應用程式或工具，則可以將其狀態切換回 Active 以重新啟用第一個存取金鑰。然後，返回步驟 Step 2 並更新此應用程式以使用新的金鑰。
6. 在您等待一段時間以確保所有應用程式和工具都已更新後，您可以使用此命令刪除第一個存取金鑰：

   - aws iam delete-access-key

更新存取金鑰 (AWS API)

您可以使用 AWS API 更新存取金鑰。

在不中斷應用程式 (AWS API) 的情況下更新存取金鑰

1. 當第一個存取金鑰仍然有效時，建立第二個存取金鑰，該索引鍵在預設情況下處於作用中。呼叫以下操作：

   - CreateAccessKey

   此時，使用者有兩個作用中的存取金鑰。
2. 更新所有應用程式和工具以使用新的存取金鑰。
3. 透過呼叫此操作判斷第一個存取金鑰是否仍在使用中：

   - GetAccessKeyLastUsed

   其中一個方法是等待幾天，然後在繼續之前檢查舊的存取金鑰以供使用。
4. 即使步驟 Step 3 表示不使用舊金鑰，我們也建議您不要立即刪除第一個存取金鑰。相反地，Inactive透過呼叫此操作將第一個存取金鑰的狀態變更為 ：

   - UpdateAccessKey
5. 僅使用新的存取金鑰來確認您的應用程式正在工作。仍然使用原始存取金鑰的任何應用程式和工具此時都會停止運作，因為他們不再能夠存取 AWS 資源。如果找到此類應用程式或工具，則可以將

其狀態切換回 `Active` 以重新啟用第一個存取金鑰。然後，返回步驟 Step 2 並更新此應用程式以使用新的金鑰。

6. 在您等待一段時間以確保所有應用程式和工具都已更新後，您可以透過呼叫此操作來刪除第一個存取金鑰：

   - `DeleteAccessKey`

保護存取金鑰的安全

擁有您存取金鑰的任何人都擁有與您執行之 AWS 資源相同的存取層級。因此， AWS 會竭盡全力保護您的存取金鑰，而且為了符合我們的共同責任模型，您也應該這麼做。

展開下列各節，以取得協助您保護存取金鑰的指引。

> **Note**
>
> 您的組織可能具有與本主題中所述不同的安全要求和政策。這裡提供的建議旨在做為一般指導方針。

移除（或不產生） AWS 帳戶根使用者 存取金鑰

保護帳戶其中一個最佳方法是，不要有 AWS 帳戶根使用者的存取金鑰。除非您必須要有根使用者存取金鑰 (少數情況下)，否則最好不要產生存取金鑰。反之，請在 中 AWS IAM Identity Center 為每日管理任務建立管理使用者。如需如何在 IAM Identity Center 中建立管理使用者的相關資訊，請參閱《IAM Identity Center 使用者指南》中的入門。

如果您已經有帳戶的根使用者存取金鑰，建議您執行以下步驟：找出應用程式中目前使用金鑰 (如果有) 的地方，以 IAM 使用者存取金鑰取代根使用者存取金鑰。然後停用並移除根使用者存取金鑰。如需有關如何更新存取金鑰的詳細資訊，請參閱 更新存取金鑰。

使用臨時安全憑證 (IAM 角色)，而不是長期存取金鑰

在許多情況下，您不必像 IAM 使用者一樣需要永遠不會過期的長期存取金鑰。反之，您可以建立 IAM 角色並產生暫時安全憑證。臨時安全登入資料包含存取金鑰 ID 和私密存取金鑰，但其中也包含指出登入資料何時到期的安全符記。

例如與 IAM 使用者和根使用者相關聯的長期存取金鑰，會一直保持有效，直到您手動撤銷為止。不過，透過 IAM 角色和其他 功能取得的臨時安全登入資料會在短時間內 AWS Security Token Service 過期。使用臨時安全登入資料，可協助降低風險，以防登入資料意外洩露。

請在以下案例中使用 IAM 角色和臨時安全憑證：

- 您有應用程式或 AWS CLI 指令碼在 Amazon EC2 執行個體上執行。請勿直接在應用程式中使用存取金鑰。請勿將存取金鑰傳遞至應用程式、將其嵌入應用程式中，或讓應用程式從任何來源讀取存取金鑰。反之，請定義具有應用程式適當權許可的 IAM 角色，然後使用 EC2 角色啟動 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。這樣做會將 IAM 角色與 Amazon EC2 執行個體相關聯。這種做法也可讓應用程式取得臨時安全憑證，以便用來對 AWS進行程式化呼叫。 AWS SDKs 和 AWS Command Line Interface (AWS CLI) 可以自動從角色取得臨時登入資料。
- 您需要授予跨帳戶存取權。使用 IAM 角色培養帳戶之間的信任，然後對一個帳戶中的使用者授予有限的許可，以存取信任的帳戶。如需詳細資訊，請參閱IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權。
- 您有一個行動應用程式。請勿使用應用程式內嵌存取金鑰，甚至是內嵌於加密儲存中。反之，請使用 Amazon Cognito 來管理應用程式中的使用者身分。此服務可讓您使用 Login with Amazon、Facebook、Google 或任何與身分提供者相容的 OpenID Connect (OIDC)，驗證使用者的身分。然後，您可以使用 Amazon Cognito 憑證供應商，管理應用程式用來向 AWS提出請求的憑證。
- 您想要與 聯合， AWS 且您的組織支援 SAML 2.0。如果您處理的組織是具有支援 SAML 2.0 的身分識別提供者的組織，請將提供者設定為使用 SAML。您可以使用 SAML 與 交換身分驗證資訊， AWS 並取回一組臨時安全登入資料。如需詳細資訊，請參閱SAML 2.0 聯合身分。
- 您想要與 聯合， AWS 而且您的組織具有內部部署身分存放區。如果使用者可以在您的組織內部進行身分驗證，您可以撰寫應用程式，向他們發出臨時安全登入資料以存取 AWS 資源。如需詳細資訊，請參閱啟用 AWS 主控台的自訂身分代理程式存取。
- 在 IAM 政策中使用條件，僅允許從預期網路進行存取。您可以透過實作具有條件的 IAM 政策來限制存取金鑰的使用位置和方式，這些條件指定並僅允許預期網路，例如公有 IP 位址或虛擬私有雲端 (VPC)。如此一來，便能確保存取金鑰僅從預期且可接受的網路使用。

> **Note**
>
> 您是否將 Amazon EC2 執行個體與需要以程式設計方式存取 AWS 資源的應用程式搭配使用？如果是，請使用適用於 EC2 的 IAM 角色。

適當管理 IAM 使用者存取金鑰

如果您必須為程式設計存取建立存取金鑰 AWS，請為 IAM 使用者建立存取金鑰，僅授予使用者所需的許可。

遵循這些預防措施，以協助保護 IAM 使用者存取金鑰：

- 請勿將存取金鑰直接嵌入程式碼。AWS SDK 和 AWS 命令列工具可讓您將存取金鑰放入已知位置，如此一來，就不必將其留在程式碼中。

  將存取金鑰放入以下其中一個位置：

  - AWS 登入資料檔案。SDKs和 AWS AWS CLI 會自動使用您在登入資料檔案中存放的 AWS 登入資料。

    如需有關使用 AWS 登入資料檔案的資訊，請參閱 SDK 的文件。範例包括《 適用於 Java 的 AWS SDK 開發人員指南》中的設定 AWS 登入資料和區域，以及AWS Command Line Interface 《 使用者指南》中的組態和登入資料檔案。

    若要存放 適用於 .NET 的 AWS SDK 和 的登入資料 AWS Tools for Windows PowerShell，建議您使用 SDK 存放區。如需詳細資訊，請參閱《適用於 .NET 的 AWS SDK 開發人員指南》中的使用 SDK Store。
  - 環境變數. 在多租用戶系統中，請選擇使用者環境變數，而不是系統環境變數。

    如需有關使用環境變數來存放憑證的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的環境變數。
- 對不同的應用程式使用不同的存取金鑰。如果它們已經公開，這麼做可以隔離許可，並撤銷個別應用程式的存取金鑰。針對不同的應用程式擁有獨立的存取金鑰，也會在 AWS CloudTrail 日誌檔中產生不同的項目。此組態可讓您更容易判斷哪個應用程式執行特定動作。
- 視需要更新存取金鑰。若有存取金鑰可能遭洩露的風險，請更新該存取金鑰並刪除先前的存取金鑰。如需詳細資訊，請參閱更新存取金鑰
- 移除未使用的存取金鑰。如果使用者離開組織，請移除相對應的 IAM 使用者，使該使用者無法再存取您的資源。若要了解上次使用存取金鑰的時間，請使用 `GetAccessKeyLastUsed` API (AWS CLI 命令：`aws iam get-access-key-last-used`)。
- 為最機密的 API 操作使用臨時憑證並設定多重驗證。利用 IAM 政策，可以指定允許使用者呼叫的 API 操作。在某些情況下，您可能希望在允許使用者執行特別敏感的動作之前，要求使用者使用 AWS MFA 進行身分驗證的額外安全性。例如，您可能擁有允許使用者執行 Amazon EC2 `RunInstances`、`DescribeInstances` 與 `StopInstances` 動作的政策。但您可能想要限制類

似 的破壞性動作，`TerminateInstances`並確保使用者只有在使用 AWS MFA 裝置進行身分驗證時，才能執行該動作。如需詳細資訊，請參閱透過 MFA 實現安全的 API 存取。

使用存取金鑰 AWS 存取行動應用程式

您可以使用 AWS 行動應用程式存取一組有限的 AWS 服務和功能。行動應用程式可協助您在外出時支援事件回應。如需詳細資訊及下載應用程式，請參閱 AWS 主控台行動應用程式。

您可以使用主控台密碼或存取金鑰登入行動應用程式。根據最佳實務，請勿使用根使用者存取金鑰。相反地，我們強烈建議您除了在行動裝置上使用密碼或生物識別鎖定之外，還要建立專門用於使用行動應用程式管理 AWS 資源的 IAM 使用者。如果您遺失行動裝置，您可以移除 IAM 使用者的存取權。

使用存取金鑰登入 (行動應用程式)

1. 在行動裝置上開啟應用程式。
2. 如果這是您第一次將身分新增至裝置，請依序選擇 Add an identity (新增身分) 及選擇 Access keys (存取金鑰)。

   如果您已經使用其他身分登入，請依序選擇功能表圖示及 Switch identity (切換身分)。然後依序選擇 Sign in as a different identity (以不同的身分登入) 及 Access keys (存取金鑰)。
3. 在 Access keys (存取金鑰) 頁面輸入您的資訊。

   - 存取金鑰 ID – 輸入您的存取金鑰 ID。
   - 私密存取金鑰 – 輸入您的私密存取金鑰。
   - 身分名稱 – 輸入將顯示在行動應用程式中的身分識別名稱。這不需要與您的 IAM 使用者名稱相符。
   - 身分識別 PIN – 建立您將在未來登入期間使用的個人識別碼 (PIN)。

   > **Note**
   >
   > 如果您為 AWS 行動應用程式啟用生物識別，系統會提示您使用您的指紋或臉部辨識進行驗證，而不是 PIN 碼。如果生物特徵辨識失敗，系統可能會提示您輸入 PIN 碼。
4. 選擇 Verify and add keys (驗證並新增金鑰)。

   您現在可以使用行動應用程式存取精選的一組資源。

相關資訊

下列主題提供設定 AWS SDKs和 AWS CLI 以使用存取金鑰的指引：

- 《適用於 Java 的 AWS SDK 開發人員指南》中的設定 AWS 登入資料和區域
- 《適用於 .NET 的 AWS SDK 開發人員指南》中的使用 SDK Store
- 《適用於 PHP 的 AWS SDK 開發人員指南》中的對 SDK 提供憑證
- Boto 3（適用於 Python 的AWS SDK) 文件中的組態
- 《AWS Tools for Windows PowerShell 使用者指南》中的使用 AWS 憑證
- 《AWS Command Line Interface 使用者指南》中的組態和憑證檔案
- 《適用於 .NET 的 AWS SDK 開發人員指南》中的使用 IAM 角色授予存取權
- AWS SDK for Java 2.x 中的為 Amazon EC2 設定 IAM 角色

使用存取金鑰和秘密金鑰憑證存取主控台

除 AWS CLI外，還可以使用存取金鑰和秘密金鑰憑證直接存取 AWS 管理主控台 。這可以使用 API AWS STS `GetFederationToken`呼叫來實現。IAM 主體只需透過使用 `GetFederationToken` 提供的臨時憑證和權杖建構主控台 URL，就能存取主控台。如需詳細資訊，請參閱啟用 AWS 主控台的自訂身分代理程式存取。

值得釐清的是，直接使用 IAM 或根使用者憑證登入主控台並啟用 MFA 時，將需要 MFA。不過，如果使用上述方法 (搭配 `GetFederationToken` 使用臨時憑證)，則不需要 MFA。

稽核存取金鑰

您可以檢閱程式碼中的 AWS 存取金鑰，以判斷金鑰是否來自您擁有的帳戶。您可以使用 `aws sts get-access-key-info` AWS CLI 命令或 `GetAccessKeyInfo` AWS API 操作傳遞存取金鑰 ID。

AWS CLI 和 AWS API 操作會傳回存取金鑰所屬 AWS 帳戶 的 ID。開頭為 `AKIA` 的存取金鑰是 IAM 使用者或 AWS 帳戶根使用者的長期登入資料。以 開頭的存取金鑰 IDs `ASIA` 是使用 AWS STS 操作建立的臨時登入資料。如果回應中的帳戶是您的，您可以根使用者的身分登入並檢閱根使用者存取金鑰。然後，您可以提取憑證報告來了解擁有金鑰的是哪位 IAM 使用者。若要了解誰請求`ASIA`存取金鑰的臨時登入資料，請檢視 CloudTrail 日誌中的 AWS STS 事件。

基於安全考量，您可以檢閱 AWS CloudTrail 日誌，以了解在其中執行動作的人員 AWS。您可以使用角色信任政策中的 `sts:SourceIdentity` 條件金鑰，請求使用者在擔任角色時指定身分。例如，您

可以請求 IAM 使用者將自己的使用者名稱指定為其來源身分。這可以協助您判斷哪位使用者在 AWS中執行了特定動作。如需詳細資訊，請參閱sts:SourceIdentity。

此操作不會指出存取金鑰的狀態。金鑰可能是作用中、非作用中或已刪除。作用中金鑰可能沒有操作的執行許可。提供已刪除的存取金鑰可能會傳回金鑰不存在的錯誤。

AWS IAM 中的多重要素驗證

為了提高安全性，我們建議您設定多重要素驗證 (MFA)，以協助保護您的 AWS 資源。您可以為 AWS 帳戶根使用者 啟用 MFA AWS 帳戶，包括獨立帳戶、管理帳戶和成員帳戶，以及您的 IAM 使用者。我們建議您盡可能使用網路釣魚防護 MFA，例如通行金鑰和安全金鑰。這些 FIDO 型驗證器使用公有金鑰密碼編譯，可抵禦網路釣魚、man-in-the-middle和重播攻擊，提供比 TOTP 型選項更強的安全性。

對於所有類型的帳戶，其根使用者均強制執行 MFA。如需詳細資訊，請參閱保護 AWS Organizations 您的帳戶根使用者登入資料。

當您為根使用者啟用 MFA 時，它僅影響根使用者憑證。帳戶中的 IAM 使用者都有自己憑證的不同身分，並且每個身分都有自己的 MFA 組態。如需有關使用 MFA 保護根使用者的詳細資訊，請參閱的多重要素驗證 AWS 帳戶根使用者。

您的 AWS 帳戶根使用者 和 IAM 使用者可以註冊最多八個任何類型的 MFA 裝置。註冊多個 MFA 裝置可以提供彈性，並協助您降低裝置遺失或損壞時存取中斷的風險。您只需要一個 MFA 裝置登入 AWS 管理主控台 ，或透過 AWS CLI建立工作階段。

> **Note**
>
> 我們建議您要求人類使用者在存取時使用臨時登入資料 AWS。您是否考慮過使用 AWS IAM Identity Center？ 您可以使用 IAM Identity Center 集中管理對多個 的存取， AWS 帳戶 並為使用者提供受 MFA 保護的單一登入存取，可從單一位置存取其所有指派的帳戶。使用 IAM Identity Center，您可以在 IAM Identity Center 中建立和管理使用者身分，或輕鬆連線至您現有的 SAML 2.0 相容身分提供者。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

MFA 增加了額外的安全性，除了使用者存取 AWS 網站或服務時登入憑證之外，還要求使用者從 AWS 支援的 MFA 機制提供唯一身分驗證。

MFA 類型

AWS 支援下列 MFA 類型：

內容

- 通行密鑰和安全金鑰
- 虛擬驗證器應用程式
- 硬體 TOTP 權杖

通行密鑰和安全金鑰

AWS Identity and Access Management 支援 MFA 的通行金鑰和安全金鑰。根據 FIDO 標準，通行金鑰使用公有金鑰密碼編譯來提供比密碼更安全的強大、網路釣魚防護身分驗證。 AWS 支援兩種類型的通行金鑰：裝置繫結的通行金鑰 （ 安全金鑰 ） 和同步的通行金鑰。

- 安全金鑰：這些是用作身分驗證的第二個因素的實體裝置，例如 YubiKey。單一安全金鑰可以支援多個根使用者帳戶和 IAM 使用者。
- 同步通行密鑰：這些使用來自提供者 (例如 Google、Apple、Microsoft 帳戶等) 和第三方服務 (例如 1Password、Dashlane 和 Bitwarden 等) 的憑證管理工具作為第二個因素。

您可以使用內建的生物識別驗證器，例如 Apple MacBooks 上的 Touch ID，解鎖您的憑證管理工具並登入 AWS。通行密鑰是透過您選擇的提供者，使用指紋、面部或裝置 PIN 碼建立的。您也可以使用一台裝置 (例如行動裝置或硬體安全金鑰) 中的跨帳戶身分驗證 (CDA) 通行密鑰，在筆記型電腦等其他裝置上登入。如需詳細資訊，請參閱 cross-device authentication (CDA)。

您可以跨裝置同步通行金鑰，以便使用 登入 AWS，增強可用性和可復原性。如需啟用通行密鑰和安全金鑰的詳細資訊，請參閱為根使用者啟用通行密鑰或安全金鑰 (主控台)。

FIDO Alliance 維護與 FIDO 規範相容的所有經 FIDO 認證的產品的清單。

虛擬驗證器應用程式

在手機或其他裝置上執行並模擬實體裝置的虛擬驗證器應用程式。虛擬驗證器應用程式實作以時間為基礎的一次性密碼 (TOTP) 算法，並且支援在單台裝置上使用多個權杖。使用者必須在登入期間出現提示時輸入裝置中的有效代碼。每個指派給使用者的權杖都必須是唯一的。使用者無法輸入另一個使用者的權杖來進行身分驗證。

我們建議您使用網路釣魚防護 MFA，例如通行金鑰或安全金鑰，以獲得最強大的保護。如果您還無法使用通行密鑰或安全金鑰，建議您在等待硬體購買核准或等待硬體送達時使用虛擬 MFA 裝置做為臨時措施。如需可以用來做為虛擬 MFA 裝置的支援應用程式清單，請參閱多重要素驗證 (MFA)。

如需為 IAM 使用者設定虛擬 MFA 裝置的說明，請參閱在 中指派虛擬 MFA 裝置 AWS 管理主控台。

> **Note**
>
> 當您透過 AWS 管理主控台 或在登入程序期間新增新的虛擬 MFA 裝置時，AWS 帳戶 將刪除中未指派的虛擬 MFA 裝置。未指派的虛擬 MFA 裝置是指位於您帳戶中但帳戶根使用者或 IAM 使用者並未在登入程序中使用的裝置。將這些裝置刪除後，便可將新的虛擬 MFA 裝置新增至您的帳戶。刪除這些裝置後，您還可以重複使用裝置名稱。
>
> - 若要檢視帳戶中未指派的虛擬 MFA 裝置，您可以使用 list-virtual-mfa-devices AWS CLI 命令或 API 呼叫。
> - 若要停用虛擬 MFA 裝置，您可以使用 deactivate-mfa-device AWS CLI 命令或 API 呼叫。停用後，裝置會變成未指派狀態。
> - 若要將未指派的虛擬 MFA 裝置連接至 AWS 帳戶 根使用者或 IAM 使用者，您需要裝置產生的驗證碼，以及 enable-mfa-device AWS CLI 命令或 API 呼叫。

硬體 TOTP 權杖

一種在以時間為基礎的一次性密碼 (TOTP) 演算法的基礎上產生六位數字程式碼的硬體裝置。使用者必須在登入期間在第二個網頁上輸入裝置中的有效程式碼。

這些字符僅用於 AWS 帳戶。您只能使用具有其安全共用之唯一字符種子的字符 AWS。權杖種子是權杖生產時產生的私密金鑰。從其他來源購買的權杖將無法與 IAM 一起運作。若要確保相容性，您必須從下列其中一個連結購買硬體 MFA 裝置：OTP 權杖或 OTP 顯示卡。

- 每個指派給使用者的 MFA 裝置都必須是唯一的。使用者無法輸入另一個使用者裝置的代碼來進行身分驗證。如需支援的硬體 MFA 裝置資訊，請參閱多重要素驗證 (MFA)。
- 如果想要使用實體 MFA 裝置，我們建議您使用安全金鑰作為硬體 TOTP 裝置的替代方案。安全金鑰沒有電池需求、可防禦網路釣魚，並支援單一裝置上的多個使用者。

您只能從 啟用通行金鑰或安全金鑰 AWS 管理主控台 ，不能從 AWS CLI 或 AWS API 啟用。在可以啟用安全金鑰之前，您必須擁有對裝置的實體存取權。

如需有關為 IAM 使用者設定硬體 TOTP 權杖的說明，請參閱在 中指派硬體 TOTP 字符 AWS 管理主控台。

> **Note**
>
> SMS 文字訊息型 MFA – 已 AWS 結束對啟用 SMS 多重要素驗證 (MFA) 的支援。我們建議擁有使用 SMS 簡訊式 MFA 的 IAM 使用者的客戶改為使用下列任一種替代方式：通行密鑰或安全金鑰、虛擬 (軟體式) MFA 裝置，或硬體 MFA 裝置。您可以使用已分配的 SMS MFA 裝置來識別帳戶中的使用者。在 IAM 主控台中，從導覽窗格選擇 Users (使用者)，然後在表格中 MFA 欄位尋找具有 SMS 的使用者。

MFA 建議

為了協助保護您的身分，請遵循這些 MFA AWS 身分驗證建議。

- 我們建議您使用網路釣魚防護 MFA，例如通行金鑰和安全金鑰，做為 MFA 裝置。這些以 FIDO 為基礎的驗證器可針對網路釣魚等攻擊提供最強大的保護。
- 建議您為 中的 AWS 帳戶根使用者 和 IAM 使用者啟用多個 MFA 裝置 AWS 帳戶。這可讓您提高 中的安全列 AWS 帳戶 ，並簡化管理高權限使用者的存取權，例如 AWS 帳戶根使用者。
- 您可以向 AWS 帳戶根使用者 和 IAM 使用者註冊最多八個目前支援 MFA 類型之任何組合的 MFA 裝置。使用多個 MFA 裝置時，您只需一個 MFA 裝置登入 ， AWS 管理主控台 或透過 以該使用者 AWS CLI 身分建立工作階段。IAM 使用者必須使用現有的 MFA 裝置進行身分驗證，才能啟用或停用其他 MFA 裝置。
- 如果發生遺失、遭竊或無法存取的 MFA 裝置，您可以使用其中一個剩餘的 MFA 裝置來存取 ， AWS 帳戶 而無需執行 AWS 帳戶 復原程序。如果 MFA 裝置遺失或遭竊，應取消 MFA 裝置與 IAM 主體的可能關聯。
- 使用多個 MFAs 可讓您的員工分散在地理位置或遠端工作，使用硬體型 MFA 存取 ， AWS 而不必協調員工之間單一硬體裝置的實體交換。
- 針對 IAM 主體使用額外的 MFA 裝置，可讓您在日常使用期間使用一或多個 MFA 裝置，同時將實體 MFA 裝置維護在安全的實體位置 (例如文件庫)，或是用於備份和備援的安全位置。

> **備註**
>
> - 您無法將安全金鑰或通行金鑰的 MFA 資訊傳遞給 AWS STS API 操作，以請求臨時憑證。您可以在使用安全金鑰或通行金鑰時，透過執行 `aws login`命令來取得登入資料，以搭配 AWS CLI 和 AWS SDKs 使用。
> - 您無法使用 AWS CLI 命令或 AWS API 操作來啟用 FIDO 安全金鑰。

- 同一個名稱不能用於多個根使用者或 IAM MFA 裝置。

其他資源

下列資源可協助您進一步了解 MFA。

- 如需使用 MFA 存取的詳細資訊 AWS，請參閱 啟用 MFA 的登入。
- 您可以使用 IAM Identity Center 啟用 AWS 存取入口網站、IAM Identity Center 整合應用程式和 的安全 MFA 存取 AWS CLI。如需詳細資訊，請參閱在 IAM Identity Center 中啟用 MFA。
- 若要確認您的使用者是否已啟用 MFA，請參閱 檢查 MFA 狀態。

在 中指派通行金鑰或安全金鑰 AWS 管理主控台

通行金鑰是一種多重要素驗證 (MFA) 裝置，可用來保護您的 AWS 資源。 AWS 支援同步通行金鑰和裝置繫結通行金鑰，也稱為安全金鑰。

同步通行密鑰可讓 IAM 使用者在許多裝置 (甚至是新裝置) 上存取 FIDO 登入憑證，而無需在每個帳戶上重新註冊每個裝置。同步通行密鑰包含 Google、Apple 和 Microsoft 等第一方憑證管理員，以及 1Password、Dashlane 和 Bitwarden 等第三方憑證管理員，作為第二個要素。您也可以使用裝置上的生物識別技術 (例如 TouchID、FaceID) 來解除鎖定您選擇的憑證管理員，以使用通行密鑰。

或者，裝置綁定的通行密鑰會綁定至 FIDO 安全金鑰，您可以將其插入電腦上的 USB 連接埠，然後在出現提示時點選以安全地完成登入程序。若您已搭配其他服務使用 FIDO 安全金鑰，並且它具備 AWS 支援的組態 (例如來自 Yubico 的 YubiKey 5)，您也可以搭配 AWS來使用它。否則，若您希望在 AWS 中使用 MFA 的 WebAuthn，您需要購買 FIDO 安全金鑰。此外，FIDO 安全金鑰可以支援同一裝置上的多個 IAM 或根使用者，從而增強其公用程式以確保帳戶安全。如需兩種裝置類型的規格及購買資訊，請參閱多重要素驗證。

您可以向 AWS 帳戶根使用者 和 IAM 使用者註冊最多八個目前支援 MFA 類型之任何組合的 MFA 裝置。使用多個 MFA 裝置時，您只需一個 MFA 裝置即可登入 ， AWS 管理主控台 或透過 以該使用者 AWS CLI 身分建立工作階段。我們建議您註冊多個 MFA 裝置。例如，您可以註冊內建驗證器，也可以註冊存放在實體安全位置的安全金鑰。如果無法使用內建驗證器，則可以使用已註冊的安全金鑰。對於驗證器應用程式，我們也建議您在這些應用程式中啟用雲端備份或同步功能，以協助避免在具有驗證器應用程式的裝置遺失或損壞時，失去對帳戶的存取權限。

> **Note**
>
> 存取 AWS時，我們建議您要求人類使用者使用暫時性憑證。您的使用者可以 AWS 與身分提供者聯合到 ，並在其中使用其公司登入資料和 MFA 組態進行身分驗證。若要管理對 AWS 和商業應用程式的存取，建議您使用 IAM Identity Center。如需詳細資訊，請參閱 IAM Identity Center 使用者指南。

主題

- 必要許可
- 為您自己的 IAM 使用者啟用通行密鑰或安全金鑰 (主控台)
- 為另一個 IAM 使用者啟用通行密鑰或安全金鑰 (主控台)
- 取代通行密鑰或安全金鑰
- 使用通行密鑰和安全金鑰的支援組態

必要許可

若要管理您自己的 IAM 使用者的 FIDO 通行密鑰，同時保護敏感的 MFA 相關動作，您必須擁有下列政策的許可：

> **Note**
>
> ARN 值是靜態值，不是哪項協定被用來註冊驗證器的指示器。我們已棄用 U2F，因此所有新的實作都採用 WebAuthn。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
```

```
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

為您自己的 IAM 使用者啟用通行密鑰或安全金鑰 (主控台)

您只能從 為您自己的 IAM 使用者啟用通行金鑰或安全金鑰 AWS 管理主控台 ，不能從 AWS CLI 或 AWS API 啟用。在可以啟用安全金鑰之前，您必須擁有對裝置的實體存取權。

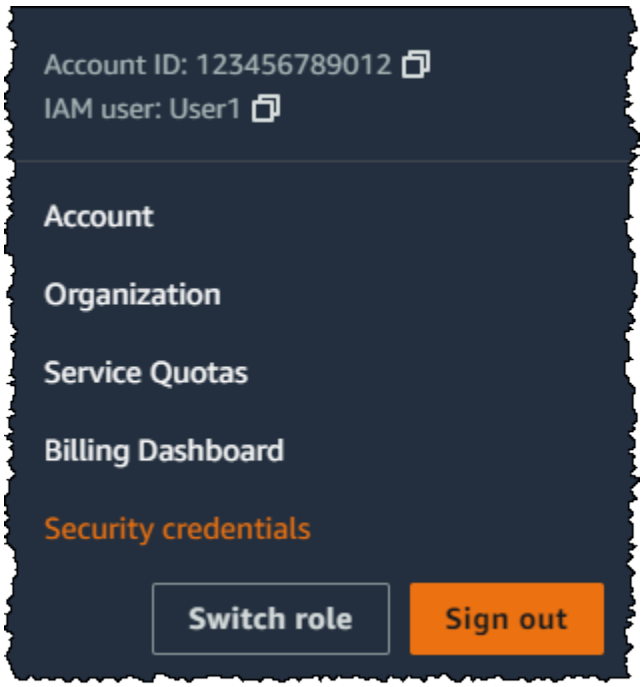
若要為您自己的 IAM 使用者啟用通行密鑰或安全金鑰 (主控台)

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

> **Note**
>
> 為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入 AWS 您的帳戶 ID 或帳戶別名，以重新導向至帳戶的 IAM 使用者登入頁面。

若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。

2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在選取的 IAM 使用者頁面上，選擇安全憑證索引標籤。
4. 在 Multi-Factor Authentication (MFA) (多重要素驗證 (MFA)) 區段下方，選擇 Assign MFA device (指派 MFA 裝置)。
5. 在 MFA 裝置名稱頁面上，輸入裝置名稱，選擇通行密鑰或安全金鑰，然後選擇下一步。
6. 在設定裝置上，設定您的通行密鑰。使用臉部或指紋等生物識別資料、裝置 PIN，或將 FIDO 安全金鑰插入電腦的 USB 連接埠並點選，由此建立通行密鑰。
7. 遵循瀏覽器上的說明進行操作，然後選擇繼續。

您現在已註冊要與 搭配使用的通行金鑰或安全金鑰 AWS。如需搭配 使用 MFA 的詳細資訊 AWS 管理主控台，請參閱 啟用 MFA 的登入。

為另一個 IAM 使用者啟用通行密鑰或安全金鑰 (主控台)

您只能從 為其他 IAM 使用者啟用通行金鑰或安全金鑰 AWS 管理主控台 ，不能從 AWS CLI 或 AWS API 啟用。

為另一個 IAM 使用者啟用通行金鑰或安全金鑰 （主控台）

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 在使用者下，選擇要為其啟用 MFA 的使用者名稱。
4. 在選取的 IAM 使用者頁面上，選擇安全憑證索引標籤。
5. 在 Multi-Factor Authentication (MFA) (多重要素驗證 (MFA)) 區段下方，選擇 Assign MFA device (指派 MFA 裝置)。
6. 在 MFA 裝置名稱頁面上，輸入裝置名稱，選擇通行密鑰或安全金鑰，然後選擇下一步。
7. 在設定裝置上，設定您的通行密鑰。使用臉部或指紋等生物識別資料、裝置 PIN，或將 FIDO 安全金鑰插入電腦的 USB 連接埠並點選，由此建立通行密鑰。
8. 遵循瀏覽器上的說明進行操作，然後選擇繼續。

您現在已為另一個 IAM 使用者註冊通行密鑰或安全金鑰，以便與 AWS搭配使用。如需搭配 使用 MFA 的詳細資訊 AWS 管理主控台，請參閱 啟用 MFA 的登入。

取代通行密鑰或安全金鑰

您的 AWS 帳戶根使用者 和 IAM 使用者一次最多可以有八個 MFA 裝置指派給使用者目前支援 MFA 類型的任何組合。如果使用者遺失 FIDO 驗證器或因為任何原因需要更換，您必須先停用舊的 FIDO 驗證器。然後，再為使用者新增新的 MFA 裝置。

- 如需停用目前與 IAM 使用者相關聯的裝置，請參閱 停用 MFA 裝置。
- 若要為 IAM 使用者新增新的 FIDO 安全性金鑰，請參閱 為您自己的 IAM 使用者啟用通行密鑰或安全金鑰 (主控台)。

如果您無法存取新的通行密鑰或安全金鑰，您可以啟用新的虛擬 MFA 裝置或硬體 TOTP 權杖。請參閱以下其中一項以取得說明：

- 在 中指派虛擬 MFA 裝置 AWS 管理主控台
- 在 中指派硬體 TOTP 字符 AWS 管理主控台

使用通行密鑰和安全金鑰的支援組態

您可以使用 FIDO2 裝置綁定的通行密鑰 (亦稱為安全金鑰)，作為使用目前支援組態的 IAM 的多重要素驗證 (MFA) 方法。其中包括 IAM 支援的 FIDO2 裝置，以及支援 FIDO2 的瀏覽器。

註冊 FIDO2 裝置之前，請檢查您使用的是最新的瀏覽器和作業系統 (OS) 版本。不同瀏覽器、驗證器和作業系統用戶端的功能可能有所不同。如果您的裝置在某個瀏覽器上註冊失敗，您可以嘗試使用其他瀏覽器註冊。

FIDO2 是 FIDO U2F 的開放認證標準，也是 FIDO U2F 的延伸，可根據公有金鑰加密技術提供相同的高度安全性。FIDO2 由 W3C Web Authentication 規範 (WebAuthn API) 和 FIDO Alliance Client-to-Authenticator Protocol (CTAP，一個應用程式層通訊協定) 組成。CTAP 可透過外部驗證器在用戶端或平台之間進行通訊，例如瀏覽器或作業系統。

當您在 中啟用 FIDO 認證驗證器時 AWS，安全金鑰會建立新的金鑰對，僅用於 AWS。首先，您需要輸入您的憑證。出現提示時，您需要點選安全金鑰，回應 AWS發出的身分驗證查問。若要進一步了解 FIDO2 標準，請參閱 FIDO2 專案。

支援的 FIDO2 裝置 AWS

IAM 支援透過 USB、Bluetooth 或 NFC 連線到裝置的 FIDO2 安全裝置。IAM 也支援平台驗證器，例如 TouchID 或 FaceID。IAM 不支援 Windows Hello 的本機通行密鑰註冊。若要建立和使用通行密鑰，Windows 使用者應該使用跨裝置身分驗證，您在其中使用諸如行動裝置等裝置的通行密鑰或硬體安全金鑰，登入諸如筆記型電腦等其他裝置。

> **Note**
>
> AWS 需要存取電腦上的實體 USB 連接埠，才能驗證 FIDO2 裝置。安全金鑰無法與虛擬機器、遠端連線或瀏覽器無痕模式搭配使用。

FIDO Alliance 會維護與 FIDO 規範相容之所有 FIDO2 產品的清單。

支援 FIDO2 的瀏覽器

在 Web 瀏覽器中執行的 FIDO2 安全裝置的可用性取決於瀏覽器和作業系統的組合。以下瀏覽器目前支援使用安全金鑰：

Web 瀏覽器	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	是	是	是	是	否
Safari	是	否	否	是	否
Edge	是	是	否	是	否
Firefox	是	是	否	是	否

> **Note**
>
> 依預設，目前支援 FIDO2 的大多數 Firefox 版本都不會啟用支援。如需在 Firefox 中啟用 FIDO2 支援的說明，請參閱 對通行密鑰和 FIDO 安全金鑰進行疑難排解。
> macOS 上的 Firefox 可能無法完全支援通行金鑰的跨裝置身分驗證工作流程。您可能會收到觸控安全金鑰的提示，而不是繼續進行跨裝置身分驗證。建議您使用不同的瀏覽器 (例如 Chrome 或 Safari)，在 macOS 上使用通行密鑰登入。

如需有關 YubiKey 等 FIDO2 認證的裝置的瀏覽器支援的詳細資訊，請參閱 FIDO2 和 U2F 的作業系統和 Web 瀏覽器支援。

瀏覽器外掛程式

AWS 僅支援原生支援 FIDO2 AWS 的瀏覽器。不支援使用外掛程式來新增 FIDO2 瀏覽器支援。有些瀏覽器外掛程式與 FIDO2 標準不相容，可能會導致未預期的 FIDO2 安全金鑰結果。

如需停用瀏覽器外掛程式和其他故障診斷提示，請參閱 我無法啟用 FIDO 安全性金鑰。

裝置認證

我們只會在註冊安全金鑰期間擷取和指派與裝置相關的認證，例如 FIPS 驗證和 FIDO 認證等級。從 FIDO Alliance Metadata Service (MDS) 中擷取裝置認證。

如果安全金鑰的認證狀態或等級發生變更，它將不會自動反映在裝置標籤中。若要更新裝置的認證資訊，請再次註冊裝置以擷取更新的認證資訊。

AWS 在裝置註冊期間提供下列認證類型作為條件索引鍵，可從 FIDO MDS 中取得：FIPS-140-2、FIPS-140-3 和 FIDO 認證等級。您可以根據認證類型和層級，限制 IAM 政策中的驗證器註冊。如需詳細資訊，請參閱以下政策。

裝置認證的範例政策

下列使用案例顯示可讓您使用 FIPS 認證註冊 MFA 裝置的範例政策。

主題

- 使用案例 1：僅允許註冊具有 FIPS-140-2 L2 認證的裝置
- 使用案例 2：允許註冊具有 FIPS-140-2 L2 和 FIDO L1 認證的裝置
- 使用案例 3：允許註冊具有 FIPS-140-2 L2 或 FIPS-140-3 L2 認證的裝置
- 使用案例 4：允許註冊具有 FIPS-140-2 L2 憑證的裝置，並支援其他 MFA 類型，例如虛擬身分驗證器和硬體 TOTP

使用案例 1：僅允許註冊具有 FIPS-140-2 L2 認證的裝置

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
```

```
                }
            }
        }
    ]
}
```

使用案例 2：允許註冊具有 FIPS-140-2 L2 和 FIDO L1 認證的裝置

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
}
```

使用案例 3：允許註冊具有 FIPS-140-2 L2 或 FIPS-140-3 L2 認證的裝置

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
            }
        }
    ]
}
```

使用案例 4：允許註冊具有 FIPS-140-2 L2 憑證的裝置，並支援其他 MFA 類型，例如虛擬身分驗證器和硬體 TOTP

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey": "Create"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey": "Activate",
          "iam:FIDO-FIPS-140-2-certification": "L2"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "Null": {
          "iam:RegisterSecurityKey": "true"
        }
      }
    }
  ]
}
```

AWS CLI 和 AWS API

AWS 僅支援在 中使用通行金鑰和安全金鑰 AWS 管理主控台。AWS CLI 和 AWS API 不支援使用通行密鑰和安全金鑰進行 MFA，或用以存取受 MFA 保護的 API 操作。

其他資源

- 如需在 中使用通行金鑰和安全金鑰的詳細資訊 AWS，請參閱 在 中指派通行金鑰或安全金鑰 AWS 管理主控台。
- 如需在 中對通行金鑰和安全金鑰進行故障診斷的說明 AWS，請參閱 對通行密鑰和 FIDO 安全金鑰進行疑難排解。
- 如需有關 FIDO2 支援的一般行業資訊，請參閱 FIDO2 專案。

在 中指派虛擬 MFA 裝置 AWS 管理主控台

> **Important**
>
> AWS 建議您 AWS盡可能使用 MFA 的通行金鑰或安全金鑰。如需詳細資訊，請參閱在 中指派通行金鑰或安全金鑰 AWS 管理主控台。

您可以使用手機或其他裝置，做為虛擬多重要素驗證 (MFA) 裝置。若要執行此操作，安裝與 RFC 6238 (以標準為基礎的 TOTP (以時間為基礎的單次密碼) 演算法) 相容的行動應用程式。這些應用程式產生六位數的身分驗證代碼。由於驗證器可以在不安全的行動裝置上執行，而且程式碼可能會與未經授權的對象共用，因此以 TOTP 為基礎的 MFA 不提供與 FIDO2 安全金鑰和通行金鑰等網路釣魚防護選項相同的安全層級。我們建議您使用 MFA 的通行金鑰或安全金鑰，以最強大的防護來防範網路釣魚等攻擊。

如果您還無法使用通行金鑰或安全金鑰，建議您在等待任何硬體購買核准或硬體送達時，使用虛擬 MFA 裝置做為臨時措施。

大多數虛擬 MFA 應用程式支援建立多個虛擬裝置，讓您可以為多個 AWS 帳戶 或 使用者使用相同的應用程式。您最多可以向 AWS 帳戶根使用者 和 IAM 使用者註冊任意組合 MFA 類型的八個 MFA 裝置。您只需一個 MFA 裝置即可登入 AWS 管理主控台 或透過 建立工作階段 AWS CLI。我們建議您註冊多個 MFA 裝置。對於驗證器應用程式，我們也建議您啟用雲端備份或同步功能，以協助避免在裝置遺失或損壞時，失去對帳戶的存取權。

AWS 需要產生六位數 OTP 的虛擬 MFA 應用程式。如需您可以使用的虛擬 MFA 應用程式清單，請參閱多重要素驗證。

主題

- 必要許可
- 針對 IAM 使用者啟用虛擬 MFA 裝置 (主控台)
- 取代虛擬 MFA 裝置

必要許可

若要為 IAM 使用者AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其自己的 MFA 裝置管理虛擬 MFA 裝置時，您必須擁有以下政策的許可：。

針對 IAM 使用者啟用虛擬 MFA 裝置 (主控台)

您可以在 中使用 IAM AWS 管理主控台 ，為帳戶中的 IAM 使用者啟用和管理虛擬 MFA 裝置。您可以將標籤連接至 IAM 資源 (包括虛擬 MFA 裝置)，以識別、整理和控制其存取權。您只能在使用 AWS CLI 或 AWS API 時標記虛擬 MFA 裝置。若要使用 AWS CLI 或 AWS API 啟用和管理 MFA 裝置，請參閱 在 AWS CLI 或 AWS API 中指派 MFA 裝置。如需標記 IAM 資源的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。

> **Note**
>
> 您必須擁有實體存取託管使用者的虛擬 MFA 裝置的硬體，才能設定 MFA。例如，您可以為使用者設定 MFA，該使用者將使用在智慧型手機上執行的虛擬 MFA 裝置。在這種情況下，您必須有可用的智慧型手機，才能完成精靈。因此，您可能想要讓使用者設定和管理自己的虛擬 MFA 裝置。在這種情況下，您必須授予使用者執行必要 IAM 動作的許可。如需詳細資訊以及授予這些許可的 IAM 政策範例，請參閱 IAM 教學課程：允許使用者管理其憑證和 MFA 設定 和範例政策 AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其自己的 MFA 裝置。

為 IAM 使用者啟用虛擬 MFA 裝置 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 在使用者清單中選擇 IAM 使用者名稱。
4. 選擇 安全憑證 標籤。在 Multi-Factor Authentication (MFA) (多重要素驗證 (MFA)) 區段下方，選擇 Assign MFA device (指派 MFA 裝置)。

5. 在精靈中輸入 Device name，然後選擇 驗證器應用程式，再選擇 下一步。

   IAM 將產生並顯示虛擬 MFA 裝置的配置資訊，包括 QR 碼圖形。此圖形代表「私密組態金鑰」，可用來在不支援 QR 碼的裝置上手動輸入。

6. 開啟您的虛擬 MFA 應用程式。如需可以用於託管虛擬 MFA 裝置的應用程式清單，請參閱多重要素驗證。

   如果虛擬 MFA 應用程式支援多個虛擬 MFA 裝置或帳戶，請選擇對應的選項以建立新的虛擬 MFA 裝置或帳戶。

7. 判定 MFA 應用程式是否支援 QR 碼，然後執行以下操作之一：

   - 從精靈中，選擇 Show QR code (顯示 QR 碼)，然後使用應用程式掃描 QR 碼。這可能是攝影機圖示或使用裝置的攝影機掃描代碼的掃描代碼選項。
   - 在精靈中，選擇 Show secret key (顯示私密金鑰)，然後在您的 MFA 應用程式中輸入私密金鑰。

   完成操作後，虛擬 MFA 裝置會開始產生一次性密碼。

8. 在 設定裝置 頁面中的 MFA 代碼 1 方塊內輸入虛擬 MFA 裝置上目前顯示的一次性密碼。請等待 30 秒，裝置將產生新的一次性密碼。然後將第二個一次性密碼輸入 MFA code 2 (MFA 代碼 2) 方塊中。選擇 Add MFA (新增 MFA)。

   > **Important**
   >
   > 產生代碼之後立即提交您的請求。如果在產生代碼後等待很長時間才提交請求，MFA 裝置會成功地與使用者建立關聯，但 MFA 裝置不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。這種情況下，您可以重新同步裝置。

虛擬 MFA 裝置現在可以與 搭配使用 AWS。如需搭配 使用 MFA 的詳細資訊 AWS 管理主控台，請參閱 啟用 MFA 的登入。

> **Note**
>
> 當您透過 AWS 管理主控台 或在登入程序期間新增新的虛擬 MFA 裝置時， AWS 帳戶 將會刪除 中未指派的虛擬 MFA 裝置。未指派的虛擬 MFA 裝置是指位於您帳戶中但帳戶根使用者或 IAM 使用者並未在登入程序中使用的裝置。將這些裝置刪除後，便可將新的虛擬 MFA 裝置新增至您的帳戶。刪除這些裝置後，您還可以重複使用裝置名稱。

- 若要檢視帳戶中未指派的虛擬 MFA 裝置，您可以使用 list-virtual-mfa-devices AWS CLI 命令或 API 呼叫。
- 若要停用虛擬 MFA 裝置，您可以使用 deactivate-mfa-device AWS CLI 命令或 API 呼叫。停用後，裝置會變成未指派狀態。
- 若要將未指派的虛擬 MFA 裝置連接至 AWS 帳戶 根使用者或 IAM 使用者，您需要裝置產生的驗證碼，以及 enable-mfa-device AWS CLI 命令或 API 呼叫。

取代虛擬 MFA 裝置

您的 AWS 帳戶根使用者 和 IAM 使用者可以註冊最多八個 MFA 裝置的任意 MFA 類型組合。如果使用者遺失裝置或基於任何原因需要更換，請停用舊裝置。然後，再為使用者加入新的裝置。

- 如需停用目前與另一個 IAM 使用者相關聯的裝置，請參閱 停用 MFA 裝置。
- 若要新增另一個 IAM 使用者的取代用虛擬 MFA 裝置，請遵循上方 針對 IAM 使用者啟用虛擬 MFA 裝置 (主控台) 程序中的步驟。
- 若要為 新增替代虛擬 MFA 裝置 AWS 帳戶根使用者，請遵循程序 中的步驟針對根使用者啟用虛擬 MFA 裝置 (主控台)。

在 中指派硬體 TOTP 字符 AWS 管理主控台

Important

AWS 建議您 AWS盡可能使用 MFA 的通行金鑰或安全金鑰。如需詳細資訊，請參閱在 中指派通行金鑰或安全金鑰 AWS 管理主控台。

硬體 TOTP 權杖會在以時間為基礎的一次性密碼 (TOTP) 演算法的基礎上產生六位數字程式碼。使用者必須在登入程序期間出現提示時輸入裝置中的有效代碼。指派給使用者的每個 MFA 裝置必須是唯一的；使用者不能從另一個使用者的裝置中輸入代碼進行身分驗證。MFA 裝置無法跨帳戶或使用者共用。

硬體 TOTP 權杖與 FIDO 安全金鑰都是您購買的實體裝置。硬體 MFA 裝置會在您登入時產生 TOTP 代碼以進行身分驗證 AWS。它們依賴電池，這可能需要 AWS 隨著時間的推移替換和重新同步。FIDO 安全金鑰使用公有金鑰加密，不需要電池並提供無縫的身分驗證程序。我們建議使用 FIDO 安全金鑰來防禦網路釣魚，這為 TOTP 裝置提供了更安全的替代方案。此外，FIDO 安全金鑰可以支援同一裝置

上的多個 IAM 或根使用者，從而增強其公用程式以確保帳戶安全。如需兩種裝置類型的規格及購買資訊，請參閱多重要素驗證。

您可以從 、 AWS 管理主控台命令列或 IAM API 為 IAM 使用者啟用硬體 TOTP 字符。若要為您的 啟用 MFA 裝置 AWS 帳戶根使用者，請參閱 針對根使用者啟用硬體 TOTP 權杖 (主控台)。

您可以向 AWS 帳戶根使用者 和 IAM 使用者註冊最多八個目前支援 MFA 類型之任何組合的 MFA 裝置。使用多個 MFA 裝置時，您只需一個 MFA 裝置即可登入 ， AWS 管理主控台 或透過 以該使用者 AWS CLI 身分建立工作階段。

> **Important**
>
> 建議您為您的使用者啟用多台 MFA 裝置，以便在一台 MFA 裝置遺失或無法存取時繼續存取您的帳戶。

> **Note**
>
> 如果您想要從命令列啟用 MFA 裝置，請使用 `aws iam enable-mfa-device`。若要使用 IAM API 啟用 MFA 裝置，請使用 `EnableMFADevice` 操作。

主題

- 必要許可
- 啟用您 IAM 使用者的硬體 TOTP 權杖 (主控台)
- 啟用另一個 IAM 使用者的硬體 TOTP 權杖 (主控台)
- 取代實體 MFA 裝置

必要許可

若要管理您 IAM 使用者的硬體 TOTP 權杖，同時保護敏感的 MFA 相關動作，您必須擁有下列政策的許可：

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

啟用您 IAM 使用者的硬體 TOTP 權杖 (主控台)

您可以從 AWS 管理主控台啟用您自己的硬體 TOTP 權杖。

> **Note**
>
> 在您可以啟用硬體 TOTP 權杖之前，您必須擁有裝置的實體存取權。

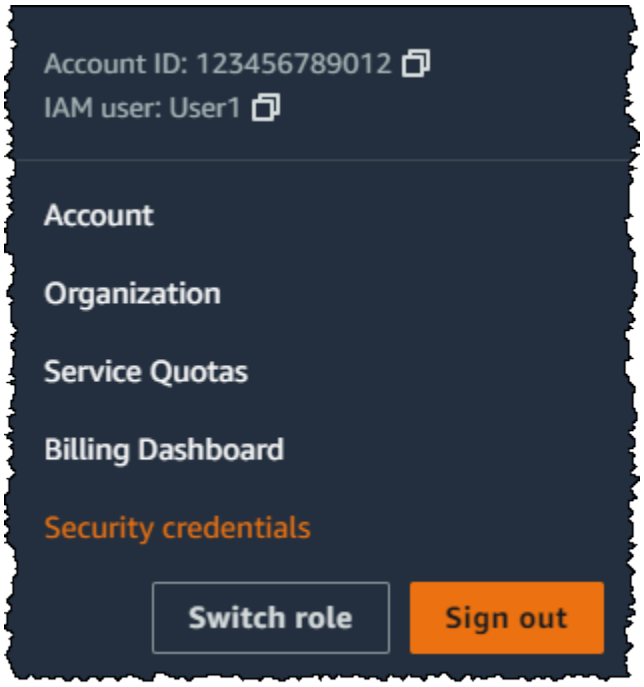
要啟用您 IAM 使用者的硬體 TOTP 權杖 (主控台)

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

> **Note**
>
> 為了方便起見， AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入 AWS 您的帳戶 ID 或帳戶別名，以重新導向至帳戶的 IAM 使用者登入頁面。

   若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。

2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在 AWS IAM credentials (IAM 憑證) 索引標籤上，在 Multi-factor authentication (MFA) (多重要素驗證 (MFA)) 區段，選擇 Assign MFA device (指派 MFA 裝置)。
4. 在精靈中，輸入一個裝置名稱，然後選取 Hardware TOTP token (硬體 TOTP 權杖)，再選擇 Next (下一步)。
5. 輸入裝置序號。序號通常位於裝置的背面。
6. 在 MFA code 1 (MFA 代碼 1) 方塊中，輸入 MFA 裝置顯示的六位數字。您可能需要按下裝置正面的按鈕以顯示數字。



7. 當裝置在重新整理代碼時，等候 30 秒時間後，在 MFA code 2 (MFA 代碼 2) 方塊中輸入六位數字。您可能需要再次按下裝置正面的按鈕以顯示第二個數字。
8. 選擇 Add MFA (新增 MFA)。

> **Important**
>
> 產生驗證代碼之後立即提交您的請求。如果在產生代碼後等待很長時間才提交請求，MFA 裝置會成功地與使用者建立關聯，但 MFA 裝置卻變成不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。這種情況下，您可以重新同步裝置。

裝置已準備好與 搭配使用 AWS。如需與 AWS 管理主控台一起使用 MFA 的詳細資訊，請參閱 啟用 MFA 的登入。

啟用另一個 IAM 使用者的硬體 TOTP 權杖 (主控台)

您可以從 AWS 管理主控台啟用另一個 IAM 使用者的硬體 TOTP 權杖。

要啟用另一個 IAM 使用者的硬體 TOTP 權杖 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 選擇要為其啟用 MFA 的使用者名稱。
4. 選擇 安全憑證 標籤。在 Multi-Factor Authentication (MFA) (多重要素驗證 (MFA)) 區段下方，選擇 Assign MFA device (指派 MFA 裝置)。
5. 在精靈中，輸入一個裝置名稱，然後選取 Hardware TOTP token (硬體 TOTP 權杖)，再選擇 Next (下一步)。
6. 輸入裝置序號。序號通常位於裝置的背面。
7. 在 MFA code 1 (MFA 代碼 1) 方塊中，輸入 MFA 裝置顯示的六位數字。您可能需要按下裝置正面的按鈕以顯示數字。



8. 當裝置在重新整理代碼時，等候 30 秒時間後，在 MFA code 2 (MFA 代碼 2) 方塊中輸入六位數字。您可能需要再次按下裝置正面的按鈕以顯示第二個數字。
9. 選擇 Add MFA (新增 MFA)。

> **Important**
>
> 產生驗證代碼之後立即提交您的請求。如果在產生代碼後等待很長時間才提交請求，MFA 裝置會成功地與使用者建立關聯，但 MFA 裝置卻變成不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。這種情況下，您可以重新同步裝置。

裝置已準備好與 搭配使用 AWS。如需與 AWS 管理主控台一起使用 MFA 的詳細資訊，請參閱 啟用 MFA 的登入。

取代實體 MFA 裝置

您的 AWS 帳戶根使用者 和 IAM 使用者一次最多可以有八個 MFA 裝置指派給使用者目前支援的 MFA 類型任意組合。如果使用者遺失裝置或基於任何原因需要汰換，您必須先停用舊裝置。然後，再為使用者加入新的裝置。

- 如需停用目前與使用者相關聯的裝置，請參閱 停用 MFA 裝置。
- 若要新增 IAM 使用者適用的替代硬體 TOTP 權杖，請遵循本主題稍早介紹的 啟用另一個 IAM 使用者的硬體 TOTP 權杖 (主控台) 程序。
- 若要為 新增替換硬體 TOTP 權杖 AWS 帳戶根使用者，請遵循本主題針對根使用者啟用硬體 TOTP 權杖 (主控台)先前程序中的步驟。

在 AWS CLI 或 AWS API 中指派 MFA 裝置

您可以使用 AWS CLI 命令或 AWS API 操作，為 IAM 使用者啟用虛擬 MFA 裝置。您無法使用 AWS 帳戶根使用者 AWS CLI、 AWS API、Tools for Windows PowerShell 或任何其他命令列工具為 啟用 MFA 裝置。不過，您可以使用 AWS 管理主控台 為根使用者啟用 MFA 裝置。

當您從 啟用 MFA 裝置時 AWS 管理主控台，主控台會為您執行多個步驟。如果您改為使用 AWS CLI、Tools for Windows PowerShell 或 AWS API 建立虛擬裝置，則必須以正確的順序手動執行步驟。例如，如果要建立虛擬 MFA 裝置，則必須建立 IAM 物件，將程式碼擷取為字串或 QR 碼圖形，然後同步該裝置並將其與 IAM 使用者建立關聯。請參閱 New-IAMVirtualMFADevice 中的範例章節以了解更多詳細資訊。對於實體裝置，您可以跳過建立步驟，直接同步該裝置並將其與使用者建立關聯。

您可以將標籤連接至 IAM 資源 (包括虛擬 MFA 裝置)，以識別、整理和控制其存取權。您只能在使用 AWS CLI 或 AWS API 時標記虛擬 MFA 裝置。

使用 SDK 或 CLI 的 IAM 使用者可以透過呼叫 `EnableMFADevice` 啟用額外的 MFA 裝置，或者透過呼叫 `DeactivateMFADevice` 停用現有的 MFA 裝置。若要成功執行此動作，這些使用者必須先使用現有的 MFA 裝置呼叫 `GetSessionToken` 並提交 MFA 代碼。此呼叫會傳回暫時的安全憑證，然後可使用此憑證簽署需要 MFA 身分驗證的 API 作業。如需要求和回應的範例，請參閱 GetSessionToken - 不受信任環境中使用者的暫時憑證。

在 IAM 中建立虛擬裝置實體來代表虛擬 MFA 裝置

這些命令提供在以下許多命令中代替序號的裝置 ARN。

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API： `CreateVirtualMFADevice`

啟用 MFA 裝置以搭配 使用 AWS

這些命令會將裝置與 同步， AWS 並將其與使用者建立關聯。如果裝置是虛擬裝置，則將虛擬裝置的 ARN 做為序號使用。

> **Important**
>
> 產生驗證代碼之後立即提交您的請求。如果在產生代碼後等待很長時間才提交請求，MFA 裝置會成功地與使用者建立關聯，但 MFA 裝置卻變成不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。如果發生這種情況，可以使用下面介紹的命令重新同步裝置。

- AWS CLI: `aws iam enable-mfa-device`
- AWS API： `EnableMFADevice`

停用裝置

這些命令將取消裝置與使用者間的關聯並停用裝置。如果裝置是虛擬裝置，則將虛擬裝置的 ARN 做為序號使用。您也必須單獨刪除虛擬裝置實體。

- AWS CLI: aws iam deactivate-mfa-device
- AWS API： DeactivateMFADevice

列出虛擬 MFA 裝置實體

使用這些命令來列出虛擬 MFA 裝置的實體。

- AWS CLI: aws iam list-virtual-mfa-devices
- AWS API： ListVirtualMFADevices

標記虛擬 MFA 裝置

使用這些指令來標記虛擬 MFA 裝置。

- AWS CLI: aws iam tag-mfa-device
- AWS API： TagMFADevice

列出虛擬 MFA 裝置的標籤

使用這些命令列出連接至虛擬 MFA 裝置的標籤。

- AWS CLI: aws iam list-mfa-device-tags
- AWS API： ListMFADeviceTags

取消標記虛擬 MFA 裝置

使用這些命令移除連接至虛擬 MFA 裝置的標籤。

- AWS CLI: aws iam untag-mfa-device
- AWS API： UntagMFADevice

重新同步 MFA 裝置

如果裝置正在產生 不接受的代碼，請使用這些命令 AWS。如果裝置是虛擬裝置，則將虛擬裝置的 ARN 做為序號使用。

- AWS CLI: `aws iam resync-mfa-device`
- AWS API： `ResyncMFADevice`

刪除 IAM 中的虛擬 MFA 裝置實體

在裝置與使用者取消關聯後，您可以刪除裝置實體。

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API： `DeleteVirtualMFADevice`

復原遺失或無法運作的虛擬 MFA 裝置

有時，主控虛擬 MFA 應用程式的使用者的裝置會發生遺失、遭到替換，或無法運作。當這種情況發生時，使用者無法自行復原。使用者必須聯絡管理員以停用裝置。如需詳細資訊，請參閱在 IAM 中復原受 MFA 保護的身分。

檢查 MFA 狀態

使用 IAM 主控台來檢查 AWS 帳戶根使用者 或 IAM 使用者是否已啟用有效的 MFA 裝置。

若要查看根使用者的 MFA 狀態

1. AWS 管理主控台 使用您的根使用者憑證登入 ，然後在 開啟 IAM 主控台https://console.aws.amazon.com/iam/。
2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。
3. 檢查 多重要素驗證 (MFA) 下方，查看 MFA 是否已啟用或停用。如果 MFA 尚未啟動，則會顯示提醒符號
   ()

如果您想要啟用 MFA 帳戶，請參閱下列其中一項：

- 針對根使用者啟用虛擬 MFA 裝置 (主控台)
- 為根使用者啟用通行密鑰或安全金鑰 (主控台)
- 針對根使用者啟用硬體 TOTP 權杖 (主控台)

查看 IAM 使用者的 MFA 狀態

1. 在 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)。
3. 如有必要，請透過完成以下步驟將 MFA 欄新增到使用者表格：
   a. 在最右側的表格上方，選擇設定圖示 ()。
   b. 在 Manage Columns (管理欄) 中，選取 MFA。
   c. (選用) 清除您不想在使用者資料表中顯示的任何資料欄標題的核取方塊。
   d. 選擇 Close (關閉) 返回使用者清單。
4. MFA 欄告訴您有關已啟用的 MFA 裝置。如果沒有給使用者作用中的 MFA 裝置，則主控台會顯示 None (無)。如果使用者已啟用 MFA 裝置，則 MFA 欄顯示啟用裝置的類型值為 Virtual (虛擬)、Security Key (安全金鑰)、Hardware (硬體) 或 SMS。

   > **Note**
   >
   > AWS 結束對啟用 SMS 多重要素驗證 (MFA) 的支援。建議擁有使用 SMS 簡訊式 MFA 的 IAM 使用者的客戶改為使用下列任一種替代方式：虛擬 (軟體式) MFA 裝置、FIDO 安全性金鑰或硬體 MFA 裝置。您可以使用已分配的 SMS MFA 裝置來識別帳戶中的使用者。為此，請前往 IAM 主控台，從導覽窗格中選擇 Users (使用者)，然後在表的 MFA 列中尋找具有 SMS 的使用者。

5. 若要查看有關使用者的 MFA 裝置的其他資訊，請選擇要檢查其 MFA 狀態的使用者名稱。然後選擇 Security credentials (安全憑證) 索引標籤。
6. 如果沒有給使用者作用中的 MFA 裝置，則主控台會顯示 無 MFA 裝置。在多重要素驗證 (MFA) 區段中指派 MFA 裝置來改善 AWS 環境的安全性。如果使用者有已啟用的 MFA 裝置，則 Multi-factor authentication (MFA) (多重要素驗證 (MFA)) 區段會顯示有關這些裝置的詳細資訊：
   - 裝置名稱
   - 裝置類型
   - 裝置識別符，例如實體裝置的序號或 AWS 虛擬裝置的 中的 ARN
   - 何時建立裝置

若要移除或重新同步裝置，請選擇裝置旁的選項按鈕，然後選取 Remove (移除) 或 Resync (重新同步)。

如需有關啟用 MFA 的詳細資訊，請參閱以下內容：

- 在 中指派虛擬 MFA 裝置 AWS 管理主控台
- 在 中指派通行金鑰或安全金鑰 AWS 管理主控台
- 在 中指派硬體 TOTP 字符 AWS 管理主控台

重新同步虛擬和硬體 MFA 裝置

您可以使用 AWS 來重新同步虛擬和硬體多重要素驗證 (MFA) 裝置。如果您的裝置在您嘗試使用時未同步，登入嘗試失敗，且 IAM 提示您重新同步裝置。

> **Note**
>
> FIDO 安全性金鑰不會失去同步。若 FIDO 安全性金鑰遺失或損壞，您可以停用它。如需停用任何 MFA 裝置類型的說明，請參閱 為另一個 IAM 使用者停用 MFA 裝置 (主控台)。

身為 AWS 管理員，如果 IAM 使用者的虛擬和硬體 MFA 裝置不同步，您可以重新同步它們。

如果您的 AWS 帳戶根使用者 MFA 裝置無法運作，您可以使用 IAM 主控台重新同步裝置，無論是否完成登入程序。如果您無法成功重新同步處理裝置，您可能需要為裝置取消關聯，再重新關聯此裝置。如需如何執行此作業的資訊，請參閱 停用 MFA 裝置 和 AWS IAM 中的多重要素驗證。

主題

- 必要許可
- 重新同步虛擬及硬體 MFA 裝置 (IAM 主控台)
- 重新同步虛擬及硬體 MFA 裝置 (AWS CLI)
- 重新同步虛擬和硬體 MFA 裝置 (AWS API)

必要許可

若要為自己的 IAM 使用者重新同步虛擬或硬體 MFA 裝置，您必須擁有以下政策的許可。此政策不允許您建立或停用裝置。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

重新同步虛擬及硬體 MFA 裝置 (IAM 主控台)

您可以使用 IAM 主控台來重新同步虛擬及硬體 MFA 裝置。

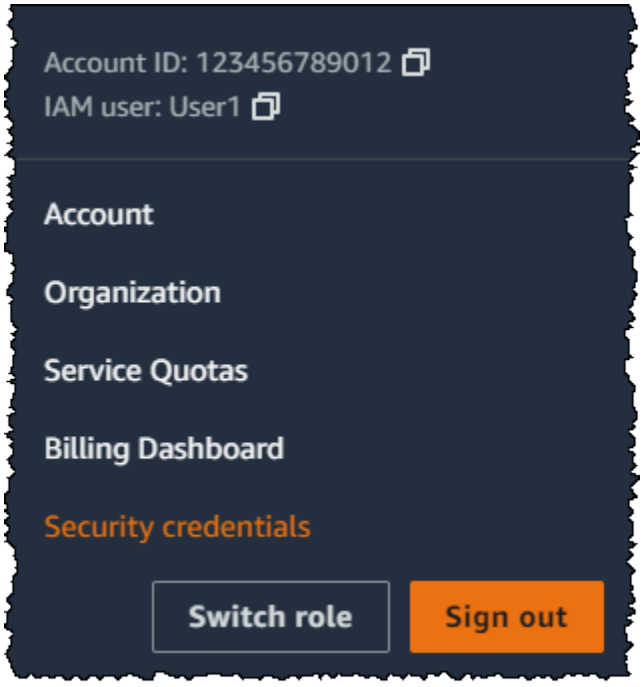
重新同步您 IAM 使用者的虛擬或硬體 MFA 裝置 (主控台)

1. 使用 AWS 您的帳戶 ID 或帳戶別名、IAM 使用者名稱和密碼登入 IAM 主控台。

   > **Note**
   >
   > 為了方便起見，AWS 登入頁面會使用瀏覽器 Cookie 來記住您的 IAM 使用者名稱和帳戶資訊。如果您先前以不同的使用者身分登入，請選擇在頁面底部附近的 Sign in to a different account (登入不同的帳戶)，返回主要登入頁面。在那裡，您可以輸入要重新導向至 AWS 您帳戶的 IAM 使用者登入頁面的帳戶 ID 或帳戶別名。

   若要取得您的 AWS 帳戶 ID，請聯絡您的管理員。

2. 在右上方的導覽列中，選擇您的使用者名稱，然後選擇 安全憑證 。



3. 在 AWS IAM 憑證 標籤的 多重要素驗證 (MFA) 區段選擇 MFA 裝置旁的選項按鈕，然後選擇 重新同步。
4. 將裝置接下來連續產生的兩個代碼輸入 MFA code 1 (MFA 代碼 1) 和 MFA code 2 (MFA 代碼 2)。然後選擇 Resync (重新同步)。

> **Important**
>
> 產生代碼之後立即提交您的請求。如果您產生代碼，然後等太久而無法提交請求、請求會出現運作，但裝置保持未同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。

重新同步另一個 IAM 使用者的虛擬或硬體 MFA 裝置 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)，然後選擇需要重新同步 MFA 裝置的使用者名稱。
3. 選擇 Security credentials (安全憑證) 索引標籤。在 多重要素驗證 (MFA) 區段中選擇 MFA 裝置旁的選項按鈕，然後選擇 重新同步。
4. 將裝置接下來連續產生的兩個代碼輸入 MFA code 1 (MFA 代碼 1) 和 MFA code 2 (MFA 代碼 2)。然後選擇 Resync (重新同步)。

> **Important**
>
> 產生代碼之後立即提交您的請求。如果您產生代碼，然後等太久而無法提交請求、請求會出現運作，但裝置保持未同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。

登入前重新同步根使用者 MFA (主控台)

1. 在 Amazon Web Services Sign In With Authentication Device (使用身分驗證裝置登入 Amazon Web Services) 頁面上，選擇 Having problems with your authentication device? (您的身分驗證裝置登有問題？)。Click here (請點選此處)。

> **Note**
>
> 您可能會看到不同的文字，例如使用 MFA 登入和對您的身分驗證裝置進行疑難排解。不過，其功能是相同的。

2. 在 Re-Sync With Our Servers (與我們的伺服器重新同步) 區段中，將裝置接下來連續產生的兩個代碼輸入 MFA code 1 (MFA 代碼 1) 和 MFA code 2 (MFA 代碼 2)。然後選擇 Re-sync authentication device (重新同步身分驗證裝置)。

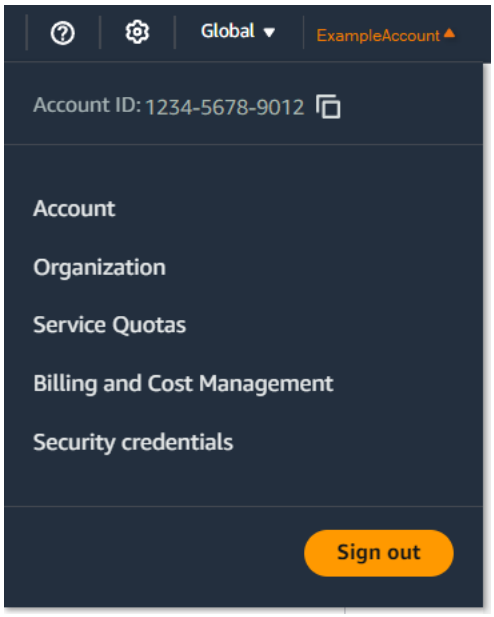
3. 如果必要，再次輸入密碼，然後選擇登入。然後，使用您的 MFA 裝置完成登入。

登入後重新同步根使用者 MFA 裝置 (主控台)

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者身分登入 IAM 主控台。在下一頁中，輸入您的密碼。

> **Note**
>
> 根使用者無法登入 以 IAM 使用者身分登入 頁面。如果您看到 以 IAM 使用者身分登入 頁面，請選擇頁面底部附近的 使用根使用者電子郵件登入。如需以根使用者身分登入的說明，請參閱AWS 登入 《 使用者指南》中的以根使用者 AWS 管理主控台 身分登入 。

2. 在導覽列右側選擇您的帳戶名稱，然後選擇 安全憑證 。如有需要，選擇 Continue to Security Credentials (繼續至安全憑證)。



3. 展開頁面上的 Multi-factor authentication (MFA) (多重要素驗證 (MFA)) 區段。
4. 選擇裝置旁的選項按鈕，然後選取 Resync (重新同步)。
5. 在 Resync MFA device (重新同步 MFA 裝置) 對話方塊中，將裝置接下來連續產生的兩個代碼輸入 MFA code 1 (MFA 代碼 1) 和 MFA code 2 (MFA 代碼 2)。然後選擇 Resync (重新同步)。

> **Important**
>
> 產生代碼之後立即提交您的請求。如果在產生代碼後等待很長時間才提交請求，MFA 裝置會成功地與使用者建立關聯，但 MFA 裝置將不同步。會發生這種情況是因為定時式的一次性密碼 (TOTP) 在過了一小段時間後就會過期。

重新同步虛擬及硬體 MFA 裝置 (AWS CLI)

您可以從 AWS CLI重新同步虛擬及硬體 MFA 裝置。

重新同步 IAM 使用者的虛擬或硬體 MFA 裝置 (AWS CLI)

在命令提示字元中，發出 aws iam resync-mfa-device 命令：

- 虛擬 MFA 裝置：將裝置的 Amazon Resource Name (ARN) 指定為序號。

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- 硬體 MFA 裝置：將硬體裝置的序號指定為序號。格式為廠商特定。例如，您可以從 Amazon 購買 gemalto 權杖。其序號通常是四個字母，後面接著四個數字。

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> 產生代碼之後立即提交您的請求。如果您產生代碼，然後因等太久而無法提交請求、請求會因代碼在不久後過期而失敗。

重新同步虛擬和硬體 MFA 裝置 (AWS API)

IAM 有一個執行同步化的 API 呼叫。在這種情況下，建議您提供您的虛擬及硬體 MFA 裝置使用者許可，讓其存取此 API 呼叫。然後根據該 API 呼叫建置工具，讓使用者在需要時能隨時重新同步他們的裝置。

重新同步 IAM 使用者 (AWS API) 的虛擬或硬體 MFA 裝置

- 傳送 ResyncMFADevice 請求。

停用 MFA 裝置

如果您無法將多重要素驗證 (MFA) 裝置作為 IAM 使用者登入，請聯絡您的管理員尋求協助。

身為管理員，您可以停用另一個 IAM 使用者的裝置。這可讓使用者在不使用 MFA 的情況下登入。在更換 MFA 裝置或裝置暫時不可用時，您可以將此做為臨時解決方案。但是，我們建議您盡快為使用者啟用新的裝置。若要了解如何啟用新的 MFA 裝置，請參閱 AWS IAM 中的多重要素驗證。

> **Note**
>
> 如果您使用 API AWS CLI 或 從中刪除使用者 AWS 帳戶，則必須停用或刪除使用者的 MFA 裝置。您將此變更做為移除使用者的過程的一部分。如需有關移除使用者的詳細資訊，請參閱 移除或停用 IAM 使用者。

主題

- 停用 MFA 裝置 (主控台)
- 停用 MFA 裝置 (AWS CLI)
- 停用 MFA 裝置 (AWS API)

停用 MFA 裝置 (主控台)

為另一個 IAM 使用者停用 MFA 裝置 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)。
3. 若要為使用者停用 MFA 裝置，請選擇要移除其 MFA 的使用者的名稱。
4. 選擇 Security credentials (安全憑證) 索引標籤。
5. 在 多重要素驗證 (MFA) 下方選擇 MFA 裝置旁的選項按鈕、選擇 移除，然後選擇 移除。

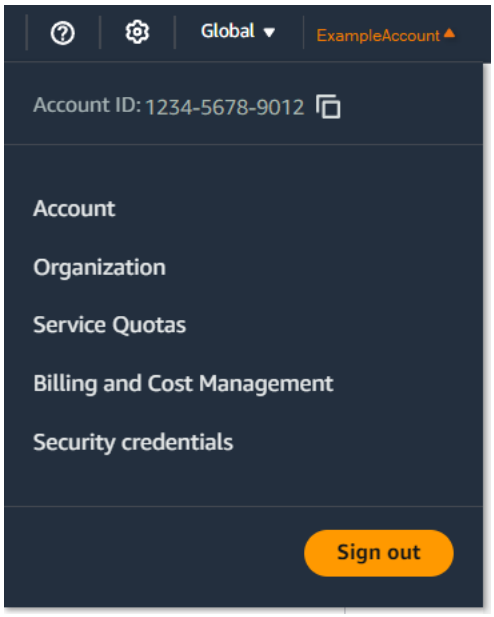
   裝置已從中移除 AWS。在重新啟用請求並與 AWS 使用者或 建立關聯之前，它無法用來登入或驗證請求 AWS 帳戶根使用者。

停用 AWS 帳戶根使用者 （ 主控台 ） 的 MFA 裝置

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者身分登入 IAM 主控台。在下一頁中，輸入您的密碼。

   > **Note**
   >
   > 根使用者無法登入 以 IAM 使用者身分登入 頁面。如果您看到 以 IAM 使用者身分登入 頁面，請選擇頁面底部附近的 使用根使用者電子郵件登入。如需以根使用者身分登入的說明，請參閱AWS 登入 《 使用者指南》中的以根使用者 AWS 管理主控台 身分登入 。

2. 在導覽列右側選擇您的帳戶名稱，然後選擇 安全憑證 。如有需要，選擇 Continue to Security Credentials (繼續至安全憑證)。



3. 在 Multi-factor authentication (MFA) (多重要素驗證 (MFA)) 區段中，選擇您想要停用的 MFA 裝置旁的選項按鈕，然後選擇 Remove (移除)。
4. 選擇移除。

   已為 AWS 帳戶停用 MFA 裝置。檢查與您的 相關聯的電子郵件， AWS 帳戶 以取得來自 Amazon Web Services 的確認訊息。該電子郵件會通知您，Amazon Web Services 多重要素驗證 (MFA) 已停用。訊息來自於 `@amazon.com` 或 `@aws.amazon.com`。

> **Note**
>
> 當您透過 AWS 管理主控台 或在登入程序期間新增新的虛擬 MFA 裝置時，AWS 帳戶 將刪除中未指派的虛擬 MFA 裝置。未指派的虛擬 MFA 裝置是指位於您帳戶中但帳戶根使用者或 IAM 使用者並未在登入程序中使用的裝置。將這些裝置刪除後，便可將新的虛擬 MFA 裝置新增至您的帳戶。刪除這些裝置後，您還可以重複使用裝置名稱。

停用 MFA 裝置 (AWS CLI)

為 IAM 使用者停用 MFA 裝置 (AWS CLI)

- 執行此命令：`aws iam deactivate-mfa-device`

停用 MFA 裝置 (AWS API)

停用 IAM 使用者的 MFA 裝置 (AWS API)

- 呼叫此操作：`DeactivateMFADevice`

在 IAM 中復原受 MFA 保護的身分

如果您的虛擬 MFA 裝置或硬體 TOTP 字符似乎正常運作，但您無法使用它來存取您的 AWS 資源，則它可能會與 不同步 AWS。如需同步虛擬 MFA 裝置或硬體 MFA 裝置的資訊，請參閱 重新同步虛擬和硬體 MFA 裝置。FIDO 安全性金鑰不會失去同步。

如果 AWS 帳戶根使用者 的 MFA 裝置遺失、損壞或無法運作，您可以復原對帳戶的存取。IAM 使用者必須聯絡管理員以停用裝置。

> **Important**
>
> 我們建議您啟用多個 MFA 裝置。註冊多個 MFA 裝置有助於確保在裝置遺失或損壞時持續存取。您的 AWS 帳戶根使用者 和 IAM 使用者可以註冊最多八個任何類型的 MFA 裝置。

先決條件 – 使用另一個 MFA 裝置

如果您的多重要素驗證 (MFA) 裝置遺失、損毀或無法運作，您可以使用另一個註冊到相同根使用者或 IAM 使用者的 MFA 裝置登入。

若要使用另一個 MFA 裝置登入

1. 使用您的 AWS 帳戶 ID 或帳戶別名和密碼登入 AWS 管理主控台。
2. 在需要其他驗證頁面或多重要素驗證頁面上，選擇嘗試其他 MFA 方法。
3. 使用您選取的 MFA 裝置類型進行驗證。
4. 下一個步驟根據您是否成功使用替代 MFA 裝置登入而有所不同。
   - 如果您已成功登入，您可以重新同步虛擬和硬體 MFA 裝置，這可能會解決此問題。如果您的 MFA 裝置遺失或損壞，您可以予以停用。如需停用任何 MFA 裝置類型的說明，請參閱 停用 MFA 裝置。
   - 如果您無法使用 MFA 登入，請使用復原根使用者 MFA 裝置或復原 IAM 使用者 MFA 裝置中的步驟來復原受 MFA 保護的身分。

復原根使用者 MFA 裝置

如果無法透過 MFA 登入，您可以使用其他身分驗證方法進行登入，即採用您帳戶中註冊的電子郵件和主要聯絡人電話號碼進行身分驗證。

確認在使用其他驗證要素以根使用者身分登入之前，您可以存取與您的帳戶關聯的電子郵件和主要聯絡人電話號碼。如果需要更新主要聯絡人電話號碼，請使用管理員存取權 (而非根使用者) 以 IAM 使用者的身分登入。如需了解有關更新帳戶聯絡人資訊的更多指示，請參閱 AWS Billing 使用者指南中的編輯聯絡人資訊。如果無權存取電子郵件和主要聯絡人電話號碼，則您必須聯絡 AWS 支援。

> **Important**
>
> 建議您保持電子郵件地址及聯絡人電話號碼與您的根使用者的關聯處於最新狀態，以成功地復原帳戶。如需詳細資訊，請參閱《AWS 帳戶管理 參考指南》中的更新您 AWS 帳戶的主要連絡人。

使用身分驗證的替代因素做為 登入 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者AWS 管理主控台身分登入 。在下一頁中，輸入您的密碼。
2. 在需要其他驗證頁面上，選取要用來進行驗證的 MFA 方法，然後選擇下一步。

> **Note**
>
> 您可能會看到替代文字，例如使用 MFA 登入、對您的驗證裝置進行故障診斷，或對 MFA 進行故障診斷，但功能是相同的。您不能使用其他驗證要素來驗證您的帳戶電子郵件地址和主要聯絡人電話號碼，請聯絡 AWS 支援 停用您的 MFA 裝置。

3. 視您使用的 MFA 類型而定，您會看到不同的頁面，但 MFA 疑難排解選項的功能相同。在需要其他驗證頁面或多重要素驗證頁面上，選擇 MFA 疑難排解。
4. 如果必要，再次輸入密碼，然後選擇 Sign in (登入)。
5. 在使用其他驗證要素登入區段中的驗證裝置疑難排解頁面上，選擇使用其他要素登入。
6. 在使用其他驗證要素登入頁面上，透過驗證電子郵件地址來驗證您的帳戶，然後選擇傳送驗證電子郵件。
7. 檢查與您的 相關聯的電子郵件 AWS 帳戶 是否有來自 Amazon Web Services (recover-mfa-no-reply@verify.signin.aws) 的訊息。請遵循電子郵件中的指示進行。

   如果您沒看到帳戶中有電子郵件，請檢查您的垃圾郵件資料夾，或返回瀏覽器，然後選擇 Resend the email (重新傳送電子郵件)。
8. 驗證電子郵件地址後，您可以繼續驗證您的帳戶。若要驗證主要聯絡人電話號碼，請選擇 Call me now (立即呼叫我)。
9. 接聽來自 的呼叫， AWS 並在出現提示時，在電話鍵盤上輸入網站上的 AWS 6 位數號碼。

   如果您沒有收到來自 的通話 AWS，請選擇登入以再次登入主控台並重新開始。或者，請參閱遺失或無法使用的多重要素驗證 (MFA) 裝置來聯絡支援中心以尋求協助。
10. 驗證您的電話號碼之後，您可以選擇 Sign in to the console (登入主控台) 登入您的帳戶。
11. 下一步取決於您使用的 MFA 類型：

    - 針對虛擬 MFA 裝置，請從您的裝置移除帳戶。然後，請前往 AWS Security Credentials (AWS 安全憑證) 頁面，並刪除舊 MFA 虛擬裝置實體，再建立新的實體。
    - 如需 FIDO 安全金鑰，請參閱AWS 安全登入資料頁面，並在啟用新金鑰之前停用舊的 FIDO 安全金鑰。
    - 針對硬體 TOTP 權杖，請聯絡第三方供應商，請其協助修復或更換裝置。您可以繼續使用其他身分驗證方法登入，直到收到新的裝置為止。擁有新的硬體 MFA 裝置後，請參閱AWS 安全登入資料頁面並刪除舊的 MFA 裝置。

> **Note**
>
> 您無須將遺失或遭竊的 MFA 裝置更換成相同類型的裝置。例如，若您損壞了 FIDO 安全金鑰並訂購了新的金鑰，您可以使用虛擬 MFA 或硬體 TOTP 權杖，直到新的 FIDO 金鑰到達。

> **Important**
>
> 如果您的 MFA 裝置遺失或遭竊，請在登入並建立取代用 MFA 裝置後變更您的根使用者密碼。攻擊者可能竊取了驗證裝置，也可能擁有您目前的密碼。如需詳細資訊，請參閱變更 的密碼 AWS 帳戶根使用者。

復原 IAM 使用者 MFA 裝置

如果您是無法使用 MFA 進行登入的 IAM 使用者，則無法自行復原 MFA 裝置。您必須聯絡管理員以停用裝置。然後，您就可以啟用新裝置。

以 IAM 使用者身分取得 MFA 裝置的相關協助

1. 請聯絡 AWS 管理員或為您提供 IAM 使用者使用者名稱和密碼的其他人員。管理員必須依 停用 MFA 裝置 中所述停用 MFA 裝置，如此您才可以登入。
2. 下一步取決於您使用的 MFA 類型：

   - 針對虛擬 MFA 裝置，請從您的裝置移除帳戶。然後依 在 中指派虛擬 MFA 裝置 AWS 管理主控台 所述啟用虛擬裝置。
   - 針對 FIDO 安全金鑰，請聯絡第三方供應商，請其協助更換裝置。當您收到新的 FIDO 安全性金鑰時，請遵循 在 中指派通行金鑰或安全金鑰 AWS 管理主控台 中所述的程序來啟用它。
   - 針對硬體 TOTP 權杖，請聯絡第三方供應商，請其協助修復或更換裝置。在擁有新的實體 MFA 裝置後，依 在 中指派硬體 TOTP 字符 AWS 管理主控台 所述啟用裝置。

> **Note**
>
> 您無須將遺失或遭竊的 MFA 裝置更換成相同類型的裝置。您最多可以有八台任意組合的 MFA 裝置。例如，若您損壞了 FIDO 安全金鑰並訂購了新的金鑰，您可以使用虛擬 MFA 或硬體 TOTP 權杖，直到新的 FIDO 金鑰到達。

3. 若您的 MFA 裝置遺失或遭竊，請同時變更您的 密碼，以防攻擊者竊取身分驗證裝置，同時還可能擁有您目前的密碼。如需詳細資訊，請參閱管理 IAM 使用者的密碼

透過 MFA 實現安全的 API 存取

利用 IAM 政策，可以指定允許使用者呼叫的 API 操作。您可以套用額外安全性，方法是先要求使用者透過多重要素驗證 (MFA) 進行身分驗證，然後再允許使用者執行特別敏感的動作。

例如，您可能擁有允許使用者執行 Amazon EC2 `RunInstances`、`DescribeInstances` 與 `StopInstances` 動作的政策。但您可能想要限制類似 的破壞性動作，`TerminateInstances`並確保使用者只有在使用 AWS MFA 裝置進行身分驗證時，才能執行該動作。

主題

- 概觀
- 使用案例：跨帳戶委派的 MFA 防護
- 使用案例：目前帳戶中 API 操作存取的 MFA 防護
- 使用案例：擁有以資源為基礎的政策之資源的 MFA 防護

概觀

新增 MFA 防護到 API 操作將包括以下任務：

1. 管理員會為每個必須提出需要 AWS MFA 身分驗證之 API 請求的使用者設定 MFA 裝置。如需詳細資訊，請參閱AWS IAM 中的多重要素驗證。
2. 管理員會為使用者建立政策，其中包含檢查使用者是否使用 AWS MFA 裝置進行身分驗證的 `Condition` 元素。
3. 使用者呼叫其中一個支援 MFA 參數的 AWS STS API 操作：AssumeRole 或 GetSessionToken。作為呼叫的一部分，使用者包含與其關聯的裝置的裝置識別碼。使用者也包含裝置產生之以時間為基

礎的單次密碼 (TOTP)。在任一情況下，使用者都會取回稍後用來向 AWS發出其他請求的臨時安全憑證。

> **Note**
>
> 只有在服務支援臨時安全憑證時，才可使用該服務的 API 操作的 MFA 防護。如需這些服務的清單，請參閱使用暫時安全憑證存取 AWS。

如果授權失敗， AWS 會傳回存取遭拒錯誤訊息 （ 如同任何未經授權的存取 ）。使用受 MFA 保護的 API 政策時，如果使用者嘗試在沒有有效 MFA 身分驗證的情況下呼叫 API 操作， 會 AWS 拒絕存取政策中指定的 API 操作。如果 API 操作請求的時間戳記在政策中指定的允許範圍之外，也會拒絕該操作。使用者必須使用 MFA 代碼和裝置序號來請求新的臨時安全性憑證，並透過 MFA 重新進行身分驗證。

含有 MFA 條件的 IAM 政策

含有 MFA 條件的政策可連接到以下項目：

- IAM 使用者或群組
- 資源，例如 Amazon S3 儲存貯體、Amazon SQS 佇列或者 Amazon SNS 主題
- 可由使用者擔任的 IAM 角色的信任政策

可使用政策中的 MFA 條件來檢查以下屬性：

- 存在 - 如果只是驗證使用者是否已使用 MFA 進行身分驗證，請檢查 `aws:MultiFactorAuthPresent` 金鑰在 `Bool` 條件中是否為 `True`。只有在使用者使用短期憑證進行驗證時，索引鍵才會存在。長期憑證，例如存取金鑰，則不包括此鍵。
- 持續時間 - 如果您只希望在 MFA 身分驗證後的指定時間內授予存取權限，請使用數值條件類型將 `aws:MultiFactorAuthAge` 索引鍵的有效期與某個值 (如 3600 秒) 進行比較。請注意，如果未使用 MFA，則 `aws:MultiFactorAuthAge` 索引鍵不會顯示。

以下範例顯示 IAM 角色的信任政策，該政策包含一個 MFA 條件，用於測試是否存在 MFA 身分驗證。使用此政策，來自 `Principal`元素中 AWS 帳戶 指定 (`ACCOUNT-B-ID`以有效的 AWS 帳戶 ID 取代) 的使用者可以擔任此政策所連接的角色。不過，如果使用者使用 MFA 進行身分驗證，這類的使用者只能擔任該角色。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

如需有關 MFA 條件類型的詳細資訊，請參閱AWS 全域條件內容索引鍵、數位條件運算子 與 用於檢查條件索引鍵是否存在的條件運算子。

在 GetSessionToken 和 AssumeRole 之間選擇

AWS STS 提供兩種 API 操作，可讓使用者傳遞 MFA 資訊： `GetSessionToken`和 `AssumeRole`。使用者呼叫以獲取臨時安全憑證的 API 操作取決於適用於以下哪個案例。

針對以下案例使用 **`GetSessionToken`**：

- 呼叫 API 操作，以存取 AWS 帳戶 與發出請求的 IAM 使用者相同的 中的資源。請注意，只有在`GetSessionToken`憑證請求中包含 MFA 資訊時，請求中的臨時憑證才能存取 IAM 和 AWS STS API 操作。由於 `GetSessionToken` 傳回的臨時憑證包含 MFA 資訊，因此您可以檢查由該憑證發出的個別 API 操作中的 MFA。
- 存取受到以資源為基礎且包含 MFA 條件的政策所保護的資源。

`GetSessionToken` 操作的目的是使用 MFA 驗證使用者的身分。您不能使用政策來控制驗證操作。

針對以下案例使用 **`AssumeRole`**：

- 呼叫存取相同的或不同的 AWS 帳戶中的資源的 API 操作。API 呼叫可以包含任何 IAM 或 AWS STS API。請注意，要保護存取，您可以在使用者擔任角色時強制執行 MFA。由 `AssumeRole` 傳回的臨時憑證未將 MFA 資訊包含在上下文中，因此您無法檢查 MFA 的單一 API 操作。這就是您必須使用 `GetSessionToken` 限制對受到以資源為基礎之政策保護的資源的存取的原因。

> **Note**
>
> AWS CloudTrail 當 IAM 使用者使用 MFA 登入時， 日誌將包含 MFA 資訊。如果 IAM 使用者擔任 IAM 角色，CloudTrail 也會將 `mfaAuthenticated: true` 記錄在使用該擔任角色執行的動作的 `sessionContext` 屬性中。不過，CloudTrail 記錄與使用擔任角色的憑證進行 API 呼叫時 IAM 所需的記錄是分開的。如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

本文件稍後將提供有關如何實施這些使用案例的詳細資訊。

有關受 MFA 保護的 API 存取的要點

瞭解 API 操作的 MFA 防護有下列數個層面非常重要：

- MFA 防護僅透過使用臨時安全性憑證供，而該憑證必須使用 `AssumeRole` 或 `GetSessionToken` 來取得。
- 您無法搭配 AWS 帳戶根使用者 登入資料使用受 MFA 保護的 API 存取。
- 無法搭配 U2F 安全性金鑰使用受 MFA 保護的 API 存取。
- 聯合身分使用者無法獲指派 MFA 裝置以搭配 AWS 服務使用，因此無法存取 MFA 控制 AWS 的資源。(請查看下一要點。)
- 傳回暫時登入資料的其他 AWS STS API 操作不支援 MFA。對於 `AssumeRoleWithWebIdentity`和 `AssumeRoleWithSAML`，使用者由外部提供者進行驗證，AWS 無法判斷該提供者是否需要 MFA。對於 `GetFederationToken`，MFA 不一定要與特定使用者相關聯。
- 同樣地，長期憑證 (IAM 使用者存取金鑰和根使用者存取金鑰) 無法用於受 MFA 保護的 API 存取，因為此類憑證不會過期。
- 也可以在沒有 MFA 資訊的情況下呼叫 `AssumeRole` 與 `GetSessionToken`。在此情況下，呼叫者將取回臨時安全性憑證，但這些臨時憑證的工作階段資訊不會顯示使用 MFA 進行身分驗證的使用者。
- 若要建立 API 操作的 MFA 防護，可將 MFA 條件加入到政策。政策必須包含 `aws:MultiFactorAuthPresent` 條件索引鍵，才能強制使用 MFA。對於跨帳戶委派，該角色的信任政策必須包含條件索引鍵。
- 當您允許另一個 AWS 帳戶 存取您帳戶中的資源時，資源的安全性取決於受信任帳戶的組態（另一個帳戶，而非您的帳戶）。即使您要求多重要素驗證，也是如此。有權建立虛擬 MFA 裝置的可信帳戶中的任何身分都可以建構 MFA 宣告，以滿足角色信任政策的該部分。在允許另一個帳戶的成員存

取需要多重要素驗證 AWS 的資源之前，您應該確保信任帳戶的擁有者遵循安全最佳實務。例如，信任的帳戶應該限制存取敏感 API 操作 (例如 MFA 裝置管理 API 操作) 至特定的信任身分。

- 如果政策包含 MFA 條件，則在以下情況下將拒絕請求：使用者未進行 MFA 身分驗證或使用者提供了無效的 MFA 裝置識別碼或無效的 TOTP。

使用案例：跨帳戶委派的 MFA 防護

在此案例中，您想要將存取權委派給另一個帳戶中的 IAM 使用者，但前提是使用者已使用 AWS MFA 裝置進行身分驗證。如需有關跨帳戶委派的詳細資訊，請參閱 角色術語和概念。

假設您有一個帳戶 A (擁有要存取的資源的信任帳戶)，其 IAM 使用者 Anaya 擁有管理員許可。她希望對帳戶 B (可信任的帳戶) 中的使用者 Richard 授予存取權，但希望確保 Richard 在擔任該角色之前已使用 MFA 進行身分驗證。

1. 在信任帳戶 A 中，Anaya 會建立名為 的 IAM 角色，`CrossAccountRole`並將角色信任政策中的委託人設定為帳戶 B 的帳戶 ID。信任政策會將 AWS STS 許可授予 `AssumeRole`動作。Anaya 也將 MFA 條件加入到信任政策中，如以下範例中所示。

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Principal": {"AWS": "ACCOUNT-B-ID"},
       "Action": "sts:AssumeRole",
       "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
     }
   }
   ```

2. Anaya 在該角色新增一個許可政策，以指定允許該角色執行的操作。具有 MFA 防護的角色許可政策與任何其他角色許可政策沒有差別。以下範例顯示 Anaya 新增到角色的政策；它允許假定的使用者在帳戶 A 中的 Books 資料表上執行任何 Amazon DynamoDB 動作。此政策也允許 `dynamodb:ListTables` 動作，而這是在主控台中執行動作的必要項目。

   > **Note**
   >
   > 該許可政策不包含 MFA 條件。了解 MFA 身分驗證僅用於確定使用者是否可以擔任此角色這點非常重要。在使用者擔任此角色後，將不會進行進一步的 MFA 檢查。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:111122223333:table/Books"
        },
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
}
```

3. 在信任的帳戶 B 中，管理員會確定 IAM 使用者 Richard 已設定 AWS MFA 裝置，而且他知道裝置的 ID。如果是硬體 MFA 裝置，則裝置 ID 是序號，或如果是虛擬 MFA 裝置，裝置是 ARN。
4. 在帳戶 B 中，管理員將以下政策連接到使用者 Richard (或該使用者所在的群組)，該政策允許使用者呼叫 `AssumeRole` 動作。資源被設定到 Anaya 在第 1 步中所建立的角色 ARN。注意，該政策不包含 MFA 條件。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/CrossAccountRole"
            ]
        }
    ]
```

```
}
```

5. 在帳戶 B 中，Richard (或 Richard 正在執行的應用程式) 呼叫 AssumeRole。API 呼叫包含要擔任角色的 ARN (arn:aws:iam::*ACCOUNT-A-ID*:role/CrossAccountRole)、MFA 裝置的 ID 和 Richard 從其裝置中取得的目前 TOTP。

   當 Richard 呼叫 時AssumeRole， AWS 判斷他是否擁有有效的登入資料，包括 MFA 的需求。如果是這種情況，Richard 將成功取得角色，並且可在使用角色的臨時憑證時對帳戶 A 中名為 Books 的表格執行任何 DynamoDB 動作。

   有關呼叫 AssumeRole 的程式之範例，請參閱 使用 MFA 身分驗證呼叫 AssumeRole。

使用案例：目前帳戶中 API 操作存取的 MFA 防護

在此案例中，您應該確保 中的使用者只有在使用 AWS MFA 裝置驗證使用者時 AWS 帳戶 ，才能存取敏感的 API 操作。

假設您擁有帳戶 A，其中包含一組需要使用 EC2 執行個體的開發人員。普通開發人員可以使用執行個體，但他們未獲得 ec2:StopInstances 或 ec2:TerminateInstances 操作的許可。您希望僅允許幾個可信任的使用者執行這些「破壞性」特權操作，因此您將 MFA 防護加入到允許這些敏感 Amazon EC2 動作的政策中。

在此使用案例中，使用者 Sofía 是可信任的使用者之一。使用者 Anaya 是帳戶 A 中的管理員。

1. Anaya 確定 Sofía 已設定 AWS MFA 裝置，且 Sofía 知道裝置的 ID。如果是硬體 MFA 裝置，則裝置 ID 是序號，或如果是虛擬 MFA 裝置，裝置是 ARN。
2. Anaya 建立一個名為 EC2-Admins 的群組並將使用者 Sofía 加入到該群組中。
3. Anaya 將以下政策連接到 EC2-Admins 群組。此政策授予使用者呼叫 Amazon EC2 StopInstances 與 TerminateInstances 動作的許可，但前提是該使用者已使用 MFA 進行身分驗證。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:StopInstances",
      "ec2:TerminateInstances"
```

```
    ],
    "Resource": ["*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

4.

> **Note**
>
> 要讓此政策生效，使用者必須先登出，然後再次登入。

如果使用者 Sofía 需要停止或終止 Amazon EC2 執行個體，她 (或執行中的應用程式) 會呼叫 `GetSessionToken`。此 API 操作傳遞 MFA 裝置的 ID，以及 Sofía 從其裝置取得的目前 TOTP。

5. 使用者 Sofía (或 Sofía 正在使用的應用程式) 使用由 `GetSessionToken` 提供的臨時憑證來呼叫 Amazon EC2 `StopInstances` 或者 `TerminateInstances` 動作。

有關呼叫 `GetSessionToken` 的程式之範例，請參閱本文件後述的 使用 MFA 身分驗證呼叫 GetSessionToken。

使用案例：擁有以資源為基礎的政策之資源的 MFA 防護

在此案例中，您是 S3 儲存貯體、SQS 佇列或 SNS 主題的擁有者。您想要確保來自任何存取資源 AWS 帳戶 的使用者都經過 AWS MFA 裝置驗證。

此使用案例介紹了一種提供跨帳戶 MFA 防護的方法，無需使用者先擔任角色。在此情況下，若符合三個條件，使用者便可存取資源。使用者必須通過 MFA 的身分驗證，能夠從 `GetSessionToken` 取得臨時安全性憑證，並且在資源政策所信任的帳戶中。

假設您在帳戶 A 中並建立一個 S3 儲存貯體。您想要將此儲存貯體的存取權授予位於數個不同 中的使用者 AWS 帳戶，但前提是這些使用者已使用 MFA 進行身分驗證。

在此方案中，使用者 Anaya 是帳戶 A 中的管理員。使用者 Nikhil 是帳戶 C 中的 IAM 使用者。

1. 在帳戶 A 中，Anaya 建立一個名為 `Account-A-bucket` 的儲存貯體。
2. Anaya 將儲存貯體政策加入到儲存貯體。該政策允許帳戶 A、帳戶 B 或帳戶 C 中的所有使用者執行儲存貯體中的 Amazon S3 `PutObject` 和 `DeleteObject` 動作。該政策包含 MFA 條件。

   JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> Amazon S3 (僅) 針對根帳戶存取提供「MFA 刪除」功能。在您設定儲存貯體的版本控制狀態時，可啟用 Amazon S3 MFA Delete 功能。Amazon S3 MFA Delete 功能不適用於 IAM 使用者，在管理時獨立於 MFA 防護的 API 存取。即使 IAM 使用者有刪除儲存貯體的許可，但在啟用 Amazon S3 MFA Delete 功能時，也無法執行刪除。如需有關 Amazon S3 MFA Delete 的詳細資訊，請參閱 MFA Delete。

3. 在帳戶 C 中，管理員確定使用者 Nikhil 以 AWS MFA 裝置設定，而且知道裝置的 ID。如果是硬體 MFA 裝置，則裝置 ID 是序號，或如果是虛擬 MFA 裝置，裝置是 ARN。
4. 在帳戶 C 中，Nikhil (或該使用者正在執行的應用程式) 呼叫 `GetSessionToken`。此呼叫包括 MFA 裝置的 ID 或 ARN 以及 Nikhil 從其裝置中取得的目前 TOTP。
5. Nikhil (或 Nikhil 正在使用的應用程式) 使用由 `GetSessionToken` 傳回的暫時憑證來呼叫 Amazon S3 `PutObject` 動作，上傳檔案到 `Account-A-bucket`。

   有關呼叫 `GetSessionToken` 的程式之範例，請參閱本文件後述的 使用 MFA 身分驗證呼叫 GetSessionToken。

> **Note**
>
> AssumeRole 傳回的暫時憑證在此案例中則不會運作。雖然使用者可以提供 MFA 資訊以取得角色，AssumeRole 傳回的暫時憑證不會包含 MFA 資訊。有了這項資訊，才能符合政策中的 MFA 條件。

範本程式碼：使用多重要素驗證請求憑證

以下範例顯示如何呼叫 GetSessionToken 和 AssumeRole 操作，並傳遞 MFA 驗證參數。呼叫 GetSessionToken 不需要許可，但您必須擁有一個可讓您呼叫 AssumeRole 的政策。傳回的憑證隨後用於列出帳戶中的所有 S3 儲存貯體。

使用 MFA 身分驗證呼叫 GetSessionToken

以下範例說明如何呼叫 GetSessionToken 並傳遞 MFA 身分驗證資訊。然後，由 GetSessionToken 操作傳回的臨時安全憑證用於列出帳戶中的所有 S3 儲存貯體。

連接到執行此程式碼的使用者的政策 (或群組中的使用者)，會針對傳回的臨時憑證提供許可。對於此範本程式碼，政策必須授予使用者許可來請求 Amazon S3 ListBuckets 操作。

下列程式碼範例示範如何使用 GetSessionToken。

CLI

AWS CLI

為 IAM 身分取得一組短期憑證

下列 get-session-token 命令會為進行呼叫的 IAM 身分擷取一組短期憑證。產生的憑證可用於政策要求多重要素驗證 (MFA) 的請求。憑證會在產生後的 15 分鐘過期。

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

輸出：

```
{
```

```
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的請求臨時安全憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetSessionToken。

PowerShell

Tools for PowerShell V4

範例 1：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為設定時間段的臨時憑證。從目前 Shell 預設值中推斷出用於請求臨時憑證的憑證。若要指定其他憑證，請使用 -ProfileName 或 -AccessKey/-SecretKey 參數。

```
Get-STSSessionToken
```

輸出：

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

範例 2：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。從指定的設定檔中取得用於提出請求的憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

輸出：

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

範例 3：使用在設定檔 'myprofilename' 中指定其憑證的帳戶關聯的 MFA 裝置識別碼和裝置提供的值，傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

輸出：

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetSessionToken。

Tools for PowerShell V5

範例 1：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為設定時間段的臨時憑證。從目前 Shell 預設值中推斷出用於請求臨時憑證的憑證。若要指定其他憑證，請使用 -ProfileName 或 -AccessKey/-SecretKey 參數。

```
Get-STSSessionToken
```

輸出：

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

```
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

範例 2：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。從指定的設定檔中取得用於提出請求的憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

輸出：

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

範例 3：使用在設定檔 'myprofilename' 中指定其憑證的帳戶關聯的 MFA 裝置識別碼和裝置提供的值，傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

輸出：

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetSessionToken。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

透過傳遞 MFA 字符取得工作階段字符，並使用它列出該帳戶的 Amazon S3 儲存貯體。

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
```

```
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetSessionToken。

使用 MFA 身分驗證呼叫 AssumeRole

以下範例說明如何呼叫 `AssumeRole` 並傳遞 MFA 身分驗證資訊。然後，由 `AssumeRole` 傳回的臨時安全憑證用於列出帳戶中的所有 Amazon S3 儲存貯體。

如需有關此案例的詳細資訊，請參閱 使用案例：跨帳戶委派的 MFA 防護。

下列程式碼範例示範如何使用 `AssumeRole`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
```

```
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);
```

```
            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 AssumeRole。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
```

```
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
##############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AssumeRole。

C++

適用於 C++ 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
```

```
                                Aws::Auth::AWSCredentials &credentials,
                                const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 AssumeRole。

CLI

AWS CLI

擔任角色

下列 assume-role 命令會為 IAM 角色 s3-access-example 擷取一組短期憑證。

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

輸出：

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

該命令的輸出包含存取金鑰、私密金鑰以及可用來向 AWS進行驗證的工作階段字符。

對於 AWS CLI 使用，您可以設定與角色相關聯的具名設定檔。當您使用設定檔時，CLI AWS 會呼叫 assume-role 並為您管理登入資料。如需詳細資訊，請參閱《CLI 使用者指南》中的在 AWS CLI 中使用 IAM 角色。 AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AssumeRole。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
```

```
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, arn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
```

```
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 AssumeRole。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立用戶端。

```
import { STSClient } from "@aws-sdk/client-sts";
```

```
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

擔任 IAM 角色。

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 AssumeRole。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);
```

```
  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 AssumeRole。

PowerShell

Tools for PowerShell V4

範例 1：傳回一組臨時登入資料（存取金鑰、私密金鑰和工作階段字符），可用於存取請求使用者通常無法存取 AWS 的資源一小時。傳回的憑證擁有所擔任角色之存取政策和提供的政策所允許的許可 (您無法使用提供的政策來授予超過所擔任角色之存取政策所定義的許可)。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

範例 2：傳回一組臨時憑證，有效期為一小時，這些憑證擁有所擔任角色之存取政策中定義的相同許可。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

範例 3：傳回一組臨時憑證，提供序號，以及與用於執行 cmdlet 之使用者憑證相關聯的 MFA 產生的權杖。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

範例 4：傳回一組臨時憑證，這些臨時憑證擔任了客戶帳戶中定義的角色。對於第三方可以擔任的每個角色，客戶帳戶必須使用識別符來建立角色，必須在每次擔任角色時在 -ExternalId 參數中傳遞該識別符。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AssumeRole。

Tools for PowerShell V5

範例 1：傳回一組臨時登入資料（存取金鑰、私密金鑰和工作階段字符），可用於存取請求使用者通常無法存取 AWS 的資源一小時。傳回的憑證擁有所擔任角色之存取政策和提供的政策所允許的許可 (您無法使用提供的政策來授予超過所擔任角色之存取政策所定義的許可)。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

範例 2：傳回一組臨時憑證，有效期為一小時，這些憑證擁有所擔任角色之存取政策中定義的相同許可。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

範例 3：傳回一組臨時憑證，提供序號，以及與用於執行 cmdlet 之使用者憑證相關聯的 MFA 產生的權杖。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

範例 4：傳回一組臨時憑證，這些臨時憑證擔任了客戶帳戶中定義的角色。對於第三方可以擔任的每個角色，客戶帳戶必須使用識別符來建立角色，必須在每次擔任角色時在 -ExternalId 參數中傳遞該識別符。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AssumeRole。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

擔任需要 MFA 字符的 IAM 角色，並使用暫時性憑證列出該帳戶的 Amazon S3 儲存貯體。

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
```

```
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 AssumeRole。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
```

```
  # @param role_arn [String] The ARN of the role that is assumed when these
 credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
 assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 AssumeRole。

Rust

適用於 Rust 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
```

```
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的AWS SDK API 參考》中的 AssumeRole。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }
```

```
            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 《適用於 Swift 的AWS SDK API 參考》中的 AssumeRole。

IAM 使用者的服務特定憑證

服務特定的登入資料是專門為特定 AWS 服務設計的身分驗證機制。相較於標準登入資料，這些 AWS 登入資料提供簡化的身分驗證，並針對個別 AWS 服務的身分驗證需求量身打造。與可在多個 AWS 服務中使用的存取金鑰不同，服務特定的登入資料僅設計用於建立它們的服務。這種具有較強針對性的方法透過限制憑證的適用範圍來增強安全性。

服務特定憑證通常包含使用者名稱與密碼對，或根據特定服務的需求經格式化的專用 API 金鑰。在建立服務特定憑證時，它們預設為處於作用中狀態，並且可以立即使用。每個 IAM 使用者的每項受支援服務最多可以有兩組服務特定憑證。此限制可讓您在在需要時輪換至新集時維持一個作用中集。 AWS 目前支援下列服務的服務特定憑證：

何時使用服務特定的登入資料

服務特定的登入資料旨在與第三方程式庫、SDKs、工具或應用程式相容，這些應用程式本質上與 AWS 登入資料、 AWS 軟體SDKs或 AWS APIs 不相容。這類使用案例包括從自我託管的基礎設施或其他供應商託管 AWS 的服務遷移至 服務。

從頭開始，並盡可能使用 AWS 臨時登入資料，例如 IAM 角色提供的臨時登入資料，以使用支援 AWS 臨時登入資料的 AWS SDK 或程式庫來驗證 AWS 服務。

輪換服務特定憑證

作為安全性最佳實務，請定期輪換服務特定憑證。若要在不中斷應用程式的情況下輪換憑證，請執行下列動作：

1. 為相同服務和 IAM 使用者再建立一組服務特定憑證
2. 更新所有應用程式以使用新憑證，並確認它們可正常運作

3. 將原始憑證的狀態變更為「非作用中」
4. 確認所有應用程式仍在正常運作
5. 確定不再需要非作用中的服務特定憑證後，將其刪除

監控服務特定憑證

您可以使用 AWS CloudTrail 來監控帳戶中 AWS 服務特定登入資料的使用情況。若要檢視與服務特定憑證使用情況相關的 CloudTrail 事件，請檢閱 CloudTrail 日誌以取得使用憑證之服務的事件。如需詳細資訊，請參閱使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail。

為提高安全性，請考慮設定 CloudWatch 警示，以便通知您特定的憑證使用模式，這些使用模式可能指示存在未經授權的存取或其他安全問題。如需詳細資訊，請參閱 AWS CloudTrail User Guide 中的 Monitoring CloudTrail Log Files with Amazon CloudWatch Logs。

下列主題提供有關服務特定憑證的資訊。

主題

- AWS 服務的 API 金鑰
- 將 IAM 與 Amazon Keyspaces (適用於 Apache Cassandra) 搭配使用

AWS 服務的 API 金鑰

除了標準 IAM 登入資料，例如臨時安全登入資料和長期存取金鑰，某些 AWS 服務還支援用於驗證程式設計請求的 API 金鑰。 AWS 提供兩種類型的 API 金鑰：

- 長期 API 金鑰 – 長期 API 金鑰與 IAM 使用者相關聯，並使用 IAM 服務特定的憑證產生。這些登入資料僅供單一 AWS 服務使用，透過限制登入資料範圍來增強安全性。您可以設定長期 API 金鑰的過期時間。若要產生長期 API 金鑰，您可以使用 IAM 或服務特定的主控台 AWS CLI、 或 AWS API。
- 短期 API 金鑰 – 短期 API 金鑰是使用 AWS Signature 第 4 版的預先簽章 URL。短期 API 金鑰與產生 API 金鑰的身分憑證具有相同的許可和過期時間，有效期最長為 12 小時或主控台工作階段的剩餘時間，以較短者為準。您可以使用 Amazon Bedrock/AWS 上的 Claude Platform 主控台、Python 和其他程式設計語言的套件來產生短期 API 金鑰。如需詳細資訊，請參閱《Amazon Bedrock 使用者指南》中的產生 Amazon Bedrock API 金鑰和AWS 上的 Claude Platform 《 使用者指南》中的身分驗證。

> **Note**
>
> 與短期 API 金鑰相比，長期 API 金鑰的安全風險更高。建議盡可能使用短期 API 金鑰或臨時安全憑證。如果使用長期 API 金鑰，建議定期輪換金鑰。

支援 API 金鑰的服務

下表列出支援 API 金鑰 AWS 的服務，以及每個服務支援的 API 金鑰類型。

#	服務	長期 API 金鑰	短期 API 金鑰	自動連接的受管政策	服務特定文件
1	Amazon Bedrock	是	是	AmazonBedrockLimitedAccess	使用 Amazon Bedrock API 金鑰
2	AWS 上的 Claude Platform	是	是	AnthropicInferenceAccess	身分驗證
3	Amazon CloudWatch	是	N/A	CloudWatchAPIKeyAccess	設定指標的承載字符身分驗證
4	Amazon CloudWatch Logs	是	N/A	CloudWatchLogsAPIKeyAccess	設定承載字符身分驗證

當您為服務產生長期 API 金鑰時，對應的 AWS 受管政策會自動連接到 IAM 使用者，授予該服務的核心操作存取權。如果您需要額外的存取權，您可以修改 IAM 使用者的許可。如需有關修改許可的資訊，請參閱the section called “新增或移除身分許可”。

若要進一步了解特定 服務的 API 金鑰，請參閱上表中的服務特定文件連結。

長期 API 金鑰的先決條件

在 IAM 主控台中產生長期 API 金鑰之前，您必須符合下列先決條件：

- 用於與長期 API 金鑰建立關聯的 IAM 使用者。如需有關建立 IAM 使用者的說明，請參閱the section called “建立使用者”。
- 您必須擁有下列 IAM 政策許可，才能管理 IAM 使用者的服務特定憑證。此範例政策授予建立、列出、更新、刪除和重設服務特定憑證的許可。以您將產生長期 API 金鑰的 IAM 使用者名稱取代資源元素中的*username*值：

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageBedrockServiceSpecificCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:UpdateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ResetServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/username"
        }
    ]
}
```

產生長期 API 金鑰（主控台）

在 IAM 主控台中為特定服務產生長期 API 金鑰

1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽面板中，選擇使用者。
3. 選擇您要為其產生長期 API 金鑰的 IAM 使用者。
4. 選擇 Security credentials (安全憑證) 索引標籤。
5. 在 API 金鑰區段中，選擇產生 API 金鑰。
6. 從AWS 服務下拉式清單中，選擇您要 API 金鑰進行身分驗證的服務。
7. 對於 API 金鑰過期，請執行下列其中一項操作：

   - 選擇 1、5、30、90 或 365 天的 API 金鑰過期持續時間。

- 選擇自訂持續時間以指定自訂 API 金鑰過期日期。
- 選擇永不過期 （不建議）。

8. 選擇產生 API 金鑰。
9. 複製或下載您的 API 金鑰。這是您唯一可以檢視 API 金鑰值的機會。

> **Important**
>
> 請妥善儲存您的 API 金鑰。關閉此對話方塊之後，您將無法再次擷取此 API 金鑰。如果您遺失或忘記 API 金鑰，則無法擷取它。反之，請產生新的 API 金鑰並將舊金鑰設為非作用中。

產生長期 API 金鑰 (AWS CLI)

若要使用 產生長期 API 金鑰 AWS CLI，請使用下列步驟：

1. 使用 create-user 命令建立將與 服務搭配使用的 IAM 使用者：

```
aws iam create-user \
    --user-name APIKeyUser_1
```

2. 使用 attach-user-policy 命令將 AWS 受管政策連接至 IAM 使用者。

   對於 Amazon Bedrock：

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AmazonBedrockLimitedAccess
```

   對於 AWS 上的 Claude Platform：

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AnthropicInferenceAccess
```

   對於 Amazon CloudWatch：

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchAPIKeyAccess
```

對於 Amazon CloudWatch Logs：

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchLogsAPIKeyAccess
```

3. 使用 create-service-specific-credential 命令產生長期 API 金鑰。

對於 Amazon Bedrock：

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name bedrock.amazonaws.com \
    --credential-age-days 30
```

對於 AWS 上的 Claude Platform：

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name aws-external-anthropic.amazonaws.com \
    --credential-age-days 30
```

對於 Amazon CloudWatch：

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name cloudwatch.amazonaws.com \
    --credential-age-days 30
```

對於 Amazon CloudWatch Logs：

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name logs.amazonaws.com \
    --credential-age-days 30
```

> **Note**
>
> `--credential-age-days` 為選用參數。您可以指定介於 1–36600 天之間的值。如果您省略此參數，API 金鑰不會過期。

回應`ServiceApiKeyValue`中傳回的 是個別服務的長期 API 金鑰。請妥善儲存 `ServiceApiKeyValue` 值，因為您之後將無法再擷取它。

列出長期 API 金鑰 (AWS CLI)

若要列出特定使用者的長期 API 金鑰中繼資料，請使用 list-service-specific-credentials 命令搭配 `--user-name` 參數：

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --user-name APIKeyUser_1
```

> **Note**
>
> `bedrock.amazonaws.com` 以適當的服務名稱取代（例如，`logs.amazonaws.com`Amazon CloudWatch Logs 或 `aws-external-anthropic.amazonaws.com` AWS 上的 Claude Platform)。

若要列出帳戶中的所有長期 API 金鑰中繼資料，請使用 list-service-specific-credentials 命令搭配 `--all-users` 參數：

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --all-users
```

更新長期 API 金鑰的狀態 (AWS CLI)

若要更新長期 API 金鑰的狀態，請使用 update-service-specific-credential 命令：

```
aws iam update-service-specific-credential \
    --user-name "APIKeyUser_1" \
    --service-specific-credential-id "ACCA1234EXAMPLE1234" \
```

```
    --status Inactive|Active
```

產生長期 API 金鑰 (AWS API)

您可以使用下列 IAM API 操作來管理任何支援服務的長期 API 金鑰：

- CreateServiceSpecificCredential
- ListServiceSpecificCredentials
- UpdateServiceSpecificCredential
- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential

短期 API 金鑰 （選取服務）

選取的服務目前支援短期 API 金鑰。

如需有關搭配 Amazon Bedrock 產生和使用短期 API 金鑰的資訊，請參閱《Amazon Bedrock 使用者指南》中的產生 API 金鑰。

如需有關為 產生和使用短期 API 金鑰的資訊 AWS 上的 Claude Platform，請參閱AWS 上的 Claude Platform 《 使用者指南》中的身分驗證。

服務特定的資訊

- 如需搭配 Amazon Bedrock 使用 API 金鑰的詳細資訊，請參閱《Amazon Bedrock 使用者指南》中的使用 Amazon Bedrock API 金鑰。
- 如需搭配 使用 API 金鑰的詳細資訊 AWS 上的 Claude Platform，請參閱AWS 上的 Claude Platform 《 使用者指南》中的身分驗證。
- 如需搭配 Amazon CloudWatch 使用 API 金鑰的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的為指標設定承載字符身分驗證。
- 如需搭配 Amazon CloudWatch Logs 使用 API 金鑰的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的設定承載字符身分驗證。

將 IAM 與 Amazon Keyspaces (適用於 Apache Cassandra) 搭配使用

Amazon Keyspaces (適用於 Apache Cassandra) 是一種可擴展、高可用性且受管的 Apache Cassandra 相容資料庫服務。您可以透過 AWS 管理主控台或以程式設計方式存取 Amazon Keyspaces。若要使用服務特定憑證以程式設計方式存取 Amazon Keyspaces，您可以使用 `cqlsh` 或

開放原始碼 Cassandra 驅動程式。服務特定的憑證包括使用者名稱和密碼，例如 Cassandra 用於身分驗證和存取管理的使用者名稱和密碼。對於每位使用者，每個支援的服務最多可以有兩組服務特定憑證。

若要使用存取金鑰以程式設計方式 AWS 存取 Amazon Keyspaces，您可以使用 AWS SDK、 AWS Command Line Interface (AWS CLI) 或開放原始碼 Cassandra 驅動程式搭配 SigV4 外掛程式。若要進一步了解，請參閱《Amazon Keyspaces （適用於 Apache Cassandra) 開發人員指南》中的建立和設定 Amazon Keyspaces 的 AWS 登入資料。

> **Note**
>
> 如果您打算只透過主控台與 Amazon Keyspaces 互動，則不需要產生服務特定的憑證。如需詳細資訊，請參閱《Amazon Keyspaces (適用於 Apache Cassandra) 開發人員指南》中的使用主控台存取 Amazon Keyspaces。

如需有關存取 Amazon Keyspaces 所需許可的詳細資訊，請參閱 Amazon Keyspaces (適用於 Apache Cassandra) 開發人員指南中的 Amazon Keyspaces (適用於 Apache Cassandra) 以身分為基礎的政策範例。

產生 Amazon Keyspaces 憑證 (主控台)

您可以使用 AWS 管理主控台 為您的 IAM 使用者產生 Amazon Keyspaces （適用於 Apache Cassandra) 憑證。

產生 Amazon Keyspaces 服務特定的憑證 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)，然後選擇需要憑證的使用者名稱。
3. 在 Security Credentials (安全憑證) 索引標籤的 Credentials for Amazon Keyspaces (for Apache Cassandra) (Amazon Keyspaces (適用於 Apache Cassandra) 憑證) 上，選擇 Generate credentials (產生憑證)。
4. 您的服務特定憑證現在可供使用。這是唯一可以檢視或下載密碼的機會。您稍後無法進行復原。不過，您可以隨時重設密碼。請將使用者和密碼儲存在安全的位置，因為稍後需要用到。

產生 Amazon Keyspaces 憑證 (AWS CLI)

您可以使用 AWS CLI 為您的 IAM 使用者產生 Amazon Keyspaces （適用於 Apache Cassandra) 憑證。

產生 Amazon Keyspaces 服務特定的憑證 (AWS CLI)

- 使用下列命令：
  - aws iam create-service-specific-credential

產生 Amazon Keyspaces 登入資料 (AWS API)

您可以使用 AWS API 為您的 IAM 使用者產生 Amazon Keyspaces （適用於 Apache Cassandra) 憑證。

產生 Amazon Keyspaces 服務特定登入資料 (AWS API)

- 請完成下列操作：
  - CreateServiceSpecificCredential

尋找未使用的 AWS 登入資料

若要提高 的安全性 AWS 帳戶，請移除不需要的 IAM 使用者登入資料 （即密碼和存取金鑰）。例如，當使用者離開您的組織或不再需要 AWS 存取時，請尋找他們正在使用的登入資料，並確保他們不再運作。在理想情況下，如果不再需要可刪除憑證。有需要時，您隨時可以在日後重新建立它們。至少您應變更密碼或停用存取金鑰，讓前任持有者無法再繼續存取。

當然，未使用的定義可能有所不同，通常表示在指定期間內未使用的憑證。

尋找未使用的密碼

您可以使用 AWS 管理主控台 檢視使用者的密碼使用資訊。如果您有大量的使用者，您可以使用主控台來下載憑證報告與有關每位使用者上次使用他們的主控台密碼的資訊。您也可以從 AWS CLI 或 IAM API 存取資訊。

尋找未使用的密碼 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)。
3. 如果必要，請將 Console last sign-in (主控台上次登入) 欄位新增到使用者表格：
   a. 在最右側的表格上方，選擇設定圖示 ( )。

b. 在 選取可見欄 中選取 主控台上次登入。

c. 選擇 確認 返回使用者清單。

4. 主控台上次登入資料欄會顯示使用者上次 AWS 透過主控台登入 的日期。您可以使用此資訊來尋找其密碼超過指定期間都沒有登入的使用者。該欄位對於其密碼從未登入的使用者顯示 Never (永不)。None (無) 指出使用者無密碼。最近未使用的密碼可能就是待移除的項目。

> **Important**
>
> 由於服務問題，上次使用密碼的資料不包含從 2018 年 5 月 3 日 22:50 (太平洋日光時間) 到 2018 年 5 月 23 日 14:08 (太平洋日光時間) 使用的密碼。這會影響顯示於 IAM 主控台中的前次登入日期，以及在 IAM 憑證報告中的最後使用密碼日期，並由 GetUser API 操作傳回。如果使用者在受影響的時間登入，傳回的上次使用密碼的日期會是使用者最後一次在 2018 年 5 月 3 日之前登入的日期。對於在 2018 年 5 月 23 日 14:08 (太平洋日光時間) 之後登入的使用者，傳回的上次使用密碼日期會是準確的。
>
> 如果您使用密碼上次使用的資訊來識別未使用的登入資料以供刪除，例如刪除 AWS 過去 90 天內未登入 的使用者，建議您調整評估時段以包含 2018 年 5 月 23 日之後的日期。或者，如果您的使用者使用存取金鑰以 AWS 程式設計方式存取 ，您可以參考存取金鑰上次使用的資訊，因為它在所有日期都是準確的。

透過下載憑證報告尋找未使用的密碼 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Credential report (憑證報告)。
3. 選擇 Download Report (下載報告) 以下載名稱為 `status_reports_<date>T<time>.csv` 的逗號分隔值 (CSV) 檔案。第五個欄位包含 `password_last_used` 欄與日期或以下其中一項：
   - N/A (無) – 完全沒有指派密碼的使用者。
   - no_information (無資訊) – 自從 IAM 在 2014 年 10 月 20 日開始追蹤密碼使用期限，未使用他們密碼的使用者。

若要尋找未使用的密碼 (AWS CLI)

執行以下命令來尋找未使用的密碼：

- `aws iam list-users` 傳回使用者清單，每個都有一個 PasswordLastUsed 值。如果缺少值，則表示使用者沒有密碼或者自從 IAM 在 2014 年 10 月 20 日開始追蹤密碼使用期限，未使用密碼。

尋找未使用的密碼 (AWS API)

呼叫以下操作來尋找未使用的密碼：

- `ListUsers` 傳回使用者集合，每個選項都有 `<PasswordLastUsed>` 值。如果缺少值，則表示使用者沒有密碼或者自從 IAM 在 2014 年 10 月 20 日開始追蹤密碼使用期限，未使用密碼。

如需關於下載憑證報告的命令的資訊，請參閱取得憑證報告 (AWS CLI)。

尋找未使用的存取金鑰

您可以使用 AWS 管理主控台 來檢視使用者的存取金鑰用量資訊。如果您有大量的使用者，您可以使用主控台來下載憑證報告，以尋找每位使用者上次使用他們的存取金鑰的時間。您也可以從 AWS CLI 或 IAM API 存取資訊。

尋找未使用的存取金鑰 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)。
3. 如有必要，請將 Access key last used (上次使用的存取金鑰) 欄位新增到使用者表格：
   a. 在最右側的表格上方，選擇設定圖示 ( )。
   b. 在 選取可見欄 中選取 存取金鑰上次使用時間。
   c. 選擇 確認 返回使用者清單。
4. 上次使用的存取金鑰欄會顯示自上次 AWS 以程式設計方式存取使用者以來的天數。您可以使用此資訊來尋找存取金鑰超過指定期間都沒有使用的使用者。該欄會向沒有存取金鑰的使用者顯示 –。最近未使用的存取金鑰可能就是待移除的項目。

透過下載憑證報告尋找未使用的存取金鑰 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Credential Report (憑證報告)。
3. 選擇 Download Report (下載報告) 以下載名稱為 `status_reports_<date>T<time>.csv` 的逗號分隔值 (CSV) 檔案。欄位 11 到 13 包含存取金鑰 1 上次使用的日期、區域和服務資訊。欄位 16 到 18 包含存取金鑰 2 的相同資訊。如果使用者沒有存取金鑰，或者使用者自從 IAM 在 2015 年 4 月 22 日開始追蹤存取金鑰使用期限均未使用存取金鑰，該值為 N/A (無)。

若要尋找未使用的存取金鑰 (AWS CLI)

執行以下命令來尋找未使用的存取金鑰：

- `aws iam list-access-keys` 傳回有關使用者的存取金鑰的資訊，包括 `AccessKeyID`。
- `aws iam get-access-key-last-used` 需要存取金鑰 ID 並會傳回輸出，其中包含上次使用存取金鑰的 `LastUsedDate`、`Region`，以及上次所請求服務的 `ServiceName`。如果缺少 `LastUsedDate`，則表示自從 IAM 在 2015 年 4 月 22 日開始追蹤存取金鑰使用期限均未使用存取金鑰。

尋找未使用的存取金鑰 (AWS API)

呼叫以下操作來尋找未使用的存取金鑰：

- `ListAccessKeys` 傳回與指定的使用者關聯之存取金鑰的 `AccessKeyID` 值清單。
- `GetAccessKeyLastUsed` 需要存取金鑰 ID 並會傳回值集合。包含有是上次使用的存取金鑰的 `LastUsedDate`、`Region`，以及上次所請求服務的 `ServiceName`。如果缺少值，則表示使用者沒有存取金鑰，或者自從 IAM 在 2015 年 4 月 22 日開始追蹤存取金鑰使用期限均未使用存取金鑰。

如需關於下載憑證報告的命令的資訊，請參閱取得憑證報告 (AWS CLI)。

為您的 產生登入資料報告 AWS 帳戶

您可以產生並下載憑證報告，其中會列出帳戶中的所有使用者，及其各種憑證的狀態，包括密碼、存取金鑰和 MFA 裝置。您可以從 AWS 管理主控台、 AWS SDKs和命令列工具或 IAM API 取得登入資料報告。

> **Note**
>
> IAM 登入資料報告僅包含下列 IAM 受管登入資料：密碼、每個使用者的前兩個存取金鑰、MFA 裝置和 X.509 簽署憑證。報告不包含服務特定的登入資料 （例如 CodeCommit 密碼、Amazon Bedrock 長期 API 金鑰或 Amazon CloudWatch Logs 長期 API 金鑰 ） 或任何其他超過前兩個的使用者存取金鑰。如需完整的登入資料可見性，請使用 ListServiceSpecificCredentials 和 ListAccessKeys APIs。

您可以使用憑證報告協助您進行稽核和合規性工作。您可以使用報告來稽核憑證生命週期要求的效果，如密碼和存取金鑰更新。您可以向外部稽核員提供報告，或向稽核員授予許可，以便該人員直接下載報告。

您可以依照每四小時一次的頻率來產生憑證報告。當您請求報告時，IAM 會先檢查是否已在過去四小時內 AWS 帳戶 產生 的報告。若是如此，便會下載最新的報告。如果帳戶的最新報告是四小時前產生的，或者如果帳戶無先前報告，則 IAM 會產生並下載新報告。

主題

- 必要許可
- 了解報告格式
- 取得憑證報告 (主控台)
- 取得憑證報告 (AWS CLI)
- 取得登入資料報告 (AWS API)

必要許可

建立和下載報告必須要有以下許可：

- 建立憑證報告：`iam:GenerateCredentialReport`
- 下載報告：`iam:GetCredentialReport`

了解報告格式

憑證報告採取逗號分隔值 (CSV) 檔案為格式。您可以使用常用試算表軟體打開 CSV 檔以執行分析，也可以建構應用程式來以程式設計方式使用 CSV 檔並執行自訂分析。

CSV 檔案內含下列欄位：

使用者

易讀的使用者名稱。

arn

使用者的 Amazon Resource Name (ARN)。如需有關 ARN 的詳細資訊，請參閱 IAM ARN。

user_creation_time

使用者建立的日期和時間，採用 ISO 8601 日期時間格式。

password_enabled

當使用者有密碼時，此值為 TRUE，否則為 FALSE。對於作為組織的一部分建立的新成員帳戶，此值為 FALSE，因為這些帳戶預設不具備根使用者憑證。

password_last_used

上次使用 AWS 帳戶根使用者 或 使用者密碼登入 AWS 網站的日期和時間，採用 ISO 8601 日期時間格式。擷取使用者上次登入時間 AWS 的網站為 AWS 管理主控台、 AWS 開發論壇和 AWS Marketplace。如果密碼在 5 分鐘的時間範圍內多次使用，則僅在此欄位中記錄此期間內的第一次使用。

- 在以下情況中，此欄位的值為 no_information：
  - 從未使用使用者密碼。
  - 沒有與密碼關聯的登入資料，例如，在 IAM 於 2014 年 10 月 20 日開始追蹤此資訊後，使用者密碼未曾使用。
- 當使用者沒有密碼時，此欄位中的值是 N/A (不適用)。

> **Important**
>
> 由於服務問題，上次使用密碼的資料不包含從 2018 年 5 月 3 日 22:50 (太平洋日光時間) 到 2018 年 5 月 23 日 14:08 (太平洋日光時間) 使用的密碼。這會影響顯示於 IAM 主控台中的前次登入日期，以及在 IAM 憑證報告中的最後使用密碼日期，並由 GetUser API 操作傳回。如果使用者在受影響的時間登入，傳回的上次使用密碼的日期會是使用者最後一次在 2018 年 5 月 3 日之前登入的日期。對於在 2018 年 5 月 23 日 14:08 (太平洋日光時間) 之後登入的使用者，傳回的上次使用密碼日期會是準確的。
> 如果您使用密碼上次使用的資訊來識別未使用的登入資料以供刪除，例如刪除 AWS 過去 90 天內未登入 的使用者，建議您調整評估時段以包含 2018 年 5 月 23 日之後的日期。或者，如果您的使用者使用存取金鑰以 AWS 程式設計方式存取 ，您可以參考存取金鑰上次使用的資訊，因為它在所有日期都是準確的。

password_last_changed

使用者密碼上次設定的日期和時間，採用 ISO 8601 日期時間格式。若使用者沒有密碼，此欄位中的值是 N/A (不適用)。

password_next_rotation

如果帳戶的密碼政策要求密碼輪換，則此欄位包含使用者需要設定新密碼的日期和時間，採用 ISO 8601 日期時間格式。 AWS 帳戶 （ 根 ） 的值一律為 not_supported。

mfa_active

當使用者啟用多重要素驗證 (MFA) 裝置時，此值為 TRUE，否則為 FALSE。

access_key_1_active

當使用者有存取金鑰且存取金鑰狀態為 Active 時，此值為 TRUE，否則為 FALSE。同時適用於帳戶根使用者和 IAM 使用者。

access_key_1_last_rotated

使用者的存取金鑰建立或前次變更的時間與日期，採用 ISO 8601 日期時間格式。若使用者沒有主動式存取金鑰，此欄位中的值是 N/A (不適用)。同時適用於帳戶根使用者和 IAM 使用者。

access_key_1_last_used_date

使用者存取金鑰最近用於登入 AWS API 請求的日期與時間，採用 ISO 8601 日期時間格式。如果存取金鑰在 15 分鐘的時間範圍內多次使用，則僅在此欄位中記錄第一次使用。同時適用於帳戶根使用者和 IAM 使用者。

在以下情況中，此欄位的值為 N/A (不適用)：

- 使用者無存取金鑰。
- 從未使用存取金鑰。
- 自 IAM 於 2015 年 4 月 22 日開始追蹤此資訊後，存取金鑰未曾使用。

access_key_1_last_used_region

最近使用過存取金鑰的 AWS 區域。如果存取金鑰在 15 分鐘的時間範圍內多次使用，則僅在此欄位中記錄第一次使用。同時適用於帳戶根使用者和 IAM 使用者。

在以下情況中，此欄位的值為 N/A (不適用)：

- 使用者無存取金鑰。
- 從未使用存取金鑰。
- 存取金鑰最後一次使用的時間為 IAM 於 2015 年 4 月 22 日開始追蹤此資訊之前。
- 上次使用的服務沒有區域限制，如 Amazon S3。

access_key_1_last_used_service

最近使用 存取金鑰存取 AWS 的服務。此欄位中的值會使用服務的命名空間 — 舉例來說，Amazon S3 為 s3、而 Amazon EC2 為 ec2。如果存取金鑰在 15 分鐘的時間範圍內多次使用，則僅在此欄位中記錄第一次使用。同時適用於帳戶根使用者和 IAM 使用者。

在以下情況中，此欄位的值為 N/A (不適用)：

- 使用者無存取金鑰。
- 從未使用存取金鑰。
- 存取金鑰最後一次使用的時間為 IAM 於 2015 年 4 月 22 日開始追蹤此資訊之前。

access_key_2_active

當使用者有第二個存取金鑰且第二個存取金鑰狀態為 Active 時，此值為 TRUE。否則為 FALSE。同時適用於帳戶根使用者和 IAM 使用者。

> **Note**
>
> 使用者最多可有兩個存取金鑰，首先更新金鑰，然後刪除先前的金鑰，從而使輪換變得更簡單。如需有關更新存取金鑰的詳細資訊，請參閱 更新存取金鑰。

access_key_2_last_rotated

使用者的第二個存取金鑰建立或前次更新的時間與日期，採用 ISO 8601 日期時間格式。若使用者沒有第二個主動式存取金鑰，此欄位中的值是 N/A (不適用)。同時適用於帳戶根使用者和 IAM 使用者。

access_key_2_last_used_date

使用者第二個存取金鑰最近用於簽署 AWS API 請求的日期和時間，採用 ISO 8601 日期時間格式。如果存取金鑰在 15 分鐘的時間範圍內多次使用，則僅在此欄位中記錄第一次使用。同時適用於帳戶根使用者和 IAM 使用者。

在以下情況中，此欄位的值為 N/A (不適用)：

- 使用者沒有第二個存取金鑰。
- 從未使用第二個存取金鑰。
- 使用者的第二個存取金鑰最後一次使用的時間為 IAM 於 2015 年 4 月 22 日開始追蹤此資訊之前。

access_key_2_last_used_region

使用者的第二個最近使用過存取金鑰的 AWS 區域。如果存取金鑰在 15 分鐘的時間範圍內多次使用，則僅在此欄位中記錄第一次使用。同時適用於帳戶根使用者和 IAM 使用者。在以下情況中，此欄位的值為 N/A (不適用)：

- 使用者沒有第二個存取金鑰。
- 從未使用第二個存取金鑰。
- 使用者的第二個存取金鑰最後一次使用的時間為 IAM 於 2015 年 4 月 22 日開始追蹤此資訊之前。
- 上次使用的服務沒有區域限制，如 Amazon S3。

access_key_2_last_used_service

最近使用使用者的第二個存取金鑰存取 AWS 的服務。此欄位中的值會使用服務的命名空間 — 舉例來說，Amazon S3 為 s3、而 Amazon EC2 為 ec2。如果存取金鑰在 15 分鐘的時間範圍內多次使用，則僅在此欄位中記錄第一次使用。同時適用於帳戶根使用者和 IAM 使用者。在以下情況中，此欄位的值為 N/A (不適用)：

- 使用者沒有第二個存取金鑰。
- 從未使用第二個存取金鑰。
- 使用者的第二個存取金鑰最後一次使用的時間為 IAM 於 2015 年 4 月 22 日開始追蹤此資訊之前。

cert_1_active

當使用者有 X.509 簽署的憑證且該憑證的狀態為 Active 時，此值為 TRUE，否則為 FALSE。

cert_1_last_rotated

使用者的簽署憑證建立或前次變更的時間與日期，採用 ISO 8601 日期時間格式。若使用者沒有主動式簽署憑證，此欄位中的值是 N/A (不適用)。

cert_2_active

當使用者有第二個 X.509 簽署的憑證且該憑證的狀態為 Active 時，此值為 TRUE，否則為 FALSE。

> **Note**
>
> 使用者最多可有兩個 X.509 簽署憑證，讓輪換更容易。

cert_2_last_rotated

使用者的第二個簽署憑證建立或前次變更的時間與日期，採用 ISO 8601 日期時間格式。若使用者沒有第二個主動式簽署憑證，此欄位中的值是 N/A (不適用)。

additional_credentials_info

當使用者有兩個以上的存取金鑰或憑證時，此值是其他存取金鑰或憑證的數量，以及可用來列出與使用者相關聯之存取金鑰或憑證的動作。

取得憑證報告 (主控台)

您可以使用 AWS 管理主控台 將登入資料報告下載為逗號分隔值 (CSV) 檔案。

下載憑證報告 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Credential report (憑證報告)。
3. 選擇 Download Report (下載報告)。

取得憑證報告 (AWS CLI)

下載憑證報告 (AWS CLI)

1. 產生登入資料報告。 會 AWS 儲存單一報告。如果報告存在，產生憑證報告會覆寫先前的報告。`aws iam generate-credential-report`
2. 檢視上次產生的報告：`aws iam get-credential-report`

取得登入資料報告 (AWS API)

下載登入資料報告 (AWS API)

1. 產生登入資料報告。 會 AWS 儲存單一報告。如果報告存在，產生憑證報告會覆寫先前的報告。`GenerateCredentialReport`
2. 檢視上次產生的報告：`GetCredentialReport`

CodeCommit 的 IAM 登入資料：Git 登入資料、SSH 金鑰和 AWS 存取金鑰

CodeCommit 是一項受管版本控制服務，可在 AWS 雲端中託管私有的 Git 儲存庫。若要使用 CodeCommit，請將 Git 用戶端設定為與 CodeCommit 儲存庫通訊。作為此組態的一部分，您提供 CodeCommit 可用於對您進行身分驗證的 IAM 憑證。IAM 支援 CodeCommit 有三種類型的憑證：

- Git 憑證，一種 IAM 產生的使用者名稱和密碼對，可用於透過 HTTPS 與 CodeCommit 儲存庫進行通訊。
- SSH 金鑰，一種本機產生的公有私有金鑰對，您可以將其與 IAM 使用者關聯，以透過 SSH 與 CodeCommit 儲存庫進行通訊。
- AWS 存取金鑰，您可以使用 隨附的登入資料協助程式 AWS CLI ，透過 HTTPS 與 CodeCommit 儲存庫通訊。

> **Note**
>
> 您無法使用 SSH 金鑰或 Git 憑證來存取另一個 AWS 帳戶中的儲存庫。若要了解如何為另一個 IAM 使用者和群組設定 CodeCommit 儲存庫的存取權 AWS 帳戶，請參閱AWS CodeCommit《 使用者指南》中的使用 角色設定 AWS CodeCommit 儲存庫的跨帳戶存取權。

如需有關每個選項的詳細資訊，請參閱下列各節。

與 CodeCommit 一起使用 Git 憑證和 HTTPS (建議)

使用 Git 憑證後，您可以為 IAM 使用者產生靜態的使用者名稱和密碼對，然後將這些憑證用於 HTTPS 連線。您還可以將這些憑證用於支援靜態的 Git 憑證的任何第三方工具或整合開發環境 (IDE)。

由於這些憑證對於所有受支援的作業系統都是通用的，並且與大多數憑證管理系統、開發環境和其他軟體開發工具相容，因此建議使用此方法。您可以隨時重設 Git 憑證的密碼。您還可以使憑證處於非作用中，或者在不再需要時刪除它們。

> **Note**
>
> 您無法為 Git 憑證選擇自己的使用者名稱或密碼。IAM 會為您產生這些登入資料，以協助確保它們符合 CodeCommit 中 AWS 和 安全儲存庫的安全標準。您只能在產生憑證時作一次下載。請確定將憑證儲存在安全的位置。如有需要，您可以隨時重設密碼，但這樣做會使用舊密碼設定的任何連線無效。在連線之前，必須重新設定連線以使用新密碼。

如需詳細資訊，請參閱下列主題：

- 若要建立 IAM 使用者，請參閱 在 中建立 IAM 使用者 AWS 帳戶。
- 若要使用 CodeCommit 產生和使用 Git 憑證，請參閱 AWS CodeCommit 使用者指南中的使用 Git 憑證的 HTTPS 使用者。

> **Note**
>
> 產生 Git 憑證後變更 IAM 使用者的名稱不會變更 Git 憑證的使用者名稱。使用者名稱和密碼保持不變，且仍然有效。

若要更新服務特定的憑證

1. 除了目前正在使用的設定之外，還要建立第二個特定於服務的憑證設定。
2. 更新您的所有應用程式，以使用新設定的憑證並驗證應用程式是否正常工作。
3. 將原始憑證的狀態變更為「非作用中」。
4. 確認您的所有應用程式仍在運作。
5. 刪除非作用中特定於服務的憑證

與 CodeCommit 一起使用的 SSH 金鑰和 SSH

使用 SSH 連接，您可以在本機電腦上建立 Git 和 CodeCommit 用於 SSH 身分驗證的公有和私有金鑰檔案。您將公有金鑰與 IAM 使用者關聯，並將私有金鑰存放在本機電腦上。如需詳細資訊，請參閱下列主題：

- 若要建立 IAM 使用者，請參閱 在 中建立 IAM 使用者 AWS 帳戶。
- 若要建立 SSH 公有金鑰，並將其與 IAM 使用者關聯的詳細資訊，請參閱在 Linux、macOS 或 Unix 上的 SSH 連線，或查看 AWS CodeCommit 使用者指南中的在 Windows 上的 SSH 連線。

> **Note**
>
> 公有金鑰必須以 ssh-rsa 格式或 PEM 格式編碼。公有金鑰的最小位元長度為 2048 位元，最長為 16384 位元。這與您上傳的檔案大小不同。例如，您可以產生 2048 位元索引鍵，產生的

PEM 檔案長度為 1679 位元組。如果以其他格式或大小提供公有金鑰，則會看到錯誤訊息，指出該金鑰格式無效。

搭配 AWS CLI 登入資料協助程式和 CodeCommit 使用 HTTPS

做為使用 Git 登入資料進行 HTTPS 連線的替代方案，只要 Git 需要向 進行身分驗證 AWS 才能與 CodeCommit 儲存庫互動，您就可以允許 Git 使用 IAM 使用者登入資料或 Amazon EC2 執行個體角色的密碼編譯簽署版本。這是 CodeCommit 儲存庫不需要 IAM 使用者的唯一連線方法。這也是使用聯合身分存取和臨時憑證的唯一方法。如需詳細資訊，請參閱下列主題：

- 若要進一步了解有關聯合身分存取的詳細資訊，請參閱 身分提供者和 的聯合身分 AWS 和 對外部驗證的使用者的存取權 (聯合身分)。
- 若要進一步了解暫時憑證，請參閱 IAM 中的暫時安全憑證 和 CodeCommit 儲存庫的暫時存取。

AWS CLI 登入資料協助程式與其他登入資料協助程式系統不相容，例如 Keychain Access 或 Windows Credential Management。當使用憑證協助程式設定 HTTPS 連線時，會有額外的組態考量。如需詳細資訊，請參閱AWS CodeCommit 《 使用者指南》中的使用 AWS CLI 登入資料協助程式進行 Linux、macOS 或 Unix 上的 HTTPS 連線，或使用AWS CLI 登入資料協助程式進行 Windows 上的 HTTPS 連線。

在 IAM 中管理伺服器憑證

若要在 中啟用網站或應用程式的 HTTPS 連線 AWS，您需要 SSL/TLS 伺服器憑證。處理 AWS Certificate Manager (ACM) 可支援區域中的憑證時，我們建議您使用 ACM 來佈建、管理和部署您的伺服器憑證。在不支援的區域中，您必須使用 IAM 作為憑證管理員。如需了解 ACM 支援哪些區域，請參閱 AWS 一般參考 中的 AWS Certificate Manager 端點和配額。

Important

ACM 為佈建、管理和部署伺服器憑證的首選工具。使用 ACM，您可以請求憑證或將現有的 ACM 或外部憑證部署至 AWS 資源。ACM 提供的憑證為免費且會自動續約。在支援的區域中，您可以使用 ACM 從主控台或以程式設計方式管理伺服器憑證。如需有關 ACM 的詳細資訊，請參閱 AWS Certificate Manager 使用者指南。如需有關請求 ACM 憑證的詳細資訊，請參閱 AWS Certificate Manager 使用者指南中的請求公有憑證或請求私有憑證。如需有關將第三方憑證匯入 ACM 的詳細資訊，請參閱《AWS Certificate Manager 使用者指南》中的匯入憑證。

只有當您必須在 ACM 不支援的區域中支援 HTTPS 連接時，才應使用 IAM 作為憑證管理器。IAM 會安全地加密您的私有金鑰並將加密的版本儲存在 IAM SSL 憑證存放區中。IAM 支援在所有區域中部署伺服器憑證，但您必須從外部供應商取得憑證以搭配 使用 AWS。不能將 ACM 憑證上傳到 IAM。此外，也無法從 IAM 主控台管理憑證，

如需有關如何將第三方憑證上傳到 IAM 的詳細資訊，請參閱下列主題。

主題

- 上傳伺服器憑證 (AWS API)
- AWS 伺服器憑證的 API 操作
- 對伺服器憑證進行疑難排解

上傳伺服器憑證 (AWS API)

若要將伺服器憑證上傳至 IAM，您必須提供憑證及相符的私有金鑰。當憑證並非自我簽署憑證時，您還必須提供憑證鏈。(上傳自簽署憑證時無需憑證連結。) 在上傳憑證前，請確保您已具有所有這些項目且滿足以下條件：

- 憑證在上傳時必須有效。您不能在憑證有效期開始 (憑證的 `NotBefore` 日期) 之前或憑證有效期到期 (憑證的 `NotAfter` 日期) 之後上傳憑證。
- 私有金鑰必須為未加密。您無法上傳受密碼或複雜密碼保護的私有金鑰。有關解密已加密的私有金鑰的說明資訊，請參閱 對伺服器憑證進行疑難排解。
- 憑證、私有金鑰和憑證鏈全都必須採用 PEM 編碼。有關將這些項目轉換為 PEM 格式的說明資訊，請參閱 對伺服器憑證進行疑難排解。

若要使用 IAM API 上傳憑證，請傳送 UploadServerCertificate 請求。以下範例顯示如何使用 AWS Command Line Interface (AWS CLI) 執行此作業。該範例假設如下：

- PEM 編碼的憑證存放在名為 `Certificate.pem` 的檔案中。
- PEM 編碼的憑證鏈存放在名為 `CertificateChain.pem` 的檔案中。
- PEM 編碼的未加密私有金鑰存放在名為 `PrivateKey.pem` 的檔案中。
- (選用) 您要使用鍵值組來標記伺服器憑證。例如，您可以新增標籤金鑰 `Department` 和標籤值 `Engineering`，以協助您識別和整理憑證。

若要使用下列範例命令，請將這些檔案名稱取代為您自己的檔案名稱。使用您上傳憑證的名稱取代 *ExampleCertificate*。如果您要標記憑證，請將 *ExampleKey* 和 *ExampleValue* 標籤鍵值組取代為您自己的值。在連續的一行中輸入命令。為方便閱讀，以下範例包含分行符號和多餘的空格。

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
                                  --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

如果上述命令執行成功，則它將傳回上傳憑證的相關中繼資料，包括其 Amazon Resource Name (ARN)、易記名稱、識別符 (ID)、過期日期、標籤等。

> **Note**
>
> 如果您要上傳用於 Amazon CloudFront 的伺服器憑證，則必須使用 `--path` 選項指定路徑。路徑必須以 `/cloudfront` 開頭，且必須包含結尾反斜線 (例如，`/cloudfront/test/`)。

若要使用 AWS Tools for Windows PowerShell 上傳憑證，請使用 Publish-IAMServerCertificate。

AWS 伺服器憑證的 API 操作

使用下列命令來檢視、標記、重新命名及刪除伺服器憑證。

- 使用 GetServerCertificate 來擷取憑證。此請求將傳回憑證、憑證鏈 (如果已上傳一個) 和有關憑證的中繼資料。

  > **Note**
  >
  > 在您上傳後，無法從 IAM 下載或擷取私有金鑰。

- 使用 Get-IAMServerCertificate 來擷取憑證。
- 使用 ListServerCertificates 來列出您已上傳的伺服器憑證。請求會傳回包含有關每個憑證的中繼資料的清單。
- 使用 Get-IAMServerCertificates 來列出您已上傳的伺服器憑證。
- 使用 TagServerCertificate 來標記現有的伺服器憑證。

- 使用 UntagServerCertificate 來取消標記伺服器憑證。
- 使用 UpdateServerCertificate 來重新命名伺服器憑證或更新其路徑。

  以下範例顯示如何使用 AWS CLI執行此作業。

  若要使用以下範例指令，請將取代舊與新的憑證名稱與憑證路徑，並在連續的一行中輸入命令。為方便閱讀，以下範例包含分行符號和多餘的空格。

```
aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                  --new-server-certificate-
name CloudFrontCertificate
                                  --new-path /cloudfront/
```

  若要使用 AWS Tools for Windows PowerShell 重新命名伺服器憑證或更新其路徑，請使用 Update-IAMServerCertificate。
- 使用 DeleteServerCertificate 來刪除伺服器憑證。

  若要使用 AWS Tools for Windows PowerShell 刪除伺服器憑證，請使用 Remove-IAMServerCertificate。

對伺服器憑證進行疑難排解

您必須先確保憑證、私有金鑰和憑證連結均使用 PEM 編碼，然後才能將憑證上傳到 IAM。您還必須確保私有金鑰為未加密。請參閱以下範例。

Example PEM 編碼憑證範例

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example PEM 編碼的未加密私有金鑰範例

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example PEM 編碼的憑證鏈範例

憑證鏈包含一或多個憑證。您可以使用文字編輯器、Windows 的 copy 命令，或 Linux cat 命令，將憑證檔案串連為憑證鏈。當您包含多個憑證時，每個憑證必須認證先前的憑證。您可以透過串連憑證 (包含上一個根 CA 憑證) 來完成此動作。

以下範例包含三個憑證，但您的憑證鏈可包含更多或更少憑證。

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

如果這些項目在上傳到 IAM 時未採用正確的格式，您可以使用 OpenSSL 將其轉換為正確的格式。

將憑證或憑證鍊從 DER 轉換為 PEM

使用 OpenSSL x509 命令，如下列範例。在以下範例命令中，將 *Certificate.der* 替換為包含您的 DER 編碼的憑證檔案的名稱。以偏好的輸出檔案名稱取代 *Certificate.pem*，以包含 PEM 編碼的憑證。

```
openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

若要將私有金鑰從 DER 轉換為 PEM

使用 OpenSSL rsa 命令，如下列範例。在以下範例命令中，將 *PrivateKey.der* 替換為包含您的 DER 編碼的私有金鑰檔案的名稱。以偏好的輸出檔案名稱取代 *PrivateKey.pem*，以包含 PEM 編碼的私有金鑰。

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

解密已加密的私有金鑰 (移除密碼或密碼短語)

使用 OpenSSL rsa 命令，如下列範例。若要使用以下範例命令，將 *EncryptedPrivateKey.pem* 替換為包含您的已加密私有金鑰檔案的名稱。以偏好的輸出檔案名稱取代 *PrivateKey.pem*，以包含 PEM 編碼的未加密私有金鑰。

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

將憑證 bundle 從 PKCS#12 (PFX) 轉換為 PEM

使用 OpenSSL pkcs12 命令，如下列範例。在以下範例命令中，將 *CertificateBundle.p12* 替換為包含您的 PKCS#12 編碼的憑證 bundle 的名稱。以偏好的輸出檔案名稱取代 *CertificateBundle.pem*，以包含 PEM 編碼的憑證 bundle。

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

將憑證 bundle 從 PKCS#7 轉換為 PEM

使用 OpenSSL pkcs7 命令，如下列範例。在以下範例命令中，將 *CertificateBundle.p7b* 替換為包含您的 PKCS#7 編碼的憑證 bundle 的名稱。以偏好的輸出檔案名稱取代 *CertificateBundle.pem*，以包含 PEM 編碼的憑證 bundle。

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

IAM 使用者群組

IAM 使用者群組是 IAM 使用者的集合。使用者群組可讓您指定多個使用者的許可，從而可以更輕鬆地管理這些使用者的許可。例如，您可以擁有一個名為 Admins 的使用者群組，並為該使用者群組提供典型的管理員許可。該使用者群組中的任何使用者都自動擁有 Admins 群組許可。如果新使用者加入您的組織並需要管理員權限，您可以透過將使用者新增到 Admins 使用者群組來指派適當的許可。如果某人在您的組織中變更工作，而不是編輯該使用者的許可，您可以將他們從舊 IAM 群組中移除，並將他們新增到適當的新 IAM 群組中。

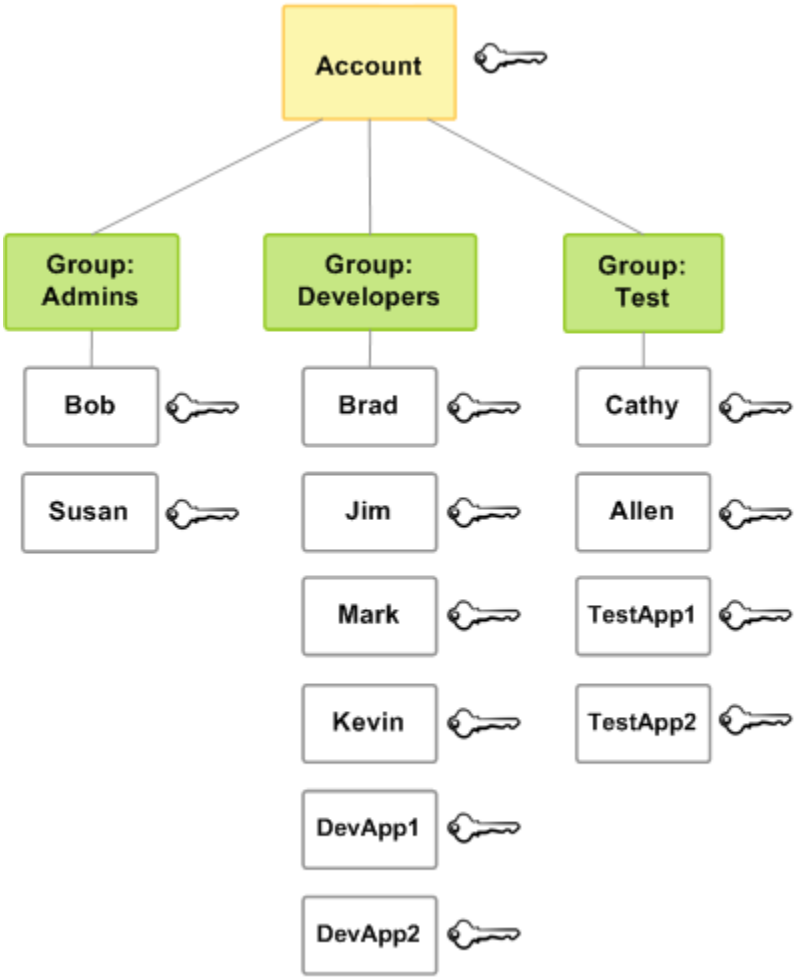
您可以將身分型政策連接至使用者群組，以便使用者群組中的所有使用者都會收到該政策的許可。您無法將使用者群組識別為政策 (例如資源型政策) 中的 `Principal`，因為群組與許可 (而非驗證) 相關，

並且主體是經過驗證的 IAM 實體。如需有關政策類型的詳細資訊，請參閱 以身分為基礎和以資源為基礎的政策。

以下是 IAM 群組的一些重要特性：

- 使用者群組可以包含許多使用者，使用者可以屬於多個使用者群組。
- 使用者群組不能為巢狀；群組只能包含使用者，而不包含其他 IAM 群組。
- 沒有預設使用者群組自動包含 AWS 帳戶中的所有使用者。如果您想擁有這樣的使用者群組，您必須建立它並指派每個新使用者到該群組。
- 中的 IAM 資源數量和大小會受到限制 AWS 帳戶，例如群組數量，以及使用者可以成為成員的群組數量。如需詳細資訊，請參閱IAM AWS STS 和配額。

下圖顯示一個小型公司的簡單範例。該公司擁有者為使用者建立 Admins 使用者群組，以便隨著公司的成長建立和管理其他使用者。所以此 Admins 使用者群組會建立 Developers 使用者群組和 Test 使用者群組。這些 IAM 群組都包含與 (Jim、Brad、DevApp1 等） 互動的使用者 AWS （人類和應用程式）。每個使用者都有一組個別的安全憑證。在這個範例中，每個使用者均屬於單一使用者群組。不過，使用者可屬於多個 IAM 群組。



建立使用者群組之後，您可以 檢視 IAM 群組、 將政策連接至 IAM 使用者群組和 重新命名 IAM 使用者群組。

建立 IAM 群組

> **Note**
>
> 最佳實務是，建議您要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 。如果遵循最佳實務，則您不用管理 IAM 使用者和群組。反之，您的使用者和群組是在 外部管理 AWS ，並且能夠以聯合身分的形式存取 AWS 資源。聯合身分是來自您的企業使用者目錄、Web 身分提供者、 AWS Directory Service、Identity Center 目錄或任何使用透過身分來源提供的登入資料存取 AWS 服務的使用者。聯合身分使用由其身分提供者定義的群組。如果您使用的是 AWS IAM Identity Center，請參閱AWS IAM Identity Center 《 使用者指南》中的

在 IAM Identity Center 中管理身分，以取得在 IAM Identity Center 中建立使用者和群組的相關資訊。

您可以建立 IAM 群組來管理具有類似角色或責任的多個使用者的存取許可。透過將政策連接至這些群組，可以授予或撤銷整組使用者的許可。您對群組許可所做的變更會自動套用至該群組的所有成員，確保存取控制保持一致，因此可以簡化安全政策的維護。建立群組之後，請根據您期望群組中 IAM 使用者要執行的工作類型向群組授予許可，然後將 IAM 使用者新增至群組。

如需有關建立 IAM 群組所需許可的資訊，請參閱存取 IAM 資源所需的許可。

建立 IAM 群組和連接政策

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中選擇 User groups (使用者群組)，然後選擇 Create group (建立群組)。
3. 針對 User group name (使用者群組名稱)，請輸入群組名稱。

   > **Note**
   >
   > AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。群組名稱可以是長達 128 個字母、數字以及這些字元的組合：加號 (+)、等號 (=)、逗號 (,)、句號 (.)、@ 符號、底線 (_) 以及連字號 (-)。名稱在帳戶中必須是唯一的。群組名稱無法透過大小寫進行區分。例如，您無法建立名為 **ADMINS** 和 **admins** 的群組。

4. 在使用者清單中，針對您要新增到群組的每個使用者，選取其核取方塊。
5. 在政策清單中，對於您要套用到群組所有成員的每個政策選取核取方塊。
6. 選擇 Create group (建立群組)。

AWS CLI

執行以下命令：

- aws iam create-group

API

呼叫以下操作：

- CreateGroup

檢視 IAM 群組

您可以列出帳戶中的所有 IAM 群組，列出使用者群組中的使用者，以及列出使用者所屬的 IAM 群組。如果您使用的是 CLI 或 API，您可以列出所有具有特定路徑字首的 IAM 群組。

Console

若要列出帳戶中的所有 IAM 群組，請執行下列動作：

- 在導覽窗格中，選擇使用者群組。

若要列出特定 IAM 群組中的 IAM 使用者，請執行下列動作：

- 在導覽窗格中，選擇 User groups (使用者群組)。然後選擇群組的名稱以開啟群組詳細資訊頁面。檢閱使用者索引標籤以查看群組成員資格。

若要列出使用者所在的 IAM 群組，請執行下列動作：

- 在導覽窗格中，選擇使用者 。然後選擇使用者名稱以開啟使用者詳細資訊頁面。選擇群組索引標籤，以查看使用者所屬的群組清單。

AWS CLI

若要列出帳戶中的所有 IAM 群組，請執行下列動作：

- aws iam list-groups

若要列出特定 IAM 群組中的使用者，請執行下列動作：

- aws iam get-group

若要列出使用者所在的 IAM 群組，請執行下列動作：

- aws iam list-groups-for-user

API

若要列出帳戶中的所有 IAM 群組，請執行下列動作：

- ListGroups

若要列出特定 IAM 群組中的使用者，請執行下列動作：

- GetGroup

若要列出使用者所在的 IAM 群組，請執行下列動作：

- ListGroupsForUser

編輯 IAM 群組中的使用者

使用 IAM 群組來將相同許可一次套用到多個使用者。然後可以從 IAM 群組新增或移除使用者。這在有人員加入和離開您的組織時很實用

檢閱政策存取

在移除群組之前，請使用群組詳細資訊頁面來檢閱群組的成員 (IAM 使用者)，在許可索引標籤上檢閱連接至群組的政策，以及使用上次存取索引標籤檢閱最近的服務層級活動。這有助於防止無意中從正在使用它的主體 (人員或應用程式) 中移除存取權。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

將 IAM 使用者新增至 IAM 群組

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 User groups (使用者群組)，然後選擇群組的名稱。
3. 選擇 Users (使用者) 索引標籤，然後選擇 Add Users (新增使用者)。選取要新增之使用者旁的核取方塊。

4. 選擇 Add users (新增使用者)。

AWS CLI

執行以下命令：

- aws iam add-user-to-group

API

呼叫以下操作：

- AddUserToGroup

從 IAM 群組中移除 IAM 使用者

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 User groups (使用者群組)，然後選擇群組的名稱。
3. 選擇 Users (使用者) 索引標籤。選取要移除之使用者旁的核取方塊，然後選擇 Remove users (移除使用者)。

AWS CLI

執行以下命令：

- aws iam remove-user-from-group

API

呼叫以下操作：

- RemoveUserFromGroup

將政策連接至 IAM 使用者群組

您可以將 AWS 提供的預先撰寫政策 受管政策連接至 AWS使用者群組，如下列步驟所述。若要連接客戶受管政策 ，即具有您建立的自訂許可的政策，必須先建立政策。如需建立客戶受管政策的詳細資訊，請參閱 使用客戶管理政策定義自訂 IAM 許可。

如需有關許可和政策的詳細資訊，請參閱 AWS 資源的存取管理。

將政策連接至 IAM 群組

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 User groups (使用者群組)，然後選擇群組的名稱。
3. 選擇 Permissions (許可) 標籤。
4. 選擇新增許可，然後選擇連接政策。
5. 連接至使用者群組的目前政策會顯示在目前的許可政策清單。在 Other permissions policies (其他許可政策) 清單中，選取要連接的政策名稱旁的核取方塊。您可以使用搜尋方塊來依類型和政策名稱篩選政策清單。
6. 選取要連接至 IAM 群組的政策，然後選擇連接政策。

AWS CLI

執行以下命令：

- `aws iam attach-group-policy`

API

呼叫以下操作：

- `AttachGroupPolicy`

重新命名 IAM 使用者群組

當您更改使用者群組名稱或路徑時，會發生以下情況：

- 連接到使用者群組的所有政策繼續採用新名稱。
- 使用者群組保留新名稱下的所有使用者。
- 使用者群組的唯一 ID 保持不變。如需有關唯一 ID 的詳細資訊，請參閱 唯一識別碼。

IAM 不會自動更新將該使用者群組視為資源以使用新名稱的政策。因此，當您重新命名使用者群組時，請務必謹慎。在重新命名使用者群組之前，必須手動檢查所有政策，以尋找按名稱提及該使用者群組的任何政策。例如，假設 Bob 是組織測試部分的經理。Bob 擁有政策連接到他的 IAM 使用者實體，可讓他從測試使用者群組中新增和移除使用者。如果管理員更改使用者群組的名稱 (或更改群組的路徑)，則管理員還需更新連接到 Bob 的政策以使用新名稱或新路徑。否則，Bob 將無法在該使用者群組中新增或移除使用者。

若要尋找將 IAM 群組作為資源來引用的政策，請執行下列動作：

1. 從 IAM 主控台的導覽窗格中，選擇 Policies (政策)。
2. 以 Type (類型) 資料欄排序，尋找 Customer managed (客戶受管) 自訂政策。
3. 選擇要編輯的政策的政策名稱。
4. 選擇許可索引標籤，然後選擇摘要。
5. 從服務清單中選擇 IAM，如果存在。
6. 在 Resource (資源) 欄位中尋找使用者群組的名稱。
7. 選擇編輯，可在政策中變更您的使用者群組名稱。

變更 IAM 使用者群組名稱

Console

1. 在導覽窗格中，選取使用者群組，然後選取群組名稱。
2. 選擇 Edit (編輯)。輸入新的使用者群組名稱，然後選擇 Save changes (儲存變更)。

AWS CLI

執行以下命令：

- aws iam update-group

API

呼叫以下操作：

- UpdateGroup

刪除 IAM 群組

當您在主控台中刪除某個 IAM 群組時，主控台會自動移除所有群組成員、分離所有連接的受管政策，並刪除所有內嵌政策。不過，由於 IAM 不會自動刪除將 IAM 群組作為資源來引用的政策，因此刪除 IAM 群組時必須小心。在刪除 IAM 群組之前，請手動檢閱政策，尋找依名稱提及該群組的政策。例如，測試團隊管理員 John 擁有與其 IAM 使用者實體相連的政策，可讓他從測試使用者群組中新增和移除使用者。如果管理員刪除該群組，則管理員還必須刪除連接到 John 的政策。否則，如果管理員重新建立已刪除的群組並為其提供相同名稱，John 的許可仍會保留，即使他離開測試團隊也一樣。

相反，在使用 CLI、SDK 或 API 刪除群組時，需先移除群組中的使用者。然後刪除嵌入在 IAM 群組中的所有內嵌政策。接下來，分離連接至該群組的所有受管政策。然後刪除該 IAM 群組本身。

Console

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 User groups (使用者群組)。
3. 在 IAM 群組清單中，選取要刪除的 IAM 群組名稱旁的核取方塊。您可以使用搜尋方塊，依類型、許可和群組名稱來篩選 IAM 群組清單。
4. 選擇 刪除。
5. 在確認方塊中，如果要刪除單一群組，請輸入群組名稱，然後選擇刪除。如果要刪除多個群組，請輸入要刪除的 IAM 群組數量，後接 **`user groups`**，然後選擇刪除。例如，如果要刪除三個群組，則輸入 **`3 user groups`**。

AWS CLI

1. 從 IAM 群組中移除所有使用者。
   - aws iam get-group (取得 IAM 群組中的使用者清單) 和 aws iam remove-user-from-group (從 IAM 群組中移除使用者)
2. 刪除嵌入在 IAM 群組中的所有內嵌政策。

- aws iam list-group-policies (取得 IAM 群組內嵌政策的清單) 和 aws iam delete-group-policy (刪除 IAM 群組的內嵌政策)

3. 分離連接至 IAM 群組的所有受管政策。

   - aws iam list-attached-group-policies (取得連接至 IAM 群組的受管政策清單) 和 aws iam detach-group-policy (從 IAM 群組中分離受管政策)

4. 刪除 IAM 群組。

   - aws iam delete-group

API

1. 從 IAM 群組中移除所有使用者。

   - GetGroup (取得 IAM 群組中的使用者清單) 和 RemoveUserFromGroup (從 IAM 群組中移除使用者)

2. 刪除嵌入在 IAM 群組中的所有內嵌政策。

   - ListGroupPolicies (取得 IAM 群組內嵌政策的清單) 和 DeleteGroupPolicy (刪除 IAM 群組的內嵌政策)

3. 分離連接至 IAM 群組的所有受管政策。

   - ListAttachedGroupPolicies (取得連接至 IAM 群組的受管政策清單) 和 DetachGroupPolicy (從 IAM 群組中分離受管政策)

4. 刪除 IAM 群組。

   - DeleteGroup

IAM 角色

IAM 角色是您可以在帳戶中建立的另一種 IAM 身分，具有特定的許可。IAM 角色類似於 IAM 使用者，因為同樣是 AWS 身分，也有許可政策可決定該身分在 AWS中可執行和不可執行的操作。但是，角色的目的是讓需要它的任何人可代入，而不是單獨地與某個人員關聯。此外，角色沒有與之關聯的標準長期憑證，例如密碼或存取金鑰。反之，當您擔任角色時，其會為您的角色工作階段提供臨時安全性憑證。

您可以使用 角色，將存取權委派給通常無法存取 AWS 資源的使用者、應用程式或服務。例如，您可能想要授予您 AWS 帳戶中的使用者存取他們通常沒有的資源，或授予使用者 AWS 帳戶 存取另一個帳戶中的資源。或者，您可能想要允許行動應用程式使用 AWS 資源，但不想在應用程式中嵌入 AWS 金鑰 （在難以更新的地方，以及使用者可以擷取的地方）。有時候，您想要提供 AWS 存取權給已在 外部定義身分的使用者 AWS，例如在您的公司目錄中。或者，您可能需要向第三方授予存取您帳戶的許可，讓他們可以對您的資源執行稽核。

對於這些案例，您可以使用 IAM 角色委派對 AWS 資源的存取。本節介紹各種角色和它們的不同使用方式，如何從不同方式中選出適合的時機與方法，以及如何建立、管理、切換到 (或擔任) 和刪除角色。

帳戶存取管理員是一項 IAM 功能，可讓您將組織帳戶中的 IAM 角色集中指派給 IAM Identity Center 使用者和群組。帳戶存取管理員透過 IAM 角色提供員工完整功能集的存取權。您可以搭配 IAM Identity Center 許可集或單獨使用。

> **Note**
>
> 當您第一次建立 時 AWS 帳戶，預設不會建立任何角色。為帳戶新增服務時，這些服務可能會新增服務連結角色，以支援其使用案例。
> 服務連結角色是連結至 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。
> 在您刪除服務連結角色之前，您必須先刪除這些角色的相關資源。這可保護您的資源，避免您不小心移除資源的存取許可。
> 如需哪些服務支援使用服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找 Service-Linked Role (服務連結角色) 欄中顯示 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。

主題

- 何時建立 IAM 使用者 (而不是角色)
- 角色術語和概念
- 其他資源
- 混淆代理人問題
- IAM 角色的常見案例
- IAM 角色建立

- IAM 角色管理
- 擔任角色的方法

何時建立 IAM 使用者 (而不是角色)

建議您僅將 IAM 使用者用於聯合身分不支援的使用案例。以下是部分使用案例：

- 無法使用 IAM 角色的工作負載 - 您可以從需要存取 AWS的位置執行工作負載。在某些情況下，您無法使用 IAM 角色提供暫時性憑證，例如針對 WordPress 外掛程式。在這些情況下，請將 IAM 使用者長期存取金鑰用於該工作負載，對 AWS進行身分驗證。
- 第三方 AWS 用戶端 – 如果您使用不支援透過 IAM Identity Center 存取的工具，例如未託管的第三方 AWS 用戶端或廠商 AWS，請使用 IAM 使用者長期存取金鑰。
- AWS CodeCommit 存取 – 如果您使用 CodeCommit 來存放程式碼，您可以使用 IAM 使用者搭配 SSH 金鑰或服務特定的登入資料，讓 CodeCommit 驗證您的儲存庫。除了將 IAM Identity Center 中的使用者用於一般身分驗證之外，我們也建議您這樣做。IAM Identity Center 中的使用者是您人力中需要存取您的 AWS 帳戶 或雲端應用程式的人員。若要將您 CodeCommit 儲存庫的存取權授予使用者，而不設定 IAM 使用者，您可以設定 git-remote-codecommit 公用程式。如需 IAM 和 CodeCommit 的詳細資訊，請參閱 CodeCommit 的 IAM 登入資料：Git 登入資料、SSH 金鑰和 AWS 存取金鑰。如需設定git-remote-codecommit公用程式的詳細資訊，請參閱AWS CodeCommit《使用者指南》中的使用輪換憑證連線至 AWS CodeCommit 儲存庫。
- Amazon Keyspaces (適用於 Apache Cassandra) 存取 – 在無法使用 IAM Identity Center 中的使用者的情況下，例如為了測試 Cassandra 相容性，您可以將 IAM 使用者搭配服務特定憑證使用，以便使用 Amazon Keyspaces 進行身分驗證。IAM Identity Center 中的使用者是您人力中需要存取您的 AWS 帳戶 或雲端應用程式的人員。您也可以使用暫時性憑證連線到 Amazon Keyspaces。如需詳細資訊，請參閱 Amazon Keyspaces (for Apache Cassandra) Developer Guide 中的 Using temporary credentials to connect to Amazon Keyspaces using an IAM role and the SigV4 plugin。
- 緊急存取：無法存取身分提供者且必須在 AWS 帳戶中採取動作的情況下。您可以將建立緊急存取 IAM 使用者列入彈性計劃中的一環。建議您使用多重要素驗證 (MFA) 嚴格控管緊急使用者憑證並加以保護。

角色術語和概念

以下是一些基本術語，可協助您開始使用角色。

Role

您可以在帳戶中建立的 IAM 身分具有特定的許可。IAM 角色與 IAM 使用者有一些相似之處。角色和使用者都是具備許可政策的 AWS 身分，可決定身分在 AWS中可執行和不可執行的操作。但是，角色的目的是讓需要它的任何人可代入，而不是單獨地與某個人員關聯。此外，角色沒有與之關聯的標準長期憑證，例如密碼或存取金鑰。反之，當您擔任角色時，其會為您的角色工作階段提供臨時安全性憑證。

角色可由以下項目擔任：

- 相同 AWS 帳戶 或另一個 中的 IAM 使用者 AWS 帳戶
- 相同帳戶中的 IAM 角色
- 服務主體，用於 AWS 服務和功能，例如：
  - 可讓您在運算服務上執行程式碼的服務，例如 Amazon EC2 或 AWS Lambda
  - 代表您對資源執行動作的功能，例如 Amazon S3 物件複寫
  - 為在 外部執行的應用程式提供臨時安全登入資料的服務 AWS，例如 IAM Roles Anywhere 或 Amazon ECS Anywhere
- 透過外部身分提供者 (IdP) 服務進行驗證的外部使用者，並且該服務與 SAML 2.0 或 OpenID Connect 相容

AWS 服務角色

服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務。

AWS 服務連結角色

服務連結角色是連結至 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

> **Note**
>
> 如果當服務開始支援服務連結的角色時您已經在使用服務，您可能會收到一封電子郵件，宣告您帳戶中的新角色。在這種情況下，服務會自動在您的帳戶中建立服務連結的角色。您不需要採取任何動作來支援此角色，而且您不應手動刪除它。如需詳細資訊，請參閱 我的 AWS 帳戶中出現新角色。

如需哪些服務支援使用服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找 Service-Linked Role (服務連結角色) 欄中顯示 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。如需詳細資訊，請參閱建立服務連結角色。

角色鏈結

當您使用一個角色來擔任第二個角色時，會發生角色鏈結。您可以 AWS 管理主控台 切換角色 AWS CLI、 或 API，透過 執行角色鏈結。例如，RoleA 擁有擔任 RoleB 的許可。您可以透過在 AssumeRole API 操作中使用長期使用者憑證，啟用 User1 擔任 RoleA。這會傳回 RoleA 短期憑證。透過角色鏈結，您可以使用 RoleA 的短期憑證，啟用 User1 擔任 RoleB。

當您擔任角色時，您可以傳遞工作階段標籤，並將標籤設為可轉移。可轉移工作階段標籤會傳遞到角色鏈中所有的後續工作階段。若要進一步了解工作階段標籤，請參閱 在 中傳遞工作階段標籤 AWS STS。

角色鏈結會將您的 AWS 管理主控台 AWS CLI 或 AWS API 角色工作階段限制為最多一小時。無論為個別角色設定的工作階段持續時間上限為何，其皆適用。當您使用 AssumeRole API 操作來擔任角色時，可以使用 DurationSeconds 參數指定角色工作階段的持續時間。您可以指定參數值高達 43200 秒 (12 小時)，這取決於您角色的最大工作階段持續時間設定 。然而，如果在您使用角色鏈結來擔任角色時，並提供大於一小時的 DurationSeconds 參數值，則操作失敗。

如需切換到 中角色的資訊 AWS 管理主控台，請參閱 從使用者切換至 IAM 角色 (主控台)。

委派

授予許可給某人，以允許存取您控制的資源。委派涉及設定兩個帳戶間的信任。第一個是擁有資源的帳戶 (信任帳戶)。第二個是包含需要存取資源使用者的帳戶 (受信任帳戶)。受信任帳戶和信任帳戶可以是以下任何一個：

- 相同帳戶。
- 都在您的組織的控制之下的個別帳戶。
- 由不同組織擁有的兩個帳戶。

若要委派許可來存取資源，您要在具有連接兩個政策的信任帳戶中建立 IAM 角色。許可政策會授予角色的使用者所需的許可，以對資源執行預定的任務。信任政策會指定允許哪些受信任帳戶成員可擔任角色。

當您建立信任政策時，您無法在委託人元素中指定萬用字元 (*) 做為 ARN 的一部分。信任政策會連接到信任帳戶中的角色，並且是許可的二分之一。另外一半是連接到受信任帳戶中使用者的許可政策，其允許切換為該使用者或擔任該角色 。擔任角色的使用者會暫時放棄其自己的許可，改為接受該角色的許可。當使用者退出或停止使用該角色時，會恢復原有的使用者許可。其他稱為外部 ID 的參數有助於確保在不是由相同組織控制的帳戶之間安全使用角色。

信任政策

JSON 政策文件，您會在其中定義您信任擔任角色的主體。角色信任政策是在 IAM 中連接至角色的以資源為基礎的必要政策。您在信任政策中可指定的主體包含使用者、角色、帳戶和服務。如需詳細資訊，請參閱 AWS 安全部落格中的 How to use trust policies in IAM roles。

跨帳戶存取的角色

將一個帳戶中的資源存取權，授予不同帳戶中受信任主體的角色。角色是授予跨帳戶存取權的主要方式。不過，某些 AWS 服務可讓您將政策直接連接到資源 （而不是使用角色做為代理）。這些稱為資源型政策，您可以使用它們來授予其他資源 AWS 帳戶 存取權中的委託人。其中一些資源包括 Amazon Simple Storage Service (S3) 儲存貯體、Amazon Glacier 保存庫、Amazon Simple Notification Service (SNS) 主題以及 Amazon Simple Queue Service (SQS) 佇列。若要了解哪些服務支援以資源為基礎的政策，請參閱 AWS 使用 IAM 的 服務。如需有關以資源為基礎的政策詳細資訊，請參閱 IAM 中的跨帳戶資源存取。

其他資源

下列資源可協助您進一步了解與 IAM 角色相關的 IAM 術語。

- 委託人是 中的實體 AWS ，可執行動作和存取資源。委託人可以是 AWS 帳戶根使用者、IAM 使用者或角色。代表 AWS 服務身分的委託人是服務委託人。使用角色信任政策中的 Principal 元素，來定義您信任能夠擔任角色的主體。

  如需您可以允許擔任角色之主體的詳細資訊和範例，請參閱 AWS JSON 政策元素： Principal。
- 聯合身分會在外部身分提供者與 之間建立信任關係 AWS。您可以使用現有的 OpenID Connect (OIDC) 或安全性聲明標記語言 (SAML) 2.0 提供者來管理誰可以存取 AWS 資源。當您使用 OIDC 和 SAML 2.0 設定這些外部身分提供者與 之間的信任關係時 AWS ，系統會將使用者指派給 IAM 角色。使用者也可以接收臨時憑證，其允許使用者存取您的 AWS 資源。

  如需有關聯合身分主體的詳細資訊，請參閱身分提供者和 的聯合身分 AWS。
- 聯合主體是來自 Directory Service企業使用者目錄或 OIDC 提供者的現有身分。當透過身分提供者請求存取時， 會將角色 AWS 指派給聯合主體。

  如需有關 SAML 和 OIDC 聯合身分主體的詳細資訊，請參閱聯合身分使用者工作階段和角色。
- 許可政策是身分型政策，可定義角色可以使用的動作和資源。文件的撰寫會根據 IAM 政策語言的規則。

  如需詳細資訊，請參閱IAM JSON 政策參考。

- 許可界限是一種進階功能，可供您使用政策，限制身分型政策可以授予角色的最大許可。您不能將許可界限用到服務連結的角色。

  如需詳細資訊，請參閱IAM 實體的許可界限。

混淆代理人問題

混淆代理人問題屬於安全性議題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。為了防止這種情況，如果您提供第三方 （稱為跨帳戶) 或其他 AWS 服務 （稱為跨服務) 存取您帳戶中的資源， AWS 會提供可協助您保護帳戶的工具。

有時，您可能需要讓第三方存取您的 AWS 資源 （委派存取權）。例如，您決定聘請名為 Example Corp 的第三方公司來監控您的 AWS 帳戶 並協助最佳化成本。為了追蹤您的每日支出，Example Corp 需要存取您的 AWS 資源。Example Corp 也可監控其他客戶的許多其他 AWS 帳戶 。您可以使用 IAM 角色，在您的 AWS 帳戶 和 Example Corp 帳戶之間建立信任關係。此案例的一個重要方面是外部 ID。外部 ID 是一個選用識別碼，您可在 IAM 角色信任政策中使用該識別碼來指定誰可以擔任該角色。外部 ID 的主要功能是解決並防止混淆代理人問題。

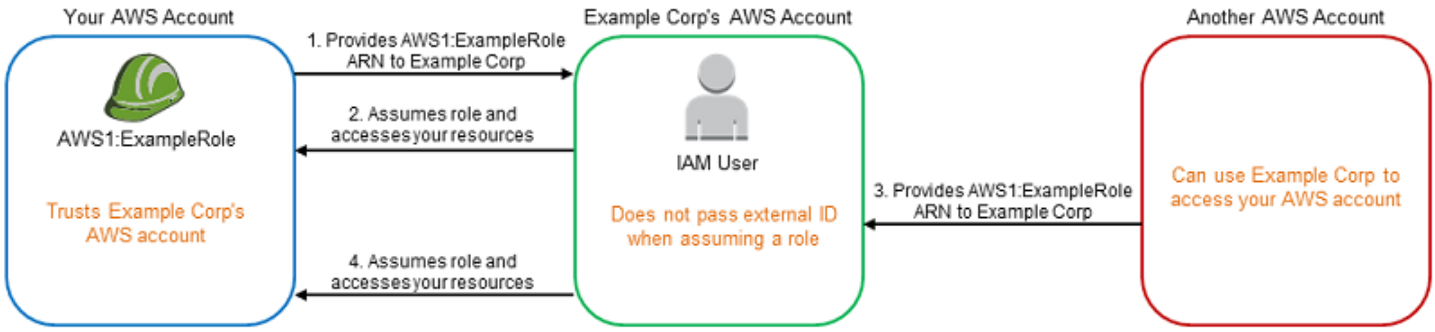
有些 AWS 服務 （呼叫 服務） 會使用其 AWS 服務主體從其他服務 AWS （稱為 服務） 存取 AWS 資源。在其中一些服務互動中，您可以設定呼叫服務，以與不同 中呼叫服務的資源通訊 AWS 帳戶。其中一個範例是 AWS CloudTrail 將 設定為寫入位於不同 的中央 Amazon S3 儲存貯體 AWS 帳戶。呼叫服務 CloudTrail 可透過新增 的允許陳述式，使用 S3 儲存貯體的政策授予對 S3 儲存貯體的存取權`cloudtrail.amazonaws.com`。

當呼叫 AWS 服務的服務委託人從呼叫的服務存取資源時，呼叫服務的資源政策只會授權 AWS 服務委託人，而不是設定呼叫服務的動作者。例如，信任 CloudTrail 服務主體且無條件的 S3 儲存貯體，可以從 AWS 帳戶 信任的管理員設定的 接收 CloudTrail 日誌，但如果 CloudTrail 知道 S3 儲存貯體的名稱 AWS 帳戶，也會從其中未經授權的演員接收 CloudTrail 日誌。

當演員使用 AWS 服務委託人的信任來存取他們不打算存取的資源時，就會發生混淆代理人問題。

預防跨帳戶混淆代理人

下圖說明了跨帳戶混淆代理人問題。



此案例假設如下：

- AWS 1 是您的 AWS 帳戶。
- AWS 1:ExampleRole 是您帳戶中的角色。此角色的信任政策透過將 Example Corp 的 AWS 帳戶指定為可擔任該角色的帳戶來信任 Example Corp。

將發生以下情況：

1. 在您開始使用 Example Corp 的服務時，您將向 Example Corp 提供 AWS 1:ExampleRole 的 ARN。
2. Example Corp 使用該角色 ARN 來取得臨時安全登入資料，以存取 中的資源 AWS 帳戶。這樣一來，您將信任 Example Corp 做為可代表您執行操作的「代理人」。
3. 另一個 AWS 客戶也開始使用 Example Corp 的服務，而此客戶也提供 AWS 1：ExampleRole 的 ARN，供 Example Corp 使用。另一個客戶可能已了解或猜到已不是機密資訊的 AWS 1:ExampleRole。
4. 當其他客戶要求 Example Corp 存取其帳戶中 AWS 的資源 （其聲稱為） 時，Example Corp 會使用 AWS 1：ExampleRole 來存取您帳戶中的資源。

這就是其他客戶可對您的資源進行未授權存取的方式。由於此客戶能夠誘使 Example Corp 無意中操作您的資源，因此 Example Corp 現在是一個「混淆代理人」。

Example Corp 可以透過要求在角色的信任政策中包含 `ExternalId` 條件檢查來解決混淆代理人問題。Example Corp 為每個客戶生成唯一的 `ExternalId` 值，並在其請求中使用該值來擔任此角色。因此，`ExternalId` 值必須在 Example Corp 的客戶中具備唯一性，並由 Example Corp 而非其客戶控制。這就是您從 Example Corp 取得該 ID 且不能自行提供該 ID 的原因。這可防止 Example Corp 成為混淆代理人，並授予另一個帳戶的 AWS 資源存取權。

在我們的方案中，假設 Example Corp 為您提供的獨有識別碼是 12345，而為另一個客戶提供的識別碼是 67890。這些識別碼已針對此方案進行簡化。通常，這些識別碼為 GUID。假定這些識別碼在 Example Corp 的客戶之間是獨有的，它們將是用於外部 ID 的有意義的值。

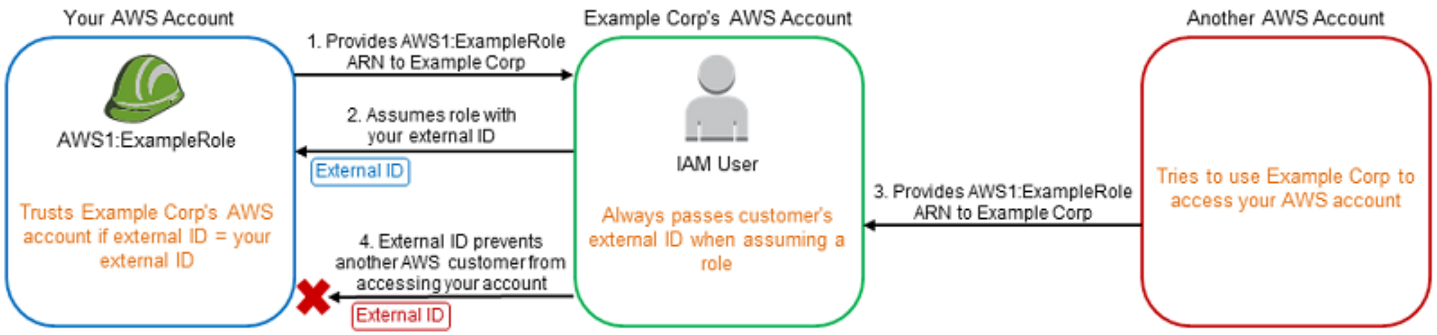
Example Corp 將為您提供外部 ID 值 12345。然後，您必須將一個 `Condition` 元素加入到角色的信任政策，該政策要求 `sts:ExternalId` 值為 12345，如下所示：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
    }
  }
}
```

此政策中的 Condition 元素允許 Example Corp 僅在 AssumeRole API 呼叫包括外部 ID 值 12345 時擔任該角色。Example Corp 會確認，只要它代表客戶擔任角色，就會一律在 AssumeRole 呼叫中包括客戶的外部 ID。即使其他客戶為您的 ARN 提供 Example Corp，也無法控制 Example Corp 在其請求中包含的外部 ID AWS。這有助於防止未經授權的客戶取得對您的資源的存取權限。

下圖說明此程序。

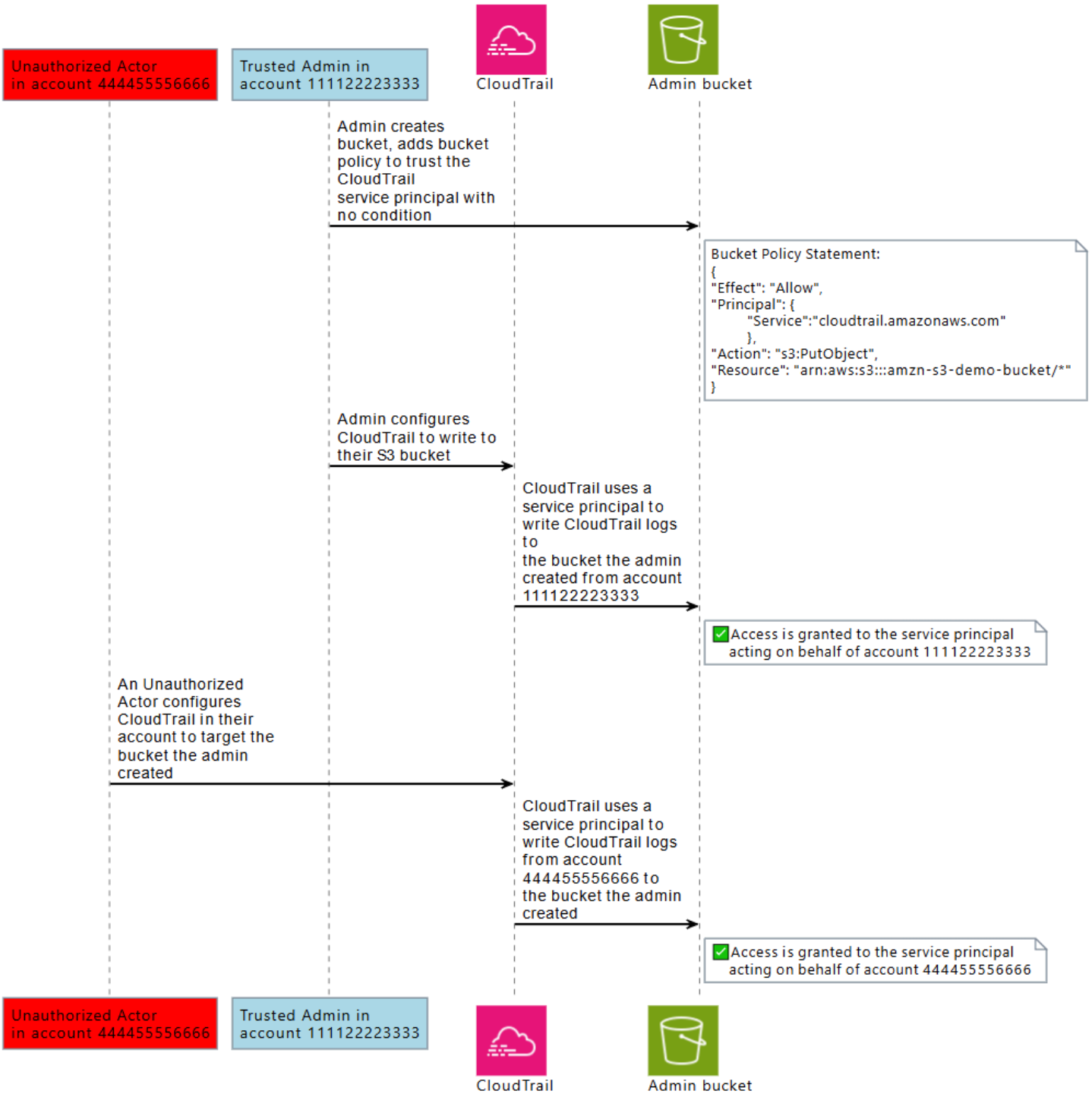


1. 與之前一樣，在您開始使用 Example Corp 的服務時，您將向 Example Corp 提供 AWS 1:ExampleRole 的 ARN。
2. 在 Example Corp 使用該角色 ARN 來擔任 AWS 1:ExampleRole 角色時，Example Corp 將在 AssumeRole API 呼叫中包含您的外部 ID (12345)。外部 ID 符合角色的信任政策，因此 AssumeRole API 呼叫成功，而 Example Corp 會取得臨時安全登入資料來存取您 中的資源 AWS 帳戶。
3. 另一個 AWS 客戶也開始使用 Example Corp 的服務，而如同之前一樣，此客戶也提供 AWS 1：ExampleRole 的 ARN，供 Example Corp 使用。
4. 但這一次，在 Example Corp 嘗試擔任角色 AWS 1:ExampleRole 時，它提供與其他客戶建立關聯的外部 ID (67890)。其他客戶無法更改此外部 ID。Example Corp 這樣做是因為另一個客戶請求使用該角色，因此 67890 表示 Example Corp 正在其中操作的環境。因為您已將具有您自己的外部 ID (12345) 的條件新增至 AWS 1:ExampleRole 的信任政策，所以 AssumeRole API 呼叫將失敗。該其他客戶不能對您帳戶中的資源進行未經授權的存取 (由圖表中的紅色 "X" 表示)。

該外部 ID 說明阻止任何其他客戶誘使 Example Corp 無意中存取您的資源。

預防跨服務混淆代理人

下圖使用 CloudTrail 和 Amazon S3 互動範例示範了跨服務混淆代理人問題，其中未經授權的執行者將 CloudTrail 日誌寫入到他們無權存取的 Amazon S3 儲存貯體中。



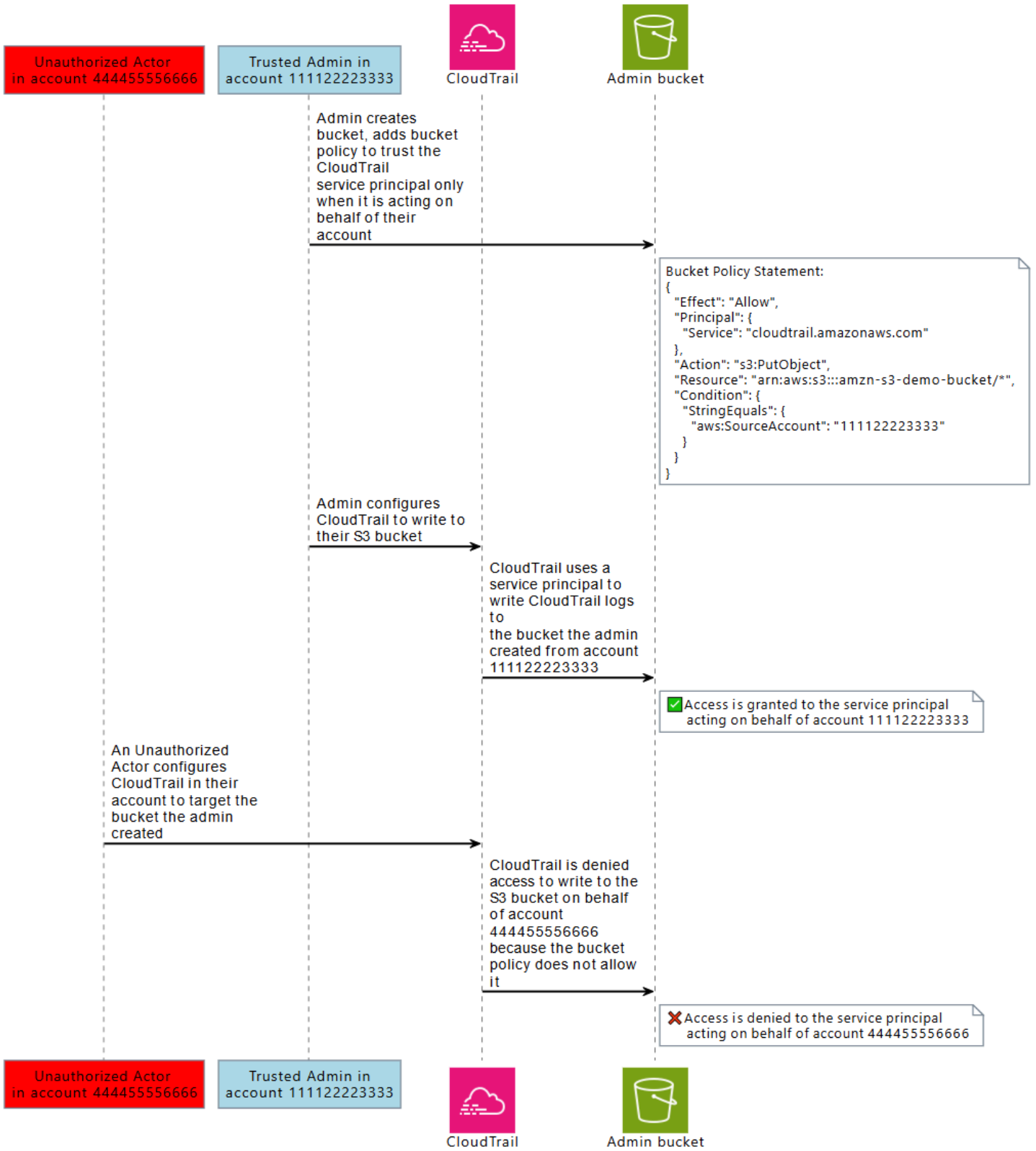
為了協助防止未經授權的演員使用委託人的信任 AWS 來存取您的資源， AWS 服務委託人包含 AWS 資源 AWS 帳戶及其代表 AWS 的組織的相關資訊。

此資訊可用於可在資源政策中使用的全域條件索引鍵值，或用於 AWS 服務主體所提出請求的資源控制政策。建議您在資源政策aws:SourceOrgPaths中使用 aws:SourceArn、aws:SourceOrgID、aws:SourceAccount或 ，只要 AWS 服務主體獲得存取其中一個資源的許可。這些條件金鑰可讓您在資

源政策或資源控制政策中進行測試，讓存取資源 AWS 的主體代表 AWS 資源執行此操作 AWS 帳戶，或您 AWS Organizations 預期執行此操作。

- 使用 `aws:SourceArn` 允許 AWS 服務主體代表特定資源存取您的資源，例如特定 AWS CloudTrail 線索或 AppStream 機群。
- 使用 `aws:SourceAccount` 允許 AWS 服務主體代表特定 存取您的 資源 AWS 帳戶。
- 使用 `aws:SourceOrgID` 允許 AWS 服務主體代表特定 存取您的 資源 AWS Organizations。
- 使用 允許`aws:SourceOrgPaths` AWS 服務主體代表特定 AWS Organizations 路徑存取您的資源。

下圖示範當資源使用`aws:SourceAccount`全域條件內容索引鍵設定時，跨服務混淆代理人案例，而來自另一個帳戶的未經授權演員嘗試存取他們不打算存取 AWS 的資源。


Unauthorized Actor in account 444455556666
Trusted Admin in account 111122223333
CloudTrail
Admin bucket
Admin creates bucket, adds bucket policy to trust the CloudTrail service principal only when it is acting on behalf of their account
Bucket Policy Statement:
{
"Effect": "Allow",
"Principal": {
"Service": "cloudtrail.amazonaws.com"
},
"Action": "s3:PutObject",
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
"Condition": {
"StringEquals": {
"aws:SourceAccount": "111122223333"
}
}
}
Admin configures CloudTrail to write to their S3 bucket
CloudTrail uses a service principal to write CloudTrail logs to the bucket the admin created from account 111122223333
Access is granted to the service principal acting on behalf of account 111122223333
An Unauthorized Actor configures CloudTrail in their account to target the bucket the admin created
CloudTrail is denied access to write to the S3 bucket on behalf of account 444455556666 because the bucket policy does not allow it
Access is denied to the service principal acting on behalf of account 444455556666
Unauthorized Actor in account 444455556666
Trusted Admin in account 111122223333
CloudTrail
Admin bucket

在政策中使用 aws:SourceArn、aws:SourceAccount、aws:SourceOrgID 和 aws:SourceOrgPaths 全域條件索引鍵，可協助您確保服務主體是代表您存取您的資源。建議您在將其中一個資源的存取權授予 AWS 服務委託人時，使用這些條件金鑰。

> **Note**
>
> 有些 AWS 服務 互動有其他控制項，可協助防止測試使用者存取資源的跨服務混淆代理人問題。例如，當 KMS 金鑰授予授予給 時 AWS 服務， AWS KMS 會使用與資源相關聯的加密內容，以及金鑰授予，以協助防範跨服務混淆代理人問題。
> 如需有關可協助避免跨服務混淆代理人風險的服務特定機制，以及 aws:SourceArn、aws:SourceAccount、aws:SourceOrgID 和 aws:SourceOrgPaths 是否受支援的詳細資訊，請參閱您所用服務的相關文件。

跨服務混淆代理人保護與資源型政策搭配使用

下列範例政策僅在服務主體代表 AWS 帳戶 111122223333 時，授予服務主體對 Amazon S3 儲存貯體 arn：aws：s3：：amzn-s3-demo-bucket1 的cloudtrail.amazonaws.com存取權。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CloudTrailAclCheck",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailWrite",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
```

```
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/[optionalPrefix]/Logs/
myAccountID/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

此範例儲存貯體政策僅在服務主體 appstream.amazonaws.com 代表指定的 Amazon AppStream 機群執行動作 (透過以 aws:SourceArn 來指定機群 ARN) 時，才授予服務主體對 s3://amzn-s3-demo-bucket2 內的 powershell 指令碼 examplefile.psh 的存取權。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "appstream.amazonaws.com"
                ]
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket2/examplefile.psh",
            "Condition": {
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:appstream:us-
east-1:111122223333:fleet/ExampleFleetName"
                }
            }
        }
    ]
}
```

跨服務混淆代理人保護與資源控制政策搭配使用

您可以使用資源控制政策 (RCP)，將跨服務混淆代理人控制套用至支援的 AWS 服務資源。RCP 讓您可以在資源上集中套用跨服務混淆代理人控制。您可以使用條件金鑰，例如 `aws:SourceOrgId`和 `aws:SourceOrgPaths` 搭配連接至您 AWS Organizations、組織單位 (OU) 或 AWS 帳戶 組織中 RCPs，而無需將陳述式新增至特定資源型政策。如需有關 RCP 和支援服務的詳細資訊，請參閱 AWS Organizations User Guide 中的 Resource control policies (RCPs)。

當 `aws:SourceOrgID` 不等於 o-ExampleOrg 時，下列範例 RCP 會拒絕 AWS 服務主體存取成員帳戶中的 Amazon S3 儲存貯體。對應的允許必須存在於 S3 儲存貯體的資源型政策中，以允許`SourceOrgID`等於 o-ExampleOrg 的 AWS 服務 主體。

此政策僅對具有 `aws:SourceAccount` 索引鍵 (`"Null": {"aws:SourceAccount": "false"}`) 的服務主體 (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) 發出的請求套用控制項，因此不需要使用條件索引鍵的服務整合與您主體的呼叫不會受到影響。如果請求內容中存在 `aws:SourceAccount` 條件索引鍵，則 Null 條件將評估為 true，從而導致強制執行 `aws:SourceOrgID`。我們在 Null 條件運算子中使用 `aws:SourceAccount` 而非 `aws:SourceOrgID`，因此如果請求來自不屬於組織的帳戶，則控制項仍然適用。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtectionForS3",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-ExampleOrg"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
```

```
        }
      }
    }
  ]
}
```

IAM 角色的常見案例

與大多數 AWS 功能一樣，您通常有兩種方式可以使用角色：在 IAM 主控台中以互動方式使用，或以程式設計方式使用 AWS CLI、Tools for Windows PowerShell 或 API。

- 使用 IAM 主控台的帳戶中的 IAM 使用者可以切換到角色，以臨時使用主控台中角色的許可。使用者放棄其原始許可並取得指派給該角色的許可。當使用者退出角色時，將恢復其原始許可。
- AWS （例如 Amazon EC2) 提供的應用程式或服務可以透過請求角色的臨時安全登入資料來擔任角色，以便向其發出程式設計請求 AWS。您以這種方式使用角色，這樣您就不必為需要存取資源的每個實體分享或維護長期安全憑證 (例如，透過建立 IAM 使用者)。

> **Note**
>
> 本指南互換使用切換到角色和擔任角色字詞。

使用角色的最簡單方法是授予 IAM 使用者切換到您在自己或另一個 AWS 帳戶中建立之角色的許可。他們可以使用 IAM 主控台輕鬆切換角色，以使用您通常不希望他們擁有的許可，然後退出角色以放棄這些許可。這有助於防止意外存取或修改敏感資源。

如需角色的更複雜用途，例如授予存取應用程式和服務，或聯合身分外部使用者，您可以呼叫 `AssumeRole` API。這個 API 呼叫會傳回一組臨時憑證，應用程式可以在後續 API 呼叫中使用這些憑證。嘗試使用臨時憑證的動作只能透過相關的角色授予許可。應用程式不需要像主控台中的使用者般「退出」角色；相反，應用程式只是停止使用臨時憑證並繼續使用原始憑證進行呼叫。

聯合身分使用者使用來自身分提供者 (IdP) 的登入資料來登入。 AWS 然後， 會提供暫時登入資料給信任的 IdP，以傳遞給使用者，以便在後續 AWS 資源請求中包含 。這些憑證提供授予指定角色的許可。

本節概述以下案例：

- 為您擁有的 IAM 使用者提供存取權 AWS 帳戶 ，以存取您擁有的另一個帳戶中的資源
- 提供對非 AWS 工作負載的存取權

- 將存取權提供給第三方擁有之 AWS 帳戶 中的 IAM 使用者
- 為 AWSAWS 資源提供的服務提供存取權
- 將存取權提供給外部驗證使用者 (聯合身分)

在您擁有的另一個 IAM 使用者中 AWS 帳戶 存取

您可以授予 IAM 使用者許可，以切換到 內的角色， AWS 帳戶 或切換到 AWS 帳戶 您擁有的其他 中定義的角色。

> **Note**
>
> 如果要授予對您未擁有或無法控制的帳戶的存取許可，請參閱本主題後面的 存取第三方 AWS 帳戶 擁有的。

假設您擁有一個對組織來說至關重要的 Amazon EC2 執行個體。您可以使用這些許可來建立角色，而非直接授予使用者終止執行個體的許可。然後，允許管理員可以在需要終止執行個體時切換為該角色。這麼做，可為執行個體加入以下幾層保護：

- 您必須向使用者明確授予擔任該角色的許可。
- 您的使用者必須使用 主動切換到角色， AWS 管理主控台 或使用 AWS CLI 或 AWS API 擔任角色。
- 您可以為角色加入多重要素驗證 (MFA) 保護，僅限登入 MFA 裝置的使用者才能擔任該角色。若要了解如何配置角色以使擔任角色的使用者必須先使用多重要素驗證 (MFA) 進行身分驗證，請參閱 透過 MFA 實現安全的 API 存取。

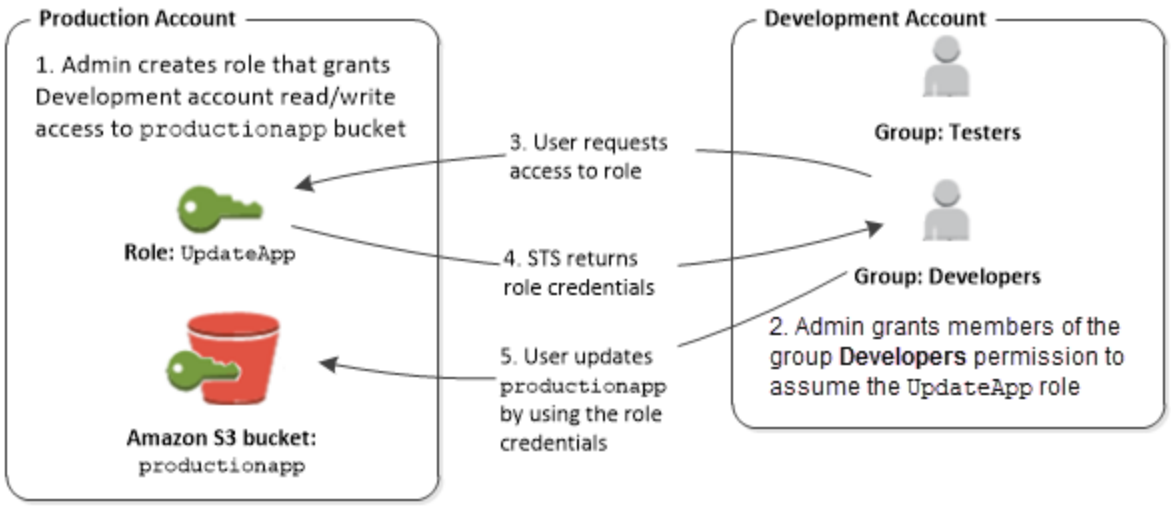
我們建議使用此方法強制實施最低權限。也就是僅限於特定任務需要時，才能使用升級的許可。透過角色，您可以協助防止對敏感環境的意外變更，特別是當您將它們與稽核結合時，以協助確保僅在需要時才使用角色。

在您出於此目的建立角色時，可在該角色的信任政策的 `Principal` 元素中依照 ID 指定其使用者需要存取許可的帳戶。隨後可以向這些其他帳戶中的特定使用者授予切換到角色的許可。若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？。

一個帳戶中的使用者可以切換為相同或不同帳戶中的角色。使用角色過程中，使用者只能執行角色允許的操作並且只能存取角色允許的資源；其原始使用者許可處於暫停狀態。使用者退出角色時，恢復原始使用者許可。

使用不同的開發和生產帳戶的範例方案

假設您的組織有多個 AWS 帳戶 來隔離開發環境與生產環境。開發帳戶中的使用者有時可能需要存取生產帳戶中的資源。例如在將更新從開發環境推廣到生產環境時，可能就需要跨帳戶存取許可。儘管您可以為在兩個帳戶中工作的使用者建立單獨的身分 (和密碼)，多個帳戶的憑證管理還是會為身分管理帶來難題。在以下圖表中，所有使用者都透過開發帳戶進行管理，但部分開發人員需要對生產帳戶進行有限存取。開發帳戶有兩個群組：測試人員和開發人員，每個群組有其專屬的政策。



1. 生產帳戶中的一名管理員使用 IAM 在該帳戶中建立 UpdateApp 角色。在角色中，管理員定義信任政策，該政策將開發帳戶指定為 Principal，這表示開發帳戶中的授權使用者可以使用 UpdateApp 角色。管理員也可以為角色定義許可政策，該政策指定名為 productionapp 之 Amazon S3 儲存貯體的讀取和寫入許可。

   然後，管理員將與需要擔任角色的任何人共用該角色的帳號和名稱。該資訊是角色的帳號和名稱（適用於 AWS 主控台使用者）或 Amazon Resource Name (ARN)（適用於 AWS CLI 或 AWS API 存取）。角色 ARN 類似於 arn:aws:iam::123456789012:role/UpdateApp，其中角色名為 UpdateApp，而且角色使用帳戶號碼 123456789012 所建立。

   > **Note**
   >
   > 管理員可以選擇是否配置角色，以便擔任角色的使用者必須先使用多重要素驗證 (MFA) 進行身分驗證。如需詳細資訊，請參閱 透過 MFA 實現安全的 API 存取。

2. 在開發帳戶中，管理員向開發人員群組的成員授予切換為角色的許可。這是透過授予開發人員群組許可來呼叫UpdateApp角色的 AWS Security Token Service (AWS STS) AssumeRole API 來完

成。開發帳戶中的開發人員群組的所有 IAM 使用者現在都可以切換為生產帳戶中的 UpdateApp 角色。不在開發人員群組中的其他使用者無權切換為該角色，因此無法存取生產帳戶中的 S3 儲存貯體。

3. 使用者請求切換到角色：
   - AWS 主控台：使用者選擇導覽列上的帳戶名稱，然後選擇切換角色。使用者指定帳戶 ID (或別名) 和角色名稱。或者，使用者可以選擇管理員透過電子郵件傳送的連結。透過該連結，使用者可以前往已填寫詳細資訊的 Switch Role (切換角色) 頁面。
   - AWS API/AWS CLI：開發帳戶開發人員群組中的使用者呼叫 AssumeRole函數以取得UpdateApp角色的登入資料。使用者將 UpdateApp 角色的 ARN 指定為呼叫的一部分。如果測試人員群組中的使用者發出相同請求，請求將失敗，因為測試人員沒有針對 AssumeRole 角色 ARN 呼叫 UpdateApp 的許可。
4. AWS STS 傳回臨時登入資料：
   - AWS console：使用角色的信任政策 AWS STS 驗證請求，以確保請求來自信任的實體 （即開發帳戶）。驗證後， 會將 AWS 臨時安全登入資料 AWS STS 傳回至 主控台。
   - API/CLI：根據角色的信任政策 AWS STS 驗證請求，以確保請求來自信任的實體 （即開發帳戶）。驗證後， 會 AWS STS 傳回暫時安全登入資料給應用程式。
5. 暫時登入資料允許存取 AWS 資源：
   - AWS 主控台： AWS 主控台會代表使用者使用暫時登入資料進行所有後續主控台動作，在此情況下會讀取和寫入儲存productionapp貯體。主控台無法存取生產帳戶中的任何其他資源。使用者退出角色時，使用者的許可恢復為切換為角色之前所擁有的原始許可。
   - API/CLI：應用程式使用臨時安全性憑證更新 productionapp 儲存貯體。應用程式只能使用臨時安全性憑證讀取和寫入 productionapp 儲存貯體，無法存取生產帳戶的任何其他資源。應用程式不必退出角色，只需在後續 API 呼叫中停止使用臨時憑證並使用原始憑證。

其他資源

如需詳細資訊，請參閱下列內容：

- IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權

非AWS 工作負載的存取

IAM 角色是 AWS Identity and Access Management (IAM) 中獲指派許可的物件。當您使用來自 外部的 IAM 身分或身分擔任該角色時 AWS，它會為您的角色工作階段提供暫時安全登入資料。您可能在資料中心或 外部的其他基礎設施中執行工作負載 AWS ，這些工作負載必須存取您的 AWS 資源。您可以使

用 AWS Identity and Access Management Roles Anywhere (IAM Roles Anywhere) 驗證非AWS 工作負載，而不是建立、分發和管理長期存取金鑰。IAM Roles Anywhere 使用來自憑證授權單位 (CA) 的 X.509 憑證來驗證身分 AWS 服務 ，並使用 IAM 角色提供的臨時憑證安全地提供對 的存取。

若要使用 IAM Roles Anywhere

1. 使用 AWS 私有憑證授權單位設定 CA，或者使用您自己的 PKI 基礎設施中的 CA。
2. 設定 CA 之後，在 IAM Roles Anywhere 中建立稱為信任錨的物件。此錨點在 IAM Roles Anywhere 和 CA 之間建立信任，以進行身分驗證。
3. 然後您可以設定現有的 IAM 角色，或建立信任 IAM Roles Anywhere 服務的新角色。
4. 使用信任錨點，透過 IAM Roles Anywhere 驗證您的非AWS 工作負載。 會將非AWS 工作負載臨時憑證 AWS 授予可存取您 AWS 資源的 IAM 角色。

其他資源

下列資源可協助您進一步了解如何提供對非AWS 工作負載的存取權。

- 如需設定 IAM Roles Anywhere 的詳細資訊，請參閱《IAM Roles Anywhere 使用者指南》中的 What is AWS Identity and Access Management Roles Anywhere (什麼是 IAM Roles Anywhere)。
- 若要了解如何為 IAM Roles Anywhere 設定公有金鑰基礎設施 (PKI)，請參閱 AWS 安全部落格中的 IAM Roles Anywhere with an external certificate authority。

存取第三方 AWS 帳戶 擁有的

當第三方需要存取您組織 AWS 的資源時，您可以使用 角色來委派存取權給他們。例如，第三方可能提供一種用於管理您的 AWS 資源之服務。透過 IAM 角色，您可以授予這些第三方存取 資源 AWS 的權限，而無需共用您的 AWS 安全登入資料。相反地，第三方可以透過擔任您在 中建立的角色來存取您的 AWS 資源 AWS 帳戶。若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？。

為了建立他們可以代入的角色，第三方必須為您提供以下資訊：

- 第三方的 AWS 帳戶 ID。為角色定義信任政策時，可將其 AWS 帳戶 ID 指定為主體。
- 與角色唯一關聯的外部 ID。外部 ID 可以是只有您和第三方知道的任何識別碼。例如，您可以使用您與該第三方之間的發票 ID，但不要使用能被猜到的內容，例如第三方的電話號碼。為角色定義信任政策時，必須指定該 ID。第三方在代入角色時必須提供該 ID。

- 第三方為使用您的 AWS 資源而請求獲得的許可。定義角色的許可政策時，必須指定這些許可。這個政策定義了他們可以執行哪些操作以及可以存取哪些資源。

建立完角色後，您必須向第三方提供該角色的 Amazon Resource Name (ARN)。他們需要使用您的角色的 ARN 來代入該角色。

> **Important**
>
> 當您授予第三方存取 資源的權限時 AWS ，他們可以存取您在政策中指定的任何資源。他們使用的資源費用將由您支付。請確保適當地限制他們對資源的使用。

第三方存取的外部 ID

外部 ID 允許正擔任該角色的使用者聲明所操作的環境。它還為帳戶擁有者提供一種方法來允許僅在特定情況下擔任該角色。外部 ID 的主要功能是解決並防止 混淆代理人問題。

> **Important**
>
> AWS 不會將外部 ID 視為秘密。在您建立存取金鑰對或密碼等秘密之後 AWS，就無法再次檢視它們。具有檢視角色許可的任何人都可看見角色的外部 ID。

我何時應使用外部 ID？

在以下情況下使用外部 ID：

- 您是 AWS 帳戶 擁有者，而且已為存取 AWS 帳戶 您 以外的其他 的第三方設定角色。您應要求第三方提供其在擔任您的角色時包含的外部 ID。然後，在您角色的信任政策中檢查該外部 ID。這樣做可確保外部方僅在代表您執行操作時才能擔任您的角色。
- 在前述情況下，您代表不同客戶 (如 Example Corp) 擔任角色。您應該為每個客戶分配一個唯一的外部 ID 並指導他們將該外部 ID 加入到其角色的信任政策。然後，您必須確保在代入角色的請求中始終包含正確的外部 ID。

  您可能已為您的每個客戶提供一個獨有識別碼，而且此獨有 ID 足以用作外部 ID。該外部 ID 不是您要明確建立或分別追蹤所需的特殊值 (僅用於此目的)。

  您應始終在您的 `AssumeRole` API 呼叫中指定外部 ID。此外，在客戶為您提供角色 ARN 時，請測試是否能在含有/不含有正確外部 ID 的情況下擔任該角色。如果可在沒有正確外部 ID 的情況下擔任

角色，則請勿在您的系統中儲存該客戶的角色 ARN。等待該客戶將角色信任政策更新為要求提供正確的外部 ID。這樣一來，您協助您的客戶執行了正確的操作，並幫助您和客戶避免了混淆代理人問題。

使用外部 ID 的範例案例

例如，假設您決定聘請名為 Example Corp 的第三方公司來監控您的 AWS 帳戶 並協助最佳化成本。為了追蹤您的每日花費，Example Corp 需要存取您的 AWS 資源。Example Corp 也可監控其他客戶的許多其他 AWS 帳戶。

請不要向 Example Corp 提供對您 AWS 帳戶中的 IAM 使用者和其長期憑證的存取權。請改用 IAM 角色及其臨時安全性憑證。IAM 角色提供一種機制，允許第三方存取您的 AWS 資源，而不需要共用長期憑證（例如 IAM 使用者存取金鑰）。

您可以使用 IAM 角色，在您的 AWS 帳戶 與 Example Corp 帳戶之間建立信任關係。建立此關係後，Example Corp 帳戶的成員可以呼叫 AWS Security Token Service AssumeRole API 來取得臨時安全登入資料。Example Corp 成員接著可以使用登入資料來存取您帳戶中 AWS 的資源。

> **Note**
>
> 如需 AssumeRole 和其他您可以呼叫以取得臨時安全登入資料之 AWS API 操作的詳細資訊，請參閱 比較 AWS STS 登入資料。

以下是此方案的更多詳細資訊：

1. 您聘請了 Example Corp，此公司將為您建立獨有的客戶識別碼。他們為您提供這個唯一的客戶 ID 及其 AWS 帳戶 號碼。您需要此資訊來在下一個步驟中建立 IAM 角色。

   > **Note**
   >
   > Example Corp 可使用其想用於 ExternalId 的任何字串值，只要該值對於每個客戶來說都是獨有的。該值可以是客戶帳戶，甚至可以是一個隨機字串，只要沒有兩個客戶擁有相同的值即可。該值不是「機密」。Example Corp 必須為每個客戶提供 ExternalId 值。關鍵在於，該值必須由 Example Corp 而非由其客戶產生，以確保每個外部 ID 都是唯一的。

2. 您登入 AWS 並建立 IAM 角色，讓 Example Corp 存取您的 資源。與任何 IAM 角色類似，該角色具有兩個政策：許可政策和信任政策。角色的信任政策指定擔任該角色的對象。在我們的範例案例中，政策會將 Example Corp AWS 帳戶 的數量指定為 `Principal`。這允許來自此帳戶的身分擔任

該角色。此外，您新增 Condition 元素到信任政策。此 Condition 測試 ExternalId 內容索引鍵，以確保它與 Example Corp 的獨有客戶 ID 一致。例如：

```
   "Principal": {"AWS": "Example Corp's AWS ## ID"},
   "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. 該角色的許可政策指定該角色允許某個人執行哪些操作。例如，您可以指定該角色允許某人只能管理您的 Amazon EC2 和 Amazon RDS 資源，但不能管理您的 IAM 使用者或群組。在我們的範例方案中，您使用許可政策為 Example Corp 授予帳戶中的所有資源的唯讀存取許可。
4. 建立完角色後，為 Example Corp 提供該角色的 Amazon Resource Name (ARN) (ARN)。
5. 當 Example Corp 需要存取您的 AWS 資源時，公司的某人會 AWS `sts:AssumeRole`呼叫 API。此呼叫包括要擔任的角色的 ARN 和與其客戶 ID 對應的 ExternalId 參數。

如果請求來自使用 Example Corp 的某人 AWS 帳戶，且角色 ARN 和外部 ID 正確，則請求會成功。然後，它會提供臨時安全登入資料，讓 Example Corp 可用來存取您的角色允許 AWS 的資源。

換言之，當角色政策包括外部 ID 時，任何需要擔任該角色的人都必須是該角色中的主體，還必須包括正確的外部 ID。

外部 ID 的要點

- 在多租戶環境中，您支援具有不同 AWS 帳戶的多個客戶，我們建議每個客戶使用一個外部 ID AWS 帳戶。此 ID 應該是由第三方產生的隨機字串。
- 如要請求第三方在取得角色時提供外部 ID，請使用您選擇的外部 ID 來更新角色的信任政策。
- 若要在擔任角色時提供外部 ID，請使用 AWS CLI 或 AWS API 擔任該角色。如需詳細資訊，請參閱 STS AssumeRole API 操作，或是 STS assume-role CLI 操作。
- `ExternalId` 值必須最少為 2 個字元，最多為 1,224 個字元。該值必須為英數字元，且不包含空格。也可以包含下列符號：加號 (+)、等號 (=)、逗號 (,)、句號 (.)、小老鼠 (@)、冒號 (:)、正斜線 (/) 和連字號 (-)。

其他資源

下列資源可協助您進一步了解如何提供對第三方擁有的 AWS 帳戶 的存取權。

- 若要了解如何允許其他人在您的 中執行動作 AWS 帳戶，請參閱 使用自訂信任政策建立角色。
- 若要了解如何授予許可以切換至角色，請參閱向使用者授予切換角色的許可

- 若要了解如何建立和提供信任使用者臨時安全登入資料，請參閱 臨時安全憑證的許可。

存取 AWS 服務

許多 AWS 服務要求您使用 角色來控制該服務可存取的內容。服務會擔任代您執行動作的角色稱為服務角色。當角色做為服務的專業用途時，它可以歸類為服務連結角色。請參閱每個服務的 AWS 文件，以查看它是否使用角色，以及了解如何指派角色以供服務使用。

如需建立角色以委派存取 提供的服務的詳細資訊 AWS，請參閱 建立角色以將許可委派給 AWS 服務。

對外部驗證的使用者的存取權 (聯合身分)

您的使用者可能已經在 外部擁有身分 AWS，例如在您的公司目錄中。如果這些使用者需要使用 AWS 資源 （或使用存取這些資源的應用程式），則這些使用者也需要 AWS 安全登入資料。您可以使用 IAM 角色為從您的組織或第三方身分提供者 (IdP) 聯合身分的使用者指定許可。

> **Note**
>
> 作為安全最佳實務，建議您使用聯合身分在 IAM Identity Center 中管理使用者存取權，而不是建立 IAM 使用者。若要了解需要 IAM 使用者的特定情形，請參閱建立 IAM 使用者 (而非角色) 的時機。

IAM Identity Center 許可設定 IAM Identity Center 管理的佈建 IAM 角色範本。帳戶存取管理員 — IAM 功能 — 可讓您將自己的自訂 IAM 角色指派給 IAM Identity Center 使用者和群組，啟用自訂信任政策、角色標籤和可設定的角色路徑等功能。您可以將它與許可集一起使用或單獨使用。

使用 Amazon Cognito 聯合行動或以 Web 為基礎的應用程式的使用者

如果您建立可存取 AWS 資源的行動或 Web 應用程式，則應用程式需要安全登入資料，才能向 發出程式設計請求 AWS。對於大多數行動應用程式藍本，建議您使用 Amazon Cognito。您可以將此服務與適用於 AWS iOS 的 Mobile SDK 和AWS 適用於 Android 和 Fire OS 的 Mobile SDK 搭配使用，為使用者建立唯一身分，並對其進行驗證，以安全地存取您的 AWS 資源。Amazon Cognito 支援與下一個部分中列出的身分提供者相同的身分提供者，並且還支援開發人員驗證身分和未經身分驗證 (訪客) 的存取。Amazon Cognito 還提供 API 操作以同步使用者資料，如此可在使用者於不同裝置之間移動時進行保留。如需詳細資訊，請參閱 用於行動應用程式的 Amazon Cognito。

使用公有身分服務提供者或 OpenID Connect 來聯合使用者

在可能的情況下，將 Amazon Cognito 用於行動和以 Web 為基礎的應用程式案例。Amazon Cognito 為您提供公有身分提供者服務的大部分幕後工作。它適用於相同的第三方服務，並且還支援匿名登入。但是，對於更進階的案例，您可以直接使用 Login with Amazon、Facebook、Google 或與 OpenID Connect (OIDC) 相容的任何 IdP 等第三方服務。如需有關使用 OIDC 聯合身分來使用其中一項服務的詳細資訊，請參閱 OIDC 聯合身分。

使用 SAML 2.0 聯合使用者

如果您的組織已使用支援 SAML 2.0（安全性聲明標記語言 2.0) 的身分提供者軟體套件，您可以在組織之間建立信任做為身分提供者 (IdP) 和 AWS 做為服務提供者。然後，您可以使用 SAML 為使用者提供 AWS 管理主控台 或 聯合身分的單一登入 (SSO)，以呼叫 AWS API 操作。例如，如果您的公司使用 Microsoft Active Directory 和 Active Directory Federation Services，則可以使用 SAML 2.0 聯合。如需有關使用 SAML 2.0 聯合使用者的詳細資訊，請參閱SAML 2.0 聯合身分。

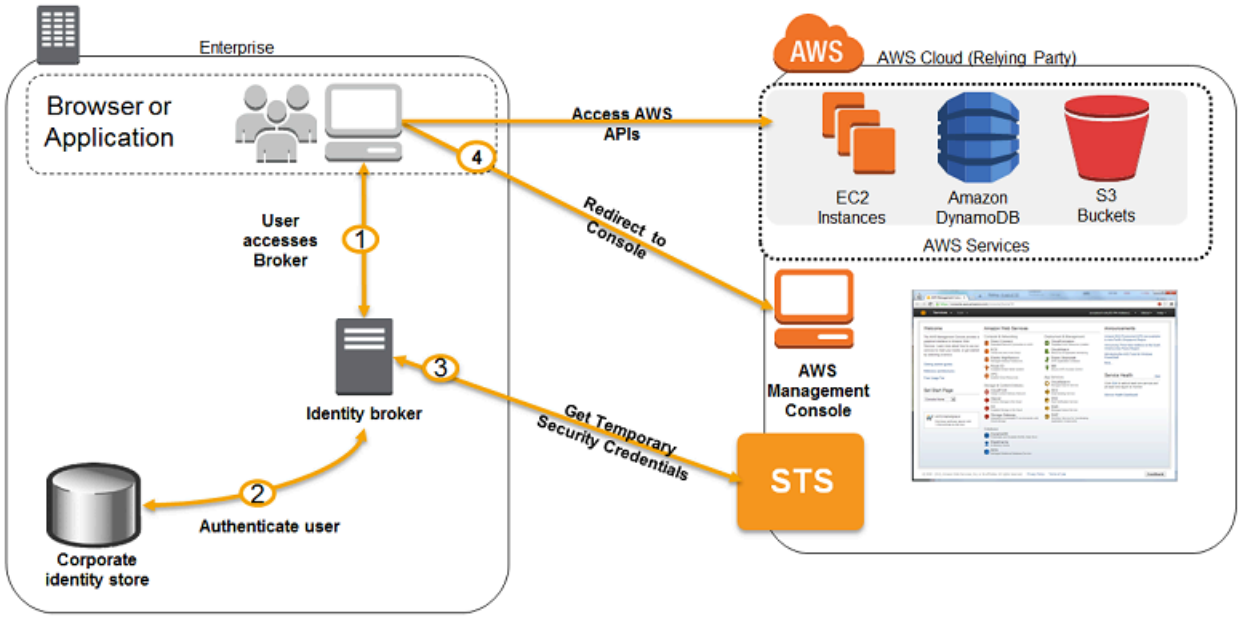
透過建立自訂身分經紀人應用程式來聯合使用者

如果您的身分存放區與 SAML 2.0 不相容，則可以建置自訂身分經紀人應用程式以執行類似的功能。代理程式應用程式會驗證使用者、請求使用者的臨時登入資料 AWS，然後將他們提供給使用者以存取 AWS 資源。

例如，Example Corp. 有許多員工需要執行存取公司 AWS 資源的內部應用程式。員工已經在公司身分和身分驗證系統中擁有身分，而 Example Corp. 不想要為每個公司員工建立個別的 IAM 使用者。

Bob 是 Example Corp. 的開發人員。 為了讓 Example Corp. 內部應用程式能夠存取公司的 AWS 資源，Bob 開發了自訂身分代理程式應用程式。該應用程式驗證員工是否已登入到現有的 Example Corp. 身分和身分驗證系統，該系統可能使用 LDAP、Active Directory 或其他系統。然後，身分經紀人應用程式取得員工的臨時安全憑證。此案例類似於前一個案例（使用自訂身分驗證系統的行動應用程式），但需要存取 AWS 資源的應用程式全都在公司網路中執行，而且公司有現有的身分驗證系統。

若要取得臨時安全憑證，身分經紀人應用程式將呼叫 `AssumeRole` 或 `GetFederationToken` 以取得臨時安全憑證，具體取決於 Bob 想要如何管理使用者政策以及臨時憑證何時過期。(如需有關這些 API 操作間差異的詳細資訊，請參閱 IAM 中的暫時安全憑證 和 臨時安全憑證的許可)。呼叫會傳回暫時性安全登入資料，其中包含 AWS 存取金鑰 ID、私密存取金鑰和工作階段字符。身分經紀人應用程式讓這些臨時安全憑證可供內部公司應用程式使用。然後，應用程式可以使用臨時憑證直接呼叫 AWS 。該應用程式快取憑證，直到過期，然後請求一組新的臨時憑證。下圖說明此情況。



此案例具有以下屬性：

- 身分經紀人應用程式有權存取 IAM 的權杖服務 (STS) API 來建立臨時安全憑證。
- 身分經紀人應用程式能夠驗證員工是否在現有身分驗證系統中進行了身分驗證。
- 使用者可以取得臨時 URL，讓他們能夠存取 AWS 管理主控台 （稱為單一登入）。

如需建立臨時安全性憑證檔案的詳細資訊，請參閱 比較 AWS STS 登入資料。如需 SAML 聯合身分主體存取 AWS 管理主控台的詳細資訊，請參閱 啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。

IAM 角色建立

若要建立角色，您可以使用 AWS 管理主控台、 AWS CLI、Tools for Windows PowerShell 或 IAM API。

如果您使用 AWS 管理主控台，精靈會引導您完成建立角色的步驟。精靈的步驟略有不同，具體取決於您是為 AWS 服務、為 建立角色 AWS 帳戶，還是為 SAML 或 OIDC 聯合委託人建立角色。

IAM 使用者的角色

建立此角色，將 或您擁有之其他 中定義的角色的許可委派 AWS 帳戶 給 AWS 帳戶 。一個帳戶中的使用者可以切換為相同或不同帳戶中的角色。使用角色過程中，使用者只能執行角色允許的操作並且只能存取角色允許的資源；其原始使用者許可處於暫停狀態。使用者退出角色時，恢復原始使用者許可。

如需詳細資訊，請參閱建立角色以將許可授予 IAM 使用者。

如需有關建立跨帳戶存取權的角色的詳細資訊，請參閱使用自訂信任政策建立角色。

AWS 服務的角色

建立此角色，以將許可委派給可以代表您執行動作的服務。您傳遞給服務的服務角色必須具有 IAM 政策，其許可允許服務執行與該服務關聯的動作。每個 AWS 服務需要不同的許可。

如需有關建立服務角色的詳細資訊，請參閱建立角色以將許可委派給 AWS 服務。

如需有關建立服務連結角色的詳細資訊，請參閱建立服務連結角色。

聯合身分的角色

建立此角色，以將許可委派給已在 AWS外部擁有身分的使用者。當您使用 身分提供者時，您不需要建立自訂登入代碼或管理自己的使用者身分，IdP 會為您處理這些工作。您的外部使用者透過 IdP 登入，您可以授予這些外部身分許可，以使用您帳戶中 AWS 的資源。身分提供者可協助保護 AWS 您的帳戶安全，因為您不必在應用程式中分發或嵌入長期安全登入資料，例如存取金鑰。

如需詳細資訊，請參閱為第三方身分提供者建立角色。

建立角色以將許可授予 IAM 使用者

您可以使用 IAM 角色來提供 AWS 資源的存取權。透過 IAM 角色，您可以在信任帳戶和其他 AWS 信任帳戶之間建立信任關係。信任帳戶擁有要存取的資源，而受信任帳戶包含需要存取資源的使用者。不過，可以讓另一個帳戶在您的帳戶中擁有資源。例如，信任的帳戶可能允許信任的帳戶來建立新的資源，例如在 Amazon S3 儲存貯體中建立新物件。在這種情況下，建立資源的帳戶擁有資源，並控制誰可以存取該資源。

建立信任關係後，來自信任帳戶的 IAM 使用者或應用程式可以使用 AWS Security Token Service (AWS STS) `AssumeRole` API 操作。此操作提供臨時安全登入資料，可讓您存取帳戶中 AWS 的資源。

兩個帳戶可都由您控制，或者含有使用者的帳戶可由第三方進行控制。如果使用者的其他帳戶是您無法控制的 AWS 帳戶 ，則您可以使用 `externalId` 屬性。外部 ID 可以是您和第三方帳戶管理員之間商定的任何文字或數字。此選項會自動新增條件到信任政策，讓使用者只在請求包含正確的 `sts:ExternalID` 時才擔任該角色。如需詳細資訊，請參閱存取第三方 AWS 帳戶 擁有的。

如需有關如何使用角色委派許可的資訊，請參閱角色術語和概念。如需使用服務角色以允許服務存取您帳戶中的資源的資訊，請參閱建立角色以將許可委派給 AWS 服務。

建立 IAM 角色 (主控台)

您可以使用 AWS 管理主控台 來建立 IAM 使用者可以擔任的角色。例如，假設您的組織有多個 AWS 帳戶 來隔離開發環境與生產環境。如需有關建立可讓開發帳戶中的使用者存取生產帳戶中資源的角色的高級別資訊，請參閱使用不同的開發和生產帳戶的範例方案。

最低許可

若要執行下列步驟，您至少必須擁有下列 IAM 許可：

- `access-analyzer:ValidatePolicy`
- `iam:AttachRolePolicy`
- `iam:CreatePolicy`
- `iam:CreateRole`
- `iam:GetAccountSummary`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:ListAccountAliases`
- `iam:ListAttachedRolePolicies`
- `iam:ListOpenIDConnectProviders`
- `iam:ListPolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`
- `iam:ListRoleTags`
- `iam:ListSAMLProviders`

Console

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在主控台的導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 選擇 AWS 帳戶 角色類型。

4. 若要為您的帳戶建立角色，請選取 This account (此帳號)。若要為其他帳戶建立角色，請選擇 Another AWS 帳戶 (另一個)，然後輸入您要對其授予資源存取權的 Account ID (帳戶 ID)。

   指定帳戶的管理員可以授予許可給該帳戶中的任何 IAM 使用者來擔任此角色。若要執行此操作，管理員要將政策連接到授予 `sts:AssumeRole` 動作之許可的使用者或群組。該政策必須指定角色的 ARN 為 `Resource`。

5. 如果您將許可授予您不控制之帳戶的使用者，而且使用者將以程式設計方式擔任此角色，則請選取 Require external ID (需要外部 ID)。外部 ID 可以是您和第三方帳戶管理員之間商定的任何文字或數字。此選項會自動新增條件到信任政策，讓使用者只在請求包含正確的 `sts:ExternalID` 時才擔任該角色。如需詳細資訊，請參閱存取第三方 AWS 帳戶 擁有的。

   > **Important**
   >
   > 選擇此選項只會透過 AWS CLI、Tools for Windows PowerShell 或 AWS API 來限制對角色的存取。這是因為您無法使用 AWS 主控台切換到在其信任政策中具有 `externalId`條件的角色。不過，您可以程式設計方式建立存取這類存取，即透過編寫指令碼或使用相關開發套件的應用程式。如需詳細資訊和範例指令碼，請參閱 AWS 安全部落格中的如何啟用跨帳戶存取 AWS 管理主控台。

6. 如果您想限制角色為使用多重要素驗證 (MFA) 登入的使用者，請選取 Require MFA (需要 MFA)。這會新增條件到角色的信任政策，以檢查 MFA 登入。想要擔任該角色的使用者必須從設定的 MFA 裝置使用臨時的一次性密碼登入。未經 MFA 身分驗證的使用者無法擔任角色。如需有關 MFA 的詳細資訊，請參閱 AWS IAM 中的多重要素驗證

7. 選擇下一步。

8. IAM 包含您帳戶中的 AWS 受管和客戶受管政策清單。選取用於許可政策的政策，或者選擇 Create policy (建立政策) 以開啟新的瀏覽器標籤，並從頭建立新的政策。如需詳細資訊，請參閱建立 IAM 政策。在您建立政策後，關閉該標籤並返回您的原始標籤。選取您希望擔任角色的任何人具有的許可政策旁的核取方塊。如果您希望，您目前可以不選取政策，稍後再將政策連接到角色。角色預設沒有任何許可。

9. (選用) 設定許可界限。這是進階功能。

   開啟 Set permissions boundary (設定許可界限) 區段，並選擇 Use a permissions boundary to control the maximum role permissions (使用許可界限來控制角色許可上限)。選取用於許可界限的政策。

10. 選擇下一步。

11. 針對 Role name (角色名稱)，輸入您的角色名稱。角色名稱在您的 中必須是唯一的 AWS 帳戶。角色名稱用在政策中或作為 ARN 的一部分時，角色名稱區分大小寫。當主控台中的客戶顯示角色名稱時 (例如在登入程序期間)，角色名稱不區分大小寫。因為有各種實體可能會參考此角色，所以建立角色之後，您就無法編輯其名稱。
12. (選用) 在 Description (說明) 中，輸入新角色的說明。
13. 在 Step 1: Select trusted entities (步驟 1：選取受信任的實體) 或者 Step 2: Add permissions (步驟 2：新增許可) 區段中選擇 Edit (編輯)，可編輯使用案例和角色許可。您將會返回先前的頁面進行編輯。
14. (選用) 藉由連接標籤作為鍵值對，將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
15. 檢閱角色，然後選擇 Create role (建立角色)。

> **Important**
>
> 請記住，這只是所需設定的前半部。您也必須以信任的帳戶許可提供給個別使用者，以切換到主控台中的角色，或程式化方式擔任角色。如需有關此步驟的詳細資訊，請參閱向使用者授予切換角色的許可。

建立 IAM 角色 (AWS CLI)

從 建立角色 AWS CLI 涉及多個步驟。當您使用 主控台建立角色時，會為您完成許多步驟，但您必須使用 自行 AWS CLI 明確執行每個步驟。您必須建立角色，然後為該角色指派許可政策。或者，您也可以設定角色的許可界限。

若要為跨帳戶存取建立角色 (AWS CLI)

1. 建立角色：aws iam create-role
2. 將受管許可政策連接到角色：aws iam attach-role-policy

   或

   為角色建立內嵌許可政策：aws iam put-role-policy
3. (選用) 透過連接標籤來將自訂屬性新增至該角色：aws iam tag-role

   如需詳細資訊，請參閱 管理 IAM 角色 (AWS CLI 或 AWS API) 上的標籤。
4. (選用) 設定角色的許可界限：aws iam put-role-permissions-boundary

許可界限控制角色可以擁有的許可上限。許可界限是一項進階 AWS 功能。

以下範例顯示前兩個最常見步驟，可在簡單的環境中建立一個跨帳戶角色。此範例允許 123456789012 帳戶中的任何使用者擔任角色，並檢視 example_bucket Amazon S3 儲存貯體。此範例也假定您使用執行 Windows 的用戶端電腦，並且已使用您的帳戶憑證和區域設定您的命令列界面。如需詳細資訊，請參閱設定 AWS 命令列界面。

當您建立角色時，這個範例在第一個命令包含下列信任政策。此信任政策允許 123456789012 帳戶中的使用者使用 AssumeRole 操作來擔任角色，但只在使用者採用 SerialNumber 和 TokenCode 參數以提供 MFA 身分驗證時。如需有關 MFA 的詳細資訊，請參閱 AWS IAM 中的多重要素驗證。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
          "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
      }
  ]
}
```

Important

如果您的 Principal 元素包含特定 IAM 角色或使用者的 ARN，則該 ARN 會在儲存政策時轉換為唯一的主體 ID。如果有人希望藉由刪除並重新建立角色或使用者來提升許可時，這麼做可有助於減少此類風險。您通常不會在主控台中看到此 ID，因為在顯示信任政策時還會反向轉換回 ARN。不過，如果您刪除角色或使用者，則主體 ID 會顯示在主控台中，因為 AWS 無法再將其對應回 ARN。因此，如果您刪除並重新建立了信任政策的 Principal 元素所引用的使用者或角色，您必須編輯角色來替換 ARN。

當您使用第二個命令的政策，您必須將現有受管政策連接到角色。下列許可政策允許任何擔任角色的人只對 example_bucket Amazon S3 儲存貯體執行 ListBucket 動作。

JSON

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

若要建立此 Test-UserAccess-Role 角色，您必須先將之前的信任政策以名稱 trustpolicyforacct123456789012.json 儲存到 policies 資料夾 (在您的本機 C: 磁碟機)。然後將先前的許可政策儲存為 中的客戶受管政策 AWS 帳戶 ，名稱為 PolicyForRole。然後，您可以使用以下命令來建立角色，並連接受管政策。

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> 請記住，這只是所需設定的前半部。您還必須提供受信任帳戶中的個別使用者許可，以切換到該角色。如需有關此步驟的詳細資訊，請參閱 向使用者授予切換角色的許可。

在您建立角色並授予其執行 AWS 任務或存取 AWS 資源的許可後，123456789012帳戶中的任何使用者都可以擔任該角色。如需詳細資訊，請參閱切換到 IAM 角色 (AWS CLI)。

建立 IAM 角色 (AWS API)

從 AWS API 建立角色涉及多個步驟。當您使用主控台建立角色時，有許多步驟會自動為您完成，但是使用 API 的話，您必須自行明確執行每個步驟。您必須建立角色，然後為該角色指派許可政策。或者，您也可以設定角色的許可界限。

在程式碼中建立角色 (AWS API)

1. 建立角色：CreateRole

   對於角色的信任政策，您可以指定一個檔案位置。

2. 將受管許可政策連接到角色：AttachRolePolicy

   或

   為角色建立內嵌許可政策：PutRolePolicy

   > **Important**
   >
   > 請記住，這只是所需設定的前半部。您還必須提供受信任帳戶中的個別使用者許可，以切換到該角色。如需有關此步驟的詳細資訊，請參閱 向使用者授予切換角色的許可。

3. （選用）透過連接標籤將自訂屬性新增至角色：TagRole

   如需詳細資訊，請參閱管理 IAM 角色 (AWS CLI 或 AWS API) 上的標籤。

4. (選用) 設定角色的許可界限：PutRolePermissionsBoundary

   許可界限控制角色可以擁有的許可上限。許可界限是一項進階 AWS 功能。

在您建立角色並授予其執行 AWS 任務或存取 AWS 資源的許可後，您必須將許可授予帳戶中的使用者，以允許他們擔任該角色。如需有關擔任角色的詳細資訊，請參閱 切換到 IAM 角色 (AWS API)。

建立 IAM 角色 (AWS CloudFormation)

如需有關在 中建立 IAM 角色的資訊 AWS CloudFormation，請參閱AWS CloudFormation 《使用者指南》中的資源和屬性參考和範例。

如需 IAM 範本的詳細資訊 AWS CloudFormation，請參閱AWS CloudFormation 《使用者指南》中的AWS Identity and Access Management 範本程式碼片段。

建立角色以將許可委派給 AWS 服務

許多 AWS 服務要求您使用 角色，以允許服務代表您存取其他 服務中的資源。服務會擔任代您執行動作的角色稱為服務角色。當角色做為服務的專業用途時，它歸類為服務連結角色。若要查看使用服務連結的角色支援哪些服務，或者服務是否支援任何形式的臨時憑證的詳細資訊，請參閱 AWS 使用 IAM 的 服務。若要了解個別服務如何使用角色，請在表格中選擇服務名稱以查看該服務的文件。

設定 PassRole 許可時，您應確保使用者傳遞的角色不會具有比您希望使用者具有之許可更多的許可。例如，可能不允許 Alice 執行任何 Amazon S3 動作。如果 Alice 可以將角色傳遞給允許 Amazon S3 動作的服務，則該服務可以在執行任務時代表 Alice 執行 Amazon S3 動作。

如需有關角色如何協助您委派許可的詳細資訊，請參閱 角色術語和概念。

服務角色許可

您必須設定許可，允許 IAM 實體 (使用者或角色) 建立或編輯服務角色。

> **Note**
>
> 服務連結角色的 ARN 包括服務主體，這在下列政策中以 *SERVICE-NAME*.amazonaws.com 形式指出。不要嘗試猜測服務主體，因為它是區分大小寫，且格式可以因各 AWS 服務而異。若要檢視服務的服務主體，請參閱該服務連結的角色文件。

允許 IAM 實體建立特定服務角色

將下列政策新增至需要建立服務角色的 IAM 實體。此政策可讓您建立所指定服務且具有特定名稱的服務角色。然後，您可以將受管或內嵌政策連接至該角色。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:PutRolePolicy"
```

```
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
}
```

允許 IAM 實體建立任何服務角色

AWS 建議您只允許管理使用者建立任何服務角色。具有建立角色和連接任何政策之許可的人員可以提升自己的許可。反之，建立一個政策，讓他們只建立所需的角色，或讓系統管理員代表他們建立服務角色。

若要連接允許管理員存取整個 的政策 AWS 帳戶，請使用 AdministratorAccess AWS 受管政策。

允許 IAM 實體編輯服務角色

將下列政策新增至需要編輯服務角色的 IAM 實體。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
            "Sid": "ViewRolesAndPolicies",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

允許 IAM 實體刪除特定服務角色

將下列陳述式新增至需要刪除所指定服務角色之 IAM 實體的許可政策。

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

允許 IAM 實體刪除任何服務角色

AWS 建議您只允許管理使用者刪除任何服務角色。反之，建立一個政策，只允許他們刪除所需的角色，或讓系統管理員代表他們刪除服務角色。

若要連接允許管理員存取整個 的政策 AWS 帳戶，請使用 AdministratorAccess AWS 受管政策。

為 AWS 服務建立角色 （主控台）

您可以使用 AWS 管理主控台 來建立 服務的角色。因為有些服務支援多個服務角色，所以請參閱服務的 AWS 文件，以查看要選擇的使用案例。您可以了解如何為角色指派必要的信任和許可政策，以便服務可以代表您擔任角色。您用來控制角色的許可的步驟可能有所不同，端視服務如何定義使用案例，以及是否建立服務連結的角色而定。

Console

為 AWS 服務 (IAM 主控台） 建立角色

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇角色，然後選擇建立角色。

3. 對於 Trusted entity type (信任的實體類型)，請選擇 AWS 服務。
4. 針對服務或使用案例，選擇服務，然後選擇使用案例。服務會定義使用案例，以包含服務所需的信任政策。
5. 選擇下一步。
6. 對於許可政策，選項取決於您選取的使用案例：

   - 如果服務定義了角色的許可，則您無法選取許可政策。
   - 從一組有限的許可政策中選取。
   - 從所有許可政策中選取。
   - 選取無許可政策，在建立角色之後建立政策，然後將政策連接到角色。
7. (選用) 設定許可界限。這是進階功能，可用於服務角色，而不是服務連結的角色。

   a. 開啟設定許可界限區段，然後選擇使用許可界限來控制角色許可上限。

      IAM 包含您帳戶中 AWS 受管和客戶受管政策的清單。

   b. 選取用於許可界限的政策。
8. 選擇下一步。
9. 對於角色名稱，選項取決於服務：

   - 如果服務定義了角色名稱，則無法編輯角色名稱。
   - 如果服務定義了角色名稱的字首，則可以輸入選用字尾。
   - 如果服務未定義角色名稱，則可以為角色命名。

   > **Important**
   >
   > 當您命名角色時，請注意下列事項：
   >
   > - 角色名稱在您的 中必須是唯一的 AWS 帳戶，而且無法依大小寫設為唯一。
   >
   >   例如，不要同時建立名為 **PRODROLE** 和 **prodrole** 的角色。當角色名稱用於政策或 ARN 的一部分時，角色名稱會區分大小寫，但是當角色名稱在主控台中顯示給客戶時，例如在登入過程中，角色名稱不會區分大小寫。
   > - 因為其他實體可能會參考角色，所以在建立角色之後，就無法編輯其名稱。
10. (選用) 在說明中，輸入角色的說明。
11. (選用) 若要編輯使用案例和角色許可，請在步驟 1：選取受信任的實體或者步驟 2：新增許可區段中選擇編輯。

12. (選用) 若要協助識別、組織或搜尋角色，請將標籤新增為索引鍵值對。如需在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的AWS Identity and Access Management 資源的標籤。
13. 檢閱角色，然後選擇 Create role (建立角色)。

為服務建立角色 (AWS CLI)

從 建立角色 AWS CLI 涉及多個步驟。當您使用 主控台建立角色時，會為您完成許多步驟，但您必須使用 自行 AWS CLI 明確執行每個步驟。您必須建立角色，然後為該角色指派許可政策。如果您使用的服務是 Amazon EC2，則還必須建立執行個體描述檔並新增該角色。或者，您也可以設定角色的許可界限。

從 建立 AWS 服務的角色 AWS CLI

1. 以下 `create-role` 命令會建立名為 Test-Role 的角色，並將信任政策連接至該角色：

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```

2. 將受管許可政策連接到角色：aws iam attach-role-policy

   例如，下列 `attach-role-policy` 命令會將名為 `ReadOnlyAccess` 的 AWS 受管政策連接至名為 `ReadOnlyRole` 的 IAM 角色：

   ```
   aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
   ReadOnlyAccess --role-name ReadOnlyRole
   ```

   或

   為角色建立內嵌許可政策：aws iam put-role-policy

   若要新增嵌許可政策，請參閱以下範例：

   ```
   aws iam put-role-policy --role-name Test-Role --policy-name
   ExamplePolicy --policy-document file://AdminPolicy.json
   ```

3. (選用) 透過連接標籤來將自訂屬性新增至該角色：aws iam tag-role

   如需詳細資訊，請參閱 管理 IAM 角色 (AWS CLI 或 AWS API) 上的標籤。

4. (選用) 設定角色的許可界限：aws iam put-role-permissions-boundary

許可界限控制角色可以擁有的許可上限。許可界限是一項進階 AWS 功能。

如果您要將角色與 Amazon EC2 或其他使用 Amazon EC2 AWS 的服務搭配使用，則必須將角色存放在執行個體描述檔中。執行個體描述檔是角色的容器，可以在啟動時連接到 Amazon EC2 執行個體。執行個體設定檔只能包含一個角色，並且無法增加該限制。如果您使用 建立角色 AWS 管理主控台，則會使用與角色相同的名稱為您建立執行個體描述檔。如需有關執行個體描述檔的詳細資訊，請參閱使用執行個體設定檔。如需有關如何使用角色啟動 EC2 執行個體的資訊，請參閱《Amazon EC2 使用者指南》中的控制對 Amazon EC2 資源的存取。

建立執行個體設定檔並將角色存放在其中 (AWS CLI)

1. 建立執行個體描述檔：aws iam create-instance-profile
2. 將角色新增至執行個體描述檔：aws iam add-role-to-instance-profile

以下 AWS CLI 範例命令集示範建立角色和連接許可的前兩個步驟。它還說明兩個建立執行個體設定檔和新增角色至描述檔的步驟。此範例信任允許 Amazon EC2 服務擔任角色並檢視 `example_bucket` Amazon S3 儲存貯體的政策。此範例也假定您在執行 Windows 的用戶端電腦上執行，並且已使用您的帳戶憑證和區域設定您的命令列界面。如需詳細資訊，請參閱設定 AWS 命令列界面。

當您建立角色時，這個範例在第一個命令包含下列信任政策。此信任政策允許 Amazon EC2 服務擔任該角色。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

當您使用第二個命令的政策，您必須將許可政策連接到角色。以下範例許可政策允許角色僅在 `example_bucket` Amazon S3 儲存貯體上執行 `ListBucket` 動作。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

若要建立此 Test-Role-for-EC2 角色，您必須先將之前的信任政策以名稱 trustpolicyforec2.json 和之前名為 permissionspolicyforec2.json 的許可政策儲存到您本機 policies 磁碟機的 C: 目錄。然後，您可以使用以下命令來建立角色、連接政策、建立執行個體設定檔，並新增角色到執行個體設定檔。

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json

# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

當您啟動 EC2 執行個體時，如果您使用 AWS 主控台，請在設定執行個體詳細資訊頁面中指定執行個體描述檔名稱。如果您使用 aws ec2 run-instances CLI 命令，指定 --iam-instance-profile 參數。

為服務建立角色 (AWS API)

從 AWS API 建立角色涉及多個步驟。當您使用主控台建立角色時，有許多步驟會自動為您完成，但是使用 API 的話，您必須自行明確執行每個步驟。您必須建立角色，然後為該角色指派許可政策。如果

您使用的服務是 Amazon EC2，則還必須建立執行個體描述檔並新增該角色。或者，您也可以設定角色的許可界限。

為 AWS 服務建立角色 (AWS API)

1. 建立角色：CreateRole

   對於角色的信任政策，您可以指定一個檔案位置。

2. 將受管許可政策連接到角色：AttachRolePolicy

   或

   為角色建立內嵌許可政策：PutRolePolicy

3. （選用）透過連接標籤將自訂屬性新增至角色：TagRole

   如需詳細資訊，請參閱管理 IAM 角色 (AWS CLI 或 AWS API) 上的標籤。

4. (選用) 設定角色的許可界限：PutRolePermissionsBoundary

   許可界限控制角色可以擁有的許可上限。許可界限是一項進階 AWS 功能。

如果您要將角色與 Amazon EC2 或其他使用 Amazon EC2 AWS 的服務搭配使用，則必須將角色存放在執行個體描述檔中。執行個體設定檔是角色的容器。每個執行個體設定檔只能包含一個角色，並且無法增加該限制。如果您在 中建立角色 AWS 管理主控台，則會使用與角色相同的名稱為您建立執行個體描述檔。如需有關執行個體描述檔的詳細資訊，請參閱 使用執行個體設定檔。如需有關如何使用角色啟動 Amazon EC2 執行個體的資訊，請參閱《Amazon EC2 使用者指南》中的控制對 Amazon EC2 資源的存取。

建立執行個體描述檔並將角色存放在其中 (AWS API)

1. 建立執行個體描述檔：CreateInstanceProfile
2. 將角色新增至執行個體描述檔：AddRoleToInstanceProfile

建立服務連結角色

服務連結角色是一種獨特的 IAM 角色類型，可直接連結到 AWS 服務。服務連結角色是由服務預先定義，並包含服務代表您呼叫其他 AWS 服務所需的所有許可。連結的服務也定義您如何建立、修改和刪除服務連結的角色。服務可能會自動建立或刪除角色。做為服務中精靈或程序一部分，它也許可讓您建

立、修改或刪除角色。或者，它可能要求您使用 IAM 來建立或刪除角色。不論採用何種方式，服務連結角色可簡化設定服務流程，因為您不必手動新增服務許可，以代表您完成動作。

> **Note**
>
> 請記得，服務角色與服務連結角色不同。服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務。服務連結角色是連結至 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

連結的服務定義其服務連結角色的許可，除非另有定義，否則僅有該服務可以擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

在您刪除角色之前，您必須首先刪除它們的相關資源。這有助於防止不小心移除資源存取許可。

> **Tip**
>
> 如需哪些服務支援使用服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找 Service-Linked Role (服務連結角色) 欄中顯示 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。

服務連結角色許可

您必須設定 IAM 實體 (使用者或角色) 的許可，允許使用者或角色建立或編輯服務連結角色。

> **Note**
>
> 服務連結角色的 ARN 包括服務主體，這在下列政策中以 *SERVICE-NAME*.amazonaws.com 形式指出。請勿嘗試猜測服務委託人，因為它區分大小寫，而且格式可能因 AWS 服務而異。若要檢視服務的服務主體，請參閱該服務連結的角色文件。

允許 IAM 實體建立特定服務連結角色

將下列政策新增至需要建立服務連結角色的 IAM 實體。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

若要允許 IAM 實體建立任何服務連結角色

將下列陳述式新增至需要建立服務連結角色的 IAM 實體的許可政策，或包含所需政策的任何服務角色。此政策陳述式不允許 IAM 實體連接政策到角色。

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

若要允許 IAM 實體編輯任何服務連結角色的說明

將下列陳述式新增至需要編輯服務連結角色說明或任何服務角色的 IAM 實體的許可政策。

```
{
```

```
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

若要允許 IAM 實體刪除特定服務連結角色

將下列陳述式新增至需要刪除服務連結角色的 IAM 實體的許可政策。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

允許 IAM 實體刪除任何服務連結角色

將下列陳述式新增至需要刪除服務連結角色 (但並非服務角色) 的 IAM 實體許可政策。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

允許 IAM 實體將現有角色傳遞到服務

有些 AWS 服務可讓您將現有角色傳遞給服務，而不是建立新的服務連結角色。若要執行此操作，使用者必須擁有傳遞角色給服務的許可。在需要傳遞角色之 IAM 實體的許可政策中，新增下列陳述式。透過這個政策陳述式，實體也可以檢視角色清單，並從中選擇要傳遞的角色。如需詳細資訊，請參閱授予使用者將角色傳遞至 AWS 服務的許可。

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
```

```
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

具有服務連結角色的間接許可

可將服務連結角色授予的許可間接轉移給其他使用者和角色。當 AWS 服務使用服務連結角色時，該服務連結角色可以使用自己的許可來呼叫其他 AWS 服務。這表示使用者和角色 (具有呼叫使用服務連結角色之服務的許可) 可能會間接存取該服務連結角色所能存取的服務。

例如，當您建立 Amazon RDS 資料庫執行個體時，RDS 的服務連結角色如果尚未存在，則會自動建立。此服務連結角色可讓 RDS 代您呼叫 Amazon EC2、Amazon SNS、Amazon SNS、Amazon CloudWatch Logs 和 Amazon Kinesis。如果您允許帳戶中的使用者和角色修改或建立 RDS 資料庫，則他們可以透過呼叫 RDS 來間接與 Amazon EC2、Amazon SNS、Amazon CloudWatch Logs 日誌和 Amazon Kinesis 資源互動，因為 RDS 會使用其服務連結角色來存取這些資源。

服務連結角色建立方法

您用來建立服務連結角色的方法取決於服務。在某些情況下，您不需要手動建立一個服務連結角色。例如，當您在服務中完成特定動作 (例如建立資源)，該服務可能會為您建立服務連結角色。或者，您若在服務開始支援服務連結角色之前已在使用該服務，則服務可能已在您的帳戶中自動建立角色。如需進一步了解，請參閱 我的 AWS 帳戶中出現新角色。

在其他情況下，服務可能支援使用服務主控台、API 或 CLI 手動建立服務連結角色。如需哪些服務支援使用服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找 Service-Linked Role (服務連結角色) 欄中顯示 Yes (是) 的服務。若要了解服務是否支援建立服務連結角色，請選擇是連結以檢視該服務的服務連結角色的文件。

如果服務不支援建立角色，則可以使用 IAM 來建立服務連結角色。

> **Important**
>
> 服務連結角色算作您的 AWS 帳戶中的 IAM 角色限制，但是如果您已達到限制，仍然可以在您的帳戶中建立服務連結角色。只有服務連結角色可以超過限制。

建立服務連結角色 (主控台)

在 IAM 中建立服務連結角色之前，了解連結的服務是否自動建立服務連結角色，此外，了解是否您可以從服務主控台、API 或 CLI 建立角色。

建立服務連結角色 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇角色。然後，選擇 Create role (建立角色)。
3. 選擇 AWS Service (服務) 角色類型。
4. 選擇服務的使用案例。服務會定義使用案例，以包含服務所需的信任政策。然後選擇下一步。
5. 選擇一或多個許可政策以連接至角色。根據您選取的使用案例，服務可能可以執行下列任何操作：

   - 定義角色使用的許可。
   - 可讓您從有限的一組許可中進行選擇。
   - 可讓您從任何許可中進行選擇。
   - 可讓您目前無法選取政策、稍後建立政策，然後將它們連接至角色。

   選取可指派您希望角色擁有之許可政策旁的核取方塊，然後選擇下一步。

   > **Note**
   >
   > 您指定的許可適用於任何使用角色的實體。角色預設沒有任何許可。

6. 針對 Role name (角色名稱)，服務會定義角色名稱自訂程度。如果服務定義角色的名稱，則此選項無法編輯。在其他情況下，服務可能會定義角色的字首，並且允許您輸入選用後綴。

   如果可能，輸入要新增至預設名稱的角色名稱後綴。此後綴可協助您識別此角色的用途。角色名稱在您的 AWS 帳戶內必須是獨一無二的。它們無法透過大小寫進行區分。例如，您無法建立名為 **`<service-linked-role-name>_SAMPLE`** 和 **`<service-linked-role-name>_sample`** 的角色。因為有各種實體可能會參照角色，所以您無法在建立角色之後編輯角色名稱。

7. (選用) 在 Description (說明) 中，編輯新服務連結角色的說明。
8. 您不能在建立角色時，將標籤連接至服務連結的角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
9. 檢閱角色，然後選擇 Create role (建立角色)。

建立服務連結角色 (AWS CLI)

在 IAM 中建立服務連結角色之前，了解連結的服務是否自動建立服務連結角色，並且了解是否您可以從服務的 CLI 建立角色。如果不支援服務 CLI，您可以使用 IAM 命令，使用服務擔任該角色所需的信任政策與內嵌政策來建立服務連結角色。

建立服務連結角色 (AWS CLI)

執行以下命令：

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

建立服務連結角色 (AWS API)

在 IAM 中建立服務連結角色之前，了解連結的服務是否自動建立服務連結角色，並且了解是否您可以從服務的 API 建立角色。如果不支援服務 API，您可以使用 AWS API 來建立具有信任政策和內嵌政策的服務連結角色，該政策是服務擔任該角色所需的。

建立服務連結角色 (AWS API)

使用 CreateServiceLinkedRole API 呼叫。在請求中指定 *SERVICE_NAME_URL*.amazonaws.com 的服務名稱。

例如，要建立 Lex Bots 服務連結的角色，使用 lex.amazonaws.com。

為第三方身分提供者建立角色

您可以使用身分提供者，而不是在 中建立 IAM 使用者 AWS 帳戶。使用身分提供者 (IdP)，您可以在外部管理您的使用者身分， AWS 並授予這些外部使用者身分存取您帳戶中 AWS 資源的許可。如需有關聯合身分與身分提供者的詳細資訊，請參閱 身分提供者和 的聯合身分 AWS。

為 OIDC 和 SAML 聯合身分主體建立角色 (主控台)

建立角色的程序取決於所選擇的第三方提供者：

- 如需 OpenID Connect (OIDC)，請參閱為 OpenID Connect 聯合身分建立角色 (主控台)。
- 關於 SAML 2.0 的詳細資訊，請參閱為 SAML 2.0 聯合身分建立角色 (主控台)。

為聯合存取建立角色 (AWS CLI)

從 AWS CLI 為支援的身分提供者 (OIDC 或 SAML) 建立角色的步驟是相同的。區別在於，您在先決條件步驟中建立的信任政策的內容不同。首先，遵循先決條件小節中針對您正在使用之提供者類型的步驟操作：

- 關於 &OIDC; 提供者的詳細資訊，請參閱建立適用於 OIDC 的角色的先決條件。
- 關於 &SAML; 提供者的詳細資訊，請參閱建立適用於 SAML 的角色的先決條件。

從 建立角色 AWS CLI 涉及多個步驟。當您使用 主控台建立角色時，會為您完成許多步驟，但您必須使用 自行 AWS CLI 明確執行每個步驟。您必須建立角色，然後為該角色指派許可政策。或者，您也可以設定角色的許可界限。

若要建立角色 (AWS CLI)

1. 建立角色：aws iam create-role
2. 將許可政策連接到角色：aws iam attach-role-policy

   或

   為角色建立內嵌許可政策：aws iam put-role-policy
3. (選用) 透過連接標籤來將自訂屬性新增至該角色：aws iam tag-role

   如需詳細資訊，請參閱 管理 IAM 角色 (AWS CLI 或 AWS API) 上的標籤。
4. (選用) 設定角色的許可界限：aws iam put-role-permissions-boundary

   許可界限控制角色可以擁有的許可上限。許可界限是一項進階 AWS 功能。

以下範例顯示前兩個最常見步驟，可在簡單的環境中建立一個身分提供者角色。此範例允許 123456789012 帳戶中的任何使用者擔任角色，並檢視 example_bucket Amazon S3 儲存貯體。此範例也假設您在執行 Windows AWS CLI 的電腦上執行 ，並已 AWS CLI 使用您的登入資料設定 。如需詳細資訊，請參閱設定 AWS Command Line Interface。

下列範例信任政策是針對使用者使用 Amazon Cognito 登入時的行動應用程式而設計。在此範例中，*us-east:12345678-ffff-ffff-ffff-123456* 代表由 Amazon Cognito 指派的身分集區 ID。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

下列許可政策允許任何擔任角色的人只對 example_bucket Amazon S3 儲存貯體執行 ListBucket 動作。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

若要建立此 Test-Cognito-Role 角色，您必須先將之前的信任政策以名稱 trustpolicyforcognitofederation.json 和之前名為 permspolicyforcognitofederation.json 的許可政策儲存到您本機 policies 磁碟機的 C: 資料夾。然後，您可以使用以下命令來建立角色，並連接內嵌政策。

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json
```

```
# Attach the permissions policy to the role to specify what it is allowed to do.
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

建立聯合存取 (AWS API) 的角色

從 AWS CLI 為支援的身分提供者 (OIDC 或 SAML) 建立角色的步驟是相同的。區別在於，您在先決條件步驟中建立的信任政策的內容不同。首先，遵循先決條件小節中針對您正在使用之提供者類型的步驟操作：

- 關於 &OIDC; 提供者的詳細資訊，請參閱建立適用於 OIDC 的角色的先決條件。
- 關於 &SAML; 提供者的詳細資訊，請參閱建立適用於 SAML 的角色的先決條件。

建立角色 (AWS API)

1. 建立角色：CreateRole
2. 將許可政策連接到角色：AttachRolePolicy

   或

   為角色建立內嵌許可政策：PutRolePolicy
3. （選用）透過連接標籤將自訂屬性新增至角色：TagRole

   如需詳細資訊，請參閱管理 IAM 角色 (AWS CLI 或 AWS API) 上的標籤。
4. (選用) 設定角色的許可界限：PutRolePermissionsBoundary

   許可界限控制角色可以擁有的許可上限。許可界限是一項進階 AWS 功能。

為 OpenID Connect 聯合身分建立角色 (主控台)

您可以使用 OpenID Connect (OIDC) 聯合身分提供者，而不是在您的 中建立 AWS Identity and Access Management 使用者 AWS 帳戶。使用身分提供者 (IdP)，您可以在 外部管理您的使用者身分， AWS 並授予這些外部使用者身分存取 AWS 您帳戶中資源的許可。如需有關聯合身分與 IdP 的詳細資訊，請參閱 身分提供者和 的聯合身分 AWS。

建立適用於 OIDC 的角色的先決條件

您必須先完成以下先決條件步驟，然後才能建立用於 OIDC 聯合身分的角色。

若要準備建立用於 OIDC 聯合身分的角色

1. 使用一或多個提供聯合 OIDC 身分的服務進行註冊。如果您要建立需要存取 AWS 資源的應用程式，您也可以使用提供者資訊來設定應用程式。當您這麼做時，提供者會將應用程式唯一的 ID 提供給您的應用程式或對象。(不同的提供者可能使用不同的術語來表達此程序。本指南則使用術語設定來表示向提供者識別您應用程式的程序)。您可以在每個提供者設定多個應用程式，或在單一應用程式設定多個提供者。檢視有關使用身分提供者的相關資訊，如下所示：

   - Login with Amazon 開發人員中心
   - Facebook 開發人員網站上的新增 Facebook 登入到您的應用程式或網站。
   - Google 開發人員網站上的登入時使用 OAuth 2.0 (OpenID Connect)。

2. 從 IdP 收到必要資訊後，請在 IAM 中建立 IdP。如需詳細資訊，請參閱在 IAM 中建立 OpenID Connect (OIDC) 身分提供者。

   > **Important**
   >
   > 如果您正在使用來自 Google、Facebook 或 Amazon Cognito 的 OIDC IdP，請勿在 AWS 管理主控台中建立單獨的 IAM IdP。這些 OIDC 身分提供者已內建於 AWS ，可供您使用。略過此步驟，並在接下來的步驟中使用您的 IdP 建立新角色。

3. 為已進行 IdP 身分驗證的使用者要擔任的角色準備政策。正如任何角色一樣，手機應用程式的角色含有兩項政策。其中一項是信任政策，其指定擔任該角色的對象。另一項政策是許可政策，其指定行動應用程式被允許或拒絕存取的 AWS 動作和資源。

   對於 Web 身分提供者，我們建議您使用 Amazon Cognito 來管理身分。在這種情況下，請使用類似於這個範例的信任政策。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
```

```
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

以 Amazon Cognito 指派給您的身分集區 ID 取代 `us-east-2:12345678-abcd-abcd-abcd-123456`。

如果您手動設定 OIDC IdP，當您建立信任政策，您必須使用三個值，以確保只有您的應用程式可擔任該角色：

- 對於 `Action` 元素，可使用 `sts:AssumeRoleWithWebIdentity` 動作。
- 如需 `Principal` 元素，請使用字串 `{"Federated":`*`providerUrl/providerArn`*`}`。
  - 對於一些常見的 OIDC IdP，*`providerUrl`* 為 URL。以下範例包含多個方式，為部分常見 idP 指定主體：

    ```
    "Principal":{"Federated":"cognito-identity.amazonaws.com"}
    ```

    ```
    "Principal":{"Federated":"www.amazon.com"}
    ```

    ```
    "Principal":{"Federated":"graph.facebook.com"}
    ```

    ```
    "Principal":{"Federated":"accounts.google.com"}
    ```

  - 對於其他的 OIDC 提供者，請使用您在 Step 2 中建立的 OIDC 身分提供者的 Amazon Resource Name (ARN)，如以下範例所示：

    ```
    "Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
    server.example.com"}
    ```

- 對於 `Condition` 元素，可使用 `StringEquals` 條件來限制許可。測試身分集區 ID (對於 Amazon Cognito) 或應用程式 ID (對於其他提供者)。身分集區 ID 應與您透過 IdP 配置應用程式時所收到的應用程式 ID 一致。ID 之間的比對可確保請求來自您的應用程式。

> **Note**
>
> Amazon Cognito 身分集區的 IAM 角色信任服務主體 `cognito-identity.amazonaws.com` 擔任該角色。此類型的角色必須包含至少一個條件索引鍵，以限制可擔任該角色的主體。

其他考量事項適用於擔任跨帳戶 IAM 角色的 Amazon Cognito 身分集區。這些角色的信任政策必須接受 cognito-identity.amazonaws.com 服務主體，且必須包含 aud 條件索引鍵，以限制來自您預期身分集區的使用者擔任角色。如果不符合此條件，信任 Amazon Cognito 身分集區的政策會產生意外身分集區中的使用者可能擔任該角色的風險。如需詳細資訊，請參閱 Amazon Cognito Developer Guide 中的 Trust policies for IAM roles in Basic (Classic) authentication。

建立類似以下其中一個範例的條件元素，其取決於您使用的 IdP：

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
"us-east:12345678-ffff-ffff-ffff-123456"}}
```

```
"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}
```

```
"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}
```

```
"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

對於 OIDC 提供者，請將 OIDC IdP 的完全合格 URL 與 aud 內容索引鍵一起使用，如以下範例所示：

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

> **Note**
>
> 角色的信任政策中的主體的值是 IdP 特有的。OIDC 的角色只能指定一個主體。因此，如果行動應用程式允許使用者從多個 IdP 登入，則為您要支援的每個 IdP 建立不同的角色。分別為每個 IdP 建立信任政策。

如果使用者使用行動應用程式從 Login with Amazon 登入，則以下範例信任政策適用。在範例中，*amzn1.application-oa2-123456* 代表使用 Login with Amazon 設定應用程式時 Amazon 指派的應用程式 ID。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

如果使用者使用行動應用程式從 Facebook 登入，則以下範例信任政策適用。在本範例中，*111222333444555* 代表 Facebook 指派的應用程式 ID。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

如果使用者使用行動應用程式從 Google 登入，則以下範例信任政策適用。在本範例中，*666777888999000* 代表由 Google 指派的應用程式 ID。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

如果使用者使用行動應用程式從 Amazon Cognito 登入，則以下範例信任政策適用。在此範例中，*us-east:12345678-ffff-ffff-ffff-123456* 代表 Amazon Cognito 指派的身分集區 ID。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"cognito-
identity.amazonaws.com:aud": "us-east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

為 OIDC 建立角色

完成先決條件後，您可在 IAM 建立角色。對於已辨識的共用 OpenID Connect (OIDC) 身分提供者 (IdP)，IAM 需要明確評估 JSON Web 權杖 (JWT) 中稱為身分提供者控制項的特定宣告。如需有關具有身分提供者控制項之 OIDC IdP 的詳細資訊，請參閱共用 OIDC 提供者的身分提供者控制項。

下列程序介紹如何在 AWS 管理主控台中建立用於 OIDC 聯合身分的角色。若要從 AWS CLI 或 AWS API 建立角色，請參閱 中的程序為第三方身分提供者建立角色。

> **Important**
>
> 如果您使用 Amazon Cognito，請使用 Amazon Cognito 主控台來設定角色。否則，請使用 IAM 主控台來為 OIDC 聯合身分建立角色。

若要為 OIDC 聯合身分建立 IAM 角色

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 選擇 Web 身分作為信任的實體類型，然後選取下一步。
4. 針對 Identity provider (身分提供者)，選擇您角色的身分提供者：
   - 如果要為個別 Web 身分提供者建立角色，請選擇 Login with Amazon、Facebook 或 Google。

     > **Note**
     >
     > 您必須為想要支援的每個身分提供者建立單獨的角色。

   - 如果想要為 Amazon Cognito 建立進階案例角色，請選擇 Amazon Cognito。

     > **Note**
     >
     > 只有在處理進階案例時，必須手動建立與 Amazon Cognito 一起使用的角色。否則，Amazon Cognito 可以為您建立角色。如需 Amazon Cognito 的詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的身分集區 (聯合身分) 外部身分提供者。

   - 如果您想要為 GitHub Actions 建立角色，您必須先將 GitHub OIDC 提供者新增至 IAM。將 GitHub OIDC 提供者新增至 IAM 之後，請選擇 token.actions.githubusercontent.com。

> **Note**
>
> 如需如何設定 AWS 將 GitHub 的 OIDC 供應商信任為聯合身分的相關資訊，請參閱 GitHub 文件 - 在 Amazon Web Services 中設定 OpenID Connect。若要針對與 GitHub 的 IAM IdP 相關聯的角色限制存取權，最佳實務的相關資訊請見本頁的設定 GitHub OIDC 身分提供者的角色。

- 如果想要為 HashiCorp Cloud Platform (HCP) Terraform 建立角色，您必須先將 Terraform OIDC 提供者新增至 IAM。將 Terraform OIDC 提供者新增至 IAM 之後，選擇 app.terraform.io。

> **Important**
>
> HashiCorp Cloud Platform (HCP) Terraform OIDC 提供者的 IAM 角色必須在角色信任政策中評估 IAM 條件索引鍵 `app.terraform.io:sub`。此條件索引鍵會限制哪些 HCP Terraform 組織、專案、工作區或執行階段能夠擔任該角色。如果沒有此條件金鑰，您的信任政策會透過組織外部的身分授予對角色 AWS 和資源的存取權，這不符合最低權限原則。
>
> 如果您為 AWS 帳戶中與 HCP Terraform OIDC 供應商相關聯的角色設定或修改角色信任政策，但未評估 IAM 條件金鑰 `app.terraform.io:sub`，您將會收到錯誤。此外，如果您的角色信任政策未評估此條件索引鍵， AWS STS 將拒絕授權請求。

5. 請求的資訊會根據所選擇的 OIDC 提供者而有所不同。

   - 輸入您的應用程式的識別碼。識別碼的標籤會因所選擇的提供者而改變：
     - 如果您要為 Login with Amazon 建立角色，請將應用程式 ID 輸入 Application ID (應用程式 ID) 方塊中。
     - 如果您要為 Facebook 建立角色，請將應用程式 ID 輸入 Application ID (應用程式 ID) 方塊中。
     - 如果您要為 Google 建立角色，請在 Audience (對象) 方塊中輸入對象名稱。
     - 如果您要為 Amazon Cognito 建立角色，請在 Identity Pool ID (身分集區 ID) 方塊中輸入您為 Amazon Cognito 應用程式建立的身分集區 ID。
   - 如果您想要為 GitHub Actions 建立角色，請輸入下列詳細資訊：
     - 針對 Audience (對象)，選擇 `sts.amazonaws.com`。

- 針對 GitHub 組織，輸入 GitHub 組織名稱。GitHub 組織名稱是必要資訊，而且必須由包含破折號 (-) 的英數字元組成。您無法在 GitHub 組織名稱中使用萬用字元 (* 和 ?)。
- (選用) 針對 GitHub 儲存庫，輸入 GitHub 儲存庫名稱。如果您不指定值，則會預設為萬用字元 (*)。
- (選用) 針對 GitHub 分支，輸入 GitHub 分支名稱。如果您不指定值，則會預設為萬用字元 (*)。

- 如果您想要為 HashiCorp Cloud Platform (HCP) Terraform 建立角色，請輸入下列詳細資訊：
  - 針對 Audience (對象)，選擇 `aws.workload.identity`。
  - 針對組織，輸入組織名稱。您可以為所有組織指定萬用字元 (*)。
  - 針對專案，輸入專案名稱。您可以為所有專案指定萬用字元 (*)。
  - 針對工作區，輸入工作區名稱。您可以為所有工作區指定萬用字元 (*)。
  - 針對執行階段，輸入執行階段名稱。您可以為所有執行階段指定萬用字元 (*)。

6. (選用) 針對條件 (選用)，選擇新增條件，以建立應用程式使用者在能夠使用角色所授予的許可之前所必須滿足的其他條件。例如，您可以新增僅針對特定 IAM 使用者 ID 授予 AWS 資源存取權的條件。您也可以在建立角色之後，將條件新增至信任政策。如需詳細資訊，請參閱更新角色信任政策。
7. 檢閱您的 OIDC 資訊，然後選擇下一步。
8. IAM 包含您帳戶中 AWS 受管和客戶受管政策的清單。選取用於許可政策的政策，或者選擇 Create policy (建立政策) 以開啟新的瀏覽器標籤，並從頭建立新的政策。如需詳細資訊，請參閱建立 IAM 政策。在您建立政策後，關閉該標籤並返回您的原始標籤。選取您希望 OIDC 使用者具有的許可政策旁的核取方塊。如果您希望，您目前可以不選取政策，稍後再將政策連接到角色。角色預設沒有任何許可。
9. (選用) 設定許可界限。這是進階功能。

   開啟 Permissions boundary (許可界限) 區段，並選擇 Use a permissions boundary to control the maximum role permissions (使用許可界限來控制角色許可上限)。選取用於許可界限的政策。
10. 選擇下一步。
11. 在 Role name (角色名稱) 中，輸入角色名稱。角色名稱在您的 中必須是唯一的 AWS 帳戶。它們不區分大小寫。例如，您無法建立名為 **`PRODROLE`** 和 **`prodrole`** 的角色。由於其他 AWS 資源可能會參考角色，因此您無法在建立角色之後編輯角色的名稱。
12. (選用) 在 Description (說明) 中，輸入新角色的說明。
13. 如要編輯使用案例和角色許可，請在 Step 1: Select trusted entities (步驟 1：選取受信任的實體) 或者 Step 2: Add permissions (步驟 2：新增許可) 區段中選擇 Edit (編輯)。

14. (選用) 若要將中繼資料新增至角色，請附加標籤做為鍵/值對。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
15. 檢閱角色，然後選擇 Create role (建立角色)。

設定 GitHub OIDC 身分提供者的角色

如果您使用 GitHub 做為 OpenID Connect (OIDC) 身分提供者 (IdP)，最佳實務是限制可擔任與 IAM IdP 相關聯角色的實體。當您在信任政策中包含條件陳述式時，可以將角色限制到特定 GitHub 組織、儲存庫或分支。您可以使用具有字符條件運算子的條件索引鍵 `token.actions.githubusercontent.com:sub` 來限制存取權。建議您將條件限制為一組特定的儲存庫或 GitHub 組織內的分支。如需如何設定 AWS 將 GitHub 的 OIDC 信任為聯合身分的相關資訊，請參閱 GitHub 文件 - 在 Amazon Web Services 中設定 OpenID Connect。

如果您在動作工作流程或 OIDC 政策中使用 GitHub 環境，我們強烈建議將保護規則新增至環境，以提升安全性。使用部署分支和標籤來限制哪些分支和索引標籤可以部署至環境。如需有關使用保護規則設定環境的詳細資訊，請參閱 GitHub 的 Using environments for deployment 文章中的 Deployment branches and tags。

當 GitHub 的 OIDC IdP 是您角色的受信任主體時，IAM 會檢查角色信任政策條件以驗證條件索引鍵 `token.actions.githubusercontent.com:sub` 確實存在，且其值不單是萬用字元 (* 和 ?) 或 null。IAM 會在建立或更新信任政策時執行此檢查。如果條件索引鍵 `token.actions.githubusercontent.com:sub` 不存在，或者鍵值不滿足上述值標準，請求將失敗且會傳回錯誤。

> **Important**
>
> 如果您未將條件金鑰限制`token.actions.githubusercontent.com:sub`為特定組織或儲存庫，則來自您控制範圍之外的組織或儲存庫的 GitHub 動作可以擔任與您 AWS 帳戶中的 GitHub IAM IdP 相關聯的角色。

下列範例信任政策會限制存取已定義的 GitHub 組織、儲存庫和分支。下列範例中，條件索引鍵 `token.actions.githubusercontent.com:sub` 值是 GitHub 所記錄的預設主旨值格式。

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

下列範例條件會限制存取已定義的 GitHub 組織和儲存庫，但會授予對儲存庫內任何分支的存取權。

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}
```

下列範例條件會限制存取已定義的 GitHub 組織中的任何儲存庫或分支。我們建議您將條件索引鍵 `token.actions.githubusercontent.com:sub` 限制為特定值，此特定值可限制從您的 GitHub 組織內部存取 GitHub 動作。

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
  }
```

```
}
```

如需有關可用於政策中條件檢查的 OIDC 聯合身分索引鍵的詳細資訊，請參閱AWS OIDC 聯合的可用金鑰。

為 SAML 2.0 聯合身分建立角色 (主控台)

您可以使用 SAML 2.0 聯合，而不是在 中建立 IAM 使用者。 AWS 帳戶使用身分提供者 (IdP)，您可以在 外部管理您的使用者身分， AWS 並授予這些外部使用者身分存取您帳戶中 AWS 資源的許可。如需有關聯合身分與身分提供者的詳細資訊，請參閱 身分提供者和 的聯合身分 AWS。

> **Note**
>
> 若要改善聯合彈性，建議您將 IdP 和 AWS 聯合設定為支援多個 SAML 登入端點。如需詳細資訊，請參閱 AWS 安全部落格文章如何使用區域 SAML 端點進行容錯移轉。

建立適用於 SAML 的角色的先決條件

您必須先完成以下先決條件步驟，然後才能建立用於 SAML 2.0 聯合身分的角色。

準備建立用於 SAML 2.0 聯合身分的角色

1. 在為以 SAML 為基礎的聯合身分建立角色之前，必須在 IAM 中建立 SAML 提供者。如需詳細資訊，請參閱 在 IAM 中建立 SAML 身分提供者。
2. 為已進行 SAML 2.0–身分驗證的使用者要擔任的角色準備政策。正如任何角色一樣，SAML 聯合身分的角色含有兩項政策。其中一項是角色信任政策，指定擔任該角色的對象。另一個是 IAM 許可政策，指定允許或拒絕 SAML 聯合主體存取 AWS 的動作和資源。

   當您為您的角色建立信任政策時，必須使用三個值，以確保只有您的應用程式可擔任該角色：

   - 對於 `Action` 元素，可使用 `sts:AssumeRoleWithSAML` 動作。
   - 如需 `Principal` 元素，請使用字串 `{"Federated":`*`ARNofIdentityProvider`*`}`。將 *`ARNofIdentityProvider`* 取代為您在 Step 1 中建立的 SAML 身分提供者 ARN。
   - 對於 `Condition` 元素，請使用 `StringEquals` 條件來測試 SAML 回應中的 `saml:aud` 屬性是否與登入主控台時瀏覽器顯示的 URL 相符。此登入端點 URL 是身分提供者的 SAML 收件人屬性。您可以在特定區域中包含登入 URLs。 AWS 建議使用區域端點而非全域端點來改善聯合彈性。如需可能 *####*值清單，請參閱AWS 登入端點中的區域欄。

如果需要 SAML 加密，登入 URL 必須包含指派給 SAML 提供者的唯一識別碼 AWS 。可以透過在 IAM 主控台中選取身分提供者來顯示詳細資訊頁面，以檢視唯一識別碼。

`https://region-code.signin.aws.amazon.com/saml/acs/IdP-ID`

以下範例信任政策是專為 SAML 聯合身分使用者設計的政策：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRoleWithSAML",
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:saml-provider/PROVIDER-
NAME"
        },
        "Condition": {
            "StringEquals": {
                "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
            }
        }
    }
}
```

將主體 ARN 取代為您在 IAM 中所建立的 SAML 提供者的實際 ARN。它會有自己的帳戶 ID 和提供者名稱。

建立 SAML 的角色

完成必要步驟後，您可以建立以 SAML 為基礎的聯合身分角色。

若要為以 SAML 為基礎的聯合身分建立角色

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 選擇 SAML 2.0 federation (SAML 2.0 聯合身分) 角色類型。

4. 針對 Select a SAML provider (選擇 SAML 提供者)，選擇您角色的提供者。
5. 選擇 SAML 2.0 存取層級的方法。

   - 選擇僅允許程式設計存取，以建立可從 AWS API 或 以程式設計方式擔任的角色 AWS CLI。
   - 選擇允許程式設計和 AWS 管理主控台 存取，以建立可以程式設計方式從 擔任的角色 AWS 管理主控台。

   同時建立類似的角色，但也可以從主控台所擔任的角色包括具有特定條件的信任政策。該條件明確確保可將 SAML 對象 (`SAML:aud` 屬性) 設定為 SAML 提供者的 AWS 登入端點。
6. 定義屬性的程序因存取類型而異。

   - 如果要為程式設計存取建立角色，請從 Attribute (屬性) 清單選取屬性。然後，在 Value (值) 方塊中，輸入包含在角色中的值。這會限制從身分提供者存取使用者的角色，其身分提供者的 SAML 身分驗證回應 (聲明) 包括您指定的屬性。您必須至少指定一個屬性，以確保您的角色僅限於組織中的一部分使用者。
   - 如果您要為程式設計和 AWS 管理主控台 存取建立角色，登入端點區段會定義您的瀏覽器在登入主控台時顯示的 URL。此端點是身分提供者的 SAML 收件人屬性，會映射至 `saml:aud` 內容索引鍵。如需詳細資訊，請參閱為身分驗證回應設定 SAML 聲明。

     a. 選擇區域端點或非區域端點。建議使用多個區域 SAML 登入端點，以便改善聯合身分彈性。

     b. 針對區域，選擇 SAML 供應商支援 AWS 登入的區域。

     c. 若要讓登入 URLs 包含唯一識別符，請選取登入端點是否包含 AWS 指派給 SAML 身分提供者的唯一識別符。加密的 SAML 聲明需要此選項。如需詳細資訊，請參閱SAML 2.0 聯合身分。
7. 若要向信任政策新增更多與屬性相關的條件，請選擇 Condition (optional) (條件 (選用))，並選取其他條件，然後指定值。

   > **Note**
   >
   > 該清單包含最常用的 SAML 屬性。IAM 支援可用於建立條件的其他屬性。如需所支援屬性的清單，請參閱 SAML 聯合身分的可用金鑰。如果您需要不在清單中的受支援 SAML 屬性的條件，您可以手動新增該條件。若要這麼做，請在建立角色後編輯信任政策。
8. 檢閱您的 SAML 2.0 信任資訊，然後選擇 Next (下一步)。

9. IAM 包含您帳戶中 AWS 受管和客戶受管政策的清單。選取用於許可政策的政策，或者選擇 Create policy (建立政策) 以開啟新的瀏覽器標籤，並從頭建立新的政策。如需詳細資訊，請參閱建立 IAM 政策。在您建立政策後，關閉該標籤並返回您的原始標籤。選取您希望 SAML 聯合身分使用者具有之許可政策旁的核取方塊。如果您希望，您目前可以不選取政策，稍後再將政策連接到角色。角色預設沒有任何許可。
10. (選用) 設定許可界限。這是進階功能。

    開啟 Permissions boundary (許可界限) 區段，並選擇 Use a permissions boundary to control the maximum role permissions (使用許可界限來控制角色許可上限)。選取用於許可界限的政策。
11. 選擇下一步。
12. 選擇下一步：檢閱。
13. 在 Role name (角色名稱) 中，輸入角色名稱。角色名稱在您的 中必須是唯一的 AWS 帳戶。它們不區分大小寫。例如，您無法建立名為 **PRODROLE** 和 **prodrole** 的角色。由於其他 AWS 資源可能會參考角色，因此您無法在建立角色之後編輯角色的名稱。
14. (選用) 在 Description (說明) 中，輸入新角色的說明。
15. 在 Step 1: Select trusted entities (步驟 1：選取受信任的實體) 或者 Step 2: Add permissions (步驟 2：新增許可) 區段中選擇 Edit (編輯)，可編輯使用案例和角色許可。
16. (選用) 藉由連接標籤作為鍵值對，將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
17. 檢閱角色，然後選擇 Create role (建立角色)。

在您建立角色之後，您透過 AWS的相關資訊設定您的身分提供者軟體來完成 SAML 信任。此資訊包含您希望 SAML 聯合身分使用者使用的角色。這是指在您的 IdP 和 AWS之間設定依賴方信任。如需詳細資訊，請參閱使用依賴方信任設定您的 SAML 2.0 IdP 並新增宣告。

使用自訂信任政策建立角色

您可以建立自訂信任政策來委派存取權，並允許其他人在您的 中執行動作 AWS 帳戶。如需詳細資訊，請參閱建立 IAM 政策。

如需有關如何使用角色委派許可的資訊，請參閱角色術語和概念。

使用自訂信任政策 (主控台) 建立 IAM 角色

您可以使用 AWS 管理主控台 來建立 IAM 使用者可以擔任的角色。例如，假設您的組織有多個 AWS 帳戶 來隔離開發環境與生產環境。如需有關建立可讓開發帳戶中的使用者存取生產帳戶中資源的角色的高階資訊，請參閱使用不同的開發和生產帳戶的範例方案。

使用自訂信任政策 (主控台) 建立角色

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在主控台的導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 選擇 Custom trust policy (自訂信任政策) 角色類型。
4. 在 Custom trust policy (自訂信任政策) 區段中，輸入或貼上角色的自訂信任政策。如需詳細資訊，請參閱建立 IAM 政策。
5. 解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。
6. (選用) 設定許可界限。這是進階功能，可用於服務角色，而不是服務連結的角色。

   開啟 Permissions boundary (許可界限) 區段，並選擇 Use a permissions boundary to control the maximum role permissions (使用許可界限來控制角色許可上限)。IAM 包含您帳戶中 AWS 受管和客戶受管政策的清單。選取用於許可界限的政策。
7. 選擇下一步。
8. 針對 Role name (角色名稱)，服務會定義角色名稱自訂程度。如果服務定義角色名稱，則無法編輯此選項。在其他情況下，服務可能會定義角色的字首，並且可允許您輸入選用後綴。有些服務可讓您指定角色的完整名稱。

   如有可能，請輸入角色名稱或角色名稱後綴。角色名稱在您的 中必須是唯一的 AWS 帳戶。它們不區分大小寫。例如，您無法建立名為 **PRODROLE** 和 **prodrole** 的角色。由於其他 AWS 資源可能會參考角色，因此您無法在建立角色之後編輯角色的名稱。
9. (選用) 在 Description (說明) 中，輸入新角色的說明。
10. (選用) 在步驟 1：選取信任的實體或步驟 2：新增許可區段中選擇編輯，以編輯角色的自訂政策和許可。
11. (選用) 藉由連接標籤作為鍵值對，將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
12. 檢閱角色，然後選擇 Create role (建立角色)。

委派存取權限的政策範例

下列範例示範如何允許或授予 AWS 帳戶 存取另一個 中的資源 AWS 帳戶。若要了解如何使用這些範例 JSON 政策文件來建立 IAM 政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

主題

- 使用 角色將存取權委派給另一個資源的資源 AWS 帳戶

- 使用政策將存取權限委託給服務
- 使用以資源為基礎的政策來指派存取另一個帳戶中的 Amazon S3 儲存貯體的權限。
- 使用以資源為基礎的政策來指派存取另一個帳戶中的 Amazon SQS 佇列的權限。
- 當拒絕存取帳戶時，不得委託存取

使用 角色將存取權委派給另一個資源的資源 AWS 帳戶

如需示範如何使用 IAM 角色授予某個帳戶中使用者存取另一個帳戶中 AWS 資源的教學課程，請參閱 IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權。

> **Important**
>
> 您可以在角色信任政策的 `Principal` 元素中包含特定角色或使用者的 ARN。當您儲存政策時， 會將 ARN AWS 轉換為唯一的委託人 ID。如果有人希望藉由刪除並重新建立角色或使用者來提升特權，這麼做可有助於減輕此類風險。您通常不會在主控台中看到此 ID，因為在顯示信任政策時還會反向轉換回 ARN。不過，如果您刪除角色或使用者，則關係會中斷。即使重新建立使用者或角色，您的政策都不再適用，因為它不符合信任政策中所儲存的主體 ID。發生這種情況時，主體 ID 會顯示在主控台中，因為 AWS 無法再將其對應回 ARN。結果是，如果您刪除並重新建立了信任政策的 `Principal` 元素所引用的使用者或角色，您必須編輯角色來替換 ARN。當您儲存政策時，它會轉換為新的主體 ID。

使用政策將存取權限委託給服務

以下範例顯示一個可以連接到角色的政策。此政策可讓 Amazon EMR 和 這兩個服務 AWS Data Pipeline擔任該角色。然後服務可以執行指派給角色的許可政策所授予的任何任務 (不顯示)。若要指定多個服務主體，不用指定兩個 `Service` 元素；您可以只使用一個該元素。實際上，您可以將一組多個服務主體做為單一 `Service` 元素的值。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
```

```
            "elasticmapreduce.amazonaws.com",
            "datapipeline.amazonaws.com"
          ]
        },
        "Action": "sts:AssumeRole"
      }
    ]
}
```

使用以資源為基礎的政策來指派存取另一個帳戶中的 Amazon S3 儲存貯體的權限。

在本範例中，帳戶 A 使用資源類型政策 (一個 Amazon S3 儲存貯體政策) 授予帳戶 B 完全存取帳戶 A 的 S3 儲存貯體。然後，帳戶 B 建立一個 IAM 使用者政策，向帳戶 B 中的一個使用者授予針對帳戶 A 的儲存貯體的存取權限。

帳戶 A 中的 S3 儲存貯體政策可能與以下政策類似。在本範例中，帳戶 A 的 S3 儲存貯體的名稱為 amzn-s3-demo-bucket，而帳戶 B 的帳戶號碼為 111122223333。它在帳戶 B 中未指定任何單一使用者或群組，僅指定帳戶本身。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

或者，帳戶 A 可使用 Amazon S3 存取控制清單 (ACL) 來授予帳戶 B 存取 S3 儲存貯體或某個儲存貯體內的單一物件。這種情況下，唯一改變的是帳戶 A 授權存取帳戶 B 的方式。如此範例的下一個部分所述，帳戶 B 仍使用一個政策委託針對帳戶 B 中的 IAM 群組的存取許可。如需有關控制 S3 儲存貯體和物件存取的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的存取控制。

帳戶 B 的管理員可能建立以下政策範例。該政策向帳戶 B 中的群組或使用者授予讀取存取許可。前一政策向帳戶 B 授予存取許可。但是，除非組或使用者政策明確授予資源存取許可，否則帳戶 B 中的單個群組和使用者不能存取資源。此政策中的許可只能是上述跨帳戶政策中的許可的一個子集。相比於帳戶 A 在第一個政策中授予帳戶 B 的許可，帳戶 B 無法向其群組和使用者授予更多許可。在此政策中，將明確定義 `Action` 元素以僅允許 `List` 操作，而且此政策的 `Resource` 元素與由帳戶 A 執行的儲存貯體政策的 `Resource` 配對。

為執行此政策，帳戶 B 使用 IAM 將其連接到帳戶 B 中的適用使用者 (或群組)。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

使用以資源為基礎的政策來指派存取另一個帳戶中的 Amazon SQS 佇列的權限。

在下列範例中，帳戶 A 具有 Amazon SQS 佇列，而此佇列使用連接至佇列的資源類型政策向帳戶 B 授予佇列存取權。然後，帳戶 B 使用 IAM 群組原則委派帳戶 B 中群組的存取權。

以下範例佇列政策為帳戶 B 授予許可，以對帳戶 A 中名為 queue1 的佇列執行 `SendMessage` 和 `ReceiveMessage` 動作，但只能在 2014 年 11 月 30 日中午 12:00 至下午 3:00 之間執行該動作。Account B 的帳戶編號為 1111-2222-3333。帳戶 A 使用 Amazon SQS 執行此政策。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
```

```
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

帳戶 B 向帳戶 B 中的組委託存取許可的政策可能類似於以下範例。帳戶 B 使用 IAM 將此政策連接到群組 (或使用者)。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
  }
}
```

在前述 IAM 使用者政策範例中，帳戶 B 使用萬用字元授權其使用者存取針對帳戶 A 的佇列的所有 Amazon SQS 操作。但是帳戶 B 可委託的範圍僅限於帳戶 B 被授權存取的範圍。擁有第二個政策的帳戶 B 群組只能在 2014 年 11 月 30 日中午至下午 3:00 之間存取該佇列。根據帳戶 A 的 Amazon SQS 佇列政策的定義，使用者只能執行 SendMessage 與 ReceiveMessage 操作。

當拒絕存取帳戶時，不得委託存取

如果另一個帳戶明確拒絕存取使用者的父帳戶，則 AWS 帳戶 無法將存取權委派給另一個帳戶的 資源。拒絕將傳播到該帳戶內的所有使用者，無論使用者的現有政策是否授予這些使用者存取許可。

例如，帳戶 A 編寫了一個針對其帳戶中 S3 儲存貯體的儲存貯體政策，其中明確拒絕了帳戶 B 存取帳戶 A 的儲存貯體。但帳戶 B 編寫了一個 IAM 使用者政策，其中對帳戶 B 中的一個使用者授予了對帳戶 A 的儲存貯體的存取權限。應用於帳戶 A 的 S3 儲存貯體的「明確拒絕」將傳播到帳戶 B 中的使

用者。它會覆蓋用於對帳戶 B 中使用者授予存取權限的 IAM 使用者政策 (有關如何計算許可的詳細資訊，請參閱 政策評估邏輯)。

帳戶 A 的儲存貯體政策可能與下列政策類似。在本範例中，帳戶 A 的 S3 儲存貯體的名稱為 amzn-s3-demo-bucket，而帳戶 B 的帳戶號碼為 1111-2222-3333。帳戶 A 使用 Amazon S3 執行此政策。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
  }
}
```

此「明確拒絕」將覆蓋帳戶 B 中所有提供帳戶 A 中 S3 儲存貯體存取許可的政策。

使用角色管理員自動建立角色

許多 AWS 服務依賴 IAM 角色代表您執行動作。IAM 角色管理員是選用的帳戶設定，會自動佈建這類角色，因此您不需要自行設定。角色管理員旨在協助您開始建置 AWS 服務，而無需額外的手動 IAM 組態。

角色管理員可在此頁面的 the section called “AWS 支援角色管理員的 服務主控台”中列出的特定 AWS 服務主控台中使用。

角色管理員的運作方式

啟用角色管理員時，它會提供適用於 服務和資源的角色，做為相關主控台工作流程的一部分。支援的服務主控台會指出角色管理員何時可用。接受角色管理員提供的角色不需要其他動作。例如，當您在主控台中建立 AWS Lambda 函數時，Lambda 預設會使用角色管理員來建立函數的執行角色。若要使用不同的角色，請選擇現有的角色或手動建立新的角色。

角色管理員的運作方式是套用角色範本，即角色的信任政策和服務和使用案例許可的定義。若要進一步了解每個服務使用的範本，請參閱 角色範本概觀。已定義任務的範本只會授予任務所需的動作。開放

式工作的範本，例如執行您自己的程式碼，會授予 AWS APIs廣泛存取權，因此您可以自行建置 ，而無需授予每個服務的存取權。

由角色管理員建立的 IAM 角色的行為類似於您建立的任何其他角色。您可以檢視、編輯或刪除它們。每個角色也會記錄其來自的角色範本，因此您可以識別角色管理員建立的角色。當角色管理員建立角色時 AWS CloudTrail 會記錄 ，以便您可以稽核建立每個角色的時間和方式。角色管理員角色建立會在 CloudTrail 中顯示為 AcquireRole 事件。

先決條件

啟用角色管理員需要 `iam:PutAccountProperties`和 `iam:CreateServiceLinkedRole`許可。停用角色管理員需要 `iam:PutAccountProperties`許可。檢視目前的設定需要 `iam:GetAccountProperties`許可。 AWS 受管政策`IAMFullAccess`包含所有這些許可。如需角色管理員所需的所有許可，請參閱 管理角色管理員的存取權。

如何啟用和停用角色管理員

您可以使用 AWS 管理主控台 AWS CLI或 AWS API 啟用或停用角色管理員。角色管理員是由`RoleManager/Enabled`您使用 `PutAccountProperties` API 設定的帳戶層級屬性 所控制。的值會`true`啟用角色管理員； 會將其`false`停用。

> **Note**
>
> 對於大多數 AWS 帳戶，角色管理員預設為停用。但是，如果您使用我們的新 AWS 體驗建立帳戶，則角色管理員預設為啟用，除非您啟用進階功能，否則無法停用。如需詳細資訊，請參閱啟用進階 AWS 功能。

啟用或停用角色管理員 （主控台）

您可以在 IAM 主控台的帳戶設定頁面上啟用或停用角色管理員。此頁面也會顯示帳戶是否已啟用角色管理員。

1. 登入 AWS 管理主控台 並開啟 IAM 主控台。
2. 在導覽窗格中，選擇帳戶設定。
3. 在角色管理員區段中，選擇啟用或停用。

當您停用角色管理員時，不會刪除角色管理員建立的角色。

啟用或停用角色管理員 (AWS CLI)

將 RoleManager/Enabled 屬性設定為 true 以啟用角色管理員，或false將其停用。

- 若要啟用角色管理員，請執行 put-account-properties：

```
aws iam put-account-properties --properties RoleManager/Enabled=true
```

- 若要停用角色管理員，請執行 put-account-properties：

```
aws iam put-account-properties --properties RoleManager/Enabled=false
```

- 若要檢視帳戶是否已啟用角色管理員，請執行 get-account-properties：

```
aws iam get-account-properties
```

啟用或停用角色管理員 (AWS API)

- 若要啟用或停用角色管理員，請呼叫 PutAccountProperties，並將 RoleManager/Enabled 屬性設定為 true或 false。
- 若要檢視目前的設定，請呼叫 GetAccountProperties。

如需詳細資訊，請參閱《IAM API 參考》中的 PutAccountProperties 和 GetAccountProperties。

AWS 支援角色管理員的 服務主控台

角色管理員支援為下列 AWS 服務主控台建立角色：

- AWS 備份
- AWS CloudFormation
- Amazon CloudWatch
- Amazon EKS
- AWS Elastic Beanstalk
- Amazon EventBridge
- AWS Glue
- AWS Lambda

- Amazon RDS
- Amazon SageMaker Unified Studio
- AWS Secrets Manager
- AWS 步驟函數

AWS 會隨著時間繼續新增服務支援。若要進一步了解每個服務使用的角色範本，請參閱 角色範本概觀。

相關資訊

- 管理角色管理員的存取權
- 將最低權限許可套用至自動建立的角色
- 角色範本概觀

管理角色管理員的存取權

您可以在兩個層級控制對角色管理員的存取。IAM 許可會決定帳戶中的哪些人員可以啟用角色管理員，以及誰可以使用其提供的角色。在組織中，服務控制政策 SCPs) 會決定成員帳戶可以執行的動作。

角色管理員的許可

若要啟用或停用角色管理員，您需要 `iam:PutAccountProperties`許可。 AWS 受管政策`IAMFullAccess`包含它。

若要使用角色管理員，使用者需要角色管理員代表他們執行之動作的 IAM 許可。角色管理員透過 `AcquireRole` API 提供角色，該 API 會根據使用者的許可評估每個基礎 IAM 動作：

- 如果符合範本的角色已存在於帳戶中， 會`AcquireRole`傳回該角色。此動作需要 `iam:GetRole` 許可。
- 如果不存在相符的角色， 會從範本`AcquireRole`建立一個。這需要 `iam:CreateRole`許可，加上範本定義內容的許可：`iam:PutRolePolicy`如果範本新增內嵌政策，以及`iam:AttachRolePolicy`如果範本連接受管政策。

具有完整 IAM 存取權的使用者擁有角色管理員所需的所有許可。若要查看特定範本所需的確切動作，請使用 擷取範本`GetRoleTemplateVersion`並檢閱其政策。

當角色管理員建立角色時， AWS CloudTrail 會將建立記錄為單一AcquireRole事件。事件會顯示呼叫 操作的人員、使用的角色範本和參數值，以及建立的角色。

下列範例顯示該角色管理員代表擔任角色 的使用者呼叫 AcquireRole操作Admin，使用PowerUserRoleTemplate角色範本建立角色 PowerUserRole。


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123EXAMPLE:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "accountId": "123456789012",
        "accessKeyId": "ASIAWZYMLEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2026-08-11T18:18:50Z",
                "mfaAuthenticated": "false"
            }
        },
        "onBehalfOf": {
            "userId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "identityStoreArn": "arn:aws:identitystore::111122223333:identitystore/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
        }
    },
    "eventTime": "2026-08-11T18:23:00Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "AcquireRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/151.0.0.0 Safari/537.36",
    "requestParameters": {
```

```
        "templateArn": "arn:aws:iam::aws:role-template/iam.amazonaws.com/
PowerUserRoleTemplate:1",
        "templateMinorVersion": 0,
        "replacementValues": {
            "AWSServiceName": {
                "values": [
                    "iam.amazonaws.com"
                ]
            }
        }
    },
    "responseElements": {
        "role": {
            "path": "/",
            "roleName": "PowerUserRole",
            "roleId": "AROA987EXAMPLE",
            "arn": "arn:aws:iam::123456789012:role/PowerUserRole",
            "createDate": "2026-08-11T18:23:00Z",
            "assumeRolePolicyDocument": "%7B%22Version%22:%222012-10-17%22,
%22Statement%22:%5B%7B%22Effect%22:%22Allow%22,%22Principal%22:%7B%22Service%22:
%22iam.amazonaws.com%22%7D,%22Action%22:%22sts:AssumeRole%22%7D%5D%7D"
        }
    },
    "requestID": "fde3dac0-73ac-491c-876b-EXAMPLE89c57",
    "eventID": "c93332cf-1429-4aee-bcaa-EXAMPLE3790d",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```


組織中的控制角色管理員

您可以使用 AWS Organizations 服務控制政策 (SCP) 來控制整個組織的角色管理員。下列範例示範如何僅封鎖啟用角色管理員、封鎖角色建立，或封鎖以範本為基礎的角色建立。

這些範例使用拒絕清單策略，這表示您也需要 FullAWSAccess或其他政策，以允許連接到組織實體的動作。您仍然需要使用身分型政策將許可授予委託人。

若要防止成員帳戶透過主控台或 API 啟用角色管理員，請拒絕 iam:PutAccountProperties許可。當您直接透過 Organizations 建立新帳戶時，我們建議您使用此政策。

```
{
    "Sid": "DenyRoleManagerEnablement",
    "Effect": "Deny",
    "Action": "iam:PutAccountProperties",
    "Resource": "*",
    "Condition": {
        "ForAnyValue:StringEquals": {
            "iam:AccountPropertyNamespaces": "RoleManager"
        }
    }
}
```

封鎖使用角色管理員的存取權

若要封鎖角色管理員的使用，請使用iam:RoleTemplateARN內容索引鍵，該索引鍵存在於從範本建立角色的 IAM 動作上。下列政策會拒絕iam:RoleTemplateARN內容索引鍵具有非空值的所有 IAM 動作。此政策不會阻止建立所有角色，只會封鎖透過角色管理員建立的角色，以及透過 SDK 建立以範本為基礎的角色。

```
{
    "Sid": "DenyTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": "iam:*",
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": false
        }
    }
}
```

封鎖角色建立，但透過角色管理員除外

若要僅允許透過角色管理員建立角色，請使用下列政策來封鎖非角色管理員角色的建立。此政策仍然允許直接透過 SDK 建立以範本為基礎的角色。

```
{
    "Sid": "DenyNonTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": [
        "iam:CreateRole",
        "iam:AttachRolePolicy",
        "iam:PutRolePolicy",
        "iam:PutRolePermissionsBoundary",
        "iam:TagRole"
    ],
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": true
        }
    }
}
```

相關資訊

- 使用角色管理員自動建立角色
- 將最低權限許可套用至自動建立的角色
- 角色範本概觀

將最低權限許可套用至自動建立的角色

在服務主控台中建立資源時，角色管理員會提供資源所需的 IAM 角色。對於某些任務，例如執行您自己的程式碼， AWS 無法事先判斷需要哪些許可。因此，角色可以允許超過資源所需的許可。在知道資源需要哪些許可之後，您可以減少角色的許可以進行比對。

停用角色管理員不會影響角色管理員先前建立的角色。您可以停用角色管理員而不減少任何許可，也可以減少角色的許可而不停用角色管理員。

何時套用最低權限許可

僅授予角色其資源所需的許可是符合最低權限原則的安全最佳實務。對於大多數角色，角色管理員會建立具有良好範圍許可的角色。對於某些角色，例如運算資源或雲端基礎設施管理的角色，它會建立具有廣泛許可的角色。

當您準備好開始縮小角色管理員建立的角色範圍時，建議您從與您最敏感的工作負載和資源相關聯的角色開始，例如處理 PII 或生產資料的任何內容。如需詳細資訊，請參閱 為最低權限許可做好準備。

如何分析未使用的存取權

在停用角色管理員之前， AWS 建議您IAM Access Analyzer用來在帳戶中建立未使用的存取分析器。分析器會將允許角色執行的動作與其已執行的動作進行比較，並報告未使用的許可。分析器會檢閱您帳戶中的 IAM 角色，包括您自行建立的角色。

問題清單的顯示位置取決於分析器的提供方式：

- 角色管理員提供的分析器：當 AWS 為您的帳戶提供未使用的存取分析器時（請參閱本節中的備註），問題清單會出現在 IAM 主控台的角色頁面上。對於每個角色，您可以查看未使用的許可數量並開啟建議。使用下列程序，從角色頁面減少角色的許可。停用角色管理員後，帳戶設定頁面會提供檢視角色按鈕，以開啟角色頁面，其中會顯示每個角色的問題清單。
- 由您建立的分析器：當您建立自己的未使用存取分析器時，問題清單和建議會顯示在 IAM Access Analyzer 主控台中。該體驗不會由角色管理員變更。若要檢閱和處理未使用的存取問題清單，請參閱 IAM Access Analyzer 問題清單。

> **Note**
>
> 如果為使用新 AWS 體驗建立的帳戶 AWS 啟用了角色管理員，則不需要在第一次停用角色管理員時建立分析器。在您啟用進階功能並停用角色管理員後， AWS 會免費提供未使用的存取分析器 90 天。分析器可讓您查看未使用的角色許可和政策縮小範圍建議，讓您可以將帳戶中的角色更新為最低權限，包括角色管理員建立的角色。

先決條件

若要檢閱 Access Analyzer 調查結果，您需要執行 `access-analyzer:ListFindings`動作的許可。若要套用建議，您需要在角色`iam:DetachRolePolicy`上執行 `iam:CreatePolicy`、`iam:AttachRolePolicy`和 的許可。

減少 主控台中角色的許可（角色管理員提供的分析器）

這些步驟會在為您的帳戶 AWS 提供未使用的存取分析器時套用。如果您建立了自己的分析器，請改為在 IAM Access Analyzer 主控台中檢閱並套用建議。請參閱 IAM Access Analyzer 調查結果。

1. 登入 AWS 管理主控台 並開啟 IAM 主控台。
2. 在導覽窗格中，選擇角色。
3. 在角色頁面的調查結果欄中，為您要減少其許可的角色選擇未使用的許可連結。

4. 在減少角色許可頁面上，檢閱針對角色報告的未使用許可。
5. 比較建議的許可與角色目前的許可，並檢閱建議移除的動作。
6. 選擇進行變更。

建議許可的運作方式

根據未使用的存取分析器調查結果，IAM 會建立一或多個客戶受管政策，其中包含建議的許可。IAM 在分離舊政策之前連接新政策，因此角色會在整個變更過程中保留其許可。

例如，請考慮 AWS Lambda 執行角色。IAM Access Analyzer 會報告哪些角色的許可未使用，並建議適用於角色所執行動作的政策，例如將日誌寫入 Amazon CloudWatch Logs。您可以針對函數執行的操作檢閱建議，確認它涵蓋函數所需的一切，並加以套用。然後，角色只允許函數所需的動作。

> **Important**
>
> 建議是以過去 30 天的活動為基礎。資源在較長時間範圍內不常使用的許可，例如用於每季任務的許可，可能會顯示為未使用。套用工作負載之前，請針對工作負載的需求檢閱建議。
> 在您變更角色的政策或信任政策之後，角色管理員將不再管理該角色。該角色不會取得最初用於建立角色的角色範本的任何版本變更。

IAM 會限制您可以連接到角色的受管政策數量。如果套用建議會超過該限制，主控台會通知您，而不會套用變更。如需此配額的相關資訊，請參閱 IAM AWS STS 和配額。

相關資訊

- 使用角色管理員自動建立角色
- 管理角色管理員的存取權
- 角色範本概觀

角色範本概觀

角色範本是用來建立新角色的角色藍圖。範本會設定角色的信任政策和許可，以便您取得服務的ready-to-use角色，而無需自行撰寫政策。

角色管理員使用 AWS受管角色範本，但您也可以在沒有角色管理員的情況下使用角色範本。此頁面說明角色範本的運作方式，並列出每個服務的範本。

角色範本的運作方式

角色範本包含信任政策和許可，並具有 Amazon Resource Name (ARN)。信任政策會定義哪些委託人可以擔任該角色。許可定義角色可以執行的操作。範本版本不可變。當 AWS 更新範本時，它會發佈新版本，而不是變更現有的版本。

角色範本可能會接受其他參數，以自訂角色的屬性和許可並加以限制。例如，範本可能RoleName針對其建立的角色採用 ，或資源識別符，例如 bucketName。當角色管理員在主控台中建立角色時，會將參數套用至角色的許可和信任政策。當您直接使用 IAM API 從範本建立角色時，您會提供這些參數。

若要從範本建立角色，請使用 iam:AcquireRole API 搭配範本的 ARN，並傳遞範本所需的參數。

角色範本的定義可能會在版本之間變更。若要查看範本目前的信任政策和許可，iam:GetRoleTemplateVersion請使用範本 ARN. AWS recommends 在依賴範本之前檢查範本，尤其是授予廣泛存取權的範本。

角色管理員如何使用 範本

在支援的服務主控台中建立資源時，角色管理員會從範本中提供符合服務和使用案例的角色。角色管理員會選擇範本並為您提供參數。結果是您會收到呼叫 的相同角色iam:AcquireRole。若要啟用角色管理員，請參閱 管理角色管理員的存取權。

角色範本目錄

下表列出角色管理員支援之每項服務的角色範本，以及其採用的範本 ARN 和參數。使用 ARN 從範本建立角色，或檢閱該服務的角色管理員授予。服務會依字母順序列出。

服務	角色範本	範本 ARN	Parameters
AWS 備份	AWSBackupDefaultServiceRoleTemplate	arn:aws:iam::aws:role-template/backup.amazonaws.com/AWSBackupDefaultServiceRoleTemplate:1	accountId、RoleName
AWS CloudFormation	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName、RoleName
Amazon EKS	AmazonEKSAutoClusterRoleTemplate	arn:aws:iam::aws:role-template/eks.amazonaws.com/	RoleName

服務	角色範本	範本 ARN	Parameters
		AmazonEKSAutoClusterRoleTemplate:1	
Amazon EKS	AmazonEKSAutoNodeRoleTemplate	arn:aws:iam::aws:role-template/ec2.amazonaws.com/AmazonEKSAutoNodeRoleTemplate:1	RoleName
AWS Elastic Beanstalk	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName、RoleName
Amazon EventBridge	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName、RoleName
AWS Glue	AWSGlueCrawlerS3PathsRoleTemplate	arn:aws:iam::aws:role-template/glue.amazonaws.com/AWSGlueCrawlerS3PathsRoleTemplate:1	RoleName、accountId、s3path
AWS Glue	AWSGluePowerUserRoleTemplate	arn:aws:iam::aws:role-template/glue.amazonaws.com/AWSGluePowerUserRoleTemplate:1	accountId、RoleName
AWS Lambda	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName、RoleName
Amazon RDS	AmazonRDSDirectoryServiceIntegrationRoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSDirectoryServiceIntegrationRoleTemplate:1	RoleName、accountId

服務	角色範本	範本 ARN	Parameters
Amazon RDS	AmazonRDS ExportToS 3RoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSExportToS3RoleTemplate:1	RoleName、accountId、bucketName
Amazon RDS	AmazonRDS ImportFro mS3RoleTe mplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSImportFromS3RoleTemplate:1	CUSTOM_KM S_KEY、Rol eName、acc ountId、 bucketNam e、keyAcco untId、key Region、 kmsKeyId、字首
Amazon RDS	AmazonRDS Monitorin gRoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSMonitoringRoleTemplate:1	RoleName、accountId
Amazon RDS	AmazonRDS ProxyRole Template	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSProxyRoleTemplate:2	PROXY_WIT H_CUSTOM_ KEY、PROXY _WITH_DB_ CONNECTIO N、PROXY_W ITH_SECRE T、RoleNam e、account Id、dbReso urceId、db User、keyI d、region、 secretId

服務	角色範本	範本 ARN	Parameters
Amazon RDS	AmazonRDSSecretAccessForDMSRoleTemplate	arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSSecretAccessForDMSRoleTemplate:1	CUSTOM_KMS_KEY、RoleName、accountId、kmsKeyId、region、secretId
Amazon SageMaker Unified Studio	AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate	arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate:1	CMK_ENABLED、RoleName、accountId、keyAccountId、keyRegion、kmsKeyId
Amazon SageMaker Unified Studio	AmazonSageMakerUserIAMPermissiveExecutionRoleTemplate	arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerUserIAMPermissiveExecutionRoleTemplate:1	RoleName、accountId
AWS Secrets Manager	AWSSecretsManagerRotationRoleTemplate	arn:aws:iam::aws:role-template/secretsmanager.amazonaws.com/AWSSecretsManagerRotationRoleTemplate:1	ADMIN_RESOURCE_ENABLED、CMK_ENABLED、RoleName、accountId、adminType、kmsKeyArn、region、resourceType

服務	角色範本	範本 ARN	Parameters
Amazon CloudWatch	PowerUser RoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName、RoleName

相關資訊

- 使用角色管理員自動建立角色
- 管理角色管理員的存取權
- IAM API 參考

IAM 角色管理

在使用者、應用程式或服務可以使用您建立的角色之前，您必須授予切換到該角色的許可。您可以使用連接至群組或使用者的任何政策，來授予必要的許可。本節說明如何授予使用者使用角色的許可。它還說明使用者如何從 AWS 管理主控台、Tools for Windows PowerShell、 AWS Command Line Interface (AWS CLI) 和 `AssumeRole` API 切換到角色。

> **Important**
>
> 當您以程式設計方式而不是在 IAM 主控台中建立角色時，則除了 `RoleName` 之外，您還可以選擇新增高達 512 個字元的 `Path`，該字元長度最多 64 個字元。不過，如果您想要在 中使用具有切換角色功能的角色 AWS 管理主控台，則合併的 `Path`和 `RoleName`不得超過 64 個字元。

主題

- 檢視角色存取
- 根據存取資訊產生政策
- 向使用者授予切換角色的許可
- 授予使用者將角色傳遞至 AWS 服務的許可
- 撤銷 IAM 角色臨時安全憑證
- 更新服務連結角色
- 更新角色信任政策

- 更新角色的許可
- 更新角色的設定
- 刪除角色或執行個體設定檔

檢視角色存取

變更角色的許可之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

根據存取資訊產生政策

您有時可能會對 IAM 實體 (使用者或角色) 授予超出其要求的許可。為了協助您精簡所授予的許可，您可以根據實體的存取活動產生 IAM 政策。IAM Access Analyzer 會檢閱您的 AWS CloudTrail 日誌，並產生政策範本，其中包含實體在指定日期範圍內已使用的許可。您可以使用範本建立具有精細許可的受管政策，然後將其連接至 IAM 實體。如此一來，您只會授予使用者或角色與特定使用案例 AWS 的資源互動所需的許可。若要進一步了解，請參閱 IAM Access Analyzer 政策產生。

向使用者授予切換角色的許可

當管理員建立用於跨帳戶存取的角色時，他們會在擁有角色和資源的帳戶 (信任帳戶) 和包含使用者的帳戶 (可信帳戶) 之間建立信任。若要執行此操作，信任帳戶的系統管理員要在角色的信任政策中將可信的帳戶編號指定為 `Principal`。這可能會允許可信帳戶中的任何使用者擔任該角色。要完成設定，信任帳戶的管理員必須為該帳戶中的特定群組或使用者提供切換到該角色的許可。

要授予許可以切換到角色

1. 身為可信帳戶的管理員，請為使用者建立新政策，或編輯現有政策來新增必要元素。如需詳細資訊，請參閱建立或編輯政策。
2. 然後，選擇您想要如何共享角色資訊：

   - 角色連結：向使用者傳送連結，讓他們進入已填寫所有詳細資訊的 Switch Role (切換角色) 頁面。
   - 帳戶 ID 或別名：為每個使用者提供角色名稱以及帳戶 ID 號碼或帳戶別名。使用者接著前往 Switch Role (切換角色) 頁面，然後手動新增詳細資訊。

   如需詳細資訊，請參閱提供資訊給使用者。

請注意，只有當您做為 IAM 使用者、SAML 聯合角色或 Web 聯合身分角色登入時，才能切換角色。如果您以 AWS 帳戶根使用者登入，則無法切換角色。

> **Important**
>
> 您無法將 中的角色切換 AWS 管理主控台 為需要 ExternalId 值的角色。您只能透過呼叫支援 `ExternalId` 參數的 `AssumeRole` API 來切換到此類角色。

> **備註**
>
> - 本主題討論使用者的政策，因為您最終會向使用者授予完成任務的許可。不過，我們不建議您直接向個別使用者授予許可。當使用者擔任角色時，他們會被指派得到與該角色關聯的許可。
> - 當您在 中切換角色時 AWS 管理主控台，主控台一律會使用您的原始登入資料來授權切換。無論您作為 IAM 使用者、SAML 聯合角色還是 Web 聯合身分角色登入，上述情形均適用。例如，如果您切換到 RoleA，IAM 會使用您的原始使用者或聯合角色憑證確定是否允許您擔任 RoleA。如果您在使用 RoleA 時嘗試切換到 RoleB，仍會使用您的原始使用者或聯合身分角色憑證對您切換到 RoleB 的嘗試進行授權。RoleA 的憑證不會用於此動作。

主題

- 建立或編輯政策
- 提供資訊給使用者

建立或編輯政策

授予使用者擔任角色的許可政策必須包含在下列項目中擁有 `Allow` 效果的陳述式：

- `sts:AssumeRole` 動作
- 在 `Resource` 元素中的角色的 Amazon Resource Name (ARN)。

取得政策的使用者被允許在所列資源上切換角色 (無論是透過群組成員還是直接連接)。

> **Note**
>
> 如果 Resource 已設定為 *，則使用者可以在信任使用者帳戶的任何帳戶中擔任任何角色。(換句話說，角色的信任政策會將使用者的帳戶指定為 Principal)。最佳實務是建議您遵循最低權限原則，並且僅為使用者所需的角色指定完整的 ARN。

以下範例顯示一個政策，該政策僅允許使用者擔任一個帳戶中的角色。此外，政策使用萬用字元 (*) 指定，如果角色名稱以字元 Test 開頭並後跟任何其他字元組合，則使用者只能切換到該帳戶中的角色。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111122223333:role/Test*"
    }
}
```

> **Note**
>
> 角色向使用者授予的許可不會新增到使用者已獲得的許可。當使用者切換到某個角色時，使用者可臨時放棄其原始許可以換取由該角色授予的許可。使用者退出該角色時，將自動恢復原始使用者許可。例如，如果使用者的許可允許使用 Amazon EC2 執行個體，但是角色的許可政策未授予這些許可。在這種情況下，在使用角色的同時，使用者不能使用在主控台中的 Amazon EC2 執行個體。此外，透過 AssumeRole 取得的暫時憑證不可透過程式設計方式使用 Amazon EC2 執行個體。

提供資訊給使用者

建立一個角色並向使用者授予切換為該角色的許可後，您必須提供使用者下列項目：

- 角色的名稱
- 包含角色的帳戶 ID 或別名

您可以向使用者發送使用帳戶 ID 和角色名稱預先配置的連結，以簡化他們的存取操作。完成建立角色精靈後，您可以選取檢視角色橫幅，或在任何跨帳戶啟用之角色的角色摘要頁面上查看角色連結。

您還可使用以下格式來手動建構連結。請用您的帳戶 ID 或別名及角色名稱來替換下列範例中的兩個參數：

```
https://signin.aws.amazon.com/switchrole?
account=your_account_ID_or_alias&roleName=optional_path/role_name
```

我們建議您將使用者導向到 從使用者切換至 IAM 角色 (主控台)，以便向他們演練該過程。若要排除您在擔任角色時可能遇到的常見問題，請參閱 我無法擔任角色。

考量事項

- 如果您以程式設計方式建立角色，則可使用路徑以及名稱來建立角色。如果您這麼做，則必須為您的使用者提供完整的路徑和角色名稱以便在 AWS 管理主控台的 Switch Role (切換角色) 頁面上輸入。例如：`division_abc/subdivision_efg/role_XYZ`。
- 如果您以程式設計方式建立角色，則除了 `RoleName` 外，您還可以新增最長 512 個字元的 `Path`。角色名稱長度上限為 64 個字元。不過，若要搭配 中的切換角色功能使用角色 AWS 管理主控台，合併 `Path`和 `RoleName`不得超過 64 個字元。

- 基於安全考量，您可以檢閱 AWS CloudTrail 日誌，以了解在其中執行動作的人員 AWS。您可以使用角色信任政策中的 `sts:SourceIdentity` 條件金鑰，請求使用者在擔任角色時指定身分。例如，您可以請求 IAM 使用者將自己的使用者名稱指定為其來源身分。這可以協助您判斷哪位使用者在 AWS中執行了特定動作。如需詳細資訊，請參閱`sts:SourceIdentity`。您亦可以使用 `sts:RoleSessionName`，請求使用者在擔任角色時指定工作階段名稱。當不同的主體使用角色時，這可協助您區分角色工作階段。

授予使用者將角色傳遞至 AWS 服務的許可

若要設定許多 AWS 服務，您必須將 IAM 角色傳遞給服務。這樣可讓該服務擔任該角色，並代表您執行動作。對於大多數服務，您只需在設定期間將角色傳遞服務一次，該服務不需要每次都要擔任角色。例如，假設您的應用程式正在 Amazon EC2 執行個體上執行。該應用程式需要暫時性憑證進行身分驗證，需要許可來授權應用程式在 AWS中執行動作。當設定應用程式時，您必須將角色傳遞到 Amazon EC2，與提供使些憑證的執行個體搭配使用。您為在執行個體上執行的應用程式定義許可，做法是將 IAM 政策連接到角色。應用程式在每次需要時擔任該角色，並在角色允許下執行動作。

若要將角色（及其許可）傳遞給 AWS 服務，使用者必須具有將角色傳遞給 服務的許可。這可幫助管理員確定只有核准的使用者可以透過授予許可的角色來設定服務。若要允許使用者將角色傳遞至 AWS 服務，您必須將 PassRole 許可授予使用者的 IAM 使用者、角色或群組。

> **Warning**
>
> - 您只能使用 PassRole許可將 IAM 角色傳遞至共用相同 AWS 帳戶的服務。若要將帳戶 A 中的角色傳遞給帳戶 B 中的服務，您必須先在帳戶 B 中建立可從帳戶 A 擔任該角色的 IAM 角色，接著帳戶 B 中的角色才可以傳遞至該服務。如需詳細資訊，請參閱IAM 中的跨帳戶資源存取。
> - 請勿嘗試透過標記角色，然後使用政策中的 ResourceTag 條件金鑰搭配 iam:PassRole 動作來控制誰可以傳遞角色。這種方法所產生的結果並不可靠。

設定 PassRole 許可時，您應確保使用者傳遞的角色不會具有比您希望使用者具有之許可更多的許可。例如，可能不允許 Alice 執行任何 Amazon S3 動作。如果 Alice 可以將角色傳遞給允許 Amazon S3 動作的服務，則該服務可以在執行任務時代表 Alice 執行 Amazon S3 動作。

當您指定服務連結角色時，您也必須擁有將該角色傳遞到服務的許可。有些服務會自動在您在該服務中執行動作時，在您的帳戶中建立服務連結角色。例如，Amazon EC2 Auto Scaling 會在您第一次建立 Auto Scaling 群組時，為您建立 AWSServiceRoleForAutoScaling 服務連結角色。如果您在建立 Auto Scaling 群組時，嘗試指定服務連結角色並且您沒有 iam:PassRole 許可，則會收到錯誤。如果您沒有明確指定角色，則不需要 iam:PassRole 許可，預設情況下，對該群組執行的所有操作都使用 AWSServiceRoleForAutoScaling 角色。若要了解哪些服務支援服務連結角色，請參閱 AWS 使用 IAM 的 服務。若要了解哪些服務在您於服務中執行動作時會自動建立服務連結角色，請選擇 Yes (是) 連結，並檢視該服務的服務連結角色文件。

使用者可以在任何 API 操作中，以參數方式傳遞角色 ARN，而該操作便是使用角色將許可指派給服務。然後，該服務會檢查該使用者是否擁有 iam:PassRole 許可。若要限制使用者只傳遞核准的角色，您可以使用 IAM 政策陳述式的 Resources 元素來篩選 iam:PassRole 許可。

您可以使用 JSON 政策中的 Condition元素來測試包含在所有請求請求內容中的金鑰值 AWS 。若要進一步了解如何在政策中使用條件索引鍵，請參閱 IAM JSON 政策元素：Condition。iam:PassedToService 條件索引鍵可用來指定能傳遞角色之服務的服務主體。若要進一步了解如何在政策中使用 iam:PassedToService 條件金鑰，請參閱 iam:PassedToService。

範例 1

假設您想要授予使用者在啟動執行個體時，能夠將任何已核准的一組角色傳遞到 Amazon EC2 服務。您需要三種元素：

- 連接到角色的 IAM 許可政策，其決定角色可以做什麼。將許可侷限在角色必須執行的動作，以及角色針對那些動作所需的資源。您可以使用 AWS 受管或客戶建立的 IAM 許可政策。

  JSON

  ```
  {
      "Version":"2012-10-17",
      "Statement": {
          "Effect": "Allow",
          "Action": [ "A list of the permissions the role is allowed to use" ],
          "Resource": [ "A list of the resources the role is allowed to access" ]
      }
  }
  ```

- 角色的「信任政策」，其允許服務擔任角色。例如，您可以將以下信任政策連接到角色及 `UpdateAssumeRolePolicy` 動作。此信任政策可讓 Amazon EC2 使用角色和連接到該角色的許可。

  JSON

  ```
  {
      "Version":"2012-10-17",
      "Statement": {
          "Sid":
   "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
          "Effect": "Allow",
          "Principal": { "Service": "ec2.amazonaws.com" },
         "Action": "sts:AssumeRole"
      }
  }
  ```

- 連接到 IAM 使用者的 IAM 許可政策，其允許使用者僅傳遞已核准的角色。通常是將 `iam:GetRole` 新增至 `iam:PassRole`，讓使用者可以取得要傳遞之角色的詳細資訊。在這個範例中，使用者可以僅傳遞存在於指定的帳戶，名稱開頭為 `EC2-roles-for-XYZ-` 的角色：

  JSON

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

```
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/EC2-roles-for-XYZ-*"
        }
    ]
}
```

現在，使用者可以使用已指派的角色來啟動 Amazon EC2 執行個體。在執行個體上執行的應用程式，可透過執行個體設定檔中繼資料來存取角色的暫時性憑證。連接到該角色的許可政策決定了執行個體可以執行哪些操作。

範例 2

Amazon Relational Database Service (Amazon RDS) 支援增強型監控功能。此功能可讓 Amazon RDS 使用代理程式來監控資料庫執行個體。同時也允許 Amazon RDS 將指標記錄到 Amazon CloudWatch Logs。若要啟用此功能，您必須建立服務角色以提供 Amazon RDS 許可來監控和寫入指標到您的日誌。

為 Amazon RDS 增強型監控建立角色

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 選擇AWS 服務角色類型，然後針對其他 的使用案例 AWS 服務，選擇 RDS 服務。選擇 RDS – Enhanced Monitoring (RDS – 增強型監控)，然後選擇 Next (下一步)。
4. 選擇 AmazonRDSEnhancedMonitoringRole 許可政策。
5. 選擇下一步。
6. 針對 Role name (角色名稱)，輸入可協助您識別此角色用途的角色名稱。角色名稱在您的 中必須是唯一的 AWS 帳戶。角色名稱用在政策中或作為 ARN 的一部分時，角色名稱區分大小寫。當主控台中的客戶顯示角色名稱時 (例如在登入程序期間)，角色名稱不區分大小寫。因為有各種實體可能會參考此角色，所以建立角色之後，您就無法編輯其名稱。
7. (選用) 在 Description (說明) 中，輸入新角色的說明。
8. (選用) 藉由連接標籤作為鍵值對，將中繼資料新增至使用者。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。

9. 檢閱角色，然後選擇 Create role (建立角色)。

角色會自動取得信任政策，授予 monitoring.rds.amazonaws.com 服務許可以擔任角色。在那之後，Amazon RDS 可以執行 AmazonRDSEnhancedMonitoringRole 政策允許的所有動作。

您要存取增強型監控的使用者需要包含陳述式的政策，該陳述式允許使用者列出 RDS 角色，以及允許使用者傳遞角色的陳述式，如下所示。使用您的帳號並將角色名稱取代為您在步驟 6 提供的名稱。

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
    "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
    "Effect": "Allow",
    "Action": [ "iam:PassRole" ],
    "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
}
```

您可以將此陳述式與在另一個政策中的陳述式併在一起，或是將它放在它自己的政策中。若要改為指定使用者可以傳遞任何開頭為 RDS- 的角色，您可以將資源 ARN 中的角色名稱取代為萬用字元，如下所示。

```
    "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

iam:PassRole AWS CloudTrail 日誌中的 動作

PassRole 不是 API 呼叫。PassRole 是一項許可，表示沒有為 IAM PassRole 產生 CloudTrail 日誌。若要檢閱在 CloudTrail AWS 服務 中傳遞哪些角色，您必須檢閱建立或修改接收角色之 AWS 資源的 CloudTrail 日誌。例如，角色會在建立時傳遞至 AWS Lambda 函數。CreateFunction 動作的日誌顯示有關被傳遞到函數之角色的記錄。

撤銷 IAM 角色臨時安全憑證

> **Warning**
>
> 如果您遵循此頁面上的步驟，則擔任角色所建立目前工作階段的所有使用者都會被拒絕存取所有 AWS 動作和資源。這會導致使用者遺失未儲存的工作。

當您允許使用者 AWS 管理主控台 以較長的工作階段持續時間（例如 12 小時）存取 時，其暫時登入資料不會快速過期。如果使用者不慎將他們的憑證向未經授權的第三方公開，該方在工作階段的持續時間均有權存取。不過，您可以立即撤銷所有在某個時間點前授予該角色的憑證許可，如果需要的話。該角色在指定時間點前發出的所有暫時性憑證變成無效。這會強迫所有使用者重新驗證和請求新的憑證。

> **Note**
>
> 您無法撤銷服務連結角色的工作階段。

當您使用此主題中的程序撤銷角色的許可時， 會將新的內嵌政策 AWS 連接到拒絕所有動作的所有許可的角色。如果使用者在您撤銷許可的時間點「之前」擔任該角色，則它只會包含套用限制條件的情況。如果使用者在撤銷許可「之後」擔任該角色，則拒絕政策不會套用至該使用者。

如需拒絕存取的詳細資訊，請參閱 停用臨時安全性憑證的許可。

> **Important**
>
> 此拒絕政策適用於指定角色的所有使用者，而不只是具主控台工作階段較長持續時間的角色。

從角色撤銷工作階段許可的最低許可

若要從角色成功地撤銷工作階段許可，您必須擁有該角色的 `PutRolePolicy` 許可。這可讓您將 `AWSRevokeOlderSessions` 內嵌政策連接至該角色。

撤銷工作階段許可

您可以從角色撤銷工作階段許可，以拒絕擔任該角色之任何使用者的所有許可。

> **Note**
>
> 您無法在 IAM 中編輯從 IAM Identity Center 許可集建立的角色。您必須撤銷 IAM Identity Center 中使用者的作用中許可集工作階段。如需詳細資訊，請參閱 IAM Identity Center User Guide 中的 Revoke active IAM role sessions created by permission sets。

立即拒絕對任何目前使用者的角色憑證的所有許可

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇角色，然後選擇您想要撤銷其許可的角色名稱 (不是核取方塊)。
3. 在所選取角色的 Summary (摘要) 頁面上，選擇 Revoke sessions (工作階段) 標籤。
4. 在 Revoke sessions (撤銷工作階段) 標籤上，選擇 Revoke active sessions (撤銷作用中的工作階段)。
5. AWS 會要求您確認動作。選取我確認我將撤銷此角色的所有作用中工作階段。核取方塊，然後在對話方塊中選擇撤銷作用中工作階段。

   然後，IAM 將名為 `AWSRevokeOlderSessions` 的政策連接至角色。在您選擇撤銷作用中工作階段後，政策會拒絕過去以及未來約 30 秒內擔任該角色的使用者的所有存取權。此未來時間選擇考量政策的傳播延遲，以便處理在更新的政策在指定區域中生效之前取得或續約的新工作階段。在您選擇「撤銷作用中的工作階段」後，擔任該角色超過大約 30 秒的任何使用者都不會受影響。若要了解變更不一定會立即顯示的原因，請參閱 我所做的變更不一定都會立刻生效。

> **Note**
>
> 如果您稍後再次選擇撤銷工作階段，則會重新整理政策中的日期和時間戳記，而且它會再次拒絕在新指定的時間之前擔任該角色之任何使用者的許可。

以這種方式呼叫工作階段的有效使用者必須獲得臨時憑證，新工作階段才能繼續運作。會 AWS CLI 快取登入資料，直到過期為止。若要強制 CLI 刪除並重新整理已失效的快取憑證，請執行以下命令之一：

Linux、macOS 或 Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

在指定時間之前撤銷工作階段許可

您也可以使用 AWS CLI 或 SDK 隨時撤銷您選擇的工作階段許可，在政策的條件元素中指定aws:TokenIssueTime金鑰的值。

在 aws:TokenIssueTime 的值早於指定的日期和時間時，此政策會拒絕所有許可。aws:TokenIssueTime 的值對應於臨時安全性憑證的確切建立時間。該aws:TokenIssueTime值僅存在於使用臨時安全登入資料簽署的 AWS 請求內容中，因此政策中的拒絕陳述式不會影響使用 IAM 使用者的長期登入資料簽署的請求。

此政策也可連接至角色。在這種情況下，該政策只會影響由該角色在指定日期和時間之前建立的臨時安全性憑證。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

以這種方式呼叫工作階段的有效使用者必須獲得臨時憑證，新工作階段才能繼續運作。會 AWS CLI 快取登入資料，直到過期為止。若要強制 CLI 刪除並重新整理已失效的快取憑證，請執行以下命令之一：

Linux、macOS 或 Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

更新服務連結角色

您用來編輯服務連結角色的方法取決於服務。有些服務也許可讓您從服務主控台、API 或 CLI 編輯服務連結角色的許可。不過，因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。您可以從 IAM 主控台、API 或 CLI 編輯任何角色的說明。

如需哪些服務支援使用服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找 Service-Linked Role (服務連結角色) 欄中顯示 Yes (是) 的服務。若要了解服務是否支援編輯服務連結角色，請選擇是連結以檢視該服務的服務連結角色的文件。

編輯服務連結角色描述 (主控台)

您可以使用 IAM 主控台來編輯服務連結角色的說明。

編輯服務連結角色的說明 (主控台)

1. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)。
2. 選擇要修改之角色的名稱。
3. 在 Role description (角色說明) 的最右邊，選擇 Edit (編輯)。
4. 在方塊中輸入新的描述，然後選擇 Save (儲存)。

編輯服務連結角色描述 (AWS CLI)

您可以使用 的 IAM 命令 AWS CLI 來編輯服務連結角色的描述。

變更服務連結角色的說明 (AWS CLI)

1. (選用) 若要檢視角色的目前說明，請執行下列命令：

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   透過 CLI 命令，使用角色名稱 (而非 ARN) 來參照角色。例如，如果角色具有下列 ARN：`arn:aws:iam::123456789012:role/myrole`，請將角色參照為 **`myrole`**。
2. 若要更新服務連結角色的說明，請執行下列命令：

   ```
   aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
   ```

編輯服務連結角色描述 (AWS API)

您可以使用 AWS API 編輯服務連結角色的描述。

變更服務連結角色 (AWS API) 的描述

1. (選用) 若要檢視角色的目前說明，請呼叫以下操作並指定角色的名稱：

   AWS API：GetRole
2. (選用) 若要更新角色的說明，請呼叫以下操作並指定角色的名稱 (和選用描述)：

   AWS API：UpdateRole

更新角色信任政策

若要變更誰可擔任該角色，您必須修改角色的信任政策。您無法修改服務連結角色的信任政策。

> **備註**
>
> - 如果使用者已於角色信任政策中列為主體，卻無法擔任該角色，請檢查該使用者的許可界限。如果已經設定使用者的許可界限，則其必須允許 `sts:AssumeRole` 動作。
> - 若要允許使用者在角色工作階段中再次擔任目前角色，請在角色信任政策中將角色 ARN 或 AWS 帳戶 ARN 指定為主體。 AWS 服務 提供運算資源，例如 Amazon EC2、Amazon ECS、Amazon EKS 和 Lambda，可提供臨時登入資料並自動更新這些登入資料。此可確保您始終擁有一組有效的憑證。對於這些服務，不需要再次擔任目前的角色即可取得臨時憑證。但是，如果您想要傳遞工作階段標籤或一個工作階段政策，則需要再次擔任目前的角色。

更新角色信任政策 (主控台)

在 中變更角色信任政策 AWS 管理主控台

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)。
3. 在您帳戶的角色清單中，選擇您想要修改的角色名稱。
4. 選擇 Trust Relationships (信任關係) 標籤，然後選擇 Edit Trust Relationship (編輯信任政策)。

5. 視需要編輯信任政策。若要新增其他可擔任該角色的主體，請在 Principal 元素中指定它們。例如，下列政策程式碼片段示範如何在 Principal元素 AWS 帳戶 中參考兩個：

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:root",
    "arn:aws:iam::444455556666:root"
  ]
},
```

如果您指定在另一個帳戶中的主體，這時在角色的信任政策中新增帳戶，只能建立一半的跨帳戶信任關係。根據預設，信任帳戶中沒有任何使用者可擔任該角色。剛信任的帳戶管理員必須授予新的使用者擔任該角色的許可。為了這樣做，管理員必須建立或編輯連接於使用者的政策，以允許使用者存取 sts:AssumeRole 動作。如需詳細資訊，請參閱下列程序或 向使用者授予切換角色的許可。

下列政策程式碼片段示範如何在 Principal元素中參考兩項 AWS 服務：

```
"Principal": {
  "Service": [
    "opsworks.amazonaws.com",
    "ec2.amazonaws.com"
  ]
},
```

6. 當您完成編輯您的信任政策，請選擇 Update policy (更新政策) 來儲存您的變更。

如需有關政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management 和 IAM JSON 政策元素參考。

允許信任外部帳戶中的使用者使用該角色 (主控台)

如需有關此程序的更多資訊和詳細資訊，請參閱 向使用者授予切換角色的許可。

1. 登入受信任的外部 AWS 帳戶。
2. 決定是否將許可連接到使用者或群組。在 IAM 主控台的導覽窗格中，相應地選擇 Users (使用者) 或 User groups (使用者群組)。
3. 選擇要授予其存取許可的使用者或群組的名稱，然後選擇 Permissions (許可) 標籤。
4. 執行下列任一步驟：

- 若要編輯客戶受管政策，請選擇政策名稱，並選擇 Edit policy (編輯政策)，然後選擇 JSON 標籤。您無法編輯具有 AWS 圖示 ( ) 的 受 AWS 管 AWS 政策。如需有關 AWS 受管政策與客戶受管政策之間差異的詳細資訊，請參閱 受管政策與內嵌政策。
- 若要編輯內嵌政策，請選擇政策名稱旁的箭頭，然後選擇 Edit policy (編輯政策)。

5. 在政策編輯器中，新增一個指定以下內容的新 `Statement` 元素：

```
{
  "Effect": "Allow",
  "Action": "sts:AssumeRole",
  "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
}
```

   將陳述式中的 ARN 取代為使用者可擔任的角色的 ARN。

6. 按照螢幕上的提示來完成政策編輯。

更新角色信任政策 (AWS CLI)

您可以使用 AWS CLI 來變更誰可以擔任角色。

修改角色信任政策 (AWS CLI)

1. (選用) 如果您不知道要修改的角色的名稱，請執行以下命令列出您帳戶中的角色：
   - aws iam list-roles
2. (選用) 若要檢視角色的目前信任政策，請執行下列命令：
   - aws iam get-role
3. 若要修改可以存取角色的信任主體，請使用更新的信任政策建立文字檔案。您可以使用任何文字編輯器來建構政策。

   例如，下列信任政策顯示如何在 `Principal` 元素中參考兩個 AWS 帳戶 。這可讓兩個不同 內的使用者 AWS 帳戶 擔任此角色。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

如果您指定在另一個帳戶中的主體，這時在角色的信任政策中新增帳戶，只能建立一半的跨帳戶信任關係。根據預設，信任帳戶中沒有任何使用者可擔任該角色。剛信任的帳戶管理員必須授予新的使用者擔任該角色的許可。為了這樣做，管理員必須建立或編輯連接於使用者的政策，以允許使用者存取 `sts:AssumeRole` 動作。如需詳細資訊，請參閱下列程序或 向使用者授予切換角色的許可。

4. 若要使用剛建立的檔案更新信任政策，請執行下列命令：

   - aws iam update-assume-role-policy

若要允許信任外部帳戶中的使用者使用該角色 (AWS CLI)

如需有關此程序的更多資訊和詳細資訊，請參閱 向使用者授予切換角色的許可。

1. 建立 JSON 檔案，其中包含擔任該角色所需的許可的許可政策。例如，以下政策包含最低必要許可：

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
```

```
        "Resource": "arn:aws:iam::111122223333:role/ROLE-NAME"
    }
}
```

將陳述式中的 ARN 取代為使用者可擔任的角色的 ARN。

2. 執行以下命令以上傳 JSON 檔案 (其中包含的信任政策) 至 IAM：

   - aws iam create-policy

   此命令的輸出結果含有該政策的 ARN。記下此 ARN，因為您需要在稍後的步驟中使用。

3. 決定將政策到連接哪位使用者或群組。如果您不知道適用的使用者或群組的名稱，請使用下列其中一個命令來列出帳戶中的使用者或群組：

   - aws iam list-users
   - aws iam list-groups

4. 使用下列命令之一將您在之前的步驟中建立的政策連接到使用者或群組：

   - aws iam attach-user-policy
   - aws iam attach-group-policy

更新角色信任政策 (AWS API)

您可以使用 AWS API 來變更誰可以擔任角色。

修改角色信任政策 (AWS API)

1. (選用) 如果您不知道要修改的角色的名稱，請呼叫以下操作列出您帳戶中的角色：

   - ListRoles

2. (選用) 若要檢視角色的目前信任政策，請呼叫下列操作：

   - GetRole

3. 若要修改可以存取角色的信任主體，請使用更新的信任政策建立文字檔案。您可以使用任何文字編輯器來建構政策。

   例如，下列信任政策顯示如何在 `Principal` 元素中參考兩個 AWS 帳戶 。這可讓兩個不同 內的使用者 AWS 帳戶 擔任此角色。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

如果您指定在另一個帳戶中的主體，這時在角色的信任政策中新增帳戶，只能建立一半的跨帳戶信任關係。根據預設，信任帳戶中沒有任何使用者可擔任該角色。剛信任的帳戶管理員必須授予新的使用者擔任該角色的許可。為了這樣做，管理員必須建立或編輯連接於使用者的政策，以允許使用者存取 `sts:AssumeRole` 動作。如需詳細資訊，請參閱下列程序或 向使用者授予切換角色的許可。

4. 若要使用剛建立的檔案更新信任政策，請呼叫下列操作：

   - UpdateAssumeRolePolicy

允許受信任外部帳戶中的使用者使用該角色 (AWS API)

如需有關此程序的更多資訊和詳細資訊，請參閱 向使用者授予切換角色的許可。

1. 建立 JSON 檔案，其中包含擔任該角色所需的許可的許可政策。例如，以下政策包含最低必要許可：

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
```

```
        "Resource": "arn:aws:iam::111122223333:role/ROLE-NAME"
    }
}
```

將陳述式中的 ARN 取代為使用者可擔任的角色的 ARN。

2. 呼叫以下操作，以上傳 JSON 檔案 (其中包含的信任政策) 至 IAM：

   - CreatePolicy

   此操作的輸出結果含有該政策的 ARN。記下此 ARN，因為您需要在稍後的步驟中使用。

3. 決定將政策到連接哪位使用者或群組。如果您不知道適用的使用者或群組的名稱，請呼叫下列其中一個操作來列出帳戶中的使用者或群組：

   - ListUsers
   - ListGroups

4. 呼叫下列操作之一將您在之前的步驟中建立的政策連接到使用者或群組：

   - API: AttachUserPolicy
   - AttachGroupPolicy

更新角色的許可

使用下列程序來更新角色的許可政策和許可界限。

先決條件：檢視角色存取

變更角色的許可之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

更新角色的許可政策

若要變更角色允許的許可，請修改角色的許可政策 (或政策)。您無法修改 IAM 中服務連結角色的許可政策。您也許可以修改該服務 (取決於角色) 內的許可政策。若要檢查服務是否支援這項功能，請參閱AWS 使用 IAM 的 服務，並且尋找 Service-linked roles (服務連結的角色) 資料列為 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。

更新角色許可政策 (主控台)

變更角色允許的許可 (主控台)

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)。
3. 選擇您要修改的角色的名稱，然後選擇 Permissions (許可) 標籤。
4. 執行下列任一步驟：

   - 若要編輯現有客戶受管政策，請選擇政策名稱，然後選擇 Edit policy (編輯政策)。

     > **Note**
     >
     > 您無法編輯具有 AWS 圖示
     > ()
     > 的 受 AWS 管 AWS 政策。如需 AWS 受管政策與客戶受管政策之間差異的詳細資訊，請參閱 受管政策與內嵌政策。

   - 若要將現有受管政策連接至角色，請選擇 Add permissions (新增許可)，然後選擇 Attach policies (連接政策)。
   - 若要編輯現有內嵌政策，請展開政策，然後選擇 Edit (編輯)。
   - 若要嵌入新的內嵌政策，請選擇 Add permissions (新增許可)，然後選擇 Create inline policy (建立內嵌政策)。
   - 若要從角色移除現有政策，請選取政策名稱旁的核取方塊，然後選擇移除。

更新角色許可政策 (AWS CLI)

若要變更角色允許的許可，請修改角色的許可政策 (或政策)。您無法修改 IAM 中服務連結角色的許可政策。您也許可以修改該服務 (取決於角色) 內的許可政策。若要檢查服務是否支援這項功能，請參閱AWS 使用 IAM 的 服務，並且尋找 Service-linked roles (服務連結的角色) 資料列為 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。

若要變更角色允許的許可 (AWS CLI)

1. (選用) 若要檢視與角色關聯的目前許可，請執行下列命令：

   1. aws iam list-role-policies 列出內嵌政策

    2. aws iam list-attached-role-policies 列出受管政策
2. 更新角色許可的命令會有所不同，具體取決於您是要更新受管政策還是內嵌政策。

   若要更新受管政策，請執行下列命令來建立新版本的受管政策：

   - aws iam create-policy-version

   若要更新內嵌政策，請執行下列命令：

   - aws iam put-role-policy

更新角色許可政策 (AWS API)

若要變更角色允許的許可，請修改角色的許可政策 (或政策)。您無法修改 IAM 中服務連結角色的許可政策。您也許可以修改該服務 (取決於角色) 內的許可政策。若要檢查服務是否支援這項功能，請參閱AWS 使用 IAM 的 服務，並且尋找 Service-linked roles (服務連結的角色) 資料列為 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。

變更角色允許的許可 (AWS API)

1. (選用) 若要檢視與角色關聯的目前許可，請呼叫下列操作：

   1. ListRolePolicies 列出內嵌政策
   2. ListAttachedRolePolicies 列出受管政策
2. 更新角色許可的操作會有所不同，具體取決於您是要更新受管政策還是內嵌政策。

   若要更新受管政策，請呼叫下列操作來建立新版本的受管政策：

   - CreatePolicyVersion

   若要更新內嵌政策，請呼叫下列操作：

   - PutRolePolicy

更新角色的許可界限

若要變更角色適用的許可上限，請修改角色的許可界限。

更新角色許可界限 (主控台)

變更用於設定角色許可界限的政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇角色。
3. 選擇您想要為其變更許可界限的角色名稱。
4. 選擇 Permissions (許可) 標籤。如有必要，開啟 許可界限 區段，然後選擇 變更界限。
5. 選擇要用於許可界限的政策。
6. 選擇 Change boundary (變更界限)。

   在下次有人擔任此角色之前，您的變更不會生效。

更新角色許可界限 (AWS CLI)

變更用於設定角色許可界限的受管政策 (AWS CLI)

1. (選用) 若要檢視角色的目前許可界限，請執行下列命令：
   - aws iam get-role
2. 若要使用不同的受管政策來更新角色的許可界限，請執行下列命令：
   - aws iam put-role-permissions-boundary

   角色只能有一個受管政策設定為許可界限。您若變更許可界限，您會變更角色適用的許可上限。

更新角色許可界限 (AWS API)

變更用於設定角色 (AWS API) 許可界限的受管政策

1. (選用) 若要檢視角色的目前許可界限，請呼叫下列操作：
   - GetRole
2. 若要使用不同的受管政策來更新角色的許可界限，請呼叫下列操作：
   - PutRolePermissionsBoundary

角色只能有一個受管政策設定為許可界限。您若變更許可界限，您會變更角色適用的許可上限。

更新角色的設定

使用下列程序來更新角色的描述，或者，變更角色的工作階段持續時間上限。

更新角色描述

若要變更描述的描述，請修改描述文字。

更新角色描述 (主控台)

變更角色的說明 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)。
3. 選擇要修改之角色的名稱。
4. 在 Summary (摘要) 區段中，選擇 Edit (編輯)。
5. 在方塊中輸入新說明，然後選擇 Save changes (儲存變更)。

更新角色描述 (AWS CLI)

變更角色的說明 (AWS CLI)

1. (選用) 若要檢視角色的目前說明，請執行下列命令：
   - aws iam get-role
2. 若要更新角色的說明，請使用說明參數執行下列命令：
   - aws iam update-role

更新角色描述 (AWS API)

變更角色的描述 (AWS API)

1. (選用) 若要檢視角色的目前說明，請呼叫下列操作：
   - GetRole

2. 若要更新角色的說明，請使用說明參數呼叫下列操作：

   - UpdateRole

更新角色的最大工作階段持續時間

若要為使用主控台 AWS CLI、 或 AWS API 擔任的角色指定工作階段持續時間上限設定，請修改工作階段持續時間上限設定值。此設定的值可介於 1 小時至 12 小時。如果未指定值，則套用預設最大值 1 小時。此設定不會限制 AWS 服務所擔任的工作階段。

更新最大角色工作階段持續時間 (主控台)

變更使用主控台 AWS CLI或 AWS API （ 主控台 ） 擔任之角色的工作階段持續時間上限設定

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)。
3. 選擇要修改之角色的名稱。
4. 在 Summary (摘要) 區段中，選擇 Edit (編輯)。
5. 在 Maximum session duration (最大工作階段持續時間) 中，選擇一個值。或者，選擇 Custom duration (自訂持續時間) 並輸入一個值 (以秒為單位)。
6. 選擇儲存變更。

   在下次有人擔任此角色之前，您的變更不會生效。若要了解如何撤銷此角色的現有工作階段，請參閱撤銷 IAM 角色臨時安全憑證。

在 中 AWS 管理主控台，IAM 使用者工作階段預設為 12 小時。在主控台中切換角色的 IAM 使用者會被授予角色最長工作階段持續時間或使用者工作階段中的剩餘時間，以較短者為準。

從 AWS CLI 或 AWS API 擔任角色的任何人都可以請求更長的工作階段，最高可達此上限。`MaxSessionDuration` 設定決定可以請求之角色工作階段的最大持續時間。

- 若要使用 指定工作階段持續時間， AWS CLI 請使用 `duration-seconds` 參數。如需詳細資訊，請參閱 切換到 IAM 角色 (AWS CLI)。
- 若要使用 AWS API 指定工作階段持續時間，請使用 `DurationSeconds` 參數。如需詳細資訊，請參閱 切換到 IAM 角色 (AWS API)。

更新角色工作階段持續時間上限 (AWS CLI)

> **Note**
>
> 從 AWS CLI 或 API 擔任角色的任何人都可以使用 `duration-seconds` CLI 參數或 `DurationSeconds` API 參數來請求更長的工作階段。`MaxSessionDuration` 設定決定可以使用`DurationSeconds` 參數請求的角色工作階段的最大持續時間。如果使用者不為 `DurationSeconds` 參數指定一個值，則使用者的安全憑證有效期為 1 小時。

使用 AWS CLI (AWS CLI) 變更所擔任角色的工作階段持續時間上限設定

1. (選用) 若要檢視角色的目前最大工作階段持續時間設定，請執行下列命令：
   - aws iam get-role
2. 若要更新角色的工作階段持續時間設定，請使用 `max-session-duration` CLI 參數或 `MaxSessionDuration` API 參數執行以下命令：
   - aws iam update-role

   在下次有人擔任此角色之前，您的變更不會生效。若要了解如何撤銷此角色的現有工作階段，請參閱撤銷 IAM 角色臨時安全憑證。

更新角色工作階段持續時間上限 (AWS API)

> **Note**
>
> 從 AWS CLI 或 API 擔任角色的任何人都可以使用 `duration-seconds` CLI 參數或 `DurationSeconds` API 參數來請求更長的工作階段。`MaxSessionDuration` 設定決定可以使用`DurationSeconds` 參數請求的角色工作階段的最大持續時間。如果使用者不為 `DurationSeconds` 參數指定一個值，則使用者的安全憑證有效期為 1 小時。

變更使用 API (AWS API) 擔任之角色的工作階段持續時間上限設定

1. (選用) 若要檢視角色的目前最大工作階段持續時間設定，請呼叫下列操作：
   - GetRole

2. 若要更新角色的工作階段持續時間設定，請使用 `max-session-duration` CLI 參數或 `MaxSessionDuration` API 參數呼叫以下操作：

   - UpdateRole

   在下次有人擔任此角色之前，您的變更不會生效。若要了解如何撤銷此角色的現有工作階段，請參閱撤銷 IAM 角色臨時安全憑證。

刪除角色或執行個體設定檔

如果您不再需要角色，建議您刪除角色及其關聯的許可。如此一來，您就沒有未主動監控或維護的未使用實體。

如果角色與 EC2 執行個體相關聯，您也可以從執行個體設定檔中移除角色，然後刪除執行個體設定檔。

> **Warning**
>
> 確保您沒有任何 Amazon EC2 執行個體與您即將刪除的角色或執行個體描述檔一起執行。若刪除與執行中的執行個體相關聯的角色或執行個體描述檔，將會中斷執行個體上執行的所有應用程式。

如果您不想永久刪除某個角色，則可以停用該角色。若要執行此操作，請變更角色的政策，然後撤銷所有目前的工作階段。例如，您可以將政策新增至拒絕存取所有 的角色 AWS。您也可以編輯信任政策，以拒絕任何嘗試擔任該角色的人存取。如需有關撤銷工作階段的詳細資訊，請參閱 撤銷 IAM 角色臨時安全憑證。

主題

- 檢視角色存取
- 刪除服務連結角色
- 刪除 IAM 角色 (主控台)
- 刪除 IAM 角色 (AWS CLI)
- 刪除 IAM 角色 (AWS API)
- 相關資訊

檢視角色存取

刪除角色之前，建議您先檢閱上次使用角色的時間。您可以使用 AWS 管理主控台、 AWS CLI或 AWS API 來執行此操作。您應該檢視此資訊，因為您不想移除正在使用該角色之某個使用者的存取權。

角色上次活動的日期可能與上次存取索引標籤中報告的上次使用日期不符。上次存取索引標籤僅報告角色許可政策允許服務的活動。角色上次活動的日期包括上次嘗試存取其中任何服務的日期 AWS。

> **Note**
>
> 角色上次活動和「上次存取」資料的追蹤期間為過去 400 天。如果您的區域在過去一年內已開始支援這些功能，此期間可能會縮短。該角色的上次使用時間可能已經超過 400 天。如需有關追蹤期間的詳細資訊，請參閱 其中 會 AWS 追蹤上次存取的資訊。

檢視上次使用角色的時間 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色)。
3. 尋找具有您要檢視活動之角色的資料列。您可以使用搜尋欄位縮減結果。檢視 Last activity (上次活動) 欄，查看自上次使用該角色後的天數。如果在追蹤期間內未曾使用該角色，資料表會顯示 None (無)。
4. 選擇角色名稱以查看詳細資訊。角色的 Summary (摘要) 頁面也包含 Last activity (上次活動)，會顯示角色上次使用的日期。如果過去 400 天內未使用過該角色，則 Last activity (上次活動) 會顯示 Not accessed in the tracking period (在追蹤期間內未存取)。

檢視上次使用角色的時間 (AWS CLI)

`aws iam get-role` - 執行此命令以傳回角色的資訊，包括 `RoleLastUsed` 物件。此物件包含 `LastUsedDate` 和上次使用角色的 `Region`。如有 `RoleLastUsed` 但不包含值，即表示未在追蹤期間內使用過該角色。

檢視上次使用角色的時間 (AWS API)

`GetRole` - 呼叫此操作以傳回角色的資訊，包括 `RoleLastUsed` 物件。此物件包含 `LastUsedDate` 和上次使用角色的 `Region`。如有 `RoleLastUsed` 但不包含值，即表示未在追蹤期間內使用過該角色。

刪除服務連結角色

您用來刪除服務連結角色的方法取決於服務。在某些情況下，您不需要手動刪除一個服務連結角色。例如，當您在服務中完成特定動作 (例如移除資源)，該服務可能會為您刪除服務連結角色。在其他情況下，服務可能支援從服務主控台、API 或 AWS CLI手動刪除服務連結角色。

請檢閱連結服務中服務連結角色的文件，以了解如何刪除該角色。您可以前往主控台的 IAM Roles (角色) 頁面，檢視您帳戶中的服務連結角色。服務連結角色會在表格的 Trusted entities (受信任實體) 欄中以 (Service-linked role) ((服務連結角色)) 顯示。在 Summary (摘要) 頁面上的橫幅，也會指出角色是一個服務連結角色。

如果服務不包含刪除服務連結角色的文件，您可以使用 IAM 主控台 AWS CLI或 API 來刪除角色。

刪除 IAM 角色 (主控台)

當您使用 AWS 管理主控台 刪除角色時，IAM 會自動分離與該角色相關聯的受管政策。它還會自動刪除與該角色關聯的內嵌政策，以及包含該角色的任何 Amazon EC2 執行個體設定檔。

> **Important**
>
> 在某些情況下，角色可能會與 Amazon EC2 執行個體描述檔相關聯，而且角色和執行個體描述檔的名稱可能完全相同。在這種情況下，您可以使用 AWS 管理主控台 來刪除角色和執行個體描述檔。此連結就會自動為您在主控台建立的角色和執行個體設定檔。如果您從 AWS CLI、Tools for Windows PowerShell 或 AWS API 建立角色，則角色和執行個體描述檔可能有不同的名稱。在該情況下，您無法使用主控台將它們刪除。反之，您必須使用 AWS CLI Tools for Windows PowerShell 或 AWS API，先從執行個體描述檔中移除角色。然後，您必須採取不同步驟刪除該角色。

刪除角色 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色)，然後勾選您要刪除之角色名稱旁的核取方塊。
3. 在頁面頂端，選擇 Delete (刪除)。
4. 在確認對話方塊中，檢閱上次存取的資訊，其中顯示每個所選角色上次存取 AWS 服務的時間。這可協助您確認角色目前是否在作用中。如果您要繼續，請在文字輸入欄位中輸入角色的名稱，並選擇 Delete (刪除)。如果您確定，可以繼續刪除，即使上次存取的資訊仍持續載入。

> **Note**
>
> 除非執行個體設定檔和角色同名，否則您不能使用主控台刪除執行個體設定檔。作為刪除角色的過程的一部分，執行個體描述檔會被刪除，如前面的程序中所述。若要刪除執行個體描述檔而不同時刪除角色，您必須使用 AWS CLI 或 AWS API。如需詳細資訊，請參閱下列區段。

刪除 IAM 角色 (AWS CLI)

當您使用 AWS CLI 刪除角色時，您必須先刪除與該角色相關聯的內嵌政策。您還必須分離與該角色關聯的受管政策。如果您要刪除包含角色的相關聯執行個體設定檔，您必須分開刪除。

刪除角色 (AWS CLI)

1. 如果您不知道想要刪除的角色名稱，請輸入以下命令列出帳戶中的角色：

   ```
   aws iam list-roles
   ```

   此清單包含每個角色的 Amazon Resource Name (ARN)。透過 CLI 命令，使用角色名稱 (而非 ARN) 來參照角色。例如，如果角色具有下列 ARN：`arn:aws:iam::123456789012:role/myrole`，請將角色參照為 **`myrole`**。
2. 從角色關聯的所有執行個體設定檔移除該角色。

   a. 若要列出所有與角色相關聯的執行個體設定檔，請輸入下列命令：

      ```
      aws iam list-instance-profiles-for-role --role-name role-name
      ```

   b. 若要從執行個體設定檔中移除該角色，請針對每個執行個體設定檔輸入下列命令：

      ```
      aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
      ```
3. 刪除所有與該角色相關聯的政策。

   a. 若要列出角色中所有的內嵌政策，請輸入下列命令：

      ```
      aws iam list-role-policies --role-name role-name
      ```

   b. 若要刪除角色中的每個內嵌政策，請針對每個政策輸入下列命令：

```
aws iam delete-role-policy --role-name role-name --policy-name policy-name
```

c. 若要列出連接至角色的所有受管政策，請輸入下列命令：

```
aws iam list-attached-role-policies --role-name role-name
```

d. 若要從角色分離每個受管政策，請針對每個政策輸入下列命令：

```
aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
```

4. 輸入下列命令以刪除角色：

```
aws iam delete-role --role-name role-name
```

5. 如果您不打算重複使用曾與角色相關聯的執行個體設定檔，您可以輸入下列命令予以刪除：

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

刪除 IAM 角色 (AWS API)

當您使用 IAM API 刪除該角色，您必須先刪除與該角色關聯的內嵌政策。您還必須分離與該角色關聯的受管政策。如果您要刪除包含角色的相關聯執行個體設定檔，您必須分開刪除。

刪除角色 (AWS API)

1. 若要列出角色關聯的所有執行個體設定檔，請呼叫 ListInstanceProfilesForRole。

   若要從執行個體設定檔移除角色，請呼叫 RemoveRoleFromInstanceProfile。您必須傳遞角色名稱和執行個體設定檔名稱。

   如果您不打算重複使用曾與角色相關聯的執行個體描述檔，請呼叫 DeleteInstanceProfile 予以刪除。

2. 若要列出角色的所有內嵌政策，請呼叫 ListRolePolicies。

   若要刪除與角色建立關聯的所有內嵌政策，請呼叫 DeleteRolePolicy。您必須傳遞角色名稱和內嵌政策名稱。

3. 若要列出連接至角色的所有受管政策，請呼叫 ListAttachedRolePolicies。

若要分離連接至該角色的受管政策，請呼叫 DetachRolePolicy。您必須傳遞角色名稱和受管政策 ARN。

4. 呼叫 DeleteRole 以刪除該角色。

相關資訊

如需有關執行個體設定檔的一般資訊，請參閱使用執行個體設定檔。

如需服務連結角色的一般資訊，請參閱 建立服務連結角色。

擔任角色的方法

在使用者、應用程式或服務可以使用您建立的角色之前，您必須授予切換到該角色的許可。您可以使用連接至群組或使用者的任何政策，來授予必要的許可。授予許可後，使用者可以從 AWS 管理主控台、Tools for Windows PowerShell、 AWS Command Line Interface (AWS CLI) 和 `AssumeRole` API 擔任角色。

> **Important**
>
> 當您以程式設計方式而不是在 IAM 主控台中建立角色時，則除了 `RoleName` 之外，您還可以選擇新增高達 512 個字元的 `Path`，該字元長度最多 64 個字元。不過，如果您想要在 中使用具有切換角色功能的角色 AWS 管理主控台，則合併的 `Path`和 `RoleName`不得超過 64 個字元。

用於擔任角色的方法決定誰可以擔任該角色以及角色工作階段可以持續多久。使用 `AssumeRole*` API 操作時，您擔任的 IAM 角色是資源。呼叫 `AssumeRole*` API 操作的使用者或角色是主體。

下表比較擔任角色的方法。

擔任角色的方法	誰可以擔任這個角色	指定憑證生命週期的方法	憑證存留期 (最小 | 最大 | 預設)
AWS 管理主控台	使用者或角色[1] (透過切換角色)	Role (角色) 摘要頁面上的 Maximum session duration	15 分鐘 | 最大工作階段持續時間設定[2] | 1 小時

擔任角色的方法	誰可以擔任這個角色	指定憑證生命週期的方法	憑證存留期 (最小 | 最大 | 預設)
		(最大工作階段持續時間)	
assume-role CLI 或 AssumeRole API 操作	使用者或角色[1]	duration-seconds CLI 或 DurationSeconds API 參數	15 分鐘 | 最大工作階段持續時間設定[2] | 1 小時
assume-role-with-saml CLI 或 AssumeRoleWithSAML API 操作	任何使用 SAML 驗證的使用者	duration-seconds CLI 或 DurationSeconds API 參數	15 分鐘 | 最大工作階段持續時間設定[2] | 1 小時
assume-role-with-web-identity CLI 或 AssumeRoleWithWebIdentity API 操作	使用 OIDC 提供者進行身分驗證的任何使用者	duration-seconds CLI 或 DurationSeconds API 參數	15 分鐘 | 最大工作階段持續時間設定[2] | 1 小時
使用 建構的主控台 URLAssumeRole	使用者或角色	SessionDuration URL 中的 HTML 參數	15 分鐘 | 12 小時 | 1 小時
使用 建構的主控台 URLAssumeRoleWithSAML	任何使用 SAML 驗證的使用者	SessionDuration URL 中的 HTML 參數	15 分鐘 | 12 小時 | 1 小時

擔任角色的方法	誰可以擔任這個角色	指定憑證生命週期的方法	憑證存留期 (最小 | 最大 | 預設)
使用 建構的主控台 URL `AssumeRoleWithWebIdentity`	使用 OIDC 提供者進行身分驗證的任何使用者	`SessionDuration` URL 中的 HTML 參數	15 分鐘 | 12 小時 | 1 小時

[1] 使用來自一個角色的憑證來擔任不同的角色稱為角色鏈結。當您使用角色鏈結時，角色的工作階段持續時間限制為一小時。這適用於 AWS 管理主控台 角色切換 AWS CLI和 API 操作。此限制不適用於從使用者憑證初始擔任角色，也不適用於在使用執行個體設定檔的 Amazon EC2 執行個體上執行的應用程式。

[2] 此設定的值可介於 1 小時至 12 小時。有關修改最大工作階段持續時間設定的詳細資訊，請參閱 IAM 角色管理。此設定決定取得角色憑證時可以請求的最大工作階段持續時間。例如，當您使用 AssumeRole* API 操作來擔任角色時，可以使用 `DurationSeconds` 參數指定工作階段長度。使用此參數指定 900 秒 (15 分鐘) 到角色的最大工作階段持續時間設定之間的角色工作階段長度。在主控台中切換角色的 IAM 使用者會被授予最長工作階段持續時間或使用者工作階段中的剩餘時間，以較短者為準。假設您在角色上設定 5 小時的最大持續時間。已登入主控台 10 小時的 IAM 使用者 (超出預設最大值 12 小時) 會切換至角色。可用的角色工作階段持續時間為 2 小時。如要了解如何查看角色的最大值，請參閱本頁後述的 更新角色的最大工作階段持續時間。

> **備註**
>
> - 工作階段持續時間設定上限不會限制由 AWS 服務擔任的工作階段。
> - Amazon EC2 IAM 角色憑證不受角色中設定的工作階段持續時間上限的限制。
> - 若要允許使用者在角色工作階段中再次擔任目前角色，請在角色信任政策中將角色 ARN 或 AWS 帳戶 ARN 指定為主體。 AWS 服務 會提供運算資源，例如 Amazon EC2、Amazon ECS、Amazon EKS 和 Lambda，可提供臨時登入資料並自動更新這些登入資料。此可確保您始終擁有一組有效的憑證。對於這些服務，不需要再次擔任目前的角色即可取得臨時憑證。但是，如果您想要傳遞工作階段標籤或一個工作階段政策，則需要再次擔任目前的角色。若要了解如何修改角色信任政策以新增主要角色 ARN 或 AWS 帳戶 ARN，請參閱 更新角色信任政策。

主題

- 從使用者切換至 IAM 角色 (主控台)
- 切換到 IAM 角色 (AWS CLI)
- 切換至 IAM 角色 (Tools for Windows PowerShell)
- 切換到 IAM 角色 (AWS API)
- 使用 IAM 角色為在 Amazon EC2 執行個體上執行的應用程式授予許可
- 使用執行個體設定檔

從使用者切換至 IAM 角色 (主控台)

當您做為 IAM 使用者、IAM Identity Center 中的使用者、SAML 聯合角色或 Web 聯合身分角色登入時，才能切換角色。角色會指定一組許可，您可以使用這些許可來存取您需要 AWS 的資源。不過，您不登入角色，但一旦以 IAM 使用者身分登入後，就可以切換到 IAM 角色。這會暫時擱置了原始使用者許可，而不是為您提供指派給該角色的許可。該角色可以在您自己的帳戶或任何其他 AWS 帳戶中。如需有關角色、其優勢以及建立方式的詳細資訊，請參閱 IAM 角色 和 IAM 角色建立。

您的使用者以及您切換到任何角色的許可都不會累計。每次只有一組許可是作用中。當您切換角色時，您會暫時放棄使用者許可並使用指派給該角色的許可。當您退出角色後，您的使用者許可會自動恢復。

當您在 中切換角色時 AWS 管理主控台，主控台一律會使用您的原始登入資料來授權切換。例如，如果您切換到 RoleA，IAM 會使用您的原始憑證確定是否允許您擔任 RoleA。如果您接著在使用 RoleA RoleA 時切換到 RoleB， AWS 仍然會使用原始登入資料來授權切換，而不是 RoleA 的登入資料。

> **Note**
>
> 以 IAM Identity Center 中的使用者、SAML 聯合角色或 Web 聯合身分角色登入時，您將在啟動工作階段時擔任 IAM 角色。例如，當 IAM Identity Center 中的使用者登入 AWS 存取入口網站時，他們可以選擇許可集或存取透過帳戶存取管理員指派的角色，然後才能存取 AWS 資源。

角色工作階段

當您切換角色時，您的 AWS 管理主控台 工作階段預設會持續 1 小時。IAM 使用者工作階段預設為 12 小時，其他使用者可能已定義不同的工作階段持續時間。在主控台中切換角色時，您會取得角色工作階段持續時間上限或使用者工作階段中的剩餘時間 (以較短者為準)。您無法透過擔任角色來延長工作階段持續時間。

例如，假設為角色設定的最大工作階段持續時間為 10 小時。您決定切換至該角色時，已登入主控台 8 小時。使用者工作階段還剩餘 4 小時，因此允許的角色工作階段持續時間為 4 小時，而不是工作階段持續時間上限 10 小時。下表顯示在主控台中切換角色時如何判斷 IAM 使用者的工作階段持續時間。

IAM 使用者工作階段剩餘時間為...	角色工作階段持續時間為...
小於角色最大工作階段持續時間	使用者工作階段中的剩餘時間
大於角色最大工作階段持續時間	最大工作階段持續時間值
等於角色最大工作階段持續時間	最大工作階段持續時間值 (近似值)

使用來自一個角色的憑證來擔任不同的角色稱為角色鏈結。使用角色鏈結時，無論為個別角色設定的工作階段持續時間上限為何，工作階段持續時間都將限制為一小時。這適用於 AWS 管理主控台 角色切換 AWS CLI和 API 操作。

> **Note**
>
> 有些 AWS 服務主控台可以在角色工作階段過期時自動續約，而無需您採取任何動作。有些主控台可能會提示您重新載入瀏覽器頁面以重新驗證您的工作階段。

考量事項

- 如果您以 身分登入，則無法切換角色 AWS 帳戶根使用者。
- 必須授予使用者依政策切換角色的許可。如需說明，請參閱向使用者授予切換角色的許可。
- 您無法將 中的角色切換 AWS 管理主控台 為需要 ExternalId 值的角色。您只能透過呼叫支援 `ExternalId` 參數的 `AssumeRole` API 來切換到此類角色。

若要切換至角色

1. 按照 AWS 登入 User Guide 中的 Sign in to the AWS 管理主控台 中適合使用者類型的登入程序操作。
2. 在 中 AWS 管理主控台，選擇右上角導覽列上的使用者名稱。它通常如下：*username*@*account_ID_number_or_alias*。
3. 選取下列其中一種方法來切換角色：
   - 選擇切換角色。
   - 若已選擇加入多工作階段支援，請選擇新增工作階段，然後選取切換角色。

   > **Note**
   >
   > 您可以在 AWS 管理主控台中的單一 Web 瀏覽器中同時登入最多五個不同的身分。如需詳細資訊，請參閱 AWS 管理主控台 Getting Started Guide 中的 Signing in to multiple accounts。

4. 在 Switch Role (切換角色) 頁面上，輸入帳戶 ID 號碼或帳戶別名以及管理員提供的角色名稱。

   > **Note**
   >
   > 如果管理員已使用路徑 (例如 `division_abc/subdivision_efg/roleToDoX`) 建立角色，則必須在 Role (角色)) 方塊中輸入該完整路徑和名稱。如果僅輸入角色名稱，或者組合的 `Path` 和 `RoleName` 超過 64 個字元，則角色切換失敗。這是存放角色名稱的瀏覽器 cookie 的限制。如果發生這種情況，請聯絡您的管理員，並要求他們減小路徑和角色名稱的大小。

5. (選用) 您可以輸入顯示名稱，然後選取在主控台導覽列中反白顯示角色的顯示顏色。
   - 在顯示名稱欄位中，鍵入當此角色處於作用中時，要在導覽列上顯示之用以取代使用者名稱的文字。根據帳戶和角色資訊建議使用名稱，但您可以將其變更為對您有意義的任何名稱。
   - 在顯示顏色欄位中，選取顏色以反白顯示顯示名稱。

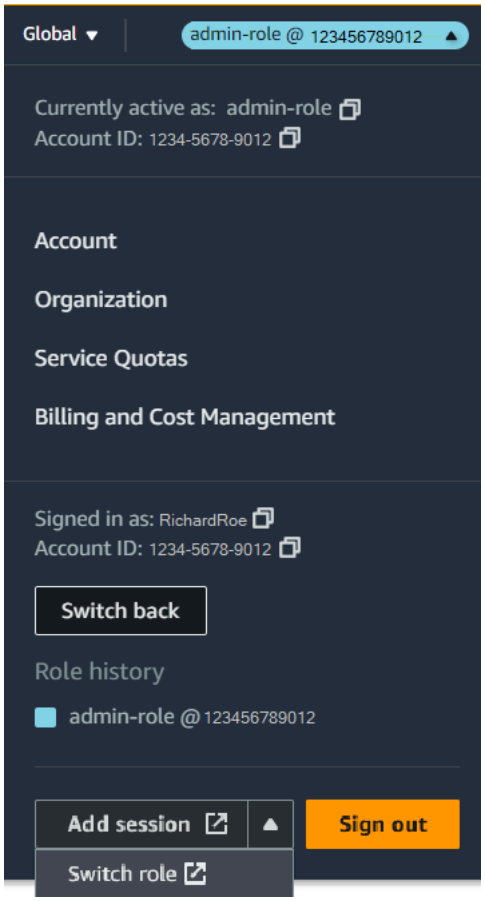
   名稱和顏色有助於在此角色處於作用中時提醒您，這會變更您的許可。例如，對於允許您存取測試環境的角色，您可以指定 **`Test`** 的 Display name (顯示名稱) 並在 Color (顏色)中選擇綠色。對於允許您存取生產的角色，您可以指定 **`Production`** 的 Display name (顯示名稱) 並在 Color (顏色) 中選擇紅色。

6. 選擇 Switch Role (切換角色)。顯示名稱和顏色將替換導覽列上的使用者名稱，您可以開始使用該角色授予您的許可。
7. 完成需要 IAM 角色的任務後，您可以切換回原始工作階段。這將移除角色提供的額外許可，並讓您返回標準許可。

   a. 在 IAM 主控台中，選擇導覽列右上角的角色 Display Name (顯示名稱)。

   b. 選擇切換回。

      例如，假設您使用使用者名稱 123456789012 登入到帳戶編號 Richard。使用過 admin-role 角色後，您想要停止使用該角色並傳回您的原始許可。若要停止使用角色，請選擇 admin-role @ 123456789012，然後選擇切換回。



> **Tip**
>
> 您使用的最後幾個角色會顯示在選單。下次想要切換到其中一個角色時，您只需選擇所需的角色。如果角色未顯示在功能表上，則只需手動鍵入帳戶和角色資訊。

其他資源

- 向使用者授予切換角色的許可
- 授予使用者將角色傳遞至 AWS 服務的許可
- 建立角色以將許可授予 IAM 使用者
- 建立角色以將許可委派給 AWS 服務
- IAM 角色疑難排解

切換到 IAM 角色 (AWS CLI)

「角色」指定一組許可，您可以使用它來存取所需的 AWS 資源。從這個意義上說，類似於 AWS Identity and Access Management中的使用者 (IAM)。當您以使用者身分登入時，您將取得一組特定的許可。不過，您不登入角色，但以使用者身分登入後，就可以切換角色。這會暫時擱置了原始使用者許可，而不是為您提供指派給該角色的許可。該角色可以在您自己的帳戶或任何其他 AWS 帳戶中。如需有關角色、其優勢以及建立和設定方式的詳細資訊，請參閱 IAM 角色 和 IAM 角色建立。要了解在擔任角色時使用的各種方法，請參閱 擔任角色的方法。

> **Important**
>
> 您的 IAM 使用者以及您擔任的任何角色的許可都不會累計。每次只有一組許可是作用中。當您擔任角色時，您會暫時放棄以前的使用者或角色許可，並使用指派給該角色的許可。當您退出角色後，您的使用者許可會自動恢復。

當您以 IAM 使用者身分登入時，您可以使用角色來執行 AWS CLI 命令。當您以已使用角色的外部身分驗證使用者 (SAML 或 OIDC) 登入時，您也可以使用角色來執行 AWS CLI 命令。此外，您可以使用角色從 Amazon EC2 執行個體執行 AWS CLI 命令，而該執行個體透過執行個體描述檔連接到角色。當您以 AWS 帳戶根使用者身分登入時，則無法擔任該角色。

角色鏈結 – 您也可以使用角色鏈結，以使用角色的許可來存取第二個角色。

在預設情況下，您的角色工作階段持續一小時。使用 `assume-role*` CLI 操作擔任此角色時，可以為 `duration-seconds` 參數指定值。此值的範圍可以從 900 秒 (15 分鐘) 到角色的最大工作階段持續時間設定的值。如果在主控台中切換角色，您工作階段的持續時間會限制為最多一小時。若要了解如何檢視角色的最大值，請參閱 更新角色的最大工作階段持續時間。

如果使用角色鏈結時，則工作階段持續時間最多限制為一小時。如果您隨後使用 `duration-seconds` 參數提供大於一小時的值，則操作會失敗。

範例案例：切換到生產角色

假設您是在開發環境中工作的 IAM 使用者。在此情況下，您偶爾需要透過 AWS CLI 在命令列中使用生產環境。您已經有存取金鑰憑證組可供您使用。這可以是指派給標準 IAM 使用者的存取金鑰對。或者，如果您以 SAML 或 OIDC 聯合身分主體的身分登入，則它可以是最初指派給您的角色的存取金鑰對。如果您目前的許可授予您擔任特定 IAM 角色的能力，則可以在 AWS CLI 組態檔案中的「設定檔」中識別該角色。然後使用指定 IAM 角色的許可來執行該命令，而不是原始身分。請注意，當您在 AWS CLI 命令中指定該設定檔時，您正在使用新角色。在這種情況下，您無法同時在開發帳戶中使用原始許可。原因是一次只有一組許可有效。

> **Note**
>
> 基於安全考量，管理員可以檢閱 AWS CloudTrail 日誌，以了解在其中執行動作的人員 AWS。當您擔任角色時，系統管理員可能需要您指定來源身分或角色工作階段名稱 。如需詳細資訊，請參閱`sts:SourceIdentity`及`sts:RoleSessionName`。

切換到生產角色 (AWS CLI)

1. 如果您從未使用 AWS CLI，則必須先設定預設的 CLI 設定檔。開啟命令提示字元並設定您的 AWS CLI 安裝，以使用來自 IAM 使用者或聯合角色的存取金鑰。如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的設定 AWS Command Line Interface。

   執行 aws configure 命令，如下所示：

   ```
   aws configure
   ```

   當出現提示時，請提供下列資訊：

   ```
   AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   Default region name [None]: us-east-2
   Default output format [None]: json
   ```

2. 在 Unix 或 Linux `.aws/config` 檔案中，或在 Windows `C:\Users\USERNAME\.aws\config` 檔案中建立角色的新描述檔。下列範例會建立稱為 `prodaccess` 的描述檔，以切換到 *`ProductionAccessRole`* 帳戶中的角色 `123456789012`。您從建立角色的帳戶管理員處取得角色 ARN。叫用此設定檔時， AWS CLI 會使用 的登入資料`source_profile`來請求角色的登

入資料。因此，做為 source_profile 參考的身分必須具有 sts:AssumeRole 中指定角色的 role_arn 許可。

```
[profile prodaccess]
    role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
    source_profile = default
```

3. 建立新設定檔之後，指定 參數的任何 AWS CLI 命令都會在連接到 IAM 角色的許可下--profile prodaccess執行，ProductionAccessRole而不是預設使用者。

```
aws iam list-users --profile prodaccess
```

如果指派給 ProductionAccessRole 的許可啟用列出目前 AWS 帳戶中的使用者，則此命令有效。

4. 若要傳回原始憑證授予的許可，請執行不帶 --profile 參數的命令。會使用您在 中設定的預設設定檔中的登入資料 AWS CLI 還原至 Step 1。

如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的擔任角色。

範例案例：允許執行個體描述檔角色來切換到另一個帳戶中的角色

假設您使用兩個 AWS 帳戶，並且您想要允許在 Amazon EC2 執行個體上執行的應用程式在兩個帳戶中執行AWS CLI命令。假設 EC2 執行個體存在於帳戶 111111111111。該執行個體包含 abcd 執行個體描述檔角色，該角色會允許應用程式在相同 111111111111 帳戶中對 amzn-s3-demo-bucket1 儲存貯體執行唯讀 Amazon S3 任務。不過，也必須允許應用程式擔任 efgh 跨帳戶角色來在帳戶 222222222222 中執行任務。若要執行此操作，abcd EC2 執行個體描述檔角色必須有以下許可政策：

帳戶 111111111111 ***abcd*** 角色許可政策

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

假設 efgh 跨帳戶角色允許在相同 222222222222 帳戶中對 amzn-s3-demo-bucket2 儲存貯體的唯讀 Amazon S3 任務。若要執行此操作，efgh 跨帳戶角色必須有以下許可政策：

帳戶 222222222222 ***efgh*** 角色許可政策

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

efgh 角色必須允許 abcd 執行個體設定檔角色擔任該角色。若要執行此操作，efgh 角色必須有以下信任政策：

帳戶 222222222222 ***efgh*** 角色信任政策

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

若要接著在帳戶 中執行 AWS CLI 命令222222222222，您必須更新 CLI 組態檔案。在 AWS CLI 組態檔案中，將 efgh 角色識別為「描述檔」且 abcd EC2 執行個體描述檔角色識別為「憑證來源」。然後，使用 efgh 角色 (而不是原始 abcd 角色) 的許可執行 CLI 命令。

> **Note**
>
> 基於安全考量，您可以使用 AWS CloudTrail 來稽核帳戶中角色的使用。若要在 CloudTrail 日誌中的不同主體使用角色時區分角色工作階段，您可以使用角色工作階段名稱。當 依本主題所述代表使用者 AWS CLI 擔任角色時，角色工作階段名稱會自動建立為 `AWS-CLI-session-nnnnnnnn`。以下 *nnnnnnnn* 是整數，代表 Unix epoch 時間中的時間 (自 1970 年 1 月 1 日午夜 UTC 的秒數)。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 CloudTrail 事件參考。

讓 EC2 執行個體設定檔角色切換到跨帳戶角色 (AWS CLI)

1. 您不需要設定預設 CLI 描述檔。您反而可以從 EC2 執行個體描述檔中繼資料載入憑證。在 `.aws/config` 檔案中為角色建立新的設定檔。下列範例會建立 `instancecrossaccount` 的描述檔，以切換到 *efgh* 帳戶中的角色 222222222222。呼叫此描述檔時， AWS CLI 會使用 EC2 執行個體描述檔中繼資料的憑證來請求角色的憑證。因此，EC2 執行個體設定檔角色必須具有在 `sts:AssumeRole` 中指定角色的 `role_arn` 許可。

   ```
   [profile instancecrossaccount]
   role_arn = arn:aws:iam::222222222222:role/efgh
   credential_source = Ec2InstanceMetadata
   ```

2. 建立新設定檔之後，指定 參數的任何 AWS CLI 命令都會在連接到帳戶 中efgh角色的許可下`--profile instancecrossaccount`執行222222222222。

   ```
   aws s3 ls amzn-s3-demo-bucket2 --profile instancecrossaccount
   ```

   如果指派給 efgh 角色的許可允許列出目前 AWS 帳戶中的使用者，則此命令有效。

3. 若要在帳戶 111111111111 中返回原始 EC2 執行個體設定檔許可，在不使用 `--profile` 參數的情形下執行 CLI 命令。

如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的擔任角色。

切換至 IAM 角色 (Tools for Windows PowerShell)

「角色」指定一組許可，您可以使用它來存取所需的 AWS 資源。從這個意義上說，類似於 AWS Identity and Access Management中的使用者 (IAM)。當您以使用者身分登入時，您將取得一組特定的許可。不過，您不登入角色，但一旦登入後就可以切換角色。這會暫時擱置了原始使用者許可，而不是為您提供指派給該角色的許可。該角色可以在您自己的帳戶或任何其他 AWS 帳戶中。如需有關角色、其優勢以及建立和設定方式的詳細資訊，請參閱 IAM 角色 和 IAM 角色建立。

> **Important**
>
> 您的 IAM 使用者以及您切換到任何角色的許可都不會累計。每次只有一組許可是作用中。當您切換角色時，您會暫時放棄使用者許可並使用指派給該角色的許可。當您退出角色後，您的使用者許可會自動恢復。

本節說明在 AWS Tools for Windows PowerShell命令列中如何切換角色。

假設您在開發環境中有一個帳戶，偶爾需要使用 Tools for Windows PowerShell 在命令列中使用生產環境。您已經有一個存取金鑰憑證組可供您使用。這些可以是指派給標準 IAM 使用者的存取金鑰對。或者，如果您以 SAML 或 OIDC 聯合身分主體身分登入，則它們可以是最初指派給您的角色的存取金鑰對。您可以使用這些憑證來執行 `Use-STSRole` cmdlet，該 cmdlet 將新角色的 ARN 做為參數傳送。該命令傳回所請求角色的臨時安全憑證。然後，您可以在後續 PowerShell 命令中使用這些憑證，並使用角色的許可存取生產中的資源。使用該角色時，您無法在開發帳戶中使用您的使用者許可，因為一次只能有一組許可有效。

> **Note**
>
> 基於安全考量，管理員可以檢閱 AWS CloudTrail 日誌，以了解在其中執行動作的人員 AWS。當您擔任角色時，系統管理員可能需要您指定來源身分或角色工作階段名稱 。如需詳細資訊，請參閱`sts:SourceIdentity`及`sts:RoleSessionName`。

請注意，所有存取金鑰和權杖僅為範例，不能如下所示般使用。以您實際環境中的適當值取代。

切換至角色 (Tools for Windows PowerShell)

1. 開啟 PowerShell 命令提示字元並設定預設設定檔以使用來自目前 IAM 使用者或聯合身分角色的存取金鑰。如果您之前已使用 Tools for Windows PowerShell，則可能已經完成。請注意，只有在以 IAM 使用者身分 (而非 AWS 帳戶根使用者) 登入時才能切換角色。

```
PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
```

如需詳細資訊，請參閱AWS Tools for PowerShell 《 使用者指南》中的使用 AWS 登入資料。

2. 若要擷取新角色的憑證，請執行下列命令以切換到 123456789012 帳戶中的 *RoleName* 角色。您從建立角色的帳戶管理員處取得角色 ARN。此命令還需要您提供工作階段名稱。您可以為此選擇任何文字。以下命令請求憑證，然後從傳回的結果物件中擷取 Credentials 屬性物件，並將其存放在 $Creds 變數中。

```
PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
RoleSessionName "MyRoleSessionName").Credentials
```

$Creds 是一個物件，現在包含您在下列步驟中所需的 AccessKeyId、 SecretAccessKey 和 SessionToken 元素。以下範例命令說明典型的值：

```
PS C:\> $Creds.AccessKeyId
AKIAIOSFODNN7EXAMPLE

PS C:\> $Creds.SecretAccessKey
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

PS C:\> $Creds.SessionToken
AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. 若要將這些憑證用於任何後續命令，請將它們包含在 -Credential 參數中。例如，以下命令使用角色中的憑證，僅在角色被授予 iam:ListRoles 許可且因此可以執行 Get-IAMRoles cmdlet 時才起作用：

```
PS C:\> get-iamroles -Credential $Creds
```

4. 若要傳回原始憑證，只需停止使用 `-Credentials $Creds` 參數，並允許 PowerShell 恢復存放在預設設定檔中的憑證。

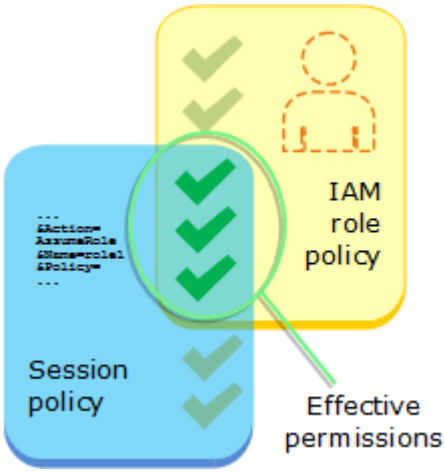
切換到 IAM 角色 (AWS API)

「角色」指定一組許可，您可以使用它來存取 AWS 資源。從這個意義上說，類似於 IAM 使用者。委託人（人員或應用程式）擔任角色來接收臨時許可，以執行必要的任務並與 AWS 資源互動。該角色可以在您自己的帳戶或任何其他 AWS 帳戶中。如需有關角色、其優勢以及建立和設定方式的詳細資訊，請參閱 IAM 角色 和 IAM 角色建立。要了解在擔任角色時使用的各種方法，請參閱 擔任角色的方法。

> **Important**
>
> 您的 IAM 使用者以及您擔任的任何角色的許可都不會累計。每次只有一組許可是作用中。當您擔任角色時，您會暫時放棄以前的使用者或角色許可，並使用指派給該角色的許可。當您退出角色後，原本的許可會自動恢復。

若要擔任角色，應用程式會 AWS STS `AssumeRole`呼叫 API 操作，並傳遞要使用的角色 ARN。此操作會建立使用臨時憑證的新工作階段。此工作階段的許可，即為以身分為基礎的政策指派給角色的許可。

當您呼叫 `AssumeRole` 時，您可以選擇性地傳遞內嵌或受管工作階段政策。工作階段政策是一種進階政策，且您會在以程式設計方式建立角色或聯合身分使用者工作階段的臨時憑證工作階段時，以參數方式傳遞。您可以使用 `Policy` 參數傳遞單一 JSON 內嵌工作階段政策文件。您可以使用 `PolicyArns` 參數，指定多達 10 個受管工作階段政策。所產生工作階段的許可會是實體的身分類型政策和工作階段政策的交集。當您需要將角色的臨時憑證提供給某人時，工作階段政策便可派上用場。他們可以在後續 AWS API 呼叫中，使用角色的臨時憑證來存取擁有該角色的帳戶中的資源。您無法使用工作階段政策來授予超出即將以身分為基礎政策所允許的許可。若要進一步了解 如何 AWS 決定角色的有效許可，請參閱 政策評估邏輯。



當您以 IAM 使用者身分登入，或者作為已使用角色的外部身分驗證使用者 (SAML 或 OIDC) 登入時，可以呼叫 AssumeRole。您還可以使用角色鏈結，它使用角色來擔任第二個角色。當您以 AWS 帳戶根使用者身分登入時，則無法擔任該角色。

在預設情況下，您的角色工作階段持續一小時。當您使用 AWS STS AssumeRole* API 操作擔任此角色時，您可以指定 DurationSeconds 參數的值。此值的範圍可以從 900 秒 (15 分鐘) 到角色的最大工作階段持續時間設定的值。若要了解如何檢視角色的最大值，請參閱 更新角色的最大工作階段持續時間。

如果使用角色鏈結時，則工作階段最多限制為一小時。如果您隨後使用 DurationSeconds 參數提供大於一小時的值，則操作會失敗。

> **Note**
>
> 基於安全考量，管理員可以檢閱 AWS CloudTrail 日誌，以了解在其中執行動作的人員 AWS。當您擔任角色時，系統管理員可能需要您指定來源身分或角色工作階段名稱 。如需詳細資訊，請參閱sts:SourceIdentity及sts:RoleSessionName。

下列程式碼範例示範如何建立使用者並擔任角色。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

- 建立沒有許可的使用者。

- 建立一個可授予許可的角色，以列出帳戶的 Amazon S3 儲存貯體。
- 新增政策，讓使用者擔任該角色。
- 使用暫時憑證，擔任角色並列出 S3 儲存貯體，然後清理資源。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }
```

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
```

```
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
```

```
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
```

```
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
```

```
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
```

```
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
```

```
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

```
    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
```

```
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
 ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
 ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
```

```
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
```

```
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();
```

```
        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
```

```
            "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
```

```
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");
```

```
        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
complete.");
        Console.WriteLine("Please wait while we clean up the resources we
created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();
```

```
        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
```

```
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }
```

```
    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
```

```
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
(IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
```

```
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI 搭配 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
```

```
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
```

```
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
```

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

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
```

```
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret

  local buckets
  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. This should not have
happened."
  else
    errecho "Because the role with permissions has not been assumed, listing
buckets failed."
  fi

  echo
  echo_repeat "*" 88
  echo "Now assume the role $iam_role_name and list the buckets."
  echo_repeat "*" 88
  echo

  local credentials

  credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. Listing buckets
succeeded because of "
   echo "the assumed role."
 else
   errecho "Failed to list buckets. This should not happen."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   export AWS_SESSION_TOKEN=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 local result=0
 export AWS_ACCESS_KEY_ID=""
 export AWS_SECRET_ACCESS_KEY=""

 echo
 echo_repeat "*" 88
```

```
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

此案例中使用的 IAM 函數。

```
###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}
```

```
  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
```

```
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user already exists, we don't want to try to create it.
if (iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name already exists in the account."
  return 1
fi

response=$(aws iam create-user --user-name "$user_name" \
  --output text \
  --query 'User.Arn')

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-user operation failed.$response"
  return 1
fi

echo "$response"
```

```
  return 0
}

################################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
```

```
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
```

```
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_policy"
  echo "Creates an AWS Identity and Access Management (IAM) policy."
  echo "  -n policy_name   The name of the IAM policy."
  echo "  -p policy_json -- The policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
```

```
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
```

```
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

################################################################################
```

```
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_delete_policy"
  echo "Deletes an AWS Identity and Access Management (IAM) policy"
  echo "  -n policy_arn -- The name of the IAM policy arn."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
```

```
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
```

```
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
```

```
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
std::endl;
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;
```

```
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
```

```
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }
```

```
        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
```

```
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());
```

```
        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
```

```
            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }
```

```
        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在命令提示中執行互動式案例。

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/aws-sdk-go-v2/service/s3"
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
```

```
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

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

```
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

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

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
```

```
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
```

```
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
```

```
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
```

```
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
```

```
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
*key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
*user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
```

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

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
 Prefix string
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

定義包裝帳號動作的結構。

```
import (
 "context"
 "log"

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

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

定義包裝政策動作的結構。

```
import (
 "context"
 "encoding/json"
 "log"

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

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
```

```
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
```

```
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

定義包裝角色動作的結構。

```
import (
 "context"
 "encoding/json"
 "log"
```

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
```

```
    Principal: map[string]string{"AWS": trustedUserArn},
    Action:    []string{"sts:AssumeRole"},
   }},
  }
  policyBytes, err := json.Marshal(trustPolicy)
  if err != nil {
   log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
  trustedUserArn, err)
   return nil, err
  }
  result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
   AssumeRolePolicyDocument: aws.String(string(policyBytes)),
   RoleName:                 aws.String(roleName),
  })
  if err != nil {
   log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
  } else {
   role = result.Role
  }
  return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
```

```
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
```

```
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
```

```
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

定義包裝使用者動作的結構。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"
```

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
```

```
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
```

```
    Resource: aws.String(roleArn),
   }},
  }
  policyBytes, err := json.Marshal(policyDoc)
  if err != nil {
   log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
  err)
   return err
  }
  _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
   PolicyDocument: aws.String(string(policyBytes)),
   PolicyName:     aws.String(policyName),
   UserName:       aws.String(userName),
  })
  if err != nil {
   log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
  err)
  }
  return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
```

```
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
```

```
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 使用者動作的函數。

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
```

```
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
 assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
 objects are read.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
```

```
        String policyName = args[1];
        String roleName = args[2];
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
```

```
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

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

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

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

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

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

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();
```

```
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
   public static void deleteRole(IamClient iam, String roleName, String polArn)
{

       try {
           // First the policy needs to be detached.
           DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                   .policyArn(polArn)
                   .roleName(roleName)
                   .build();

           iam.detachRolePolicy(rolePolicyRequest);

           // Delete the policy.
           DeletePolicyRequest request = DeletePolicyRequest.builder()
                   .policyArn(polArn)
                   .build();

           iam.deletePolicy(request);
           System.out.println("*** Successfully deleted " + polArn);

           // Delete the role.
           DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           iam.deleteRole(roleRequest);
           System.out.println("*** Successfully deleted " + roleName);

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

   public static void deleteKey(IamClient iam, String username, String
accessKey) {
       try {
           DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                   .accessKeyId(accessKey)
                   .userName(username)
                   .build();

           iam.deleteAccessKey(request);
```

```
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

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

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

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

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立一個可授予許可的 IAM 使用者和角色，以列出 Amazon S3 儲存貯體。使用者只有擔任該角色的權利。擔任角色後，請使用暫時性憑證列出該帳戶的儲存貯體。

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";

/**
```

```
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
```

```
 }

 // Create an access key. This key is used to authenticate the new user to
 // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
(AWS STS).
 // It's not best practice to use access keys. For more information, see
https://aws.amazon.com/iam/resources/best-practices/.
 const createAccessKeyResponse = await iamClient.send(
   new CreateAccessKeyCommand({ UserName: userName }),
 );

 if (
   !createAccessKeyResponse.AccessKey?.AccessKeyId ||
   !createAccessKeyResponse.AccessKey?.SecretAccessKey
 ) {
   throw new Error("Access key not created");
 }

 const {
   AccessKey: { AccessKeyId, SecretAccessKey },
 } = createAccessKeyResponse;

 let s3Client = new S3Client({
   credentials: {
     accessKeyId: AccessKeyId,
     secretAccessKey: SecretAccessKey,
   },
 });

 // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
 // thrown while the user and access keys are still stabilizing.
 await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
   try {
     return await listBuckets(s3Client);
   } catch (err) {
     if (err instanceof Error && err.name === "InvalidAccessKeyId") {
       throw err;
     }
   }
 });

 // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
 // is thrown while the user and access keys are still stabilizing.
```

```
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
      ),
  );

  if (!Role) {
    throw new Error("Role not created");
  }

  // Create a policy that allows the user to list S3 buckets.
  const { Policy: listBucketPolicy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Action: ["s3:ListAllMyBuckets"],
            Resource: "*",
          },
        ],
      }),
      PolicyName: policyName,
```

```
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
```

```
      accessKeyId: Credentials.AccessKeyId,
      secretAccessKey: Credentials.SecretAccessKey,
      sessionToken: Credentials.SessionToken,
    },
  });

  // List the S3 buckets again.
  // Retry the list buckets operation until it succeeds. AccessDenied might
  // be thrown while the role policy is still stabilizing.
  await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
    listBuckets(s3Client),
  );

  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
```

```
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 使用者動作的函數。

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
```

```
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
```

```
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
```

```
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
```

```
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
```

```
    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
```

```
    return jsonParser.parse(reader)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
```

```
use Iam\IAMService;

echo("\n");
echo("--------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("--------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
```

```
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
```

```
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立一個可授予許可的 IAM 使用者和角色，以列出 Amazon S3 儲存貯體。使用者只有擔任該角色的權利。擔任角色後，請使用暫時性憑證列出該帳戶的儲存貯體。

```
import json
import sys
import time
```

```
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
```

```
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
```

```
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
"
            f"{error.response['Error']['Message']}"
        )
```

```
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
```

```
                              grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
```

```
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
```

```
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- 如需 API 的詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立一個可授予許可的 IAM 使用者和角色，以列出 Amazon S3 儲存貯體。使用者只有擔任該角色的權利。擔任角色後，請使用暫時性憑證列出該帳戶的儲存貯體。

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
```

```
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
```

```
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
```

```
 def create_user_policy(policy_name, user, role)
   policy_document = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Action: 'sts:AssumeRole',
       Resource: role.arn
     }]
   }.to_json
   @iam_client.put_user_policy(
     user_name: user.user_name,
     policy_name: policy_name,
     policy_document: policy_document
   )
   puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
     count -= 1
```

```
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == 'AccessDenied'
      puts('Attempt to list buckets with no permissions: AccessDenied.')
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
  # This is separated into a factory function so that it can be mocked for unit
testing.
  #
  # @param key_id [String] The ID of the access key used by the STS client.
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end

  # Deletes a role. If the role has policies attached, they are detached and
  # deleted before the role is deleted.
```

```
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    @iam_client.list_attached_role_policies(role_name:
 role_name).attached_policies.each do |policy|
      @iam_client.detach_role_policy(role_name: role_name, policy_arn:
 policy.policy_arn)
      @iam_client.delete_policy(policy_arn: policy.policy_arn)
      @logger.info("Detached and deleted policy #{policy.policy_name}.")
    end
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Role deleted: #{role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
```

```
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy

- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
```

```
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
```

```
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
```

```
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
```

```
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
```

```
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

使用 IAM 角色為在 Amazon EC2 執行個體上執行的應用程式授予許可

在 Amazon EC2 執行個體上執行的應用程式必須在 AWS API 請求中包含 AWS 登入資料。您可以讓您的開發人員直接將 AWS 登入資料存放在 Amazon EC2 執行個體中，並允許該執行個體中的應用程式使用這些登入資料。不過，開發人員將必須管理憑證，並確實將憑證安全地傳遞給每個執行個體，並在需要更新憑證時更新每個 Amazon EC2 執行個體。這是許多額外的工作。

與其這麼做，您可以 (也應該) 使用 IAM 角色來管理 Amazon EC2 執行個體上所執行應用程式的臨時憑證。使用角色時，您不必將長期憑證 (例如登入憑證或存取金鑰) 分發給 Amazon EC2 執行個體。相反地，該角色會提供暫時許可，供應用程式在呼叫其他 AWS 資源時使用。啟動 Amazon EC2 執行個體時，指定要與執行個體相關聯的 IAM 角色。然後，在執行個體上執行的應用程式可以使用角色提供的暫時憑證來簽署 API 請求。

如要透過角色為 Amazon EC2 執行個體上執行的應用程式授予許可，需搭配一些額外的組態。在 Amazon EC2 執行個體上執行的應用程式是由 AWS 虛擬化作業系統從 抽象化。由於此額外區隔，您需要額外的步驟，才能將 AWS 角色及其相關許可指派給 Amazon EC2 執行個體，並將其提供給其應用程式。此額外步驟是建立連接到執行個體的執行個體設定檔。執行個體描述檔包含角色，可以將角色的臨時憑證提供給在執行個體上執行的應用程式。然後，可以在應用程式的 API 呼叫中使用這些臨時憑證來存取資源，並限制僅存取角色指定的那些資源。

> **Note**
>
> 一次只能指派一個角色給 Amazon EC2 執行個體，執行個體上的所有應用程式會共用相同的角色和許可。當您利用 Amazon ECS 來管理 Amazon EC2 執行個體時，您可以向 Amazon ECS 任務指派角色，這些角色和執行 Amazon EC2 執行個體所使用的角色有所區別。為每項任務指派一個角色的做法符合最低權限存取原則，並且允許對動作和資源進行更精細的控制。
> 如需詳細資訊，請參閱《Amazon Elastic Container Service 最佳實務指南》中的對 Amazon ECS 任務使用 IAM 角色。

以這種方式使用角色有幾個好處。由於角色憑證是臨時的並且是自動更新的，因此您不需要管理憑證，也不必擔心長期的安全風險。此外，如果對多個執行個體使用單一角色，則可以對該角色進行變更，並將變更自動傳播到所有執行個體。

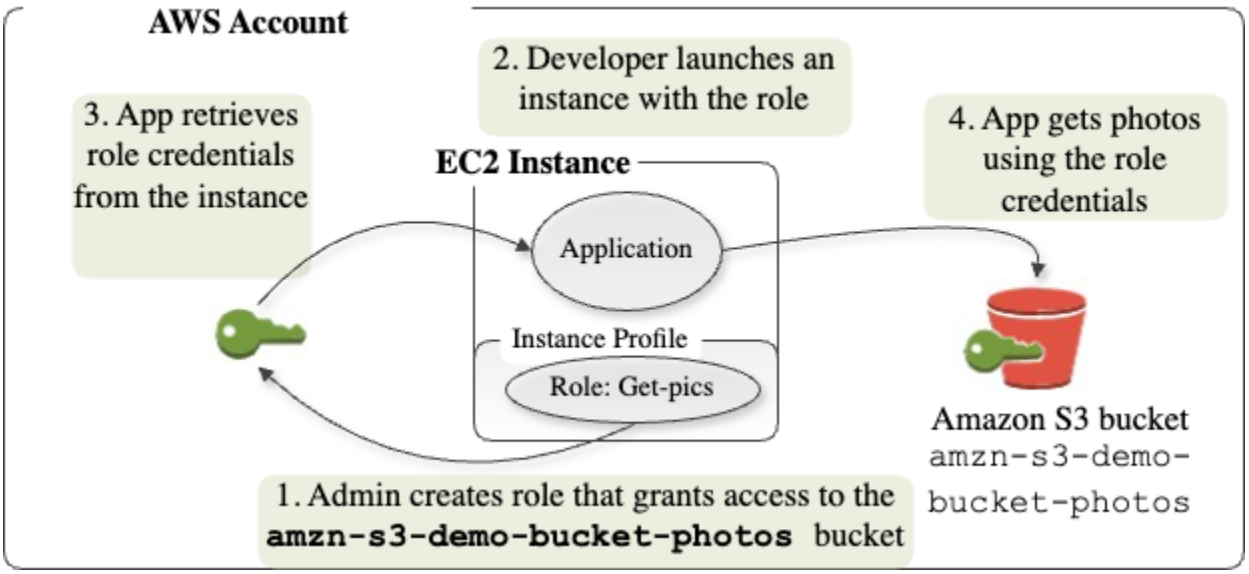
> **Note**
>
> 雖然在啟動角色時通常就會將角色指派給 Amazon EC2 執行個體，但角色也可以與目前執行的 Amazon EC2 執行個體連接。若要了解如何將角色連接到正在執行的執行個體，請參閱 Amazon EC2 的 IAM 角色。

主題

- 適用於 Amazon EC2 執行個體的角色如何運作？
- 使用 Amazon EC2 角色所需的許可
- 我該如何開始？
- 相關資訊

適用於 Amazon EC2 執行個體的角色如何運作？

下圖中，開發人員在 Amazon EC2 執行個體上執行應用程式，該應用程式需要存取名為 `amzn-s3-demo-bucket-photos` 的 S3 儲存貯體。管理員建立 `Get-pics` 服務角色，並將角色連接到 Amazon EC2 執行個體。該角色包括許可政策，該政策授予對指定的 S3 儲存貯體的唯讀存取許可。角色還包含信任政策，該政策可允許 Amazon EC2 執行個體擔任角色並擷取臨時憑證。當應用程式在執行個體上執行時，它可以使用角色的臨時憑證來存取照片儲存貯體。管理員不必授予開發人員存取照片儲存貯體的許可，開發人員也不必共用或管理憑證。



1. 管理員使用 IAM 建立 **Get-pics** 角色。在角色的信任政策中，管理員指定只有 Amazon EC2 執行個體可以擔任該角色。在角色的許可政策中，管理員為 amzn-s3-demo-bucket-photos 儲存貯體指定唯讀許可。
2. 開發人員啟動 Amazon EC2 執行個體，並將 Get-pics 角色指派給該執行個體。

> **Note**
>
> 如果您使用 IAM 主控台，則會為您管理執行個體描述檔，並且對您來說幾乎是透明的。不過，如果您使用 AWS CLI 或 API 來建立和管理角色和 Amazon EC2 執行個體，則必須建立執行個體描述檔，並將角色指派為個別步驟。然後，當您啟動執行個體時，必須指定執行個體設定檔名稱，而不是角色名稱。

3. 應用程式執行時，會從 Amazon EC2 執行個體中繼資料中取得暫時安全憑證，如從執行個體中繼資料中擷取安全憑證中所述。這些是代表角色的暫時安全憑證，並且在有限的時間段內有效。

   透過一些 AWS 開發套件，開發人員可以使用透明地管理暫時安全憑證的提供者。(AWS SDKs的文件說明該 SDK 支援用於管理登入資料的功能。)

   或者，應用程式可以直接從 Amazon EC2 執行個體的執行個體中繼資料中取得臨時憑證。您可以從中繼資料的 iam/security-credentials/*role-name* 類別 (在本例中為 iam/security-credentials/Get-pics) 中取得憑證和相關的值。如果應用程式從執行個體中繼資料取得憑證，則可以快取憑證。
4. 使用擷取到的臨時憑證，應用程式存取照片儲存貯體。基於連接至 **Get-pics** 角色的政策，應用程式具有唯讀許可。

   在執行個體上提供的臨時安全憑證在到期之前會自動更新，以便有效設定為永久可用。應用程式只需要確保在目前中繼資料到期之前，它從執行個體中繼資料中取得一組新的憑證。您可以使用 AWS SDK 來管理登入資料，因此應用程式不需要包含額外的邏輯來重新整理登入資料。例如，使用執行個體描述檔憑證提供者建立用戶端。不過，如果應用程式從執行個體中繼資料取得臨時安全憑證並快取它們，則應每小時或在目前設定過期之前至少 15 分鐘取得重新整理的憑證集。過期時間包含在 iam/security-credentials/*role-name* 類別中所傳回的資訊。

使用 Amazon EC2 角色所需的許可

若要使用角色啟動執行個體，開發人員必須具有啟動 Amazon EC2 執行個體的許可，以及傳遞 IAM 角色的許可。

下列範例政策允許使用者使用 AWS 管理主控台 來啟動具有 角色的執行個體。此政策包含萬用字元 (*)，以允許使用者傳遞任何角色，並執行列出的 Amazon EC2 動作。`ListInstanceProfiles` 動作讓使用者查看 AWS 帳戶中可用的所有角色。

Example範例政策，授予使用者使用 Amazon EC2 主控台啟動具有任何角色的執行個體的許可

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

限制哪些角色可以傳遞給 Amazon EC2 執行個體 (使用 PassRole)

您可以使用 `PassRole` 許可，限制使用者在啟動執行個體時可以將哪個角色傳遞給 Amazon EC2 執行個體。這有助於防止使用者執行具有比授予使用者更多許可的應用程式，也就是說，能夠獲得更高的

許可。例如，假設使用者 Alice 只擁有啟動 Amazon EC2 執行個體以及使用 Amazon S3 儲存貯體的許可，但她傳遞給 Amazon EC2 執行個體的角色具有使用 IAM 和 Amazon DynamoDB 的許可。在這種情況下，Alice 或許可以啟動執行個體、登入執行個體、取得暫時安全憑證，然後執行她未授權的 IAM 或 DynamoDB 動作。

若要限制使用者可以將哪些角色傳遞給 Amazon EC2 執行個體，則要建立允許 `PassRole` 動作的政策。然後，您需要將政策連接到預計啟動 Amazon EC2 執行個體的使用者 (或使用者所屬的 IAM 群組)。在政策的 `Resource` 元素中，您需列出允許使用者傳遞給 Amazon EC2 執行個體的角色。當使用者啟動執行個體並將角色與其關聯時，Amazon EC2 檢查是否允許使用者傳送該角色。當然，您也應當確保使用者可以傳遞的角色不包含比使用者應具有的許可更多的許可。

> **Note**
>
> `PassRole` 不是與`RunInstances` 或 `ListInstanceProfiles` 相同的 API 動作。反之，它是一種許可，每當角色 ARN 做為參數傳遞至 API 時（或主控台會代表使用者執行此操作），就會進行 AWS 檢查。它可協助管理員控制哪些使用者可以傳遞哪些角色。在這種情況下，它確保允許使用者將特定角色連接到 Amazon EC2 執行個體。

Example此範例政策授予使用者許可，使其可以啟動具有特定角色的 Amazon EC2 執行個體

以下範例政策可允許使用者使用 Amazon EC2 API 來啟動具有角色的執行個體。`Resource` 元素指定角色的 Amazon Resource Name (ARN)。透過指定 ARN，政策授予使用者僅傳遞 `Get-pics` 角色的許可。如果使用者在啟動執行個體時嘗試指定不同的角色，則動作會失敗。使用者沒有執行任何執行個體的許可，無論其是否有傳遞角色。

JSON

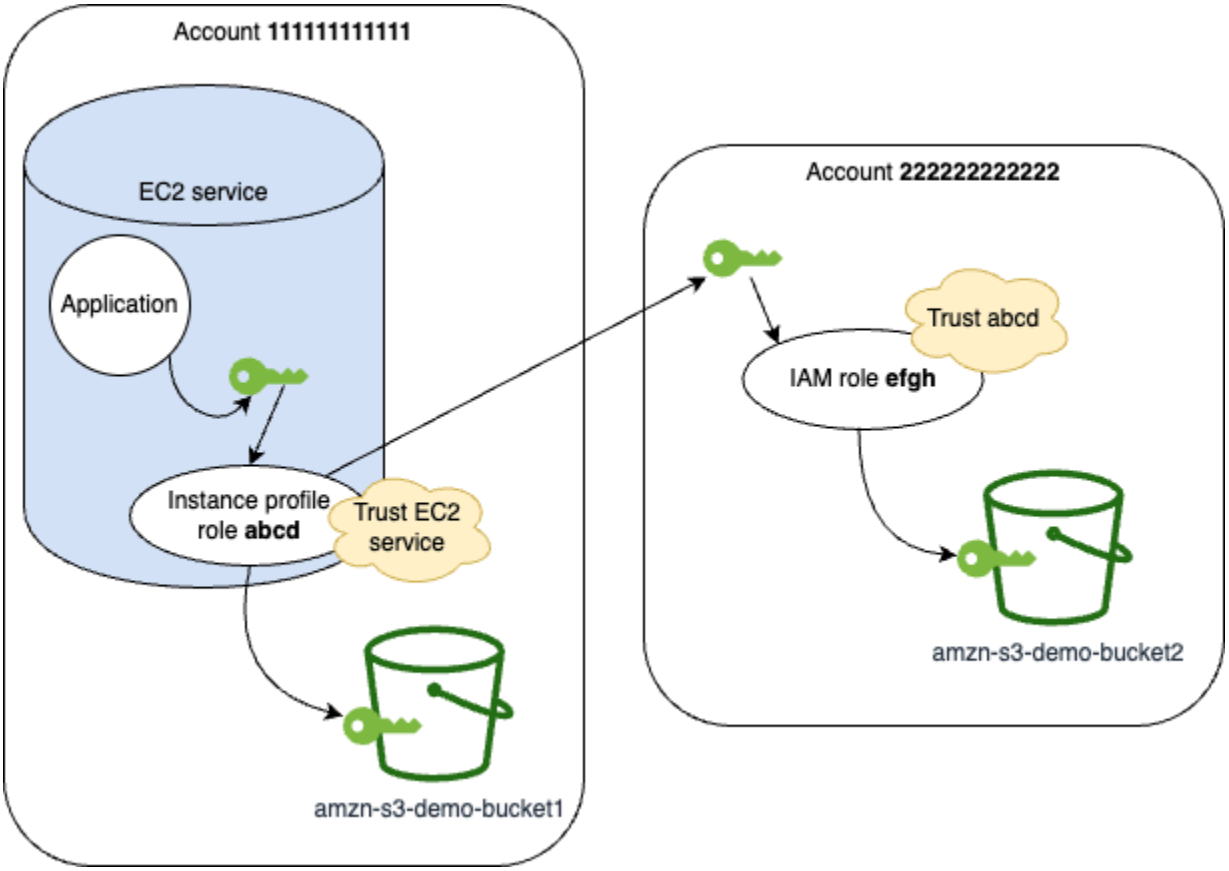
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
```

```
            "Resource": "arn:aws:iam::111122223333:role/Get-pics"
        }
    ]
}
```

允許執行個體設定檔角色切換到另一個帳戶中的角色

您可以允許在 Amazon EC2 執行個體上執行的應用程式在另一個帳戶中執行命令。若要執行此操作，您必須允許將第一個帳戶中的 Amazon EC2 執行個體角色切換到第二個帳戶中的角色。

假設您使用兩個 ，AWS 帳戶 而且您想要允許在 Amazon EC2 執行個體上執行的應用程式在兩個帳戶中執行AWS CLI命令。假設 Amazon EC2 執行個體存在於帳戶 111111111111。該執行個體包含 abcd 執行個體描述檔角色，該角色會允許應用程式在相同 111111111111 帳戶中對 amzn-s3-demo-bucket1 儲存貯體執行唯讀 Amazon S3 任務。不過，也必須允許應用程式擔任 efgh 跨帳戶角色來在帳戶 222222222222 中存取 amzn-s3-demo-bucket2Amazon S3 儲存貯體。



abcd Amazon EC2 執行個體設定檔角色必須具有以下許可政策，以允許應用程式存取 amzn-s3-demo-bucket1 Amazon S3 儲存貯體：

帳戶 111111111111 ***abcd*** 角色許可政策

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

abcd 角色必須信任 Amazon EC2 服務擔任該角色。若要執行此操作，abcd 角色必須有以下信任政策：

帳戶 111111111111 ***abcd*** 角色信任政策

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
}
```

假設 efgh 跨帳戶角色允許在相同 222222222222 帳戶中對 amzn-s3-demo-bucket2 儲存貯體的唯讀 Amazon S3 任務。若要執行此操作，efgh 跨帳戶角色必須有以下許可政策：

帳戶 222222222222 ***efgh*** 角色許可政策

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
```

```
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

efgh 角色必須信任 abcd 執行個體設定檔角色擔任該角色。若要執行此操作，efgh 角色必須有以下信任政策：

帳戶 222222222222 ***efgh*** 角色信任政策

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

我該如何開始？

若要了解角色如何使用 Amazon EC2 執行個體，您需要使用 IAM 主控台建立角色，啟動使用該角色的 Amazon EC2 執行個體，然後檢查正在執行的執行個體。您可以檢查執行個體中繼資料以查看角色的暫時憑證如何可用於執行個體。您還可以查看在執行個體上執行的應用程式如何使用該角色。使用以下資源以進一步了解。

•

- SDK 演練。軟體 AWS 開發套件文件包含逐步解說，顯示在 Amazon EC2 執行個體上執行的應用程式，該執行個體使用角色的臨時登入資料來讀取 Amazon S3 儲存貯體。以下每個演練都使用不同的程式設計語言提供類似的步驟：
  - 《適用於 Java 的 AWS SDK 開發人員指南》中的使用 SDK for Java 設定 Amazon EC2 的 IAM 角色
  - 《適用於 .NET 的 AWS SDK 開發人員指南》中的使用適用於 .NET 的開發套件啟動 Amazon EC2 執行個體
  - 《適用於 Ruby 的 AWS SDK 開發人員指南》中的使用 SDK for Ruby 建立 Amazon EC2 執行個體

相關資訊

如需建立角色或為 Amazon EC2 執行個體建立角色的詳細資訊，請參閱以下資訊：

- 如需有關搭配 Amazon EC2 執行個體使用 IAM 角色 的詳細資訊，請前往《Amazon EC2 使用者指南》。
- 若要建立角色，請參閱 IAM 角色建立
- 如需有關使用暫時安全憑證的詳細資訊，請參閱 IAM 中的暫時安全憑證。
- 如果使用 IAM API 或 CLI，則必須建立和管理 IAM 執行個體描述檔。如需有關執行個體設定檔的詳細資訊，請參閱 使用執行個體設定檔。
- 如需有關執行個體中繼資料中角色之臨時安全憑證的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Retrieving Security Credentials from Instance Metadata。

使用執行個體設定檔

使用執行個體描述檔將 IAM 角色傳遞給 EC2 執行個體。如需詳細資訊，請參閱 Amazon EC2 User Guide 中的 IAM roles for Amazon EC2。

管理執行個體設定檔 (主控台)

如果您使用 AWS 管理主控台 為 Amazon EC2 建立角色，主控台會自動建立執行個體描述檔，並為其提供與該角色相同的名稱。然後，當您使用 Amazon EC2 主控台啟動具有 IAM 角色的執行個體時，可以選擇要與執行個體關聯的角色。在主控台中，顯示的清單實際上是執行個體設定檔名稱的清單。主控台不會為與 Amazon EC2 無關的角色建立執行個體描述檔。

如果角色和執行個體描述檔具有相同的名稱，您可以使用 AWS 管理主控台 刪除 Amazon EC2 的 IAM 角色和執行個體描述檔。若要進一步了解刪除執行個體描述檔，請參閱 刪除角色或執行個體設定檔。

> **Note**
>
> 若要更新執行個體的許可，請取代其執行個體設定檔。不建議從執行個體設定檔中移除角色，因為最多需要延遲一小時，此變更才生效。

管理執行個體描述檔 (AWS CLI 或 AWS API)

如果您從 AWS CLI 或 AWS API 管理您的角色，您可以將角色和執行個體描述檔建立為個別動作。由於角色和執行個體描述檔可以有不同的名稱，因此您必須知道執行個體描述檔的名稱以及它們包含的角色的名稱。這樣，您可以在啟動 EC2 執行個體時選擇正確的執行個體描述檔。

您可以將標籤連接至 IAM 資源 (包括執行個體描述檔)，以識別、整理和控制其存取。您只能在使用 AWS CLI 或 AWS API 時標記執行個體描述檔。

> **Note**
>
> 執行個體描述檔只能包含一個 IAM 角色，但角色可以包含在多個執行個體描述檔中。每個執行個體設定檔的一個角色的限制無法增加。您可以移除現有角色，然後將不同角色新增到執行個體描述檔。然後 AWS ，由於最終一致性，您必須等待變更出現在所有 中。若要強制變更，必須取消關聯執行個體描述檔，然後關聯執行個體描述檔，或者可以停止執行個體，然後重新啟動它。

管理執行個體設定檔 (AWS CLI)

您可以使用下列 AWS CLI 命令來使用 帳戶中的 AWS 執行個體描述檔。

- 建立執行個體描述檔: `aws iam create-instance-profile`
- 標記執行個體描述檔：`aws iam tag-instance-profile`
- 列出執行個體描述檔的標籤：`aws iam list-instance-profile-tags`
- 取消標記執行個體描述檔：`aws iam untag-instance-profile`
- 將角色新增至執行個體描述檔: `aws iam add-role-to-instance-profile`
- 列出執行個體描述檔: `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- 取得有關執行個體描述檔的資訊: `aws iam get-instance-profile`

- 從執行個體描述檔中移除角色: aws iam remove-role-from-instance-profile
- 刪除執行個體描述檔: aws iam delete-instance-profile

您還可以使用以下命令將角色連接到已在執行的 EC2 執行個體。如需詳細資訊，請參閱 Amazon EC2 的 IAM 角色。

- 將具有角色的執行個體描述檔連接至已停止或正在執行的 EC2 執行個體: aws ec2 associate-iam-instance-profile
- 取得有關連接至 EC2 執行個體的執行個體描述檔的資訊: aws ec2 describe-iam-instance-profile-associations
- 使用已停止或正在執行的 EC2 執行個體中的角色分開執行個體描述檔：aws ec2 disassociate-iam-instance-profile

管理執行個體描述檔 (AWS API)

您可以呼叫下列 AWS API 操作，以在 中使用執行個體描述檔 AWS 帳戶。

- 建立執行個體描述檔: CreateInstanceProfile
- 標記執行個體描述檔：TagInstanceProfile
- 列出執行個體描述檔的標籤：ListInstanceProfileTags
- 取消標記執行個體描述檔：UntagInstanceProfile
- 將角色新增至執行個體描述檔: AddRoleToInstanceProfile
- 列出執行個體描述檔: ListInstanceProfiles, ListInstanceProfilesForRole
- 取得有關執行個體描述檔的資訊: GetInstanceProfile
- 從執行個體描述檔中移除角色: RemoveRoleFromInstanceProfile
- 刪除執行個體描述檔: DeleteInstanceProfile

您還可以呼叫以下操作將角色連接到已在執行的 EC2 執行個體。如需詳細資訊，請參閱 Amazon EC2 的 IAM 角色。

- 將具有角色的執行個體描述檔連接至已停止或正在執行的 EC2 執行個體: AssociateIamInstanceProfile
- 取得有關連接至 EC2 執行個體的執行個體描述檔的資訊: DescribeIamInstanceProfileAssociations

- 使用已停止或正在執行的 EC2 執行個體中的角色分開執行個體描述檔：`DisassociateIamInstanceProfile`

身分提供者和 的聯合身分 AWS

最佳實務是，建議您要求人類使用者使用聯合身分提供者來存取 AWS 資源，而不是在您的 中建立個別的 IAM 使用者 AWS 帳戶。使用身分提供者 (IdP)，您可以在 外部管理您的使用者身分， AWS 並提供這些外部使用者身分許可，以使用您帳戶中 AWS 的資源。如果您的組織已有自己的身分系統，例如公司使用者目錄，這個方式便很管用。如果您要建立需要存取 AWS 資源的行動應用程式或 Web 應用程式，這也很有用。

> **Note**
>
> 您也可以使用外部 SAML 身分提供者 (而不是在 IAM 中使用 SAML 聯合身分)，在 IAM Identity Center 中管理人類使用者。IAM Identity Center 與身分提供者聯合可讓您讓人員存取組織中的多個 AWS 帳戶和多個 AWS 應用程式。若要了解需要 IAM 使用者的特定情形，請參閱建立 IAM 使用者 (而非角色) 的時機。
> 您也可以使用帳戶存取管理員 — IAM 功能，可讓您將組織帳戶中現有的 IAM 角色指派給 IAM Identity Center 使用者和群組。您可以將它與許可集一起使用或單獨使用。

如果您偏好使用單一 AWS 帳戶而不啟用 IAM Identity Center，您可以使用 IAM 搭配外部 IdP，該 IdP 會使用 AWS OpenID Connect (OIDC) 或 SAML 2.0（安全性聲明標記語言 2.0) 提供身分資訊給。OIDC 會將未在 AWS AWS 資源上執行的應用程式，例如 GitHub Actions。知名 SAML 身分提供者的範例包含 Shibboleth 和 Active Directory Federation Services。

當您使用 身分提供者時，您不需要建立自訂登入代碼或管理自己的使用者身分，IdP 會為您處理這些工作。IdP 會為您提供這些功能。您的外部使用者透過 IdP 登入，您可以授予這些外部身分許可，以使用您帳戶中 AWS 的資源。身分提供者有助於保護您的 AWS 帳戶 安全，因為您不必在應用程式中分發或嵌入長期安全登入資料，例如存取金鑰。

檢閱下表，以協助判斷哪種 IAM 聯合身分類型最適合您的使用案例：IAM、IAM Identity Center 或 Amazon Cognito。下列摘要和資料表提供 方法的概觀，您的使用者可用來取得 AWS 資源的聯合存取。

IAM 聯合身分類型	Account type (帳戶類型)	存取管理…	支援的身分來源
使用 IAM Identity Center 的聯合	由 管理的多個帳戶 AWS Organizations	員工的人類使用者	• SAML 2.0 • 受管的 Active Directory • Identity Center 目錄
使用 IAM 的聯合	單一獨立帳戶	• 短期、小規模部署中的人類使用者 • 電腦使用者	• SAML 2.0 • OIDC
使用 Amazon Cognito 身分池的聯合	任何	需要 IAM 授權才能存取資源的應用程式使用者	• SAML 2.0 • OIDC • 選取 OAuth 2.0 社交身分提供者

使用 IAM Identity Center 的聯合

如需人類使用者的集中式存取管理，我們建議您使用 IAM Identity Center 管理您帳戶的存取權和這些帳戶內的許可。IAM Identity Center 中的使用者會獲得 AWS 資源的短期憑證。您可以使用 Active Directory、外部身分提供者 (IdP) 或 IAM Identity Center 目錄做為使用者和群組的身分來源，以指派 AWS 資源的存取權。

IAM Identity Center 支援與 SAML （安全性聲明標記語言） 2.0 的聯合身分，為有權在存取入口網站內使用應用程式的使用者提供聯合單一登入 AWS 存取。然後，使用者可以單一登入支援 SAML 的服務，包括 AWS 管理主控台 和第三方應用程式，例如 Microsoft 365、SAP Concur 和 Salesforce。

使用 IAM 的聯合

雖然我們強烈建議在 IAM Identity Center 中管理人類使用者，但您可以在短期、小規模部署中為人類使用者啟用使用 IAM 的聯合身分主體存取權。IAM 可讓您使用個別的 SAML 2.0 和 Open ID Connect (OIDC) IdP，並使用聯合身分主體屬性進行存取控制。透過 IAM，您可以將使用者屬性，例如成本中心、標題或地區設定，從您的 IdPs 傳遞至 AWS，並根據這些屬性實作精細的存取許可。

工作負載是可提供商業價值的資源和程式碼的集合，例如應用程式或後端程序。您的工作負載可能需要 IAM 身分才能向 AWS 服務、應用程式、操作工具和元件提出請求。這些身分包括在 AWS 環境中執行的機器，例如 Amazon EC2 執行個體或 AWS Lambda 函數。

您可以管理需要存取權的外部當事人的機器身分。若要將存取權授予機器身分，您可以使用 IAM 角色。IAM 角色具有特定許可，並透過使用角色工作階段 AWS 依賴臨時安全登入資料來提供存取的方法。此外，您可能在 外部擁有需要存取您 AWS 環境 AWS 的機器。對於在 外部執行的機器 AWS ，您可以使用 IAM Roles Anywhere。如需角色的詳細資訊，請參閱IAM 角色。如需如何使用角色來委派跨 存取的詳細資訊 AWS 帳戶，請參閱 IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權。

若要將 IdP 直接連結至 IAM，您可以建立身分提供者實體，以在 AWS 帳戶 和 IdP 之間建立信任關係。IAM 支援 IdP，其與 OpenID Connect (OIDC) 或 SAML 2.0 (安全性聲明標記語言 2.0) 相容。如需搭配 使用其中一個 IdPs 的詳細資訊 AWS，請參閱下列各節：

- OIDC 聯合身分
- SAML 2.0 聯合身分

使用 Amazon Cognito 身分池的聯合

Amazon Cognito 專為想要在行動應用程式和 Web 應用程式中為使用者進行驗證和授權的開發人員而設計。Amazon Cognito 使用者集區會在應用程式中新增登入和註冊功能，而身分池則提供 IAM 憑證，讓使用者可以存取您在 AWS中管理的受保護資源。身分池會透過 `AssumeRoleWithWebIdentity` API 操作取得臨時工作階段的憑證。

Amazon Cognito 可與支援 SAML 和 OpenID Connect 的外部身分提供者以及社交身分提供者 (如 Facebook、Google 和 Amazon) 搭配運作。應用程式可以使用使用者集區或外部 IdP 登入使用者，然後透過 IAM 角色中的自訂臨時工作階段代表使用者擷取資源。

其他資源

- 如需如何使用 AWS 管理主控台 組織的身分驗證系統建立啟用單一登入 (SSO) 的自訂聯合代理的示範，請參閱 啟用 AWS 主控台的自訂身分代理程式存取。
- 如需啟用 AWS 工作負載存取外部服務而不存放登入資料的指示，請參閱 將 AWS 身分聯合到外部服務。

常用案例

> **Note**
>
> 我們建議您要求人類使用者在存取時使用臨時登入資料 AWS。您是否考慮過使用 AWS IAM Identity Center？ 您可以使用 IAM Identity Center 集中管理對多個 的存取， AWS 帳戶 並為使用者提供受 MFA 保護的單一登入存取，可從單一位置存取其所有指派的帳戶。使用 IAM Identity Center，您可以在 IAM Identity Center 中建立和管理使用者身分，或輕鬆連線至您現有的 SAML 2.0 相容身分提供者。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

您可以使用外部身分提供者 (IdP) 來管理 AWS 和外部 IdP 以外的使用者身分。外部 IdP 可以使用 AWS OpenID Connect (OIDC) 或安全聲明標記語言 (SAML) 將身分資訊提供給 。當未在 上執行的應用程式 AWS 需要存取 AWS 資源時，通常會使用 OIDC。

當您想要使用外部 IdP 設定聯合時，您可以建立 IAM 身分提供者，以通知 AWS 外部 IdP 及其組態。這會在您的 AWS 帳戶 和外部 IdP 之間建立信任。下列主題提供使用 IAM 身分提供者的常見案例。

主題

- 用於行動應用程式的 Amazon Cognito
- 行動應用程式的 OIDC 聯合身分

用於行動應用程式的 Amazon Cognito

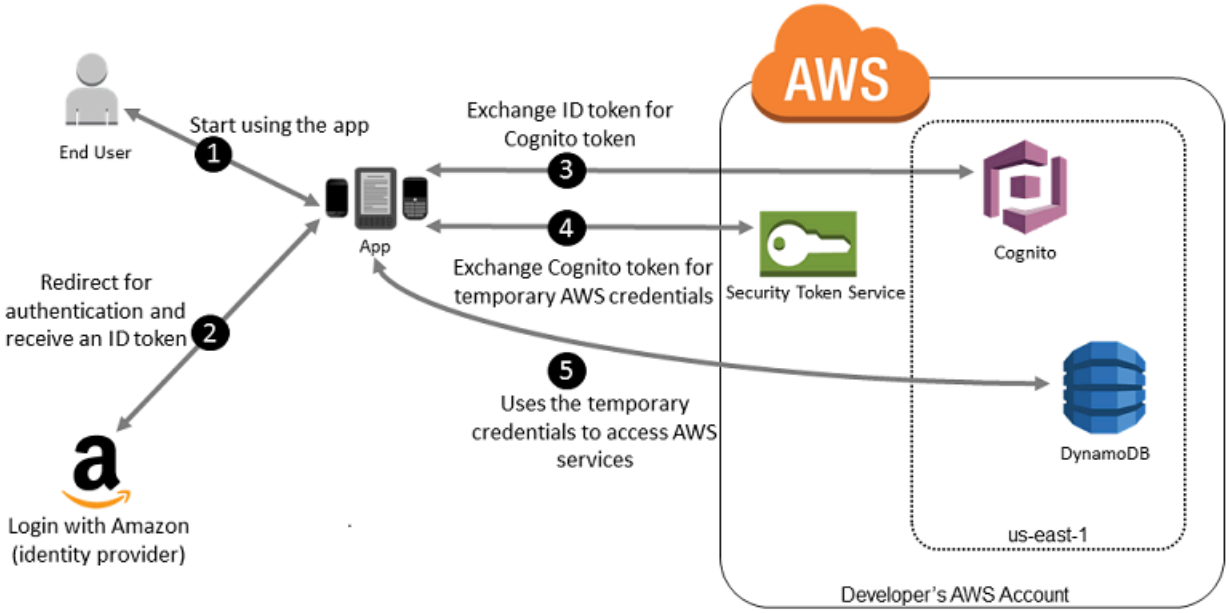
使用 OIDC 聯合身分的偏好方式是使用 Amazon Cognito。例如，開發人員 Adele 正在為行動裝置建置遊戲，其中分數和描述檔這類使用者資料都存放在 Amazon S3 和 Amazon DynamoDB 中。Adele 也會將此資料存放在裝置本機，並使用 Amazon Cognito 跨多個裝置保持同步。她了解為了安全和維護之故，長期的 AWS 安全憑證不應隨遊戲發散。她也了解遊戲可能有大量的使用者。針對所有這些原因，她不想要針對每個玩家在 IAM 建立新的使用者身分。反之，她會設計遊戲，讓使用者可以使用他們透過知名的外部身分提供者 (IdP) 建立的身分登入，例如 Login with Amazon、Facebook、Google 或任何 OpenID Connect (OIDC) 相容的 IdP。她的遊戲可善用其中一個提供者的身分驗證機制，藉以驗證使用者的身分。

為了讓行動應用程式能夠存取她的 AWS 資源，Adele 首先會使用她選擇的 IdPs 註冊開發人員 ID。她也會使用這些提供者的每一個來設定應用程式。在包含遊戲的 Amazon S3 儲存貯體和 DynamoDB 資料表 AWS 帳戶 的她中，Adele 會使用 Amazon Cognito 來建立 IAM 角色，以精確定義遊戲所需的許

可。如果她使用 OIDC IdP，她也會建立 IAM OIDC 身分提供者實體，以在她 AWS 帳戶 和 IdP 中的 Amazon Cognito 身分集區之間建立信任。

在應用程式的程式碼中，Adele 呼叫用於之前設定的 IdP 的登入介面。IdP 會處理讓使用者登入的所有詳細資訊，而應用程式會從提供者取得 OAuth 存取權杖或 OIDC ID 權杖。Adele 的應用程式可以將此身分驗證資訊交換為一組臨時安全登入資料，其中包含 AWS 存取金鑰 ID、私密存取金鑰和工作階段字符。然後，應用程式可以使用這些登入資料來存取 提供的 Web 服務 AWS。該應用程式僅限於許可，其定義於其擔任的角色。

下圖使用 Login with Amazon 做為 IdP，顯示一個有關如何運作的簡化流程。對於步驟 2，應用程式也可使用 Facebook、Google 或任何 OIDC 相容的 IdP，但此處未顯示。



1. 客戶在行動裝置啟動您的應用程式。該應用程式要求使用者登入。
2. 該應用程式會使用 Login with Amazon 資源來接受使用者的憑證。
3. 應用程式使用 Amazon Cognito API 操作 `GetId` 和 `GetCredentialsForIdentity`，用 Login with Amazon 字符交換 Amazon Cognito ID 字符。Amazon Cognito 已設定為信任您的 Login with Amazon 專案，會產生一個權杖，它與 AWS STS交換暫存工作階段憑證。
4. 該應用程序從 Amazon Cognito 接收暫時安全憑證。您的應用程式也可以使用 Amazon Cognito 中的基本（傳統）工作流程，從 AWS STS 使用 擷取字符`AssumeRoleWithWebIdentity`。如需詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的身分集區 (聯合身分) 驗證流程。
5. 透過應用程式可使用暫時性安全憑證存取操作應用程式所需的任何 AWS 資源。與暫時性安全憑證相關聯的角色及指派政策，決定了可以存取哪些內容。

使用下列程序來設定您的應用程式以使用 Amazon Cognito 來驗證使用者，並讓應用程式存取 AWS 資源。如需關於完成此案例的特定步驟，請參閱 Amazon Cognito 的說明文件。

1. (選用) 使用 Login with Amazon、Facebook、Google 或其他任何 OpenID Connect (OIDC) 相容的 IdP 註冊為開發人員，並使用提供者設定一個或多個應用程式。此步驟是可選的，因為 Amazon Cognito 也支援未經驗證的使用者存取權 (訪客)。
2. 前往 中的 Amazon Cognito AWS 管理主控台。使用 Amazon Cognito 精靈來建立身分集區，而該集區是一種容器，可讓 Amazon Cognito 用於為您的應用程式整理最終使用者身分。您可以在應用程式之間分享身分集區。當您設定身分集區時，Amazon Cognito 會建立一或兩個 IAM 角色 (一個用於已驗證的身分，一個用於未經驗證的「訪客」身分)，其為 Amazon Cognito 使用者定義許可。
3. 將 AWS Amplify 與您的應用程式整合，並匯入所需檔案以使用 Amazon Cognito。
4. 建立 Amazon Cognito 憑證提供者的執行個體，並傳遞身分集區 ID、您的 AWS 帳戶 號碼，以及與身分集區建立關聯之角色的 Amazon Resource Name (ARN)。中的 Amazon Cognito 精靈 AWS 管理主控台 提供範例程式碼，協助您開始使用。
5. 當您的應用程式存取 AWS 資源時， 會將登入資料提供者執行個體傳遞至用戶端物件，該物件會將臨時安全登入資料傳遞至用戶端。憑證的許可是根據您稍早定義的一個或多個角色。

如需詳細資訊，請參閱下列內容：

- AWS Amplify 架構文件中的登入 (Android)。
- AWS Amplify 架構文件中的登入 (iOS)。

行動應用程式的 OIDC 聯合身分

為獲得最佳效果，請使用 Amazon Cognito 作為幾乎所有 OIDC 聯合身分方案的身分經紀人。Amazon Cognito 易於使用，並提供額外功能，如匿名 (未經身分驗證) 存取，以及跨裝置和提供者的同步使用者資料。但是，如果您已經透過手動呼叫 `AssumeRoleWithWebIdentity` API 建立了一個使用 OIDC 聯合身分的應用程式，則可以繼續使用它，您的應用程式仍然可以正常工作。

在沒有 Amazon Cognito 的情況下使用 OIDC 聯合身分的程序遵循下列一般概述：

1. 使用 IdP 註冊為開發人員，並使用外部身分提供者 (IdP) 設定您的應用程式，IdP 為您的應用程式提供唯一 ID。(不同的 IdP 可能使用不同的術語來表達此程序。此大綱則使用術語設定來表示向 IdP 識別您應用程式的程序)。每個 IdP 均提供一個對於該 IdP 來說是獨有的應用程式 ID，因此如果根據多個 IdP 的要求設定同一應用程式，則該應用程式將會有多個應用程式 ID。可依照每個提供商的要求配置多個應用程式。

下列外部連結提供有關使用其中一些常用身分提供者 (IdP) 的資訊：

- Login with Amazon 開發人員中心
- Facebook 開發人員網站上的新增 Facebook 登入到您的應用程式或網站。
- Google 開發人員網站上的登入時使用 OAuth 2.0 (OpenID Connect)。

> **Important**
>
> 如果您使用 Google、Facebook 或 Amazon Cognito 的 OIDC 身分提供者，請勿在 中建立個別的 IAM 身分提供者 AWS 管理主控台。 AWS 內建這些 OIDC 身分提供者，可供您使用。略過下列步驟，並直接移至使用您的身分提供者建立新角色。

2. 如果您使用與 OIDC 相容的 Google、Facebook 或 Amazon Cognito 以外的 IdP，請為其建立 IAM 身分提供者實體。
3. 在 IAM 中，建立一或多個角色。對於每個角色，定義可以擔任角色的人員 (信任政策) 以及應用程式使用者擁有的許可 (許可政策)。一般而言，您可以為應用程式所支援的每個 IdP 建立一個角色。例如，您可以建立使用者透過 Login with Amazon 登入之應用程式可擔任的角色，使用者透過 Facebook 登入之相同應用程式可擔任的次要角色，以及使用者透過 Google 登入之應用程式可擔任的第三個角色。對於信任關係，請將 IdP (像是 Amazon.com) 指定為`Principal` (信任實體) 和包含符合 IdP 指派的應用程式 ID 的 `Condition` 。為第三方身分提供者建立角色 中會說明不同提供者的角色範例。
4. 在您的應用程式中，使用 IdP 對您的使用者進行身分驗證。有關如何執行此動作的具體情況會根據您所使用的 IdP (Login with Amazon、Facebook 或 Google) 以及您的應用程式執行的平台而有所不同。例如，Android 應用程式的身分驗證方法可能與 iOS 應用程式或以 JavaScript 為基礎的 Web 應用程式不同。

   一般而言，如果使用者尚未登入，則 IdP 會負責顯示登入頁面。在 IdP 驗證使用者之後，IdP 會向您的應用程式傳回帶有使用者資訊的身分驗證權杖。所包含的資訊取決於 IdP 公開的內容和使用者願意共用的資訊。您可以在您的應用程式中使用此資訊。
5. 在您的應用程式中，對 `AssumeRoleWithWebIdentity` 動作進行未簽署呼叫以請求臨時安全性憑證。在請求中，您會傳遞 IdP 的身分驗證字符，並指定您為該 IdP 建立之 IAM 角色的 Amazon Resource Name (ARN)。 會 AWS 驗證字符是否受信任且有效，如果是， 會將臨時安全登入資料傳回至具有您在請求中指定之角色許可的應用程式。回應還包含來自 IdP 的有關使用者的中繼資料，例如 IdP 與使用者關聯的唯一使用者 ID。
6. 使用`AssumeRoleWithWebIdentity`回應中的臨時安全登入資料，您的應用程式會向 AWS API 操作提出簽章請求。來自 IdP 的使用者 ID 資訊可以區分應用程式中的使用者。例如，您可以將物件放

入包含使用者 ID 作為字首或尾碼的 Amazon S3 資料夾。這可讓您建立鎖定該資料夾的存取控制政策，以便只有具有該 ID 的使用者才能存取它。如需詳細資訊，請參閱AWS STS 聯合身分使用者主體。

7. 您的應用程式必須快取臨時安全憑證，如此每次應用程式需要向 AWS發出請求時都不必取得新的憑證。在預設情況下，憑證可以使用一小時。當憑證過期 (或在此之前)，您再次呼叫 `AssumeRoleWithWebIdentity` 以取得一組新的臨時安全憑證。根據 IdP 以及如何管理權杖，您可能需要在對 `AssumeRoleWithWebIdentity` 進行新呼叫之前重新整理 IdP 的權杖，因為 IdP 的權杖通常也會在固定時間後過期。如果您使用適用於 iOS 的 AWS SDK 或適用於 Android 的 AWS SDK，您可以使用 AmazonSTSCredentialsProvider 動作來管理 IAM 臨時憑證，包括視需要重新整理憑證。

OIDC 聯合身分

假設您正在建立存取 AWS 資源的應用程式，例如使用工作流程存取 Amazon S3 和 DynamoDB 的 GitHub 動作。

當您使用這些工作流程時，您可以向必須使用 AWS 存取金鑰簽署的 AWS 服務提出請求。不過，我們強烈建議您不要將 AWS 登入資料長期存放在 以外的應用程式中 AWS。反之，請設定您的應用程式，在需要時使用 OIDC 聯合動態請求臨時 AWS 安全登入資料。提供的臨時登入資料會對應至只有執行應用程式所需任務所需許可 AWS 的角色。

有了 OIDC 聯合身分，您就不需要建立自訂登入代碼，或管理您自己的使用者身分。相反，您可以在 GitHub Actions 或任何其他 OpenID Connect (OIDC) 相容 IdP 等應用程式中使用 OIDC，以透過 AWS 進行驗證。他們會收到稱為 JSON Web 權杖 (JWT) 的身分驗證權杖，然後將該權杖交換為 中的臨時安全登入資料 AWS ，該登入資料對應至具有在您的 中使用特定資源許可的 IAM 角色 AWS 帳戶。使用 IdP 可協助您保持 AWS 帳戶 安全，因為您不必將長期安全登入資料與應用程式一起嵌入和分發。

OIDC 聯合身分支援機器對機器身分驗證 (例如 CI/CD 管道、自動化指令碼和無伺服器應用程式) 和人類使用者身分驗證。對於需要管理使用者註冊、登入和使用者設定檔的人類使用者身分驗證案例，請考慮使用 Amazon Cognito 作為身分經紀人。如需有關將 Amazon Cognito 與 OIDC 搭配使用的詳細資訊，請參閱用於行動應用程式的 Amazon Cognito。

> **Note**
>
> OpenID Connect (OIDC) 身分提供者發行的 JSON Web 權杖 (JWT) 在 `exp` 宣告中包含過期時間，指定權杖何時過期。IAM 提供超過 JWT 中所指定過期時間的五分鐘時段，以考量 OpenID

Connect (OIDC) Core 1.0 標準允許的時鐘誤差。這意味著 IAM 在過期時間後，但在此五分鐘時段內收到的 OIDC JWT 被接受，以進行進一步評估和處理。

主題

- OIDC 聯合身分的其他資源
- 在 IAM 中建立 OpenID Connect (OIDC) 身分提供者
- 取得 OpenID Connect 身分提供者的指紋
- 共用 OIDC 提供者的身分提供者控制項

OIDC 聯合身分的其他資源

下列資源可協助您進一步了解 OIDC 聯合身分：

- 透過在 Amazon Web Services 中設定 OpenID Connect，在 GitHub 工作流程內使用 OpenID Connect
- 適用於 Android 的 Amplify 程式庫指南中的 Amazon Cognito 身分和適用於 Swift 的 Amplify 程式庫指南中的 Amazon Cognito 身分。
- 如何在AWS 安全部落格上授予 AWS 資源的存取權時使用外部 ID，提供安全設定跨帳戶存取和外部聯合身分的指導。

在 IAM 中建立 OpenID Connect (OIDC) 身分提供者

IAM OIDC 身分提供者是 IAM 中的實體，負責描述可支援 OpenID Connect (OIDC) 標準的外部身分提供者 (IdP) 服務，例如，Google 或 Salesforce。如果想要在 OIDC 相容的 IdP 和您的 AWS 帳戶之間建立信任，請使用 IAM OIDC 身分提供者。這在建立需要存取 AWS 資源的行動應用程式或 Web 應用程式時非常有用，但您不想建立自訂登入碼或管理自己的使用者身分。如需有關此案例的詳細資訊，請參閱 the section called “OIDC 聯合身分”。

您可以使用 AWS 管理主控台、 AWS Command Line Interface、Tools for Windows PowerShell 或 IAM API 來建立和管理 IAM OIDC 身分提供者。

在您建立 IAM OIDC 身分提供者之後，您必須建立一個或多個 IAM 角色。角色是 中的身分 AWS ，沒有自己的登入資料 （與使用者相同）。但在此情況中，角色會動態指派給 OIDC 聯合身分主體，而該主體由組織的 IdP 進行驗證。角色可允許您組織的 IdP 請求暫時性安全憑證以存取 AWS。指派給角色

的政策會決定允許使用者在其中執行的操作 AWS。若要為第三方身分提供者建立角色，請參閱 為第三方身分提供者建立角色。

> **Important**
>
> 當您針對支援 `oidc-provider` 資源的動作設定以身分識別型原則時，IAM 會評估完整的 OIDC 身分識別提供者 URL，包含任何指定的路徑。如果您的 OIDC 身分識別提供者 URL 具有路徑，請務必在 `oidc-provider` ARN 中包含該路徑作為 `Resource` 要素值。您也可以選擇在 URL 路徑中的任何位置附加正斜線和萬用字元 (`/*`) 至 URL 網域，或使用萬用字元 (`*` 和 `?`)。如果要求中的 OIDC 身分識別提供者 URL 與政策的 `Resource` 要素中設定的值不相符，則請求會失敗。

若要疑難排解 IAM OIDC 聯合的常見問題，請參閱解決 re：Post 上的 OIDC 相關錯誤。 AWS

主題

- 先決條件：驗證身分提供者的組態
- 建立及管理 OIDC 提供者 (主控台)
- 建立及管理 IAM OIDC 身分提供者 (AWS CLI)
- 建立和管理 OIDC Identity Provider (AWS API)

先決條件：驗證身分提供者的組態

您必須從 IdP 取得下列資訊，然後才能建立 IAM OIDC 身分提供者。如需有關取得 OIDC 提供者組態資訊的詳細資訊，請參閱 IdP 的文件。

1. 判斷 OIDC 身分提供者的公開可用的 URL。URL 必須以 https:// 開頭。根據 OIDC 標準，允許路徑元件，但不允許查詢參數。通常，URL 僅包含主機名稱，例如 https://server.example.org 或 https://example.com。URL 不應包含連接埠號碼。
2. 將 /.well-known/openid-configuration 新增至 OIDC 身分提供者 URL 的結尾，以查看提供者的公開可用的組態文件和中繼資料。您必須具有 JSON 格式的探索文件，其中包含提供者的組態文件和中繼資料，這些文件可從 OpenID Connect 提供者探索端點 URL 擷取。
3. 確認下列值包含在提供者的組態資訊中。如果 openid-configuration 遺失其中任何欄位，您必須更新探索文件。此程序可能會因您的身分提供者而有所不同，因此請遵循 IdP 的文件來完成此任務。

    - 發行者：網域的 URL。

- jwks_uri：IAM 取得公有金鑰的 JSON Web 金鑰集 (JWKS) 端點。您的身分提供者必須在 openid-configuration 中包含 JSON Web 金鑰集 (JWKS) 端點。此 URI 會定義從何處取得您的公有金鑰，這些金鑰用於驗證身分提供者簽署的權杖。

  > **Note**
  >
  > JSON Web 金鑰集 (JWKS) 必須包含至少一個金鑰，且最多可以有 100 個 RSA 金鑰和 100 個 EC 金鑰。如果 OIDC 身分提供者的 JWKS 包含超過 100 個 RSA 金鑰或 100 個 EC 金鑰，則在將 AssumeRoleWithWebIdentity API 操作與由超過 100 個金鑰限制的金鑰類型簽署的 JWT 搭配使用時，將傳回 `InvalidIdentityToken` 例外狀況。例如，如果使用 RSA 演算法簽署 JWT，且提供者的 JWKS 中有超過 100 個 RSA 金鑰，則將傳回 `InvalidIdentityToken` 例外狀況。

- claims_supported：使用者的相關資訊，可協助您確保來自 IdP 的 OIDC 身分驗證回應包含 IAM 政策中 AWS 用於檢查 OIDC 聯合主體許可的必要屬性。如需可用於宣告的 IAM 條件索引鍵清單，請參閱AWS OIDC 聯合的可用金鑰。
  - aud：您必須確定 IdP 在 JSON Web 權杖 (JWT) 中發出的對象宣告值。對象 (aud) 宣告是應用程式特定的，並識別權杖的預期收件人。當您向 OpenID Connect 提供者註冊行動或 Web 應用程式時，其會建立一個用於識別該應用程式的用戶端 ID。用戶端 ID 是應用程式的唯一識別碼，會在 aud 宣告中傳遞以進行身分驗證。建立 IAM OIDC 身分提供者時，aud 宣告必須與對象值相符。
  - iat：宣告必須包含 `iat` 的值，表示發出 ID 權杖的時間。
  - iss：身分提供者的 URL。URL 必須以 https:// 開頭，且應對應至提供給 IAM 的提供者 URL。根據 OIDC 標準，允許路徑元件，但不允許查詢參數。通常，URL 僅包含主機名稱，例如 https://server.example.org 或 https://example.com。URL 不應包含連接埠號碼。
- response_types_supported：id_token
- subject_types_supported：公有
- id_token_signing_alg_values_supported：RS256、RS384、RS512、ES256、ES384、ES512

> **Note**
>
> 您可以在以下範例中包含其他宣告，如 `my_custom_claim` ；不過， AWS STS 會忽略宣告。

```
{
  "issuer": "https://example-domain.com",
```

```
  "jwks_uri": "https://example-domain.com/jwks/keys",
  "claims_supported": [
    "aud",
    "iat",
    "iss",
    "name",
    "sub",
    "my_custom_claim"
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256",
    "RS384",
    "RS512",
    "ES256",
    "ES384",
    "ES512"
  ],
  "subject_types_supported": [
    "public"
  ]
}
```

建立及管理 OIDC 提供者 (主控台)

按照這些指示，在 AWS 管理主控台中建立和管理 IAM OIDC 身分提供者。

> **Important**
>
> 如果您正在使用來自 Google、Facebook 或 Amazon Cognito 的 OIDC 身分提供者，請勿使用此程序建立個別的 IAM 身分提供者。這些 OIDC 身分提供者已內建於 AWS ，可供您使用。請依照下列步驟為您的身分提供者建立新角色，請參閱 為 OpenID Connect 聯合身分建立角色 (主控台)。

建立 IAM OIDC 身分提供者 (主控台)

1. 在建立 IAM OIDC 身分提供者之前，您必須使用 IdP 來註冊您的應用程式，才能接收用戶端 ID。用戶端 ID (也稱為觀眾) 是您在向 IdP 註冊應用程式時向您發佈的應用程式的唯一識別符。如需有關取得用戶端 ID 的詳細資訊，請參閱文件以取得您的 IdP。

   > **Note**
   >
   > AWS 使用我們的信任根憑證授權機構 (CAsIdPs) 的通訊，以驗證 JSON Web Key Set (JWKS) 端點的 TLS 憑證。如果您的 OIDC IdP 依賴的憑證不是由這些信任CAs 之一簽署，則我們只會使用 IdP 組態中設定的指紋進行安全通訊。如果我們無法擷取 TLS 憑證或如果需要 TLS v1.3， AWS 將返回指紋驗證。

2. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
3. 在導覽窗格中，請選擇 Identity providers (身分提供者)，然後選擇 Add provider (新增提供者)。
4. 對於 Configure provider (設定提供者)，選擇 OpenID Connect (OpenID 連線)。
5. 針對 Provider URL (提供者 URL)，輸入 IdP 的 URL。URL 必須符合這些限制：
   - URL 區分大小寫。
   - URL 必須以 **`https://`** 開頭。
   - URL 不應包含連接埠號碼。
   - 在您的 中 AWS 帳戶，每個 IAM OIDC 身分提供者都必須使用唯一的 URL。如果您嘗試提交已在 中用於 OpenID Connect 提供者的 URL AWS 帳戶，您會收到錯誤。
6. 針對對象，輸入您向 IdP 註冊並在 中接收之應用程式的用戶端 IDStep 1，以及提出請求的用戶端 ID AWS。如果此 IdP 有其他用戶端 ID (也稱觀眾)，則稍後可以在提供者詳細資訊頁面上新增它們。

   > **Note**
   >
   > 如果您的 IdP JWT 權杖包含 `azp` 宣告，請輸入此值作為對象值。
   > 如果您的 OIDC 身分提供者在權杖中同時設定 `aud`和 `azp` 宣告， AWS STS 將使用`azp`宣告中的 值作為`aud`宣告。

7. (選用) 對於 Add tags (新增標籤)，您可以新增鍵值組，以協助您識別和整理 IdP。您也可以使用標籤來控制對 AWS 資源的存取。若要進一步了解如何標記 IAM OIDC 身分提供者，請參閱 標記 OpenID Connect (OIDC) 身分提供者。選擇 Add tag (新增標籤)。輸入每個標籤鍵值組的值。

8. 請確認您提供的資訊。完成後，請選擇 Add provider (新增提供者)。IAM 會嘗試擷取並使用 OIDC IdP 伺服器憑證的頂層中繼 CA 指紋，來建立 IAM OIDC 身分提供者。

   > **Note**
   >
   > OIDC 身分提供者的憑證鏈必須以網域或發行者 URL 開頭，然後是中繼憑證，最後以根憑證結尾。如果憑證鏈順序不同或包含重複或其他憑證，則您會收到簽章不相符錯誤，且 STS 無法驗證 JSON Web 權杖 (JWT)。更正從伺服器傳回的鏈結中的憑證順序，以解決錯誤。如需有關憑證鏈標準的詳細資訊，請參閱 RFC 系列網站上的 RFC 5246 中的 certificate_list。

9. 將 IAM 角色指派給您的身分提供者，以提供身分提供者管理的外部使用者身分許可，以存取您帳戶中 AWS 的資源。若要深入了解如何建立聯合身分角色，請參閱 為第三方身分提供者建立角色。

   > **Note**
   >
   > 角色信任政策中使用的 OIDC IdP 必須與信任它的角色位於相同帳戶中。

新增或移除 IAM OIDC 身分提供者的指紋 (主控台)

> **Note**
>
> AWS 使用我們的信任根憑證授權機構 (CAsIdPs) 的通訊，以驗證 JSON Web Key Set (JWKS) 端點的 TLS 憑證。如果您的 OIDC IdP 依賴的憑證不是由這些信任CAs 之一簽署，則我們只會使用 IdP 組態中設定的指紋進行安全通訊。如果我們無法擷取 TLS 憑證或如果需要 TLS v1.3， AWS 將返回指紋驗證。

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，請選擇 Identity providers (身分提供者)。然後選擇您要更新的 IAM 身分提供者名稱。
3. 選擇端點驗證索引標籤，然後在指紋區段中，選擇管理。若要輸入新的指紋值，請選擇 Add thumbprint 新增指紋。若要移除指紋，請選擇要移除的指紋旁的 Remove (移除)。

> **Note**
>
> IAM OIDC 身分提供者必須至少有 1 個和最多可有 5 個指紋。

完成後，請選擇 Save changes (儲存變更)。

新增 IAM OIDC 身分提供者的對象 (主控台)

1. 在導覽窗格中，選擇 Identity providers (身分提供者)，然後選擇您所要更新 IAM 身分提供者的名稱。
2. 在 Audiences (對象) 區段中，選擇 Actions (動作) 並選取 Add audience (新增對象)。
3. 輸入您向 IdP 註冊並在 中接收的應用程式的用戶端 IDStep 1，該應用程式將向其發出請求 AWS。然後選擇 Add audiences (新增對象)。

> **Note**
>
> IAM OIDC 身分提供者必須至少有 1 個和最多可有 100 個對象。

移除 IAM OIDC 身分提供者的對象 (主控台)

1. 在導覽窗格中，選擇 Identity providers (身分提供者)，然後選擇您所要更新 IAM 身分提供者的名稱。
2. 在 Audiences (對象) 區段中，選取您要移除之對象旁邊的選項按鈕，然後選取 Actions (動作)。
3. 選擇 Remove audience (移除對象)。新的視窗將開啟。
4. 如果您移除對象，與對象聯合的身分無法擔任與對象相關聯的角色。在視窗中，閱讀警告，並在欄位中輸入單字 `remove`，確認您要移除對象。
5. 選擇 Remove (移除) 以移除對象。

刪除 IAM OIDC 身分提供者 (主控台)

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，請選擇 Identity providers (身分提供者)。
3. 在您要刪除的 IAM 身分提供者旁邊，勾選核取方塊。新的視窗將開啟。

4. 在欄位中輸入單字 delete，確認您要刪除提供者。再選擇 Delete (刪除)。

建立及管理 IAM OIDC 身分提供者 (AWS CLI)

您可以使用下列 AWS CLI 命令來建立和管理 IAM OIDC 身分提供者。

建立 IAM OIDC 身分提供者 (AWS CLI)

1. (選用) 若要取得 AWS 帳戶中所有 IAM OIDC 身分提供者的清單，請執行下列命令：
   - aws iam list-open-id-connect-providers
2. 若要建立新的 IAM OIDC 身分提供者，請執行下列命令：
   - aws iam create-open-id-connect-provider

更新現有 IAM OIDC 身分提供者的伺服器憑證指紋清單 (AWS CLI)

- 若要更新現有 IAM OIDC 身分提供者的伺服器憑證指紋清單，請執行下列命令：
  - aws iam update-open-id-connect-provider-thumbprint

標記現有的 IAM OIDC 身分提供者 (AWS CLI)

- 若要標記現有 IAM OIDC 身分提供者，請執行下列命令：
  - aws iam tag-open-id-connect-provider

列出現有 IAM OIDC 身分提供者 (AWS CLI) 的標籤

- 若要列出現有 IAM OIDC 身分提供者的標籤，請執行下列命令：
  - aws iam list-open-id-connect-provider-tags

移除 IAM OIDC 身分提供者 (AWS CLI) 的標籤

- 若要移除 IAM OIDC 身分提供者的標籤，請執行下列命令：
  - aws iam untag-open-id-connect-provider

從現有 IAM OIDC 身分提供者新增或移除用戶端 ID (AWS CLI API)

1. (選用) 若要取得 AWS 帳戶中所有 IAM OIDC 身分提供者的清單，請執行下列命令：
   - aws iam list-open-id-connect-providers
2. (選用) 若要取得有關 IAM OIDC 身分提供者的詳細資訊，請執行下列命令：
   - aws iam get-open-id-connect-provider
3. 若要將新的用戶端 ID 新增到現有 IAM OIDC 身分提供者，請執行下列命令：
   - aws iam add-client-id-to-open-id-connect-provider
4. 若要從現有 IAM OIDC 身分提供者中移除用戶端，請執行下列命令：
   - aws iam remove-client-id-from-open-id-connect-provider

若要刪除 IAM OIDC 身分提供者 (AWS CLI)

1. (選用) 若要取得 AWS 帳戶中所有 IAM OIDC 身分提供者的清單，請執行下列命令：
   - aws iam list-open-id-connect-providers
2. (選用) 若要取得有關 IAM OIDC 身分提供者的詳細資訊，請執行下列命令：
   - aws iam get-open-id-connect-provider
3. 若要刪除 IAM OIDC 身分提供者，請執行下列命令：
   - aws iam delete-open-id-connect-provider

建立和管理 OIDC Identity Provider (AWS API)

您可以使用以下 IAM API 命令來建立及管理 OIDC 提供者。

建立 IAM OIDC 身分提供者 (AWS API)

1. (選用) 若要取得 AWS 帳戶中所有 IAM OIDC 身分提供者的清單，請呼叫下列操作：
   - ListOpenIDConnectProviders
2. 若要建立新的 IAM OIDC 身分提供者，請呼叫下列操作：
   - CreateOpenIDConnectProvider

更新現有 IAM OIDC 身分提供者 (AWS API) 的伺服器憑證指紋清單

- 若要更新現有 IAM OIDC 身分提供者的伺服器憑證指紋清單，請呼叫下列操作：
  - UpdateOpenIDConnectProviderThumbprint

標記現有的 IAM OIDC 身分提供者 (AWS API)

- 若要標記現有 IAM OIDC 身分提供者，請叫用下列操作：
  - TagOpenIDConnectProvider

列出現有 IAM OIDC 身分提供者 (AWS API) 的標籤

- 若要列出現有 IAM OIDC 身分提供者的標籤，請叫用下列操作：
  - ListOpenIDConnectProviderTags

移除現有 IAM OIDC 身分提供者 (AWS API) 上的標籤

- 若要移除現有 IAM OIDC 身分提供者的標籤，請叫用下列操作：
  - UntagOpenIDConnectProvider

從現有的 IAM OIDC 身分提供者 (AWS API) 新增或移除用戶端 ID

1. (選用) 若要取得 AWS 帳戶中所有 IAM OIDC 身分提供者的清單，請呼叫下列操作：
   - ListOpenIDConnectProviders
2. (選用) 若要取得有關 IAM OIDC 身分提供者的詳細資訊，請呼叫下列操作：
   - GetOpenIDConnectProvider
3. 若要將新的用戶端 ID 新增到現有 IAM OIDC 身分提供者，請呼叫下列操作：
   - AddClientIDToOpenIDConnectProvider
4. 若要從現有 IAM OIDC 身分提供者中移除用戶端 ID，請呼叫下列操作：
   - RemoveClientIDFromOpenIDConnectProvider

刪除 IAM OIDC 身分提供者 (AWS API)

1. (選用) 若要取得 AWS 帳戶中所有 IAM OIDC 身分提供者的清單，請呼叫下列操作：
   - ListOpenIDConnectProviders
2. (選用) 若要取得有關 IAM OIDC 身分提供者的詳細資訊，請呼叫下列操作：
   - GetOpenIDConnectProvider
3. 若要刪除 IAM OIDC 身分提供者，請呼叫下列操作：
   - DeleteOpenIDConnectProvider

取得 OpenID Connect 身分提供者的指紋

在 IAM 中建立 OpenID Connect (OIDC) 身分提供者時，IAM 需要頂層中繼憑證授權機構 (CA) 的指紋，該授權機構會簽署外部身分提供者 (IdP) 使用的憑證。指紋是 CA 憑證的簽章，該簽章會用來發出與 OIDC 相容的 IdP 憑證。當您建立 IAM OIDC 身分提供者時，您會信任該 IdP 驗證的身分可以存取您的 AWS 帳戶。透過使用 CA 的憑證指紋，您信任該 CA 發出的任何憑證，該 CA 會使用與任何登錄之指紋相同 DNS 名稱來發出憑證。這讓您在續約 IdP 的簽署憑證時，無需在每個帳戶中更新信任。

> **Important**
>
> 在大多數情況下，聯合伺服器會使用兩個不同的憑證：
>
> - 第一個 會在 AWS 和您的 IdP 之間建立 HTTPS 連線。此憑證應該由知名的公有根 CA 發行，例如 AWS Certificate Manager。這可讓用戶端檢查憑證的可靠性和狀態。
> - 第二個憑證用於加密令牌，且應由私有或公有根 CA 簽署。

您可以使用 AWS Command Line Interface、Tools for Windows PowerShell 或 IAM API 建立 IAM OIDC 身分提供者。使用這些方法時，您可以選擇手動提供指紋。如果您選擇不包含指紋，IAM 將擷取 OIDC IdP 伺服器憑證的頂層中繼 CA 指紋。如果選擇包含指紋，您必須手動取得指紋並將其提供給 AWS。

當您使用 IAM 主控台建立 OIDC 身分提供者時，IAM 會嘗試為您擷取 OIDC IdP 伺服器憑證的頂層中繼 CA 指紋。

我們建議您同時手動取得您的 OIDC IdP 的使用指紋，並且驗證 IAM 已擷取到正確的指紋。如需有關取得憑證指紋的詳細資訊，請參閱下列各節。

> **Note**
>
> AWS 使用我們的信任根憑證授權機構 (CAsIdPs) 的通訊，以驗證 JSON Web Key Set (JWKS) 端點的 TLS 憑證。如果您的 OIDC IdP 依賴的憑證不是由這些信任CAs 之一簽署，則我們只會使用 IdP 組態中設定的指紋進行安全通訊。如果我們無法擷取 TLS 憑證或如果需要 TLS v1.3， AWS 將返回指紋驗證。

> **Important**
>
> 如果您的 OIDC 身分提供者對探索端點 （ 發行者 URL) 和 JWKS 端點 (`jwks_uri`) 使用不同的主機，您必須在 OIDC 提供者的指紋清單中包含兩個端點的指紋。當返回指紋驗證時， 會 AWS 驗證兩個端點的指紋。如果設定的清單中遺失任一指紋，身分驗證會失敗。
> 大多數 OIDC 提供者在同一網域上託管兩個端點，因此單一指紋涵蓋兩者。只有在您的提供者為兩個端點使用不同的主機或憑證時，才需要新增單獨的指紋。

取得憑證指紋

您可以使用 Web 瀏覽器和 OpenSSL 命令列工具，來取得 OIDC 提供者的憑證指紋。但是，您不需要手動取得憑證指紋即可建立 IAM OIDC 身分提供者。您可以使用下列程序來取得 OIDC 提供者的憑證指紋。

取得 OIDC IdP 的指紋

1. 您需要先取得 OpenSSL 命令列工具，然後才能取得 OIDC IdP 的指紋。您可使用此工具下載 OIDC IdP 憑證鍊並產生憑證連結中最終憑證的指紋。如果您需要安裝和設定 OpenSSL，請根據 安裝 OpenSSL 與 設定 OpenSSL 中的說明操作。
2. 從 OIDC IdP URL 開始 (例如，`https://server.example.com`)，然後新增 `/.well-known/openid-configuration` 以組成該 IdP 的組態文件的 URL，如下所示：

   `https://server.example.com/.well-known/openid-configuration`

   在 Web 瀏覽器中開啟此 URL，並將 *`server.example.com`* 取代為 IdP 的伺服器名稱。
3. 在顯示的文件中，使用您的 web 瀏覽器 Find (尋找) 功能來尋找文字 `"jwks_uri"`。文字 `"jwks_uri"` 後面會有一個冒號 (:)，然後是一個 URL。複製 URL 的完整網域名稱。不包括 `https://` 或在頂層網域後的任何路徑。

```
{
 "issuer": "https://accounts.example.com",
 "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
 "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
 "token_endpoint": "https://oauth2.exampleapis.com/token",
 "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
 "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
 "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
...
```

> **Note**
>
> 如果您的探索和 JWKS 端點使用不同的主機，請注意發行者 URL （您在 中使用的 URLStep 2，不含 `/.well-known/openid-configuration` 路徑） 的完整網域名稱。如果此網域與您在此步驟中複製的`jwks_uri`網域不同，Step 6請對兩個網域執行 Step 4 到 ，並在建立 OIDC 供應商時包含兩個指紋。

4. 使用 OpenSSL 命令列工具來執行以下命令。將 *keys.example.com* 取代為您在Step 3 中取得的網域名稱。

```
openssl s_client -servername keys.example.com -showcerts -
connect keys.example.com:443
```

5. 在命令視窗中向上滾動，直至看到類似於以下範例的憑證。如果您看到多個憑證，請找到顯示的最後一個憑證 (在命令輸出末尾)。這包含憑證授權機構鏈中的頂層中繼 CA 憑證。

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
 nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
```

```
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----
```

複製憑證 (包括 `-----BEGIN CERTIFICATE-----` 與 `-----END CERTIFICATE-----` 行) 並將其貼上文字檔中。接著儲存檔案，並將其命名為 **`certificate.crt`**。

> **Note**
>
> OIDC 身分提供者的憑證鏈必須以網域或發行者 URL 開頭，包含任何中繼憑證 (如果有)，並以根憑證結尾。如果憑證鏈順序不同或包含重複或其他憑證，您會收到簽章不相符錯誤，且 STS 無法驗證 JSON Web 權杖 (JWT)。更正從伺服器傳回的鏈結中的憑證順序，以解決錯誤。如需有關憑證鏈標準的詳細資訊，請參閱 RFC 系列網站上的 RFC 5246 中的 certificate_list。

6. 使用 OpenSSL 命令列工具來執行以下命令。

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

您的命令視窗將顯示類似於以下範例的憑證指紋：

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

從此字串中去掉冒號 (:) 字元以產生最終指紋，如：

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

> **Note**
>
> 如果您的探索端點使用不同於 JWKS 端點的主機，Step 6請使用探索端點網域 （ 您的發行者 URL 網域 ） 重複 Step 4 到 。使用 CreateOpenIDConnectProvider 或 UpdateOpenIDConnectProviderThumbprint 建立或更新 OIDC 身分提供者時，請包含這兩個指紋。

7. 如果您要使用 AWS CLI、Tools for Windows PowerShell 或 IAM API 建立 IAM OIDC 身分提供者，則提供指紋是選用的。如果您選擇在建立期間不包含指紋，IAM 將擷取 OIDC IdP 伺服器憑證的頂層中繼 CA 指紋。在建立 IAM OIDC 身分提供者之後，您可以將此指紋與 IAM 擷取的指紋進行比較。

如果您要在 IAM 主控台中建立 IAM OIDC 身分提供者，主控台會嘗試為您擷取 OIDC IdP 伺服器憑證的頂層中繼 CA 指紋。您可以將此指紋與 IAM 擷取的指紋進行比對。建立 IAM OIDC 身分提供者之後，您可以在 OIDC 提供者摘要主控台頁面上的端點驗證索引標籤中檢視 IAM OIDC 身分提供者的指紋。

> **Important**
>
> 如果您取得的指紋與您在 IAM OIDC 身分提供者指紋詳細資訊中看到的指紋不相符，則不應該使用該 OIDC 提供者。反之，您應刪除建立的 OIDC 提供者，然後在經過一段時間後再次嘗試建立 OIDC 提供者。在使用提供者之前，驗證指紋是否相符。如果這些指紋在第二次嘗試後仍然不相符，則請使用 IAM 論壇與 AWS聯絡。

安裝 OpenSSL

如果您尚未安裝 OpenSSL，請遵循此節中的說明操作。

在 Linux 或 Unix 上安裝 OpenSSL

1. 移至 OpenSSL: Source, Tarballs (https://openssl.org/source/)。
2. 下載最新來源並建置套件。

在 Windows 安裝 OpenSSL

1. 請前往 OpenSSL: Binary Distributions (https://wiki.openssl.org/index.php/Binaries)，以取得您可以從中安裝 Windows 版本的網站清單。
2. 依照您所選網站上的指示開始安裝。
3. 如果系統要求您安裝 Microsoft Visual C++ 2008 可轉散發套件，並且您的系統上尚未安裝此版本，請選擇適合您環境的下載連結。請遵照 Microsoft Visual C++ 2008 可轉散發套件安裝精靈所提供的指示執行。

> **Note**
>
> 如果您不確定 Microsoft Visual C++ 2008 可轉散發套件是否已經安裝在您的系統上，您可以嘗試先安裝 OpenSSL。如果尚未安裝 Microsoft Visual C++ 2008 可轉散發套件，則

OpenSSL 安裝程式將會顯示提醒。請確定您安裝的架構 (32 位元或 64 位元) 符合您所安裝的 OpenSSL 版本。

4. 安裝 Microsoft Visual C++ 2008 可轉散發套件後，請選取適用於您環境的 OpenSSL 二進位檔版本並將檔案儲存在本機。啟動 OpenSSL 安裝精靈。
5. 遵循 OpenSSL 安裝精靈中所述的指示。

設定 OpenSSL

在使用 OpenSSL 命令之前，您必須先設定作業系統，使其具有 OpenSSL 安裝位置的相關資訊。

在 Linux 或 Unix 上設定 OpenSSL

1. 在命令列，將 OpenSSL_HOME 變數設定為 OpenSSL 安裝的位置：

```
$ export OpenSSL_HOME=path_to_your_OpenSSL_installation
```

2. 設定要包含 OpenSSL 安裝的路徑：

```
$ export PATH=$PATH:$OpenSSL_HOME/bin
```

> **Note**
>
> 您使用 `export` 命令對環境變數做出的任何變更，僅適用於目前的工作階段。您可以在 shell 組態檔案中設定環境變數，對環境變數進行持續變更。如需詳細資訊，請參閱適用於您作業系統的文件。

在 Windows 上設定 OpenSSL

1. 開啟命令提示視窗。
2. 將 OpenSSL_HOME 變數設定為 OpenSSL 安裝的位置：

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. 將 OpenSSL_CONF 變數設定為 OpenSSL 安裝中組態檔案的位置：

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. 設定要包含 OpenSSL 安裝的路徑：

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> 您在命令提示視窗中對 Windows 環境變數做出的任何變更，僅適用於目前命令列工作階段。您可以將環境變數設定為系統屬性，對環境變數進行持續變更。確切的程序將取決於您索使用的 Windows 版本。(例如，在 Windows 7 中，開啟控制面板、系統與安全、系統。 然後選擇進階系統設定、進階索引標籤、環境變數。) 如需詳細資訊，請參閱 Windows 文件。

共用 OIDC 提供者的身分提供者控制項

對於已辨識的共用 OpenID Connect (OIDC) 身分提供者 (IdP)，IAM 需要明確評估角色信任政策中的特定宣告。這些必要宣告稱為身分提供者控制項，由 IAM 在角色建立和信任政策更新期間進行評估。如果角色信任政策未評估共用 OIDC IdP 所需的控制項，角色建立或更新將會失敗。這可確保只有來自預期組織的授權身分才能擔任角色和存取 AWS 資源。當 OIDC 提供者在多個 AWS 客戶之間共用時，此安全控制項至關重要。

對於現有的 OIDC 角色信任政策，IAM 不會評估身分提供者控制項。現有 OIDC 角色若要對角色信任政策進行任何修改，IAM 將要求其在角色信任政策中包含身分提供者控制項。

OIDC 提供者類型

IAM 將 OIDC 身分提供者分類為兩種不同的類型：私有和共用。私有 OIDC IdP 可由單一組織擁有和管理，也可以是 SaaS 提供者的租用戶，其 OIDC 發行者 URL 可作為該組織特有的唯一識別碼。相比之下，共用的 OIDC IdP 可在多個組織之間使用，對於使用該共用身分提供者的所有組織，OIDC 發行者 URL 可能相同。

下表概述了私有和共用 OIDC 提供者之間的主要差異：

特性	私有 OIDC 提供者	共用 OIDC 提供者
發行者	組織獨有	在多個組織間共用
租用資訊	透過唯一發行者傳達	透過 JWT 中的宣告傳達

特性	私有 OIDC 提供者	共用 OIDC 提供者
信任政策需求	不需要評估特定宣告	需要評估特定宣告

具有身分提供者控制項的共用 OIDC 身分提供者

在 IAM 中建立或修改 OIDC 提供者時，系統會自動識別並評估已辨識之共用 OIDC 提供者所需的宣告。如果未在角色信任政策中設定身分提供者控制項，則角色建立或更新將會失敗，並發生 MalformedPolicyDocument 錯誤。

下表列出了需要在角色信任政策中使用身分提供者控制項的共用 OIDC 提供者，以及有助於設定身分提供者控制項的其他資訊。

OIDC IdP	OIDC URL	租用宣告	必要宣告
Amazon Cognito	`cognito-identity.amazonaws.com`	aud	`cognito-identity.amazonaws.com:aud`
Azure Sentinel	`https://sts.windows.net/33e01921-4d64-4f8c-a055-5bdaffd5e33d`	`sts:RoleSessionName`	`sts:RoleSessionName`
Buildkite	`https://agent.buildkite.com`	sub	`agent.buildkite.com:sub`
Codefresh SaaS	`https://oidc.codefresh.io`	sub	`oidc.codefresh.io:sub`

OIDC IdP	OIDC URL	租用宣告	必要宣告
DVC Studio	`https://studio.datachain.ai/api`	sub	`studio.datachain.ai/api:sub`

GitHub actions	`https://token.actions.githubusercontent.com`	sub 或 job_workflow_ref	`token.actions.githubusercontent.com:sub`
			或
			`token.actions.githubusercontent.com:job_workflow_ref`

GitHub audit log streaming	`https://oidc-configuration.audit-log.githubusercontent.com`	sub	`oidc-configuration.audit-log.githubusercontent.com:sub`
GitHub vstoken	`https://vstoken.actions.githubusercontent.com`	sub	`vstoken.actions.githubusercontent.com:sub`
GitLab	`https://gitlab.com`	sub	`gitlab.com:sub`

OIDC IdP	OIDC URL	租用宣告	必要宣告
IBM Turbonomic SaaS*	• https://rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08iidd0nt4hekbn7 • https://rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr • https://rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg • https://rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26	sub	• rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08iidd0nt4hekbn7:sub • rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr:sub • rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg:sub • rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26:sub

OIDC IdP	OIDC URL	租用宣告	必要宣告
	• https://oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb • https://oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu		• oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb:sub • oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu:sub
Pulumi Cloud	https://api.pulumi.com/oidc	aud	api.pulumi.com/oidc:aud
sandboxes.cloud	https://sandboxes.cloud	aud	sandboxes.cloud:aud
Scalr	https://scalr.io	sub	scalr.io:sub
Shisho Cloud	https://tokens.cloud.shisho.dev	sub	tokens.cloud.shisho.dev:sub
Terraform Cloud	https://app.terraform.io	sub	app.terraform.io:sub

OIDC IdP	OIDC URL	租用宣告	必要宣告
Upbound	`https://proidc.upbound.io`	sub	`proidc.upbound.io:sub`
Vercel global endpoint	`https://oidc.vercel.com`	aud	`oidc.vercel.com:aud`

* IBM Turbonomic 會隨平台新版本定期更新其 OIDC 發行者 URL。我們會視需要將範圍內的其他 Turbonomic OIDC 發行者新增為共用提供者。

對於 IAM 識別為共用的任何新 OIDC IdP，將以類似方式記錄和強制執行角色信任政策所需的身分提供者控制項。

其他資源

其他資源：

- 如需有關為 OIDC 聯合身分建立 IAM 角色的詳細資訊，請參閱為 OpenID Connect 聯合身分建立角色 (主控台)。
- 如需可用於宣告的 IAM 條件索引鍵清單，請參閱AWS OIDC 聯合的可用金鑰。

SAML 2.0 聯合身分

AWS 支援與 SAML 2.0（安全性聲明標記語言 2.0) 的聯合身分，這是一種許多身分提供者 (IdPs使用的開放標準。此功能可啟用聯合單一登入 (SSO)，讓使用者可以登入 AWS 管理主控台 或呼叫 AWS API 操作，而無需為您組織中的每個人建立 IAM 使用者。透過使用 SAML，您可以簡化使用 設定聯合的程序 AWS，因為您可以使用 IdP 的服務，而不是編寫自訂身分代理程式碼。

> **Note**
>
> IAM SAML 聯合身分支援來自 SAML 型聯合身分提供者 (IdP) 的加密 SAML 回應。IAM Identity Center 和 Amazon Cognito 不支援來自 IAM SAML 身分提供者的加密 SAML 聲明。您可以間接將對加密 SAML 聲明的支援新增至 Amazon Cognito 身分集區與 Amazon Cognito 使用者集區的聯合。使用者集區具有獨立 IAM SAML 聯合身分的 SAML 聯合身分，並支援 SAML 簽署和加密。雖然此功能不會直接延伸至身分集區，但使用者集區可以是身分集區的

IdP。若要將 SAML 加密與身分集區搭配使用，請將具有加密功能的 SAML 提供者新增至作為身分集區的 IdP 的使用者集區。
您的 SAML 提供者必須能夠使用使用者集區提供的金鑰來加密 SAML 聲明。使用者集區不會接受使用 IAM 提供的憑證加密的聲明。

IAM 聯合身分支援這些使用案例：

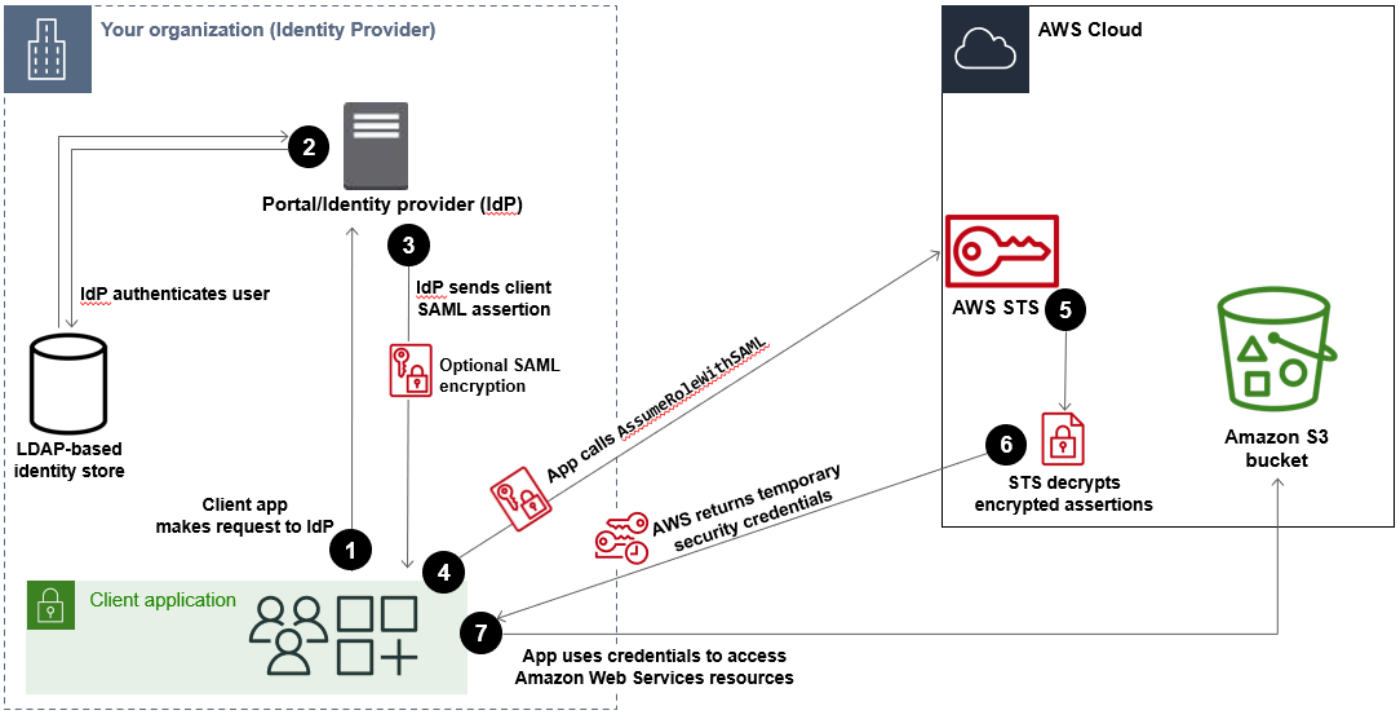
- 聯合存取，以允許組織中的使用者或應用程式呼叫 AWS API 操作。下一章節會討論此使用案例。您可以使用組織內產生的 SAML 聲明 (身分驗證回應的一部分) 獲得臨時安全性憑證。此方案類似於 IAM 支援的其他聯合身分方案，如 請求臨時安全憑證 和 OIDC 聯合身分 中說明之情況。但是，組織中以 SAML 2.0 為基礎的 IdP 可以在執行時間裡處理許多詳細資訊，以用於執行身分驗證和授權檢查。
- AWS 管理主控台 來自您組織的 Web 型單一登入 (SSO) 至 。使用者可以登入由 SAML 2.0 相容 IdP 託管的組織中的入口網站，選取要前往的選項 AWS，並重新導向至主控台，而無需提供其他登入資訊。您可以使用第三方 SAML IdP 建立對主控台的 SSO 存取，或您還可以建立自訂 IdP 來支援外部使用者的主控台存取。如需有關建構自訂 IdP 的詳細資訊，請參閱 啟用 AWS 主控台的自訂身分代理程式存取。

主題

- 使用 SAML 型聯合存取 API AWS
- 配置以 SAML 2.0 為基礎的聯合身分驗證的概觀
- 允許 SAML 聯合存取 AWS 資源的角色概觀
- 單獨辨識以 SAML 為基礎的聯合身分中的使用者
- 在 IAM 中建立 SAML 身分提供者
- 使用依賴方信任設定您的 SAML 2.0 IdP 並新增宣告
- 將第三方 SAML 解決方案供應商與 整合 AWS
- 為身分驗證回應設定 SAML 聲明
- 啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台
- 在瀏覽器中檢視 SAML 回應

使用 SAML 型聯合存取 API AWS

假設您想要為員工提供一種方式，使其能將資料從他們的電腦中複製到備份資料夾。您可以建構一個可在使用者的電腦上執行的應用程式。該應用程式可在後端的 Amazon S3 儲存貯體中讀寫物件。使用者無法直接存取 AWS。而應使用以下程序：



1. 組織中的使用者會使用用戶端應用程式來請求獲得組織 IdP 的身分驗證。
2. IdP 根據組織的身分存放區對使用者進行身分驗證。
3. IdP 會建構一個具有使用者相關資訊的 SAML 聲明，並將此聲明發送到用戶端應用程式。當您為 IAM SAML IdP 啟用 SAML 加密時，此聲明由外部 IdP 加密。
4. 用戶端應用程式會 AWS STS `AssumeRoleWithSAML`呼叫 API，傳遞 SAML 供應商的 ARN、要擔任的角色的 ARN，以及來自 IdP 的 SAML 聲明。如果啟用加密，透過用戶端應用程式傳遞的聲明會在傳輸中保持加密狀態。
5. （選用）AWS STS 使用您從外部 IdP 上傳的私有金鑰來解密加密的 SAML 聲明。
6. API 對用戶端應用程式的回應包括臨時性安全憑證。
7. 用戶端應用程式使用臨時安全性憑證來呼叫 Amazon S3 API 操作。

配置以 SAML 2.0 為基礎的聯合身分驗證的概觀

您必須先將組織的 IdP 和 AWS 帳戶 設定為彼此信任，才能如上述案例和圖表所述使用 SAML 2.0 型聯合。以下步驟介紹了用於配置此信任關係的一般過程。組織內部必須有支援 SAML 2.0 的 IdP，例如 Microsoft Active Directory Federation Service (AD FS，Windows Server 的一部分)、Shibboleth 或其他相容的 SAML 2.0 提供者。

> **Note**
>
> 若要改善聯合彈性，建議您將 IdP 和 AWS 聯合設定為支援多個 SAML 登入端點。如需詳細資訊，請參閱 AWS 安全部落格文章如何使用區域 SAML 端點進行容錯移轉。

將組織的 IdP 和 AWS 設定為彼此信任

1. 向組織的 IdP 註冊 AWS 為服務提供者 (SP)。使用 `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` 中的 SAML 中繼資料文件

   如需可能####值清單，請參閱AWS 登入端點中的區域欄。

   您可以選擇從 使用 SAML 中繼資料文件`https://signin.aws.amazon.com/static/saml-metadata.xml`。
2. 使用組織的 IdP 時，會產生一個同等的 SAML 中繼資料 XML 檔案，該檔案可將 IdP 描述為 AWS 中的 IAM 身分提供者。它必須包含發行者名稱、建立日期、過期日期，以及 AWS 可用來驗證組織中身分驗證回應 （ 聲明 ） 的金鑰。

   如果您允許從外部 IdP 傳送加密的 SAML 聲明，您必須使用組織的 IdP 產生私有金鑰檔案，並以 .pem 檔案格式將此檔案上傳至您的 IAM SAML 組態。 AWS STS 需要此私有金鑰來解密對應至上傳到 IdP 之公有金鑰的 SAML 回應。

   > **Note**
   >
   > 如 SAML V2.0 Metadata Interoperability Profile Version 1.0 所定義，IAM 既不會評估 SAML 中繼資料文件中的 X.509 憑證是否過期，也不會對其採取動作。如果您擔心過期的 X.509 憑證，建議您監控憑證過期日期，並根據組織的治理和安全政策輪換憑證。
3. 在 IAM 主控台中，您可以建立一個 SAML 身分提供者。在此過程中，您將上傳由組織的 IdP 在Step 2 中產生的 SAML 中繼資料文件和私有解密金鑰。如需詳細資訊，請參閱在 IAM 中建立 SAML 身分提供者。

4. 在 IAM 中，建立一或多個 IAM 角色。在角色的信任政策中，您將 SAML 提供者設定為委託人，這會在您的組織與 之間建立信任關係 AWS。該角色的許可政策將決定允許您組織的使用者在 AWS 中執行的操作。如需詳細資訊，請參閱為第三方身分提供者建立角色。

   > **Note**
   >
   > 角色信任政策中使用的 SAML IDP 必須與角色所在的帳戶相同。

5. 在您組織的 IdP 中，定義可將組織中的使用者或群組映射到 IAM 角色的聲明。請注意，組織中不同的使用者和群組可能映射到不同的 IAM 角色。執行映射的確切步驟取決於您使用的 IdP。在使用者的 Amazon S3 資料夾的早期方案中，可能出現所有使用者映射到提供 Amazon S3 許可的同一角色的情況。如需詳細資訊，請參閱為身分驗證回應設定 SAML 聲明。

   如果您的 IdP 啟用 AWS 主控台的 SSO，則您可以設定主控台工作階段的最長持續時間。如需詳細資訊，請參閱啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。

6. 在您建立的應用程式中，您可以呼叫 AWS Security Token Service `AssumeRoleWithSAML` API，將您在 中建立的 SAML 提供者的 ARNStep 3、角色的 ARN 傳遞給 API，以假設您在 中建立Step 4，以及關於您從 IdP. AWS makes 取得之目前使用者的 SAML 聲明，確保擔任角色的請求來自 SAML 提供者中參考的 IdP。

   如需詳細資訊，請參閱 AWS Security Token Service API 參考中的 AssumeRoleWithSAML。

7. 如果請求成功，API 會傳回一組臨時安全性憑證，您的應用程式即可用其向 AWS發出已簽署的請求。您的應用程式具有有關目前使用者的資訊並可存取 Amazon S3 中使用者特定的資料夾，如上一方案中所述。

允許 SAML 聯合存取 AWS 資源的角色概觀

IAM 中建立的角色定義了組織中允許在 AWS中執行操作的 SAML 聯合身分主體。當您為角色建立信任政策時，您可以將先前建立的 SAML 提供者指定為 `Principal`。此外，您還可以使用 `Condition` 設定信任政策的範圍，以便僅允許與特定 SAML 屬性相符的使用者存取角色。例如，您可以指定僅允許 SAML 從屬關係為 `staff` (在 https://openidp.feide.no 中聲明) 的使用者存取角色，如以下範例政策所示：

JSON

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

```
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://us-east-1.signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws-cn:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:aud": "https://cn-north-1.signin.amazonaws.cn/saml",
                    "saml:iss": "https://openidp.feide.no"
                },
                "ForAllValues:StringLike": {
                    "saml:edupersonaffiliation": [
                        "staff"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> 角色信任政策中使用的 SAML IDP 必須與角色所在的帳戶相同。

政策中的 `saml:aud` 內容索引鍵指定了登入主控台時瀏覽器顯示的 URL。此登入端點 URL 必須與身分提供者的 recipient 屬性相符。您可以在特定區域內包含登入 URLs。 AWS 建議使用區域端點而非全域端點來改善聯合彈性。如果您只設定了一個端點，則在極少數情況下，端點無法使用時，您將無法聯合到 AWS 。如需可能 *####*值清單，請參閱AWS 登入端點中的區域欄。

下列範例顯示具有選用 的登入 URL 格式`region-code`。

`https://`*`region-code`*`.signin.aws.amazon.com/saml`

如果需要 SAML 加密，登入 URL 必須包含 AWS 指派給 SAML 提供者的唯一識別碼，您可以在「身分提供者詳細資訊」頁面上找到此識別碼。在下列範例中，登入 URL 包含 IdP 唯一識別碼，這需要在登入路徑後附加 /acs/。

`https://`*`region-code`*`.signin.aws.amazon.com/saml/acs/`*`IdP-ID`*

對於該角色中的許可政策，您可以像任何角色一樣指定許可。例如，如果允許您組織的使用者管理 Amazon Elastic Compute Cloud 執行個體，您必須在許可政策中明確允許 Amazon EC2 動作 (例如 AmazonEC2FullAccess 受管政策中的動作)。

如需有關您可以在政策中簽署的 SAML 索引鍵的詳細資訊，請參閱 SAML AWS STS 型聯合的可用金鑰。

單獨辨識以 SAML 為基礎的聯合身分中的使用者

在 IAM 中建立存取政策時，可根據使用者的身分指定許可，這一點通常很有用。舉例來說，對於已使用 SAML 聯合身分的使用者，應用程式可能希望使用如下的結構保留 Amazon S3 中的資訊：

```
amzn-s3-demo-bucket/app1/user1
amzn-s3-demo-bucket/app1/user2
amzn-s3-demo-bucket/app1/user3
```

您可以透過 Amazon S3 主控台或 建立儲存貯體 (`amzn-s3-demo-bucket`) 和資料夾 (`app1`) AWS CLI，因為這些是靜態值。不過，使用者特定的資料夾 (*`user1`*、*`user2`*、*`user3`* 等等) 必須在執行階段使用程式碼建立，因為在使用者首次透過聯合身分程序登入時，用來識別使用者的值未知。

若要編寫在資源名稱中引用特定於使用者的詳細資訊的政策，必須在可以用於政策條件的 SAML 索引鍵中提供使用者身分。以下索引鍵可供以 SAML 2.0 為基礎的聯合身分在 IAM 政策中使用。您可以使用以下索引鍵傳回的值為資源 (如 Amazon S3 資料夾) 建立唯一的使用者識別碼。

- `saml:namequalifier`. 雜湊值，以 `Issuer` 回應值 (`saml:iss`)、含 AWS 帳戶 ID 的字串與 IAM 中 SAML 提供者的易記名稱 (ARN 的最後一部分) 的串聯為基礎。帳戶 ID 與 SAML 提供者的易記名稱的串聯可作為索引鍵 `saml:doc` 供 IAM 政策使用。帳戶 ID 與提供者名稱必須使用「/」分隔，例如「123456789012/provider_name」。如需詳細資訊，請參閱 `saml:doc` 中的 SAML AWS STS 型聯合的可用金鑰 索引鍵。

  NameQualifier 與 Subject 的組合可用於唯一識別 SAML 聯合身分主體。下列虛擬程式碼顯示這個值是如何計算出來的。在此虛擬程式碼中，+ 表示串聯，SHA1 代表使用 SHA-1 產生訊息摘要的功能，Base64 64 代表產生雜湊輸出的 Base-64 編碼版本的功能。

  ```
  Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
  MySAMLIdP" ) )
  ```

  如需有關可用於以 SAML 為基礎的聯合身分政策索引鍵的詳細資訊，請參閱 SAML AWS STS 型聯合的可用金鑰。
- `saml:sub` (string). 這是該陳述的主題，其中包含單獨辨識組織中某個使用者的值 (例如 `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`)。
- `saml:sub_type` (string). 此索引鍵可以是 `persistent`、`transient` 或在您的 SAML 聲明中使用的 `Format` 與 `Subject` 元素的完整 `NameID` URI。`persistent` 的值表示在所有工作階段中使用者的 `saml:sub` 值是相同的。如果值為 `transient`，則使用者在每個工作階段中擁有不同的 `saml:sub` 值。如需 `NameID` 元素的 `Format` 屬性的詳細資訊，請參閱 為身分驗證回應設定 SAML 聲明。

以下範例顯示了一個許可政策，該政策使用上述索引鍵為 Amazon S3 中的使用者特定資料夾授予許可。該政策假設 Amazon S3 物件使用同時包含 `saml:namequalifier` 與 `saml:sub` 的字首識別。請注意，`Condition` 元素包括一個測試，用於確保 `saml:sub_type` 設為 `persistent`。如果已設為 `transient`，每個工作階段使用者的 `saml:sub` 值可能不同，因此不應使用值的組合來辨識使用者特定的資料夾。

JSON

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

```
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

如需有關從 IdP 映射聲明到政策索引鍵的詳細資訊，請參閱 為身分驗證回應設定 SAML 聲明。

在 IAM 中建立 SAML 身分提供者

IAM SAML 2.0 身分提供者是 IAM 中的實體，其負責描述支援 SAML 2.0 (安全性聲明標記語言 2.0) 標準的外部身分提供者 (IdP) 服務。當您想要在 Shibboleth 或 Active Directory Federation Services 和 等 SAML 相容 IdP 之間建立信任時 AWS，您可以使用 IAM 身分提供者，以便您的使用者可以存取 AWS 資源。在 IAM 信任政策中，IAM SAML 身分提供者將作為主體。

如需有關此案例的詳細資訊，請參閱 SAML 2.0 聯合身分。

您可以在 中或使用 Tools for Windows PowerShell 或 AWS API 呼叫 AWS 管理主控台 AWS CLI來建立和管理 IAM 身分提供者。

在您建立 SAML 提供者之後，您必須建立一個或多個 IAM 角色。角色是 中的身分 AWS ，沒有自己的登入資料 （與使用者相同）。但在此情況中，角色會動態指派給 SAML 聯合身分主體，而該主體由 IdP 進行驗證。該角色允許您的 IdP 請求暫時性安全憑證以存取 AWS。指派給角色的政策會決定允許使用者在其中執行的操作 AWS。若要為 SAML 聯合身分建立角色，請參閱 為第三方身分提供者建立角色。

最後，在建立角色之後，您可以使用有關 的資訊 AWS 和您希望 SAML 聯合委託人使用的角色來設定 IdP，以完成 SAML 信任。這是指在您的 IdP 和 AWS之間設定依賴方信任。若要設定依賴方信任，請參閱使用依賴方信任設定您的 SAML 2.0 IdP 並新增宣告。

主題

- 先決條件
- 建立並管理 IAM SAML 身分提供者 (主控台)
- 管理 SAML 加密金鑰
- 建立並管理 IAM SAML 身分提供者 (AWS CLI)
- 建立和管理 IAM SAML 身分提供者 (AWS API)
- 後續步驟

先決條件

您必須從 IdP 取得下列資訊，然後才能建立 SAML 身分提供者。

- 從 IdP 取得 SAML 中繼資料文件。此文件包括發行者的名稱、到期資訊以及可用於驗證從 IdP 接收的 SAML 身分驗證回應 (宣告) 的金鑰。若要產生中繼資料文件，請使用您的外部 IdP 提供的身分管理軟體。

> **Important**
>
> 此中繼資料檔案包括發行者名稱、過期資訊以及可用於驗證從 IdP 接收的 SAML 身分驗證回應 (聲明) 的金鑰。中繼資料檔案必須以 UTF-8 格式編碼，並且不含位元組順序記號 (BOM)。若要移除 BOM，您可以使用文字編輯工具，例如 Notepad++，將檔案編碼為 UTF-8。
>
> 包含在 SAML 中繼資料文件中的 X.509 憑證必須使用大小至少有 1024 位元的金鑰。此外，X.509 憑證也必須沒有任何重複的擴充。您可以使用擴充，但這些擴充只能在憑證中出現一次。如果 X.509 憑證不符合任一條件，IdP 建立會失敗，並傳回 "Unable to parse metadata" 錯誤。
>
> 如 SAML V2.0 Metadata Interoperability Profile Version 1.0 所定義，IAM 既不會評估 SAML 中繼資料文件中的 X.509 憑證是否過期，也不會對其採取動作。如果您擔心過期的 X.509 憑證，建議您監控憑證過期日期，並根據組織的治理和安全政策輪換憑證。

- 如果選擇啟用 SAML 加密，必須使用 IdP 產生私有金鑰檔案，並以 .pem 檔案格式將此檔案上傳至 IAM SAML 組態。 AWS STS 需要此私有金鑰，才能解密對應於上傳至 IdP 之公有金鑰的 SAML 回應。支援下列演算法：
  - 加密演算法
    - AES-128
    - AES-256
    - RSA-OAEP
  - 金鑰傳輸演算法
    - AES-CBC
    - AES-GCM

  如需產生私有金鑰的步驟，請參閱身分提供者的文件。

> **Note**
>
> IAM Identity Center 和 Amazon Cognito 不支援來自 IAM SAML 身分提供者的加密 SAML 聲明。您可以間接將對加密 SAML 聲明的支援新增至 Amazon Cognito 身分集區與 Amazon Cognito 使用者集區的聯合。使用者集區具有獨立 IAM SAML 聯合身分的 SAML 聯合身分，並支援 SAML 簽署和加密。雖然此功能不會直接延伸至身分集區，但使用者集區可以是身分集區的 IdP。若要將 SAML 加密與身分集區搭配使用，請將具有加密功能的 SAML 提供者新增至作為身分集區的 IdP 的使用者集區。

您的 SAML 提供者必須能夠使用使用者集區提供的金鑰來加密 SAML 聲明。使用者集區不會接受使用 IAM 提供的憑證加密的聲明。

如需如何設定許多可用 IdPs 以使用的指示 AWS，包括如何產生所需的 SAML 中繼資料文件，請參閱將第三方 SAML 解決方案供應商與 整合 AWS。

如需 SAML 聯合身分的協助，請參閱對 SAML 聯合身分進行疑難排解。

建立並管理 IAM SAML 身分提供者 (主控台)

您可以使用 AWS 管理主控台 來建立、更新和刪除 IAM SAML 身分提供者。如需 SAML 聯合身分的協助，請參閱對 SAML 聯合身分進行疑難排解。

建立 IAM SAML 身分提供者 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，請選擇 Identity providers (身分提供者)，然後選擇 Add provider (新增提供者)。
3. 在 Configure provider (設定提供者) 中，選擇 SAML。
4. 輸入身分提供者的名稱。
5. 對於 Metadata document (中繼資料文件)，選擇 Choose file (選擇檔案)，並指定您在 the section called “先決條件” 下載的 SAML 中繼資料文件。

   > **Note**
   >
   > 身分提供者顯示的有效截止日期由 IAM 在提供者建立時間設定。它不會反映 SAML 中繼資料文件中的 `validUntil`或 `cacheDuration` 屬性。IAM 不會使用這些中繼資料屬性來判斷有效截止日期。
   >
   > 在 SAML 身分驗證期間，IAM 不會對 SAML 中繼資料文件或 SAML 中繼資料文件中的 X.509 憑證過期進行評估或採取動作。SAML 身分驗證有效性會在宣告層級強制執行，並在宣告的 `Conditions`和 `SubjectConfirmationData`元素中使用 `NotOnOrAfter` 屬性。如果您擔心過期的 X.509 憑證，建議您監控憑證過期日期，並根據組織的治理和安全政策輪換憑證。

6. (選用) 在 SAML 加密欄位中，選擇選擇檔案，然後選取您在 the section called “先決條件” 中建立的私有金鑰檔案。選擇需要加密以僅接受來自 IdP 的加密請求。

7. (選用) 對於 Add tags (新增標籤)，您可以新增鍵值組，以協助您識別和整理 IdP。您也可以使用標籤來控制對 AWS 資源的存取。若要進一步了解如何標記 SAML 身分提供者，請參閱標記 IAM SAML 身分提供者。

   選擇 Add tag (新增標籤)。輸入每個標籤鍵值組的值。

8. 請確認您提供的資訊。完成後，請選擇 Add provider (新增提供者)。
9. 將 IAM 角色指派給您的身分提供者。此角色提供身分提供者管理的外部使用者身分許可，以存取您帳戶中 AWS 的資源。若要深入了解如何建立聯合身分角色，請參閱 為第三方身分提供者建立角色。

   > **Note**
   >
   > 角色信任政策中使用的 SAML IDP 必須與角色所在的帳戶相同。

刪除 SAML 提供者 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，請選擇 Identity providers (身分提供者)。
3. 在您要刪除的身分提供者旁邊，選取選項按鈕。
4. 選擇 Delete (刪除)。新的視窗將開啟。
5. 在欄位中輸入單字 `delete`，確認您要刪除提供者。再選擇 Delete (刪除)。

管理 SAML 加密金鑰

您可以將 IAM SAML 供應商設定為從外部 IdP 接收 SAML 回應中的加密聲明。使用者可以在 AWS 中使用加密的 SAML 聲明擔任角色，方法是呼叫 `sts:AssumeRoleWithSAML`。

SAML 加密可確保聲明在透過中介或第三方傳遞時是安全的。此外，此功能可協助您符合 FedRAMP 或任何強制要求 SAML 聲明加密的內部合規政策要求。

若要設定 IAM SAML 身分提供者，請參閱在 IAM 中建立 SAML 身分提供者。如需 SAML 聯合身分的協助，請參閱對 SAML 聯合身分進行疑難排解。

輪換 SAML 加密金鑰

IAM 使用您上傳至 IAM SAML 提供者的私有金鑰，從 IdP 解密加密的 SAML 聲明。您可以為每個身分提供者儲存最多兩個私有金鑰檔案，讓您視需要輪換私有金鑰。儲存兩個檔案時，每個請求會先嘗試以最新新增日期解密，然後 IAM 會嘗試以最舊新增日期解密請求。

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，請選擇身分提供者，然後從清單中選取提供者。
3. 選擇 SAML 加密索引標籤，然後選擇新增金鑰。
4. 選取選擇檔案，並上傳您從 IdP 下載的私有金鑰作為 .pem 檔案。然後，選擇新增金鑰。
5. 在 SAML 解密的私有金鑰區段中，選取過期的私有金鑰檔案，然後選擇移除。建議在新增私有金鑰之後移除過期的私有金鑰，確保首次嘗試解密聲明能夠成功。

建立並管理 IAM SAML 身分提供者 (AWS CLI)

您可以使用 AWS CLI 來建立、更新和刪除 SAML 供應商。如需 SAML 聯合身分的協助，請參閱對 SAML 聯合身分進行疑難排解。

建立 IAM 身分提供者並上傳中繼資料文件 (AWS CLI)

- 執行此命令：`aws iam create-saml-provider`

若要更新 IAM SAML 身分提供者 (AWS CLI)

您可以更新中繼資料檔案、SAML 加密設定，以及輪換 IAM SAML 提供者的私有金鑰解密檔案。若要輪換私有金鑰，請在個別請求中新增私有金鑰，然後移除舊金鑰。如需有關輪換私有金鑰的詳細資訊，請參閱 管理 SAML 加密金鑰。

- 執行此命令：`aws iam update-saml-provider`

標記現有 IAM 身分提供者 (AWS CLI)

- 執行此命令：`aws iam tag-saml-provider`

列出現有 IAM 身分提供者的標籤 (AWS CLI)

- 執行此命令：`aws iam list-saml-provider-tags`

移除現有 IAM 身分提供者的標籤 (AWS CLI)

- 執行此命令：`aws iam untag-saml-provider`

刪除 IAM SAML 身分提供者 (AWS CLI)

1. (選用) 若要列出所有提供者的資訊，例如 ARN、建立日期和過期日期，請執行下列命令：
   - `aws iam list-saml-providers`
2. (選用) 若要取得有關特定提供者的資訊 (例如 ARN、建立日期、過期日期、加密設定和私有金鑰資訊)，請執行下列命令：
   - `aws iam get-saml-provider`
3. 若要刪除 IAM 身分提供者，請執行下列命令：
   - `aws iam delete-saml-provider`

建立和管理 IAM SAML 身分提供者 (AWS API)

您可以使用 AWS API 來建立、更新和刪除 SAML 供應商。如需 SAML 聯合身分的協助，請參閱對 SAML 聯合身分進行疑難排解。

建立 IAM 身分提供者並上傳中繼資料文件 (AWS API)

- 呼叫此操作：`CreateSAMLProvider`

更新 IAM SAML 身分提供者 (AWS API)

您可以更新中繼資料檔案、SAML 加密設定，以及輪換 IAM SAML 提供者的私有金鑰解密檔案。若要輪換私有金鑰，請在個別請求中新增私有金鑰，然後移除舊金鑰。如需有關輪換私有金鑰的詳細資訊，請參閱 管理 SAML 加密金鑰。

- 呼叫此操作：`UpdateSAMLProvider`

標記現有的 IAM 身分提供者 (AWS API)

- 呼叫此操作：`TagSAMLProvider`

列出現有 IAM 身分提供者 (AWS API) 的標籤

- 呼叫此操作：ListSAMLProviderTags

移除現有 IAM 身分提供者 (AWS API) 上的標籤

- 呼叫此操作：UntagSAMLProvider

刪除 IAM 身分提供者 (AWS API)

1. (選用) 若要列出所有 IdP 的資訊，例如 ARN、建立日期和過期，請呼叫下列操作：
   - ListSAMLProviders
2. (選用) 若要取得有關特定提供者的資訊 (例如 ARN、建立日期、過期日期、加密設定和私有金鑰資訊)，請呼叫下列操作：
   - GetSAMLProvider
3. 若要刪除 IdP，請呼叫下列操作：
   - DeleteSAMLProvider

後續步驟

建立 SAML 身分提供者之後，使用 IdP 設定依賴方信任。您也可以在政策中使用來自 IdP 身分驗證回應的宣告，來控制對角色的存取。

- 您必須告知 IdP AWS 作為服務提供者。這稱為在 IdP 和 AWS之間新增依賴方信任。新增依賴方信任的過程，取決於您使用哪種 IdP。如需詳細資訊，請參閱使用依賴方信任設定您的 SAML 2.0 IdP 並新增宣告。
- 當 IdP 傳送包含宣告的回應至 時 AWS，許多傳入宣告會對應至 AWS 內容索引鍵。您可以利用 Condition 元素在 IAM 政策中使用這些內容索引鍵，來控制對角色的存取。如需詳細資訊，請參閱為身分驗證回應設定 SAML 聲明

使用依賴方信任設定您的 SAML 2.0 IdP 並新增宣告

當您為 SAML 存取權建立 IAM 身分提供者和角色時，基本上是在告知 AWS 有關允許身分提供者 (IdP) 和其使用者執行哪些動作。您的下一個步驟是告知 IdP AWS 作為服務提供者。這稱為在 IdP 和 之間

新增依賴方信任 AWS。新增依賴方信任的過程，取決於您使用哪種 IdP。如需詳細資訊，請參閱身分管理軟體的文件。

許多 IdP 都可以讓您指定 URL，以供 IdP 從其讀取包含依賴方資訊和憑證的 XML 文件。對於 AWS，請使用登入端點 URL。下列範例顯示具有選用 的 URL 格式`region-code`。

```
https://region-code.signin.aws.amazon.com/static/saml-metadata.xml
```

如果需要 SAML 加密，URL 必須包含 AWS 指派給 SAML 供應商的唯一識別符，您可以在身分提供者詳細資訊頁面上找到該識別符。下列範例展示了包含唯一識別碼的區域登入 URL。

```
https://region-code.signin.aws.amazon.com/static/saml/IdP-ID/saml-metadata.xml
```

如需可能####值清單，請參閱AWS 登入端點中的區域欄。對於 AWS 值，您也可以使用非區域端點`https://signin.aws.amazon.com/saml`。

如果您無法直接指定 URL，請從先前的 URL 下載 XML 文件，然後匯入到您的 IdP 軟體。

您也需要在 IdP 中建立適當的宣告規則，將 指定 AWS 為依賴方。當 IdP 傳送 SAML 回應至 AWS 端點時，它會包含包含一或多個宣告的 SAML 聲明。宣告是和使用者及其群組相關的資訊。宣告規則將該資訊對應到 SAML 屬性。這可讓您確保來自 IdP 的 SAML 身分驗證回應包含 IAM 政策 AWS 中使用的必要屬性，以檢查 SAML 聯合身分主體的許可。如需詳細資訊，請參閱下列主題：

- 允許 SAML 聯合存取 AWS 資源的角色概觀。此主題討論在 IAM 政策中使用 SAML 特定金鑰，以及如何使用這些政策來限制 SAML 聯合身分主體的許可。
- 為身分驗證回應設定 SAML 聲明. 此主題討論如何設定 SAML 宣告，其包含有關使用者的資訊。聲明已捆綁在 SAML 聲明中，並且包含在傳送到 AWS的 SAML 回應中。您必須確保 AWS 政策所需的資訊以 AWS 可識別和使用的形式包含在 SAML 聲明中。
- 將第三方 SAML 解決方案供應商與 整合 AWS。 本主題提供第三方組織提供有關如何整合身分解決方案的文件連結 AWS。

> **Note**
>
> 若要改善聯合彈性，建議您將 IdP 和 AWS 聯合設定為支援多個 SAML 登入端點。如需詳細資訊，請參閱 AWS 安全部落格文章如何使用區域 SAML 端點進行容錯移轉。

將第三方 SAML 解決方案供應商與 整合 AWS

> **Note**
>
> 我們建議您要求人類使用者在存取時使用臨時登入資料 AWS。您是否考慮過使用 AWS IAM Identity Center？ 您可以使用 IAM Identity Center 集中管理對多個 的存取， AWS 帳戶 並為使用者提供受 MFA 保護的單一登入存取，可從單一位置存取其所有指派的帳戶。使用 IAM Identity Center，您可以在 IAM Identity Center 中建立和管理使用者身分，或輕鬆連線至您現有的 SAML 2.0 相容身分提供者。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

下列連結可協助您設定第三方 SAML 2.0 身分提供者 (IdP) AWS 解決方案以使用聯合。請洽詢您的身分提供者，以判斷他們是否支援 SAML 權杖加密。如需 SAML 加密要求，請參閱管理 SAML 加密金鑰。

> **Tip**
>
> AWS 支援工程師可以透過涉及第三方軟體的一些整合任務，協助擁有業務和企業支援計劃的客戶。如需所支援平台和應用程式的最新清單，請參閱 AWS Support 常見問答集中的支援哪些第三方軟體？

解決方案	其他資訊
Auth0	與 Amazon Web Services 整合 – Auth0 文件網站上的此頁面提供 資源的連結，說明如何使用 設定單一登入 (SSO)，AWS 管理主控台 並包含 JavaScript 範例。您可以設定 Auth0 來傳遞工作階段標籤。如需詳細資訊，請參閱 Auth0 宣布與 AWS for IAM 工作階段標籤的合作夥伴關係。
Microsoft Entra	教學課程：Microsoft Entra SSO 與 AWS 單一帳戶存取整合 – Microsoft 網站上的本教學課程說明如何使用 SAML 聯合將 Microsoft Entra （ 先前稱為 Azure AD) 設定為身分提供者 (IdP)。

解決方案	其他資訊
Centrify	設定 Centrify 並使用 SAML for SSO to AWS – Centrify 網站上的此頁面說明如何將 Centrify 設定為使用 SAML for SSO to AWS。
CyberArk	設定 CyberArk 以提供 Amazon Web Services (AWS) 存取許可，讓使用者從 CyberArk 使用者入口網站透過 SAML 單一登入 (SSO) 進行登入。
ForgeRock	ForgeRock Identity Platform 與 整合 AWS。您可以設定 ForgeRock 來傳遞工作階段標籤。如需詳細資訊，請參閱 Amazon Web Services 的屬性型存取控制。
Google Workspace	Amazon Web Services 雲端應用程式 – Google Workspace Admin Help 網站上的這篇文章說明如何使用 作為服務提供者 AWS ，將 Google Workspace 設定為 SAML 2.0 IdP。
IBM	您可以設定 IBM 傳遞工作階段標籤。如需詳細資訊，請參閱 IBM Cloud Identity IDaaS，這是支援 AWS 工作階段標籤的其中一個。
JumpCloud	使用 Amazon 透過 IAM 角色授予單一登入 (SSO) 存取 AWS – JumpCloud 網站上的這篇文章說明如何根據 IAM 角色設定和啟用 SSO AWS。
Matrix42	MyWorkspace 入門指南 – 本指南說明如何將 AWS 身分服務與 Matrix42 MyWorkspace 整合。

解決方案	其他資訊
Microsoft Active Directory Federation Services (AD FS)	欄位備註：將 Active Directory Federation Service 與 整合 AWS IAM Identity Center – AWS 架構部落格上的這篇文章說明 AD FS 和 AWS IAM Identity Center (IAM Identity Center) 之間的身分驗證流程。IAM Identity Center 使用 SAML 2.0 支援聯合身分，可與 AD FS 解決方案整合。使用者可透過其公司憑證登入 IAM Identity Center 入口網站，減少在 IAM Identity Center 上維護單獨憑證的管理開銷。您也可以設定 AD FS 來傳遞工作階段標籤。如需詳細資訊，請參閱搭配 AD FS 使用屬性型存取控制，以簡化 IAM 許可管理。
miniOrange	SSO for AWS – miniOrange 網站上的此頁面說明如何 AWS 為企業建立安全存取，並完全控制應用程式的存取 AWS 。
Okta	使用 Okta 整合 Amazon Web Services 命令列介面 – 從 Okta 支援網站的這個頁面，您可以了解如何設定 Okta 以與 AWS一起使用。您可以設定 Okta 傳遞工作階段標籤。如需詳細資訊，請參閱 Okta 和 AWS 合作夥伴透過工作階段標籤簡化存取。
Okta	AWS 帳戶聯合 – Okta 網站上的本節說明如何設定和啟用 IAM Identity Center AWS。
OneLogin	從 OneLogin 知識庫中，搜尋文章**SAML AWS**清單，說明如何在 OneLogin 和 之間設定 IAM Identity Center 功能，AWS 以尋找單一角色和多角色案例。您可以設定 OneLogin 傳遞工作階段標籤。如需詳細資訊，請參閱 OneLogin 和工作階段標籤：以屬性為基礎的 AWS 資源存取控制。
Ping 身分	PingFederate AWS Connector – 檢視 PingFederate AWS Connector 的詳細資訊，這是一種快速連線範本，可輕鬆設定單一登入 (SSO) 和佈建連線。閱讀文件並下載最新的 PingFederate AWS Connector 以進行整合 AWS。您可以設定 Ping 身分以傳遞工作階段標籤。如需詳細資訊，請參閱宣佈 Ping Identity 支援 AWS中的屬性型存取控制。

解決方案	其他資訊
RadiantLogic	Radiant Logic Technology Partners – Radiant Logic 的 RadiantOne 聯合身分服務與 整合 AWS ，為 SAML 型 SSO 提供身分中樞。
RSA	Amazon Web Services - RSA Ready Implementation Guide 提供整合 AWS 和 RSA 的指引。如需有關 SAML 組態的詳細資訊，請參閱 Amazon Web Services - SAML My Page SSO Configuration - RSA Ready Implementation Guide。
Salesforce.com	如何設定從 Salesforce 到 的 SSO AWS – https：//Salesforce.com 開發人員網站上的此教學文章說明如何在 Salesforce 中設定身分提供者 (IdP)，並將 AWS 設定為服務提供者。
SecureAuth	AWS - SecureAuth SAML SSO – SecureAuth 網站上的這篇文章說明如何為 SecureAuth 設備設定與 AWS 的 SAML 整合。
Shibboleth	如何使用 Shibboleth 進行 SSO 到 AWS 管理主控台 - AWS 安全部落格上的這個項目提供step-by-step教學，說明如何設定 Shibboleth 並將其設定為 的身分提供者 AWS。您可以設定 Shibboleth 傳遞工作階段標籤。

如需詳細資訊，請參閱 AWS 網站上的 IAM 合作夥伴頁面。

為身分驗證回應設定 SAML 聲明

驗證組織中的使用者身分後，外部身分提供者 (IdP) 會將身分驗證回應傳送至 AWS 登入端點 URL。此回應是一個包含 SAML 權杖的 POST 請求，該權杖遵循適用於 SAML 2.0 的 HTTP POST 繫結標準，其中包含下列元素或宣告。您可在與 SAML 相容的 IdP 中配置這些聲明。請參考 IdP 文件，以了解有關如何輸入這些聲明的說明。

當 IdP 將包含宣告的回應傳送到 時 AWS，許多傳入宣告會映射到 AWS 內容索引鍵。可以在 IAM 政策中使用 `Condition` 元素來檢查這些內容索引鍵。可用映射的清單將根據 映射 SAML 屬性以 AWS 信任政策內容索引鍵 中所示。

Subject 與 NameID

回應必須恰好包含一個 SubjectConfirmation 元素，而該元素所包含的 SubjectConfirmationData 元素須同時具備 NotOnOrAfter 屬性和 Recipient 屬性。收件人屬性必須包含符合 AWS 登入端點 URL 的值。您的 IdP 可能會使用 ACS、 Recipient或 一詞Target來參考此屬性。

如果需要 SAML 加密，登入 URL 必須包含 AWS 指派給 SAML 提供者的唯一識別碼，您可以在「身分提供者詳細資訊」頁面上找到此識別碼。下列範例顯示具有選用 的登入 URL 格式region-code。

https://*region-code*.signin.aws.amazon.com/saml

在下列範例中，登入 URL 包含唯一識別碼，這需要在登入路徑後附加 /acs/。

https://*region-code*.signin.aws.amazon.com/saml/acs/*IdP-ID*

如需可能####值清單，請參閱AWS 登入端點中的區域欄。對於 AWS 值，您也可以使用全域登入端點 https://signin.aws.amazon.com/saml。

NameID 元素的值可以是持久的、暫時的，也可以由 IdP 解決方案提供的完整格式 URI 組成。持久的值表示 NameID 中的值對於工作階段之間的使用者是相同的。如果值為暫時的，則使用者在每個工作階段中擁有不同的 NameID 值。單一登入互動支援下列類型的識別碼：

- urn:oasis:names:tc:SAML:2.0:nameid-format:persistent
- urn:oasis:names:tc:SAML:2.0:nameid-format:transient
- urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress
- urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified
- urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName
- urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName
- urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos
- urn:oasis:names:tc:SAML:2.0:nameid-format:entity

下列的摘錄顯示了範例。用您自己的值替代標記值。

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
```

```
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://region-code.signin.aws.amazon.com/saml/SAMLSP4SHN3UIS2D558H46"/>
  </SubjectConfirmation>
</Subject>
```

> **Important**
>
> saml:aud 內容索引鍵來自 SAML recipient 屬性，因其為等同於 OIDC 觀眾欄位 (例如 accounts.google.com:aud) 的 SAML。

PrincipalTag SAML 屬性

(選用) 您可以使用 Attribute 元素，並將 Name 屬性設為 https://aws.amazon.com/SAML/Attributes/PrincipalTag:*{TagKey}*。此元素可讓您在 SAML 聲明中將屬性做為工作階段標籤傳遞。如需有關工作階段標籤的詳細資訊，請參閱 在 中傳遞工作階段標籤 AWS STS。

若要將屬性做為工作階段標籤傳遞，請包含指定標籤值的 AttributeValue 元素。例如，若要傳遞標籤金鑰/值對 Project = Marketing 和 CostCenter = 12345，請使用以下屬性。為每個標籤包含個別的 Attribute 元素。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

如要將以上標籤設為可轉移，請在其中包含另一個 Attribute 元素，並將其 Name 屬性設為 https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys。這是一個選用的多重值屬性，會將您的工作階段標籤設為可轉移。可轉移標籤會在您於 AWS中使用 SAML 聲明擔任另一個角色時保存。這就是所謂的角色鏈接。例如，若要將 Principal 和 CostCenter 標籤設為可轉移，請使用以下屬性來指定金鑰。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

Role SAML 屬性

您可以使用 Attribute 元素，將 Name 屬性設為 https://aws.amazon.com/SAML/Attributes/Role。此元素包含一或多個 AttributeValue 元素，這些元素可列出將由您的 IdP 對應之使用者的 IAM 身分提供者和角色。該 IAM 角色和 IAM 身分提供者被指定為逗號分隔的 ARN 組，所採用的格式與傳遞到 AssumeRoleWithSAML 的 RoleArn 與 PrincipalArn 參數的格式相同。此元素必須包含至少一個角色提供者對 (AttributeValue 元素)，並且可以包含多個角色提供者對。如果該元素包含多個角色提供者組，則使用者將需要選擇其使用 WebSSO 登入 AWS 管理主控台時要擔任的角色。

> **Important**
>
> Name 標籤中 Attribute 屬性的值區分大小寫。必須精準設定為 https://aws.amazon.com/SAML/Attributes/Role。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

RoleSessionName SAML 屬性

您可以使用 Attribute 元素，將 Name 屬性設為 https://aws.amazon.com/SAML/Attributes/RoleSessionName。此元素包含一個為暫時憑證提供識別符的 AttributeValue 元素，而這些憑證是在擔任角色時所發出。您可以使用此功能將臨時憑證與正在使用應用程式的使用者建立關聯。此元素用於在 中顯示使用者資訊 AWS 管理主控台。AttributeValue 元素中的值長度必須介於 2 到 64 個字元之間，只能包含字母數位字元、底線和以下字元：. , + = @ - (連字號)。其中不可含有空格。該值通常是使用者 ID (john) 或電子郵件地址 (johndoe@example.com)。該值不應包含空格，如使用者的顯示名稱 (John Doe)。

> **Important**
>
> Name 標籤中 Attribute 屬性的值區分大小寫。必須精準設定為 https://aws.amazon.com/SAML/Attributes/RoleSessionName。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

SessionDuration SAML 屬性

(選用) 您可以使用 Attribute 元素，並將 Name 屬性設為 https://aws.amazon.com/SAML/Attributes/SessionDuration。此元素包含一個AttributeValue元素，指定使用者在必須請求新的臨時登入資料 AWS 管理主控台 之前，可以存取 的時間長度。該值是一個表示工作階段秒數的整數。該值的範圍是 900 秒 (15 分鐘) 到 43200 秒 (12 小時)。如果此屬性不存在，則憑證有效時間為 1 小時 (DurationSeconds API 的 AssumeRoleWithSAML 參數的預設值)。

若要使用此屬性，您必須設定 SAML 供應商， AWS 管理主控台 透過位於 的主控台登入 Web 端點提供 的單一登入存取https://*region-code*.signin.aws.amazon.com/saml。如需可能####值清單，請參閱AWS 登入端點中的區域欄。您可以選擇性地使用以下 URL：https://signin.aws.amazon.com/static/saml。請注意，此屬性只會將工作階段擴展到 AWS 管理主控台。它不能延長其他憑證的有效期。但是，如果其存在於 AssumeRoleWithSAML API 呼叫中，則可以用來縮短工作階段的持續時間。呼叫傳回的憑證預設存留期是 60 分鐘。

另請注意，如果同時定義了 SessionNotOnOrAfter 屬性，則兩個屬性的較小值 (SessionDuration 或 SessionNotOnOrAfter) 將建立主控台工作階段的持續時間上限。

在啟用具有更長持續時間的主控台工作階段時，可能會導致憑證洩露的風險。為了幫助緩解這種風險，您可以在 IAM 主控台的 Role Summary (角色摘要) 頁面上選擇 Revoke Sessions (撤銷工作階段) 來立即停用所有角色的使用中的主控台工作階段。如需詳細資訊，請參閱 撤銷 IAM 角色臨時安全憑證。

> **Important**
>
> Name 標籤中 Attribute 屬性的值區分大小寫。必須精準設定為 https://aws.amazon.com/SAML/Attributes/SessionDuration。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

`SourceIdentity` SAML 屬性

(選用) 您可以使用 `Attribute` 元素，並將 `Name` 屬性設為 `https://aws.amazon.com/SAML/Attributes/SourceIdentity`。此元素包含一個 `AttributeValue` 元素，提供使用 IAM 角色之人員或應用程式的識別碼。當您使用 SAML 工作階段在 AWS 稱為角色鏈結的 中擔任另一個角色時，來源身分的值會持續存在。針對在角色工作階段期間所採取的每個動作，來源身分的值會出現在請求中。設定的值無法在角色工作階段期間變更。然後，管理員可以使用 AWS CloudTrail 日誌來監控和稽核來源身分資訊，以判斷誰使用共用角色執行動作。

`AttributeValue` 元素中的值長度必須介於 2 到 256 個字元之間，只能包含英數字元、底線和下列字元：. ， + = @ - （連字號）。其中不可含有空格。此值通常是與使用者相關聯的屬性，例如使用者 ID (john) 或電子郵件地址 (johndoe@example.com)。該值不應包含空格，如使用者的顯示名稱 (John Doe)。如需有關使用來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。

> **Important**
>
> 如果您的 SAML 聲明設定為使用 `SourceIdentity` 屬性，則您的信任政策也必須包含 `sts:SetSourceIdentity` 動作，否則擔任角色操作將失敗。如需有關使用來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。

若要傳遞來源身分屬性，請包含指定來源身分值的 `AttributeValue` 元素。例如，若要傳遞來源身分 `Diego`，請使用下列屬性。

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>Diego</AttributeValue>
```

映射 SAML 屬性以 AWS 信任政策內容索引鍵

本部分中的表列出了常用的 SAML 屬性以及它們在 AWS中映射到信任政策條件內容索引鍵的方式。您可以使用這些金鑰來控制角色的存取。若要執行此作業，請將金鑰與包含在 SAML 存取請求隨附聲明中的值進行比較。

> **Important**
>
> 這些索引鍵僅在 IAM 信任政策 (可決定誰能擔任角色的政策) 中可用，並且不適用於許可政策。

在 eduPerson 和 eduOrg 屬性工作表中，值是以字串或字串清單的形式輸入的。對於字串值，您可以使用 `StringEquals` 或 `StringLike` 條件測試 IAM 信任政策中的這些值。對於包含字串清單的值，您可以使用 `ForAnyValue` 與 `ForAllValues` 政策集運算子來測試信任政策中的這些值。

> **Note**
>
> 每個 AWS 內容索引鍵應該只包含一個宣告。如果包含多個聲明，將僅映射其中一個。

下表顯示 eduPerson 和 eduOrg 屬性。

eduPerson 或 eduOrg 屬性 (**Name** 索引鍵)	對應至此 AWS 內容索引鍵 (**FriendlyName** 索引鍵)	Type
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPerson Affiliation`	字串清單
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	字串清單
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	字串
`urn:oid:1.3.6.1.4.1.5923.1.1.1.4`	`eduPerson OrgUnitDN`	字串清單
`urn:oid:1.3.6.1.4.1.5923.1.1.1.5`	`eduPerson PrimaryAf filiation`	字串
`urn:oid:1.3.6.1.4.1.5923.1.1.1.6`	`eduPerson PrincipalName`	字串
`urn:oid:1.3.6.1.4.1.5923.1.1.1.7`	`eduPerson Entitlement`	字串清單
`urn:oid:1.3.6.1.4.1.5923.1.1.1.8`	`eduPerson PrimaryOrgUnitDN`	字串

eduPerson 或 eduOrg 屬性 (**Name** 索引鍵)	對應至此 AWS 內容索引鍵 (**FriendlyName** 索引鍵)	Type
urn:oid:1.3.6.1.4.1.5923.1.1.1.9	eduPerson ScopedAff iliation	字串清單
urn:oid:1.3.6.1.4.1.5923.1. 1.1.10	eduPerson TargetedID	字串清單
urn:oid:1.3.6.1.4.1.5923.1. 1.1.11	eduPerson Assurance	字串清單
urn:oid:1.3.6.1.4.1.5923.1.2.1.2	eduOrgHomePageURI	字串清單
urn:oid:1.3.6.1.4.1.5923.1.2.1.3	eduOrgIde ntityAuth NPolicyURI	字串清單
urn:oid:1.3.6.1.4.1.5923.1.2.1.4	eduOrgLegalName	字串清單
urn:oid:1.3.6.1.4.1.5923.1.2.1.5	eduOrgSuperiorURI	字串清單
urn:oid:1.3.6.1.4.1.5923.1.2.1.6	eduOrgWhi tePagesURI	字串清單
urn:oid:2.5.4.3	cn	字串清單

下表顯示 Active Directory 屬性。

AD 屬性	對應至此 AWS 內容索引鍵	Type
http://schemas.xmlsoap.org/ws/2005/0 5/identity/claims/name	name	字串

AD 屬性	對應至此 AWS 內容索引鍵	Type
http://schemas.xmlsoap.org/claims/CommonName	commonName	字串
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname	givenName	字串
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname	surname	字串
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress	mail	字串
http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid	uid	String

下表顯示 X.500 屬性。

X.500 屬性	對應至此 AWS 內容索引鍵	Type
2.5.4.3	commonName	字串
2.5.4.4	surname	字串
2.4.5.42	givenName	字串
2.5.4.45	x500UniqueIdentifier	字串
0.9.2342.19200300100.1.1	uid	字串
0.9.2342.19200300100.1.3	mail	字串
0.9.2342.19200300.100.1.45	organizationStatus	字串

啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台

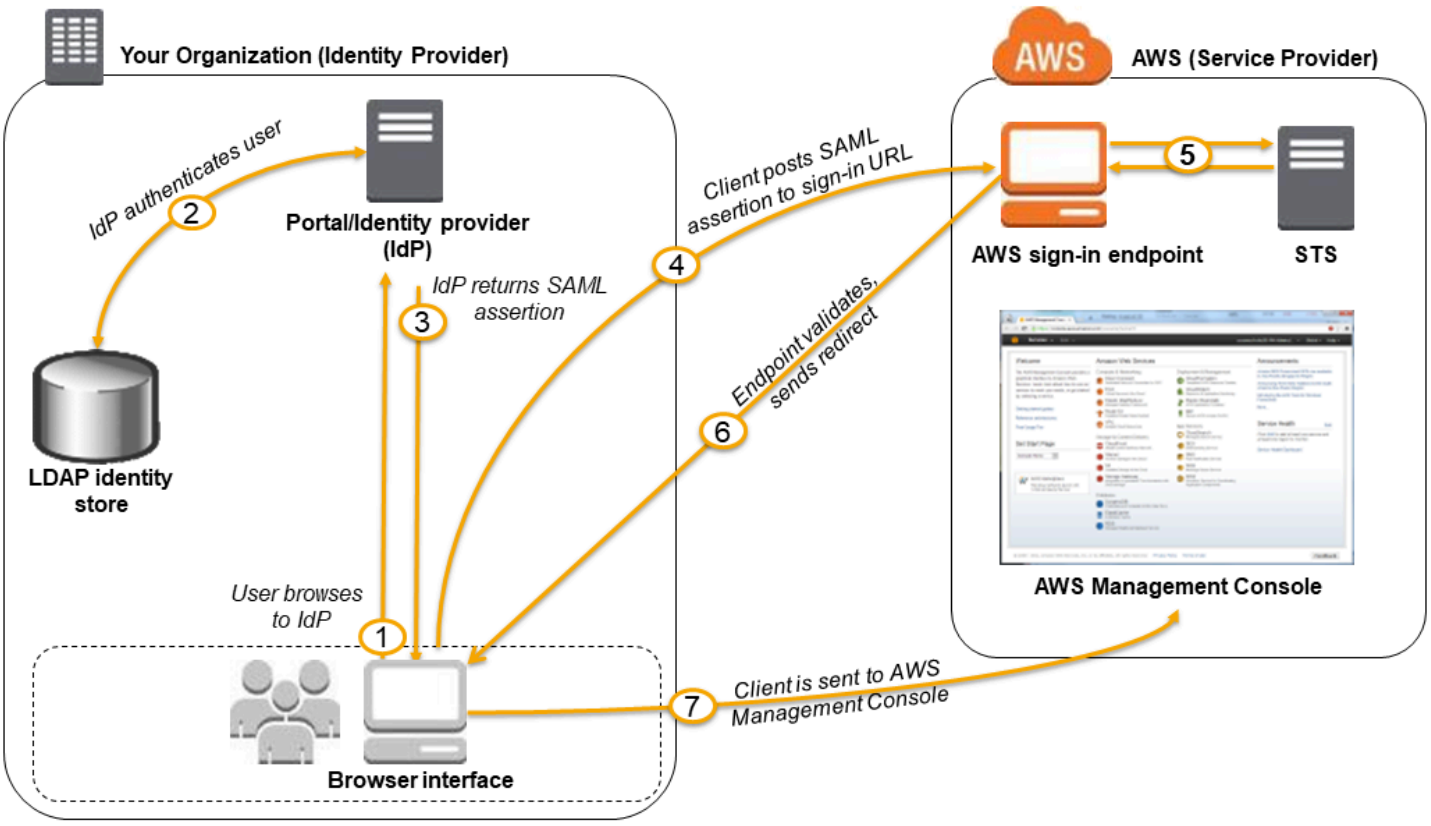
您可以使用角色來設定 SAML 2.0 相容身分提供者 (IdP) AWS ，並允許 SAML 聯合身分主體存取 AWS 管理主控台。該角色為使用者授予了在主控台中執行任務的許可。如果您想要為 SAML 聯合身分主體提供其他存取方式 AWS，請參閱下列其中一個主題：

- AWS CLI: 切換到 IAM 角色 (AWS CLI)
- Tools for Windows PowerShell：切換至 IAM 角色 (Tools for Windows PowerShell)
- AWS API： 切換到 IAM 角色 (AWS API)

概觀

下圖說明了啟用了 SAML 的單一登入的流程。

> **Note**
>
> 此 SAML 的特定用途與 中說明的更一般用途不同，SAML 2.0 聯合身分因為此工作流程 AWS 管理主控台 會代表使用者開啟 。這需要使用 AWS 登入端點，而不是直接呼叫 `AssumeRoleWithSAML` API。該端點將為使用者呼叫 API 並傳回將使用者的瀏覽器自動重新引導到 AWS 管理主控台的 URL。



此圖說明了下列步驟：

1. 使用者瀏覽到您的組織的入口網站，並選擇前往 AWS 管理主控台的選項。在您的組織中，入口網站通常是 IdP 的函數，可處理組織與 之間的信任交換 AWS。例如，在 Active Directory Federation Services 中，入口網站的 URL 是：`https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`
2. 該入口網站可驗證您的組織使用者的身分。
3. 入口網站會產生一個 SAML 身分驗證回應，其中包括辨識使用者身分的聲明以及使用者的相關屬性。您也可以配置 IdP 以包含一個名為 `SessionDuration` 的 SAML 聲明屬性，該屬性指定主控台工作階段的有效時間長度。您也可以設定 IdP，將屬性做為工作階段標籤傳遞。該入口網站將此回應傳送到用戶端瀏覽器。
4. 用戶端瀏覽器會重新導向至 AWS 單一登入端點，並發佈 SAML 聲明。
5. 端點代表使用者請求臨時安全性憑證，並建立一個使用這些憑證的主控台登入 URL。
6. AWS 會將登入 URL 傳回用戶端做為重新導向。
7. 該用戶端瀏覽器將重新引導到 AWS 管理主控台。如果 SAML 身分驗證回應包含映射到多個 IAM 角色的屬性，則將首先提示使用者選取要用於存取主控台的角色。

從使用者的角度來看，程序透明地發生：使用者從組織的內部入口網站開始 AWS 管理主控台，最後在 ，而不需要提供任何 AWS 登入資料。

有關如何配置此行為的概述以及指向詳細步驟的連結，請參閱以下章節。

將您的網路設定為 的 SAML 供應商 AWS

在您的組織的網路內部，將您的身分存放區 (如 Windows Active Directory) 設定為與 Windows Active Directory 聯合服務、Shibboleth 等以 SAML 為基礎的身分提供者 (IdP) 一起使用。使用 IdP 產生一個中繼資料文件，該文件將您的組織描述為身分提供者 (IdP) 並包含身分驗證索引鍵。您也可以設定組織的入口網站，將 的使用者請求路由 AWS 管理主控台 至 AWS SAML 端點，以使用 SAML 聲明進行身分驗證。配置您的 IdP 來產生 metadata.xml 文件之方法將根據您的 IdP 而定。請參閱您的 IdP 文件以獲得說明，或參閱 將第三方 SAML 解決方案供應商與 整合 AWS 以取得包含許多支援的 SAML 提供者之 Web 文件連結。

在 IAM 中建立 SAML 提供者

接著，您登入 AWS 管理主控台 並前往 IAM 主控台。您將在此建立一個新的 SAML 提供者，該提供者是 IAM 中包含您組織的身分提供者 (IdP) 的相關資訊的實體。在此過程中，您可以上傳在前一章節中由組織中的 IdP 軟體所產生的中繼資料文件。如需詳細資訊，請參閱在 IAM 中建立 SAML 身分提供者。

在 中 AWS 為 SAML 聯合主體設定許可

下一個步驟是建立 IAM 角色，以便建立 IAM 和您組織 IdP 之間的信任關係。基於聯合目的，這個角色必須將您的 IdP 識別為主體 (信任的實體)。此角色也會定義允許由組織 IdP 驗證的使用者在其中執行的操作 AWS。您可使用 IAM 主控台來建立此角色。當您建立信任政策，指出可以擔任角色的人員時，您可以指定您先前在 IAM 中建立的 SAML 提供者。您也可以指定一或多個 SAML 屬性，讓使用者在符合這些屬性的情況下，才能允許擔任角色。例如，您可以指定只允許其 SAML `eduPersonOrgDN` 值為 `ExampleOrg` 的使用者登入。角色精靈會自動新增一個測試 `saml:aud` 屬性的條件，確保僅以登入 AWS 管理主控台為目的來擔任該角色。

如果需要 SAML 加密，登入 URL 必須包含 AWS 指派給 SAML 提供者的唯一識別碼，您可以在「身分提供者詳細資訊」頁面上找到此識別碼。該角色的信任政策可能如下所示：

JSON

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

```
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:edupersonorgdn": "ExampleOrg",
                    "saml:aud": "https://region-code.signin.aws.amazon.com/saml/
acs/SAMLSP4SHN3UIS2D558H46"
                }
            }
        }
    ]
}
```

> **Note**
>
> 角色信任政策中使用的 SAML IDP 必須與角色所在的帳戶相同。

建議對 `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` 中的 `saml:aud` 屬性使用區域端點。如需可能 ####值清單，請參閱AWS 登入端點中的區域欄。

對於該角色中的許可政策，您可以像任何角色、使用者或群組一樣指定許可。例如，如果允許您組織中的使用者管理 Amazon EC2 執行個體，您可以在許可政策中明確允許 Amazon EC2 動作。您可以指派受管政策 (如 Amazon EC2 完整存取受管政策) 來執行此操作。

有關建立用於 SAML IdP 的角色的詳細資訊，請參閱 為 SAML 2.0 聯合身分建立角色 (主控台)。

完成組態並建立 SAML 聲明

安裝位於 `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`或 `https://signin.aws.amazon.com/static/saml-metadata.xmlsaml-metadata.xml`的檔案，以通知 AWS 做為服務供應商的 SAML IdP。如果需要 SAML 加密，可在 找到 檔案`https://region-code.signin.aws.amazon.com/static/saml/SAMLSP4SHN3UIS2D558H46/saml-metadata.xml`。

如需可能####值清單，請參閱AWS 登入端點中的區域欄。

安裝該檔案的方式取決於您的 IdP。部分提供者為您提供了輸入該 URL 的選項，此時，IdP 將為您取得並安裝該檔案。另一些提供者則要求您從該 URL 處下載檔案，然後將其做為本機檔案提供。請參閱您的 IdP 文件以獲得詳細資訊，或參閱 將第三方 SAML 解決方案供應商與 整合 AWS 以取得包含許多支援的 SAML 提供者之 Web 文件連結。

您也可設定一條資訊，說明您希望 IdP 在身分驗證回應期間將此資訊做為 SAML 屬性傳遞到 AWS 。大多數此資訊會在 中顯示 AWS 為條件內容索引鍵，您可以在政策中評估這些索引鍵。這些條件索引鍵會確保只有經過授權且位於正確內容中的使用者，才能獲得授予存取您 AWS 資源的許可。您可以指定限制可使用主控台時間的時段。您也可以指定使用者在重新整理其憑證前，可存取主控台的時間上限 (最多 12 小時)。如需詳細資訊，請參閱為身分驗證回應設定 SAML 聲明。

在瀏覽器中檢視 SAML 回應

以下程序介紹在排除與 SAML 2.0 相關的故障問題時，如何從您的瀏覽器中查看服務供應商的 SAML 回應。

對於所有瀏覽器，請前往可以重現問題的頁面。然後，針對適用的瀏覽器執行以下步驟：

主題

- Google Chrome
- Mozilla Firefox
- Apple Safari
- 如何處理 Base64 編碼的 SAML 回應

Google Chrome

在 Chrome 中查看 SAML 回應

這些步驟已使用 Google Chrome 版本 106.0.5249.103 (正式版本) (arm64) 進行了測試。如果您使用其他版本，則可能需要根據情況調整步驟。

1. 按下 F12 來啟動 Developer Tools (開發人員工具) 主控台。
2. 選取 Network (網路) 標籤，然後選取 Developer Tools (開發人員工具) 視窗左上角的 Preserve log (保留記錄)。
3. 重現問題。
4. （選用）如果開發人員工具網路日誌窗格中看不到方法欄，請開啟任何資料欄標籤的內容（按一下滑鼠右鍵）選單，然後選擇新增資料欄的方法。

5. 在 Developer Tools (開發人員工具) 的 Network (網路) 記錄窗格中尋找 SAML Post (SAML 文章)。選擇該列，然後檢視頂端的 Payload (承載) 標籤。尋找包含加密請求的 SAMLResponse 元素。關聯值為 Base64 編碼的回應。

Mozilla Firefox

在 Firefox 中查看 SAML 回應

此程序已在 Mozilla Firefox 版本 105.0.3 (64 位元) 上進行了測試。如果您使用其他版本，則可能需要根據情況調整步驟。

1. 按下 F12 來啟動 Web Developer Tools (Web 開發人員工具) 主控台。
2. 選取 Network (網路) 標籤。
3. 在 Web Developer Tools (Web 開發人員工具) 視窗右上角選擇選項 (小齒輪圖示)。選取 Persist logs (持續記錄)。
4. 重現問題。
5. （選用）如果方法欄不會顯示在 Web 開發人員工具網路日誌窗格中，請開啟任何資料欄標籤的內容（按一下滑鼠右鍵）選單，然後選擇新增資料欄的方法。
6. 尋找表格中的 POST SAML。選取該列，然後檢視 Request 標籤並尋找 SamlResponse 元素。關聯值為 Base64 編碼的回應。

Apple Safari

在 Safari 中查看 SAML 回應

這些步驟已使用 Apple Safari 版本 16.0 (17614.1.25.9.10, 17614) 進行了測試。如果您使用其他版本，則可能需要根據情況調整步驟。

1. 在 Safari 中啟用 Web Inspector。開啟 Preferences (偏好設定) 視窗，選取 Advanced (進階) 標籤，然後選取 Show Develop menu in the menu bar (在選單列顯示開發選單)。
2. 現在您可以開啟 Web Inspector。在選單列中選擇 Develop (開發)，然後選取 Show Web Inspector (顯示 Web Inspector)。
3. 選取 Network (網路) 標籤。
4. 在 Web Inspector 視窗右上角選擇選項 (包含三條水平線的小圓形圖示)。選取 Preserve logs (保留記錄)。

5. （選用）如果 Web Inspector Network 日誌窗格中看不到方法欄，請開啟任何資料欄標籤的內容（按一下滑鼠右鍵）選單，然後選擇新增資料欄的方法。
6. 重現問題。
7. 尋找表格中的 POST SAML。選擇該列，然後檢視 Headers (標題) 標籤。
8. 尋找包含加密請求的 SAMLResponse 元素。向下捲動到找到含有名稱 `SAMLResponse` 的 `Request Data`。關聯值為 Base64 編碼的回應。

如何處理 Base64 編碼的 SAML 回應

在瀏覽器中找到 Base64-encoded的 SAML 回應元素之後，請複製它並使用您最愛的 Base-64 解碼工具來擷取 XML 標記的回應。

> **安全提示**
>
> 由於您查看的 SAML 回應資料可能包含敏感安全資料，我們建議您不要使用線上 base64 解碼器。而是使用安裝在本機電腦上，不會透過網路傳送 SAML 資料的工具。

適用於 Windows 系統的內建選項 (PowerShell)：

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

適用於 MacOS 與 Linux 系統的內建選項：

```
$ echo "base64encodedtext" | base64 --decode
```

檢閱解碼檔案中的值

檢閱已解碼 SAML 回應檔案中的值。

- 確認 saml:NameID 屬性的值符合已驗證使用者的使用者名稱。
- 檢閱 https://aws.amazon.com/SAML/Attributes/Role 的值。ARN 和 SAML 提供者區分大小寫，且 ARN 必須符合您帳戶中的資源。
- 檢閱 https://aws.amazon.com/SAML/Attributes/RoleSessionName 的值。該值必須符合宣告規則中的值。

- 如果設定電子郵件地址或帳戶名稱的屬性值，請確定這些值正確無誤。這些值必須對應至已驗證使用者的電子郵件地址或帳戶名稱。

檢查錯誤並確認組態

檢查值是否包含錯誤，並確認下列組態正確無誤。

- 宣告規則符合必要元素，且所有 ARN 皆正確。如需詳細資訊，請參閱使用依賴方信任設定您的 SAML 2.0 IdP 並新增宣告。
- 您已將最新的中繼資料檔案從 IdP 上傳至 SAML 供應商 AWS 中的 。如需詳細資訊，請參閱啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。
- 您已正確設定 IAM 角色的信任政策。如需詳細資訊，請參閱擔任角色的方法。

將 AWS 身分聯合到外部服務

IAM 傳出聯合身分可讓您的 AWS 工作負載安全地存取外部服務，而無需存放長期憑證。您的 AWS 工作負載可以呼叫 GetWebIdentityToken API，向 AWS Security Token Service () 請求短期 JSON Web Token JWTs AWS STS)。這些字符以密碼編譯方式簽署，可公開驗證，並包含一組完整的宣告，可向外部服務宣告 AWS 工作負載的身分。您可以搭配各種第三方雲端提供者、SaaS 平台和自我託管應用程式使用這些字符。外部服務會使用在已知端點上發佈 AWS的驗證金鑰來驗證字符的真偽，並使用字符中的資訊進行身分驗證和授權決策。

傳出聯合身分不需要在應用程式程式碼或環境變數中存放長期登入資料，例如 API 金鑰或密碼，可改善您的安全狀態。您可以使用 IAM 政策控制權杖產生和強制執行權杖屬性的存取，例如簽署演算法、允許的受眾和持續時間。所有字符請求都會登入 AWS ，為安全性監控和合規報告提供完整的稽核線索。您也可以使用顯示為自訂宣告的標籤來自訂權杖，讓外部服務實作精細的屬性型存取控制。

如需設定傳出聯合的指示，請參閱 開始使用傳出聯合身分。如需字符內容的詳細資訊，請參閱 了解字符宣告。如需控制哪些工作負載可以請求字符的指示，請參閱 使用 IAM 政策控制存取。

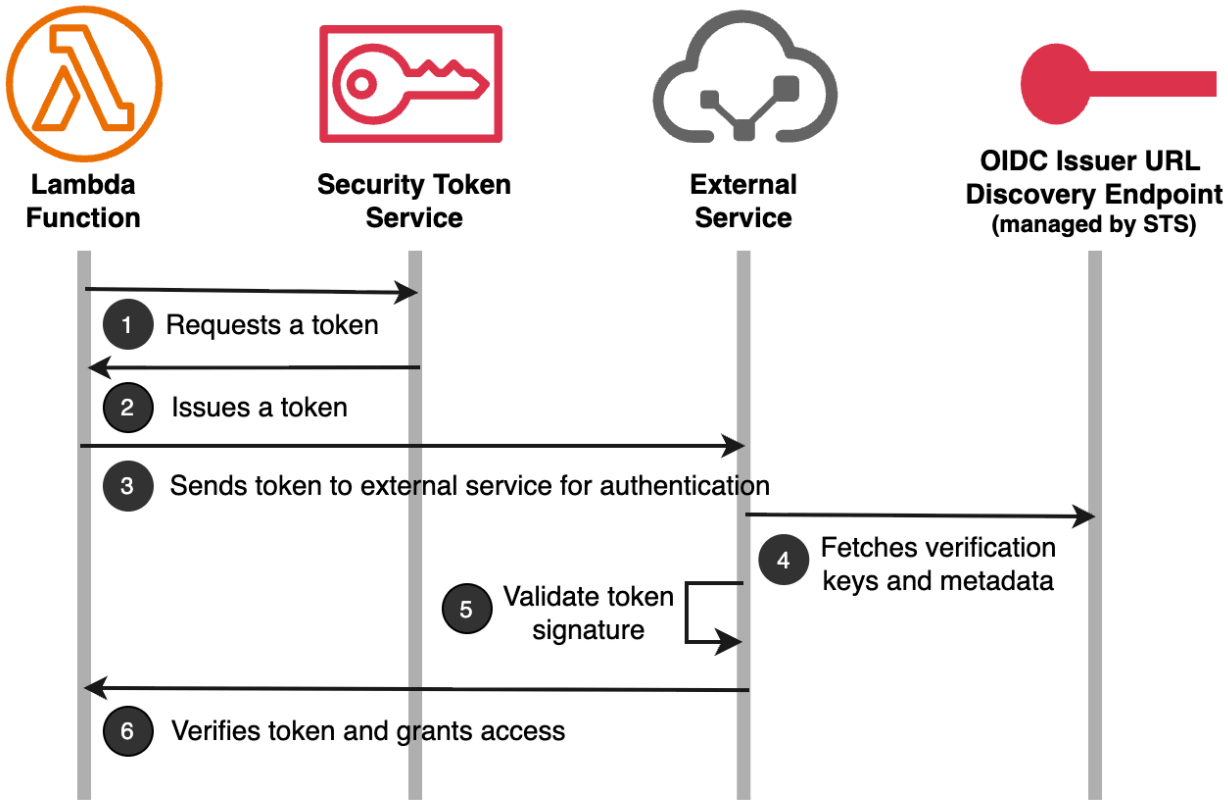
常用案例

使用傳出聯合身分，您的 AWS 工作負載可以安全地：

- 存取外部雲端提供者中的資源和服務。例如，Lambda 函數處理資料可以將結果寫入外部雲端提供者的儲存服務，或查詢其資料庫。
- 與外部software-as-a-service(SaaS) 供應商整合，用於分析、資料處理、監控等。例如，您的 Lambda 函數可以將指標傳送至可觀測性平台。

- 使用託管在 AWS外部雲端提供者或內部部署資料中心上的應用程式進行驗證，以啟用安全的混合和多雲端架構。例如，您的 AWS 工作負載可以與內部部署 Kubernetes 叢集中執行的容器化應用程式互動。

運作方式



1. Lambda 函數會呼叫 GetWebIdentityToken API，向 AWS Security Token Service () 請求 JSON Web Token (JWT)AWS STS。
2. AWS STS 會驗證請求，並將已簽署的 JWT 傳回 Lambda 函數。
3. Lambda 函數會將 JWT 傳送至外部服務。
4. 外部服務會從權杖擷取發行者 URL、驗證是否符合已知信任的發行者，以及從 OIDC 探索端點擷取 AWS驗證金鑰和中繼資料。
5. 外部服務使用驗證金鑰來驗證字符的簽章，並驗證過期時間、主旨和對象等宣告。
6. 成功驗證後，外部服務會授予 Lambda 函數的存取權。

開始使用傳出聯合身分

本指南說明如何啟用 AWS 帳戶的傳出聯合身分，並取得您的第一個 JSON Web Token (JWT)（使用 GetWebIdentityToken API)。您將啟用此功能、與外部服務建立信任關係、設定 IAM 許可，以及使用 AWS CLI 或 AWS SDK for Python (Boto3) 請求權杖。

先決條件

開始之前，請確定您已：

- 已安裝最新版本的 AWS CLI 或 Python 3.8（或更新版本）和 Boto3（適用於 AWS SDK 範例）
- 外部服務帳戶，您可以在其中設定信任關係（例如外部雲端提供者、SaaS 提供者或測試應用程式）

> **Note**
>
> - 此 `GetWebIdentityToken` API 無法在 STS 全域端點上使用。
> - `GetWebIdentityToken` API 產生的 JSON Web 權杖 (JWTs) 無法用於 AWS（透過 AssumeRoleWithWebIdentity API) 的 OpenID Connect (OIDC) 聯合。

為您的帳戶啟用傳出聯合身分

您必須先啟用傳出聯合身分，才能請求字符。您可以使用 AWS 管理主控台啟用此功能，或使用 EnableOutboundWebIdentityFederation API 以程式設計方式啟用此功能。

使用 AWS CLI

```
aws iam enable-outbound-web-identity-federation
```

使用適用於 Python 的 AWS SDK

```
import boto3

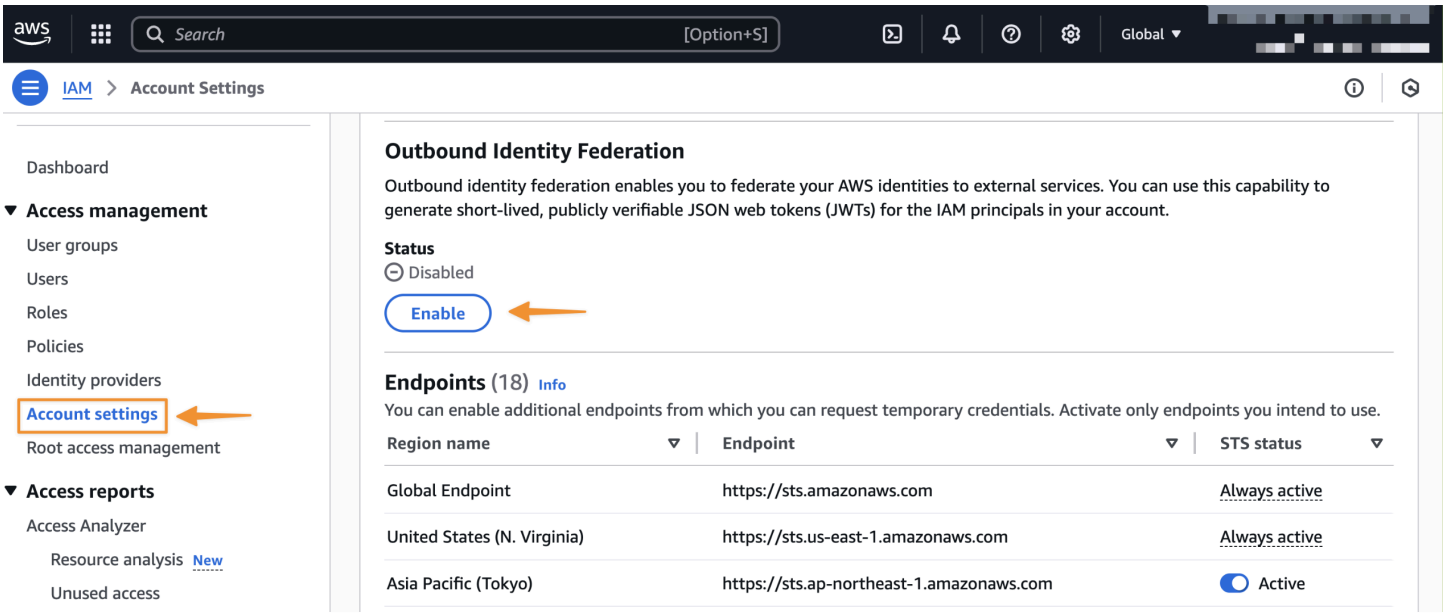
# Create IAM client
iam_client = boto3.client('iam')

# Enable outbound identity federation
response = iam_client.enable_outbound_web_identity_federation()
```

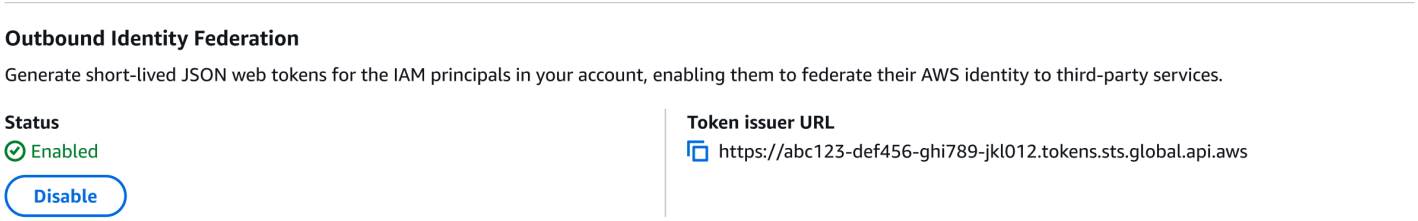
```
print(f"Feature enabled. Issuer URL: {response['IssuerUrl']}")
print(f"Status: {response['Status']}")
```

使用 AWS 主控台

導覽至 IAM，然後在左側導覽功能表的存取管理區段下選取帳戶設定



啟用此功能後，請記下您的帳戶特定發行者 URL。在外部服務中設定信任關係時，您將使用此 URL。您也可以視需要使用 GetOutboundWebIdentityFederationInfo API 擷取此發行者 URL。



在外部服務中建立信任關係

設定外部服務以信任和接受 AWS 帳戶發行的字符。具體步驟因服務而異，但通常涉及：

- 將 AWS 您的帳戶發行者 URL 註冊為信任的身分提供者
- 設定要驗證的宣告（對象、主體模式）
- 將字符宣告映射至外部服務中的許可

如需詳細的組態指示，請參閱外部服務文件。

設定 IAM 許可

建立 IAM 政策，授予呼叫 GetWebIdentityToken API 的許可，並將政策連接至需要產生字符的 IAM 角色。

此範例政策授予具有特定限制的權杖產生存取權。它只允許以「https://api.example.com」為對象請求權杖，並強制執行權杖生命週期上限為 5 分鐘 (300 秒）。如需可用來強制執行權杖屬性的條件金鑰the section called “IAM 條件鍵”清單，請參閱 。

IAM 政策範例

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "sts:IdentityTokenAudience": "https://api.example.com"
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                }
            }
        }
    ]
}
```

請求您的第一個 JSON Web 權杖 (JWT)

您可以使用 GetWebIdentityToken API 請求 JSON Web 權杖。您可以在呼叫 API 時指定下列參數：

- 對象 （必要）：字符的預期收件人。此值會在 JWT 中填入「aud」宣告。外部服務會驗證此宣告，以確保權杖適用於它們。
- SigningAlgorithm （必要）：用於簽署字符的密碼編譯演算法。有效值為 ES384 和 RS256。使用 ES384 獲得最佳安全性和效能，或 RS256 與不支援 ECDSA 的系統具有更廣泛的相容性。
- DurationSeconds （選用）：字符生命週期，以秒為單位。有效值的範圍為 60 到 3600。預設值為 300 (5 分鐘)。我們建議縮短字符生命週期，以提高安全性。

- 標籤（選用）：要做為自訂宣告包含在字符中的鍵值對清單。外部服務可以使用這些宣告進行精細授權。

API 會傳回下列欄位：

- IdentityToken：已簽署的 JWT 做為 base64url 編碼字串。在外部服務的請求中包含此字符。
- 過期：字符過期時的 UTC 時間戳記。

使用 AWS CLI

```
aws sts get-web-identity-token \
    --audience "https://api.example.com" \
    --signing-algorithm ES384 \
    --duration-seconds 300 \
    --tags Key=team,Value=data-engineering \
           Key=environment,Value=production \
           Key=cost-center,Value=analytics
```

使用適用於 Python 的 AWS SDK

```
import boto3

sts_client = boto3.client('sts')

response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
    DurationSeconds=300,
    SigningAlgorithm='RS256',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'environment', 'Value': 'production'},
        {'Key': 'cost-center', 'Value': 'analytics'}
    ]
)

token = response['WebIdentityToken']
```

您也可以解碼 JWT，使用 PyJWT、Python-jose for Python、Nimbus JOSE+JWT for Java 或 jwt.io 等偵錯工具等標準 JWT 程式庫來檢查其內容。如需字符中包含之宣告the section called “了解字符宣告”的詳細資訊，請參閱 。

搭配外部服務使用字符

收到字符後，將其包含在外部服務的請求中。方法因服務而異，但大多數服務接受授權標頭中的字符。外部服務應實作權杖驗證邏輯，從發行者的已知端點擷取 JWKS 金鑰、驗證權杖的簽章，並在授予 AWS 工作負載存取權之前驗證基本宣告。

從 OpenID Connect (OIDC) 端點擷取驗證金鑰和中繼資料

您 AWS 帳戶的唯一發行者 URL 託管 OpenID Connect (OIDC) 探索端點，其中包含驗證金鑰和字符驗證所需的中繼資料。

OIDC 探索端點 URL 包含一些提供者用來驗證字符的中繼資料。可在以下位置取得：

```
{issuer_url}/.well-known/openid-configuration
```

JWKS (JSON Web 金鑰集） 端點包含用於驗證字符簽章的金鑰。可在以下位置取得：

```
{issuer_url}/.well-known/jwks.json
```

使用 curl 擷取 JWKS

```
curl https://{issuer_url}/.well-known/jwks.json
```

回應：

```
{
  "keys": [
    {
      "kty": "EC",
      "use": "sig",
      "kid": "key-id-1",
      "alg": "ES384",
      "crv": "P-384",
      "x": "base64-encoded-x-coordinate",
      "y": "base64-encoded-y-coordinate"
    },
    {
      "kty": "RSA",
      "use": "sig",
      "kid": "key-id-2",
      "n": "base64-encoded-modulus",
```

```
      "e": "AQAB"
    }
  ]
}
```

使用適用於 Python 的 AWS SDK

```
import requests

# Fetch Openid Configuration
open_id_config_response = requests.get("https://{issuer_url}/.well-known/openid-
configuration")
open_id_config = open_id_config_response.json()

# Fetch JWKS
jwks_response = requests.get("https://{issuer_url}/.well-known/jwks.json")
jwks = jwks_response.json()
```

我們建議您快取這些金鑰，以避免在每次權杖驗證時擷取它們。

基本宣告驗證

- 主旨 (sub)：確認主旨宣告包含預期的 IAM 主體 ARN 模式。
- 過期 （過期）：請確定字符尚未過期。JWT 程式庫通常會自動處理。
- 對象 (aud)：確認對象符合您預期的值。這可防止其他 服務的字符與您的 搭配使用。
- 發行者 (iss)：確認發行者符合您信任 AWS 的帳戶 (s)。維護信任的發行者 URLs清單。

您應該盡可能驗證額外的 AWS特定宣告，以在外部服務中實作精細存取控制。例如，驗證 org_id 宣告以限制對 AWS 組織中 IAM 主體的存取、檢查 principal_tags 以強制執行屬性型存取控制 （例如僅允許生產環境或特定團隊），或驗證工作階段內容宣告，例如 lambda_source_function_arn 或 ec2_instance_source_vpc，以根據運算資源限制存取。如需字符中包含的完整聲明清單，請參閱了解字符宣告。

了解字符宣告

當您呼叫 GetWebIdentityToken API 時， AWS Security Token Service 會傳回已簽署的 JSON Web Token (JWT)，其中包含一組代表 IAM 主體身分的宣告。這些字符符合 RFC 7519。了解這些字符的結構和內容可協助您實作安全身分驗證流程、在外部服務中設定適當的宣告驗證，以及有效地使用自訂宣告進行精細存取控制。

JWT 包含標準 OpenID Connect (OIDC) 宣告，例如主體 ("sub")、對象 ("aud")、發行者 ("iss")，以促進不同外部服務的互通性。 會在適用時將 AWS 身分特定宣告 （ 例如 AWS 帳戶 ID 和主體標籤 ） 和工作階段內容宣告 （ 例如 EC2 執行個體 ARNs) AWS STS 填入權杖。您也可以將自訂宣告做為請求標籤傳遞至 GetWebIdentityToken API，藉此將自訂宣告新增至權杖。 AWS 身分特定的宣告、工作階段內容宣告和自訂宣告會巢狀在權杖中的「https://sts.amazonaws.com/」命名空間下。

如需字符中包含的宣告清單，請參閱以下範例字符。請注意，所有這些宣告可能不會同時出現在字符中。

```
{
  "iss": "https://abc123-def456-ghi789-jkl012.tokens.sts.global.api.aws",
  "aud": "https://api.example.com",
  "sub": "arn:aws:iam::123456789012:role/DataProcessingRole",
  "iat": 1700000000,
  "exp": 1700000900,
  "jti": "xyz123-def456-ghi789-jkl012",
  "https://sts.amazonaws.com/": {
    "aws_account": "123456789012",
    "source_region": "us-east-1",
    "org_id": "o-abc1234567",
    "ou_path": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/",
    "principal_tags": {
      "environment": "production",
      "team": "data-engineering",
      "cost-center": "engineering"
    },
    "lambda_source_function_arn": "arn:aws:lambda:us-
east-1:123456789012:function:process-data",
    "request_tags": {
        "job-id": "job-2024-001",
        "priority": "high",
        "data-classification": "sensitive"
    }
  }
}
```

標準宣告

字符中存在的標準 OIDC 宣告有助於與各種外部服務互通。這些宣告可以使用大多數 JWT 程式庫進行驗證。

取得	名稱	描述	範例值
iss	發行者	您的帳戶特定發行者 URL。外部服務會驗證此宣告，以確保其符合其信任的發行者。	https://abc123-def456-ghi789-jkl012.tokens.sts.global.api.aws
aud	目標對象	GetWebIdentityToken 請求中指定字符的預期收件人。	https://api.example.com
sub	主旨	請求權杖之 IAM 主體的 ARN。	arn：aws：iam：：123456789012：role/DataProcessingRole
iat	在 發行	識別發出 JWT 時間的 NumericDate 值。	1700000000
exp	Expiration	識別過期時間的 NumericDate 值，在此時間之後，JWT 不得接受處理。	1700000900
jti	JWT ID	此字符執行個體的唯一識別符。	xyz123-def456-ghi789-jkl012

自訂宣告

除了標準 OIDC 宣告之外， 還會在適用時 AWS STS 新增有關身分和工作階段內容的宣告。您也可以將自己的宣告做為請求標籤傳遞至權杖。自訂宣告會巢狀在 https://sts.amazonaws.com/ 命名空間下。

AWS 身分宣告

這些宣告提供有關 AWS 您的帳戶、組織結構和 IAM 主體的詳細資訊。

取得	說明	條件索引鍵的映射	範例值
aws_account	AWS 您的帳戶 ID	aws：PrincipalAccount	123456789012

取得	說明	條件索引鍵的映射	範例值
source_region	請求字符 AWS 的區域	aws:RequestedRegion	us-east-1
org_id	您的 AWS Organizations ID（如果您的帳戶是組織的一部分）	aws:PrincipalOrgID	o-abc1234567
ou_path	您的組織單位路徑（如適用）	aws:PrincipalOrgPaths	o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
principal_tags	連接至 IAM 主體或擔任角色工作階段的標籤。當請求的 IAM 主體同時具有委託人標籤和工作階段標籤時，請求權杖時，工作階段標籤會出現在 JWT 中。	aws：PrincipalTag/<tag-key>	{"環境"： "生產"、"團隊"： "資料工程"、"成本中心"："工程"}

工作階段內容宣告

這些宣告提供有關字符請求發起的運算環境和工作階段的資訊。根據請求主體的工作階段內容，當適用時， AWS STS 會自動包含這些宣告。

取得	說明	條件索引鍵的映射	範例值
original_session_exp	原始角色工作階段憑證何時過期（適用於擔任的角色）	N/A	2024-01-15T10：00：00Z
federated_provider	聯合工作階段的身分提供者名稱	aws:FederatedProvider	arn：aws：iam：：111122223333：oidc-provider/your_oidc_provider

取得	說明	條件索引鍵的映射	範例值
identity_store_user_id	IAM Identity Center 使用者 ID	identitystore：UserId	user-abc123def456
identity_store_arn	Identity Center 身分存放區的 ARN	identitystore：IdentityStoreArn	arn：aws：identitystore：：123456789012：identitystore/d-abc1234567
ec2_source_instance_arn	請求 EC2 執行個體的 ARN	ec2：SourceInstanceArn	arn：aws：ec2：us-east-1：123456789012：instance/i-abc123def456
ec2_instance_source_vpc	交付 EC2 角色登入資料的 VPC ID	aws：Ec2InstanceSourceVpc	vpc-abc123def456
ec2_instance_source_private_ipv4	EC2 執行個體的私有 IPv4 地址	aws：Ec2InstanceSourcePrivateIPv4	10.0.1.25
ec2_role_delivery	執行個體中繼資料服務版本	ec2：RoleDelivery	2
source_identity	委託人設定的來源身分	aws：SourceIdentity	admin-user
lambda_source_function_arn	呼叫 Lambda 函數的 ARN	lambda：SourceFunctionArn	arn：aws：lambda：us-east-1：123456789012：function：my-function
glue_credential_issuing_service	AWS Glue 任務的 Glue 服務識別符	glue：CredentialIssuingService	glue.amazonaws.com

請求標籤

您可以在 GetWebIdentityToken API 請求中指定標籤，將自訂宣告新增至權杖。這些宣告會出現在字符的 request_tags 欄位下，可讓您傳遞外部服務可用於精細授權決策的特定資訊。每個請求最多可以指定 50 個標籤。

範例請求：

```
response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
    SigningAlgorithm='ES384',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'cost-center', 'Value': 'analytics'},
        {'Key': 'environment', 'Value': 'production'}
    ]
)
```

權杖中產生的宣告：

```
{
  "request_tags": {
    "team": "data-engineering",
    "cost-center": "analytics",
    "environment": "production"
  }
}
```

使用 IAM 政策控制存取

IAM 提供多種政策類型，以控制對傳出聯合身分功能的存取。您可以使用身分型政策來控制哪些 IAM 主體可以請求權杖，並強制執行特定權杖屬性，例如對象、生命週期和簽署演算法。服務控制政策 SCPs) 可讓您對 Organizations 中的所有帳戶強制執行 AWS 整個組織的字符產生限制。資源控制政策 RCPs) 控制資源層級的存取。您也可以使用 VPC 端點政策來限制哪些主體可以透過 VPC 端點存取 API，將 AWS STS GetWebIdentityToken網路層級控制新增至您的安全狀態。本節說明如何使用這些政策類型和條件索引鍵實作精細存取控制。

若要請求身分字符，IAM 主體必須具有 sts:GetWebIdentityToken許可。透過連接至 IAM 使用者或角色的身分政策授予此許可。若要允許標籤（金鑰、值對）傳遞給 GetWebIdentityToken 呼叫，IAM 主體必須具有 sts:TagGetWebIdentityToken許可。

- 使用 sts：IdentityTokenAudience 條件金鑰來限制哪些外部服務可以接收字符。
- 使用 sts：DurationSeconds 條件金鑰來強制執行字符生命週期上限。
- 使用 sts：SigningAlgorithm 條件金鑰來要求特定的密碼編譯演算法。
- 使用 aws：RequestTag 條件金鑰，將請求中傳遞的標籤鍵值對與您在政策中指定的標籤對進行比較。
- 使用 aws：TagKeys 條件索引鍵，將請求中的標籤索引鍵與您在政策中指定的索引鍵進行比較。

請參閱 IAM 和 AWS STS 條件金鑰，進一步了解 IAM 政策中可用的條件金鑰。

此範例身分政策結合了多個條件索引鍵：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowTokenGenerationWithRestrictions",
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api1.example.com",
                        "https://api2.example.com"
                    ]
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                },
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

最佳實務

遵循這些建議，將 AWS 您的身分安全地聯合到外部 服務。

- 使用較短的字符生命週期：請求具有最短生命週期的字符，以符合您的操作需求。
- 使用 IAM 政策實作最低權限存取並限制權杖屬性：僅將 `sts:GetWebIdentityToken`許可授予需要它的 IAM 主體。視需要使用條件索引鍵來指定簽署演算法、允許的權杖對象和權杖生命週期上限。
- 驗證外部服務中的宣告：為了安全起見，請一律驗證相關宣告，例如主旨 ("sub")、受眾 ("aud") 等，以確保符合您預期的值。盡可能驗證自訂宣告，以在外部服務中啟用精細的授權決策。

IAM 中的暫時安全憑證

您可以使用 AWS Security Token Service (AWS STS) 來建立和提供信任的使用者暫時安全登入資料，以控制對 AWS 資源的存取。暫時性安全憑證的作用幾乎與長期存取金鑰憑證完全相同，而其差異如下：

- 暫時安全憑證是「短期」的，如其名稱所暗示。其可設定為在任何地方持續幾分鐘到幾小時不等。登入資料過期後， AWS 就無法再辨識這些登入資料，也無法允許從使用它們提出的 API 請求進行任何類型的存取。
- 暫時性安全憑證不會與使用者一起儲存，但會動態產生並在請求時提供給使用者。當暫時性安全憑證到期時 (或者即使在此之前)，使用者可以請求新的憑證，只要使用者的請求仍具有可這麼做的許可。

因此，相較於長期憑證，暫時性憑證具有下列優點：

- 您不需要向應用程式分發或嵌入長期 AWS 安全登入資料。
- 您可以為使用者提供 AWS 資源的存取權，而不必為其定義 AWS 身分。臨時憑證是角色和聯合身分的基礎。
- 臨時安全憑證的存留期有限，因此當不再需要時，您不需要更新它們或明確予以撤銷。暫時性安全憑證到期之後，就無法重複使用。您可以指定憑證的有效期，達到最長限制。

AWS STS 和 AWS 區域

AWS STS所產生暫時性安全憑證。根據預設， AWS STS 是位於 的單一端點的全域服務`https://sts.amazonaws.com`。不過，您也可以選擇對任何其他支援區域中的端點進行 AWS STS API 呼叫。這可透過從地理位置較靠近您的區域的伺服器傳送請求，來降低延遲發生機率 (伺服器延遲)。無論您的憑證來自哪個區域，都能全域使用。如需詳細資訊，請參閱 在 AWS STS 中管理 AWS 區域。

暫時性憑證的常見案例

暫時性憑證在涉及聯合身分、委派、跨帳戶存取和 IAM 角色的案例中非常有用。

聯合身分

您可以在 外部的外部系統中管理您的使用者身分， AWS 並授予從這些系統登入的使用者執行 AWS 任務和存取 資源 AWS 的存取權。IAM 支援兩種類型的聯合身分。在這兩種情況下，身分都會存放在 外部 AWS。差別是外部系統所在地點是您的資料中心或外部第三方的 Web 上。若要比較聯合身分的臨時安全憑證的功能，請參閱比較 AWS STS 登入資料。

如需關於外部身分提供者的詳細資訊，請參閱身分提供者和 的聯合身分 AWS。

- OpenID Connect (OIDC) 聯合 – 您可以讓使用者使用知名的第三方身分提供者登入，例如 Login with Amazon、Facebook、Google 或任何與 OIDC 相容的提供者。對於您的行動或 Web 應用程式，您不需要建立自訂登入碼或管理自己的使用者身分。使用 OIDC 聯合可協助您保持 AWS 帳戶 安全，因為您不必將 IAM 使用者存取金鑰等長期安全登入資料與您的應用程式一起分發。如需詳細資訊，請參閱OIDC 聯合身分。

  AWS STS OIDC 聯合支援 Login with Amazon、Facebook、Google 和任何與 OpenID Connect (OIDC) 相容的身分提供者。

  > **Note**
  >
  > 對於行動應用程式，我們建議您使用 Amazon Cognito。您可以使用此服務搭配 AWS SDKs 進行行動開發，為使用者建立唯一身分，並對其進行驗證，以安全地存取您的 AWS 資源。Amazon Cognito 支援與 相同的身分提供者 AWS STS，也支援未經驗證的 （訪客） 存取，並可讓您在使用者登入時遷移使用者資料。Amazon Cognito 還提供 API 操作以同步使用者資料，如此可在使用者於不同裝置之間移動時進行保留。如需詳細資訊，請參閱 Amplify 文件 中的 Amplify 身分驗證。

- SAML 聯合 – 您可以對組織網路中的使用者進行身分驗證，然後為這些使用者提供 的存取權， AWS 而無需為其建立新的 AWS 身分，並要求他們使用不同的登入憑證登入。這稱為臨時存取的單一登入方法。 AWS STS 支援開放標準，例如安全性聲明標記語言 (SAML) 2.0，您可以使用 Microsoft AD FS 來利用您的 Microsoft Active Directory。您也可以使用 SAML 2.0 來管理您自己的聯合使用者身分的解決方案。如需詳細資訊，請參閱SAML 2.0 聯合身分。
  - 自訂聯合代理程式 – 您可以使用組織的身分驗證系統來授予 AWS 資源的存取權。如需範例案例，請參閱啟用 AWS 主控台的自訂身分代理程式存取。

- 使用 SAML 2.0 的聯合 – 您可以使用您組織的身分驗證系統和 SAML，來授予 AWS 資源的存取。如需詳細資訊和範例案例，請參閱 SAML 2.0 聯合身分。

跨帳戶存取的角色

許多組織維護多個 AWS 帳戶。您可以使用角色和跨帳戶存取，來定義一個帳戶中的使用者身分，並使用那些身分來存取屬於您組織的其他帳戶中的 AWS 資源。這就是所謂暫時存取的「委派」方法。如需有關建立跨帳戶角色的詳細資訊，請參閱 建立角色以將許可授予 IAM 使用者。若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？。

Amazon EC2 的角色

如果您在 Amazon EC2 執行個體上執行應用程式，並且那些應用程式需要存取 AWS 資源，則您可以在您啟動它們時提供暫時安全憑證。這些暫時性安全憑證可供在執行個體上執行的所有應用程式使用，因此您不需要在執行個體上儲存任何長期憑證。如需詳細資訊，請參閱使用 IAM 角色為在 Amazon EC2 執行個體上執行的應用程式授予許可。

若要進一步了解 IAM Amazon EC2 角色憑證，請參閱 Amazon Elastic Compute Cloud User Guide 中的 IAM roles for Amazon EC2。

AWS 其他服務

您可以使用臨時安全登入資料來存取大多數 AWS 服務。如需接受暫時性安全性憑證的服務的清單，請參閱AWS 使用 IAM 的 服務。

使用臨時憑證的應用程式範例

您可以使用 AWS Security Token Service (AWS STS) 來建立信任的使用者，並提供暫時安全登入資料，以控制對 AWS 資源的存取。如需 的詳細資訊 AWS STS，請參閱 IAM 中的暫時安全憑證。若要了解如何使用 AWS STS 來管理臨時安全登入資料，您可以下載下列範例應用程式，以實作完整的範例案例：

- AWS 使用 Windows Active Directory、ADFS 和 SAML 2.0 將聯合啟用至 。示範如何使用企業聯合將存取權委派給 AWS 使用 Windows Active Directory (AD)、Active Directory Federation Services (ADFS) 2.0 和 SAML （ 安全性聲明標記語言 ） 2.0 的 。
- 啟用 AWS 主控台的自訂身分代理程式存取。示範如何建立啟用單一登入 (SSO) 的自訂聯合代理程式，使現有 Active Directory 使用者能夠登入 AWS 管理主控台。

- 如何使用 Shibboleth 進行 的單一登入 AWS 管理主控台。示範如何使用 Shibboleth 與 SAML 來提供使用者存取 AWS 管理主控台的單一登入 (SSO) 許可。

OIDC 聯合身分的範例

下列範例應用程式說明如何搭配 Login with Amazon、Amazon Cognito、Facebook 或 Google 等供應商使用 OIDC 聯合。您可以從這些供應商交易身分驗證，以取得臨時 AWS 安全登入資料以存取 AWS 服務。

- Amazon Cognito 教學課程 – 我們建議您使用 Amazon Cognito 搭配 AWS SDKs進行行動開發。Amazon Cognito 是進行行動應用程式身分管理的最便捷的方式，並提供了同步和跨裝置身分驗證等連接功能。如需有關 Amazon Cognito 的詳細資訊，請參閱 Amplify 文件 中的 Amplify 身分驗證。

暫時性安全憑證的其他資源。

以下案例和應用程式可以指導您使用暫時性安全憑證：

- 如何將 AWS STS SourceIdentity 與您的身分提供者整合。此文章說明如何在使用 AWS STS `SourceIdentity` Okta、Ping 或 OneLogin 做為 IdP 時設定 屬性。
- OIDC 聯合身分。本節討論如何在使用 OIDC 聯合身分和 `AssumeRoleWithWebIdentity` API 時設定 IAM 角色。
- 透過 MFA 實現安全的 API 存取. 本主題說明如何使用角色來要求多重要素驗證 (MFA) 以保護您的帳戶中的敏感 API 動作。

如需 中政策和許可的詳細資訊， AWS 請參閱下列主題：

- AWS 資源的存取管理
- 政策評估邏輯.
- Amazon Simple Storage Service 使用者指南中的管理 Amazon S3 資源的存取許可。
- 若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？。

比較 AWS STS 登入資料

下表比較傳回暫時安全登入資料之 中 API AWS STS 操作的功能。若要了解可用來在擔任角色時請求暫時性安全憑證的不同方法，請參閱擔任角色的方法。若要了解可讓您傳遞工作階段標籤的不同 AWS STS API 操作，請參閱 在 中傳遞工作階段標籤 AWS STS。

> **Note**
>
> 您可以將 AWS STS API 呼叫傳送至全域端點或其中一個區域端點。如果您選擇比較靠近您的端點，您可以減少延遲並改善您的 API 呼叫的效能。如果可以不再與原始端點通訊，您也可以選擇將您的呼叫導向至其他區域性端點。如果您使用的是其中一個各種 AWS SDKs，請在進行 API 呼叫之前使用該 SDK 方法來指定區域。如果您手動建構 HTTP API 請求，則必須您自己將請求直接導向到正確的端點。如需詳細資訊，請參閱區域與端點的AWS STS 一節和 在 AWS STS 中管理 AWS 區域。

AWS STS API	誰可以呼叫	憑證存留期 (最小 | 最大 | 預設)	MFA 支援[1]	工作階段政策支援[2]	產生的暫時憑證限制
AssumeRe	具現有暫時性安全憑證的 IAM 使用者或 IAM 角色	15 分鐘 | 最大工作階段持續時間設定[3] | 1 小時	是	是	無法呼叫 `GetFederationToken` 或 `GetSessionToken`。
AssumeReWithSAM	任何使用者；呼叫者必須傳遞 SAML 身分驗證回應，指出通過已知身分提供者的身分驗證	15 分鐘 | 最大工作階段持續時間設定[3] | 1 小時	否	是	無法呼叫 `GetFederationToken` 或 `GetSessionToken`。

AWS STS API	誰可以呼叫	憑證存留期 (最小 | 最大 | 預設)	MFA 支援[1]	工作階段政策支援[2]	產生的暫時憑證限制
AssumeRoleWithWebIdentity	任何使用者；發起人必須傳遞符合 OIDC 規範的 JWT 權杖，指出通過已知身分提供者的驗證	15 分鐘 | 最大工作階段持續時間設定[3] | 1 小時	否	是	無法呼叫 GetFederationToken 或 GetSessionToken 。
GetFederationToken	IAM 使用者或 AWS 帳戶根使用者	IAM 使用者：15 分鐘 | 36 小時 | 12 小時 根使用者：15 分鐘 | 1 小時 | 1 小時	否	是	無法使用 AWS CLI 或 AWS API 呼叫 IAM 操作。此限制不適用主控台工作階段。 無法呼叫 GetCallerIdentity .4 以外的 AWS STS 操作 允許 SSO 到主控台。[5]

AWS STS API	誰可以呼叫	憑證存留期 (最小 | 最大 | 預設)	MFA 支援[1]	工作階段政策支援[2]	產生的暫時憑證限制
GetSessionToken	IAM 使用者或 AWS 帳戶根使用者	IAM 使用者：15 分鐘 | 36 小時 | 12 小時 根使用者：15 分鐘 | 1 小時 | 1 小時	是	否	除非請求包含 MFA 資訊，否則無法呼叫 IAM API 操作。 無法呼叫 `AssumeRole` 或 `GetCallerIdentity` 以外的 AWS STS API 操作。 不允許 SSO 到主控台。[6]

[1] MFA 支援。您可以在您呼叫 AssumeRole 和 GetSessionToken API 操作時，包含有關 多重要素驗證 (MFA) 裝置的資訊。這可確保 API 呼叫所產生的暫時性安全憑證，僅可由使用 MFA 裝置進行身分驗證的使用者使用。如需詳細資訊，請參閱 透過 MFA 實現安全的 API 存取。

[2] 工作階段政策支援。當您以程式設計方式為角色或 AWS STS 聯合身分使用者工作階段建立暫時工作階段時，工作階段政策是做為參數傳遞的政策。這個政策會限制工作階段獲派自角色或使用者之以身分為基礎政策的許可。所產生工作階段的許可會是實體的身分類型政策和工作階段政策的交集。工作階段政策不能用來授予超出即將擔任角色之以身分為基礎政策所允許的許可。如需有關角色工作階段許可的詳細資訊，請參閱 工作階段政策。

[3] 最大工作階段持續時間設定。使用 `DurationSeconds` 參數來指定 900 秒 (15 分鐘) 到角色的最大工作階段持續時間設定之間的角色工作階段持續時間。若要了解如何檢視角色的最大值，請參閱 更新角色的最大工作階段持續時間。

[4] GetCallerIdentity。執行此操作不需要任何許可。如果管理員將明確拒絕存取 `sts:GetCallerIdentity` 動作的政策新增到 IAM 使用者或角色，您仍然可以執行此操作。不需要許可，因為當 IAM 使用者或角色被拒絕存取時，會傳回相同的資訊。若要檢視範例回應，請參閱 我未獲得授權，不得執行：iam:DeleteVirtualMFADevice。

[5] Single sign-on (SSO) 至主控台。若要支援 SSO，請 AWS 呼叫聯合端點 (`https://signin.aws.amazon.com/federation`) 並傳遞臨時安全登入資料。端點會傳回權杖，您可用來建構 URL，直接將使用者登入主控台，而不需要密碼。如需詳細資訊，請參閱 AWS 安全部落格中的 啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台 和如何啟用 AWS 管理主控台的跨帳戶存取。

[6] 在您擷取暫時憑證後，您無法透過將憑證傳遞到聯合單一登入端點，來存取 AWS 管理主控台 。如需詳細資訊，請參閱啟用 AWS 主控台的自訂身分代理程式存取。

服務持有人權杖

有些 AWS 服務需要您取得 AWS STS 服務承載字符的許可，才能以程式設計方式存取其資源。這些服務支援的協定會要求您使用持有人權杖，而不是使用傳統的 AWS API 請求的簽章版本 4。當您執行需要承載字符的 AWS CLI 或 AWS API 操作時， AWS 服務會代表您請求承載字符。該服務會提供您權杖，接著您就可以使用該權杖在該服務中執行後續操作。

AWS STS 服務承載字符包含原始主體身分驗證中可能影響您許可的資訊。此資訊可能包括主體標籤、工作階段標籤和工作階段政策。權杖的存取金鑰 ID 會以 `ABIA` 字首開頭。這可協助您在 CloudTrail 日誌中識別出使用服務持有人權杖來執行的操作。

> **Important**
>
> 持有人權杖只能用於對產生該權杖之服務的呼叫，以及產生該權杖的區域中。您無法在其他服務或區域中使用持有人權杖執行操作。

支援承載字符的服務範例是 AWS CodeArtifact。您必須先呼叫 `aws codeartifact get-authorization-token`操作，才能使用 NPM、Maven 或 PIP 等套件管理員與 AWS CodeArtifact 互動。此操作會傳回承載字符，您可以用來執行 AWS CodeArtifact 操作。或者，您也可以使用能完成相同操作並自動設定您用戶端的 `aws codeartifact login` 命令。

如果您在為您產生承載字符的 AWS 服務中執行 動作，您必須在 IAM 政策中擁有下列許可：

JSON

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

```
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

如需服務持有人權杖範例，請參閱 AWS CodeArtifact 使用者指南中的將以身分為基礎的政策用於 AWS CodeArtifact。

請求臨時安全憑證

若要請求臨時安全登入資料，您可以在 AWS API 中使用 AWS Security Token Service (AWS STS) 操作。這包括建立和提供信任使用者暫時安全登入資料的操作，這些登入資料可以控制對 AWS 資源的存取。如需 的詳細資訊 AWS STS，請參閱 IAM 中的暫時安全憑證。若要了解可用來在擔任角色時請求暫時性安全憑證的不同方法，請參閱擔任角色的方法。

若要呼叫 API 操作，您可以使用其中一個 AWS 軟體開發套件。適用於多種程式設計語言和環境的軟體開發套件，包括 Java、.NET、Python、Ruby、Android 和 iOS。開發套件會負責的工作諸如以密碼演算法簽署請求，必要時重試請求，以及處理錯誤回應。您也可以使用 AWS STS 查詢 API，如 AWS Security Token Service API 參考中所述。最後，兩個命令列工具支援 AWS STS 命令： AWS Command Line Interface和 AWS Tools for Windows PowerShell。

AWS STS API 操作會使用包含存取金鑰對和工作階段字符的臨時安全登入資料建立新的工作階段。存取金鑰對包含存取金鑰 ID 和秘密金鑰。使用者 (或使用者執行的應用程式) 可以使用這些憑證來存取您的資源。

您可以建立角色工作階段，並使用 AWS STS API 操作以程式設計方式傳遞工作階段政策和工作階段標籤。所產生工作階段許可會是角色的以身分為基礎的政策和工作階段政策的交集。如需有關工作階段政策的詳細資訊，請參閱 工作階段政策。如需有關工作階段標籤的詳細資訊，請參閱 在 中傳遞工作階段標籤 AWS STS。

> **Note**
>
> AWS STS API 操作傳回的工作階段字符大小不固定。我們強烈建議您不對大小上限做出任何假設。一般權杖大小小於 4096 個位元組，但有可能會不同。

使用 AWS STS 搭配 AWS 區域

您可以將 AWS STS API 呼叫傳送至全域端點或其中一個區域端點。如果您選擇比較靠近您的端點，您可以減少延遲並改善您的 API 呼叫的效能。如果可以不再與原始端點通訊，您也可以選擇將您的呼叫導向至其他區域性端點。

如果您使用的是其中一個各種 AWS SDKs，請在進行 API 呼叫之前使用該 SDK 方法來指定區域。如果您手動建構 HTTP API 請求，則必須您自己將請求直接導向到正確的端點。如需詳細資訊，請參閱區域與端點的AWS STS 一節和 在 AWS STS 中管理 AWS 區域。

以下是您可以用來取得臨時登入資料的 API 操作，以便在 AWS 環境和應用程式中使用。

透過自訂身分經紀人請求跨帳戶委派和聯合身分的憑證

`AssumeRole` API 操作有助於讓現有的 IAM 使用者存取他們尚未存取 AWS 的資源。例如，使用者可能需要存取另一個 AWS 帳戶中的資源。它也能夠用來暫時獲得特殊許可，例如，提供多重要素驗證 (MFA)。您必須使用作用中的登入資料呼叫此 API。若要了解誰可以呼叫此操作，請參閱 比較 AWS STS 登入資料。如需詳細資訊，請參閱建立角色以將許可授予 IAM 使用者及透過 MFA 實現安全的 API 存取。

透過自訂身分經紀人請求跨帳戶委派和聯合身分的臨時安全憑證

1. 使用您的 AWS 安全登入資料進行驗證。必須使用有效的 AWS 安全憑證進行此呼叫。
2. 呼叫 `AssumeRole` 操作。

以下範例顯示使用 `AssumeRole` 的範例請求及回應。此範例請求假設指定時間內的 `demo` 角色具有已包含的工作階段政策、工作階段標籤、外部 ID 和來源身分。產生的工作階段名為 `John-session`。

Example範例請求

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws:iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17       %22%2C%22Statement%22%3A%5B%7B%22Sid
%22%3A%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*
%22%2C%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
```

```
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

在上述範例中顯示的政策值是以下政策的 URL 編碼版本：

JSON

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

範例中的 AUTHPARAMS 參數是簽章的預留位置。簽章是您在 AWS HTTP API 請求中必須包含的身分驗證資訊。我們建議使用 AWS 開發套件來建立 API 請求。執行此作業的其中一個好處是開發套件會為您處理請求簽署。如果您必須手動建立和簽署 API 請求，請參閱《》中的使用簽章版本 4 簽署 AWS 請求，以Amazon Web Services 一般參考了解如何簽署請求。

除了暫時性安全憑證外，回應包含聯合身分使用者的 Amazon Resource Name (ARN) 和憑證過期時間。

Example回應範例

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
```

```
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>34</PackedPolicySize>
<SessionTokenSize>1428</SessionTokenSize>
<SessionTokenUtilization>34</SessionTokenUtilization>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> 工作階段政策和工作階段標籤會新增至工作階段字符大小。如果字符超過大小限制，即使您的純文字在其他限制內，您的請求也會失敗。SessionTokenUtilization 回應元素會顯示字符使用的大小限制百分比。PackedPolicySize 回應元素已棄用。請改用 SessionTokenUtilization。

透過 OIDC 提供者請求憑證

AssumeRoleWithWebIdentity API 操作會傳回一組臨時 AWS 安全憑證，以換取 JSON Web 權杖 (JWT)。其中包括公有身分提供者 (例如 Login with Amazon、Facebook、Google)，以及發行與 OpenID Connect (OIDC) 探索相容的 JWT 的提供者 (例如 GitHub actions 或 Azure Devops)。如需詳細資訊，請參閱OIDC 聯合身分。

> **Note**
>
> AssumeRoleWithWebIdentity 請求未經簽名，且不需要 AWS 憑證。

透過 OIDC 提供者請求憑證

1. 呼叫 AssumeRoleWithWebIdentity 操作。

   當您呼叫 時AssumeRoleWithWebIdentity， 會使用 IdP 的 JSON Web 金鑰集 (JWKS) 提供的公有金鑰驗證數位簽章，以 AWS 驗證顯示的字符。如果字符有效，且符合 IAM 角色信任政策中規定的所有條件， 會 AWS 傳回下列資訊給您：

- 一組暫時性安全憑證。這些包含存取金鑰 ID、私密存取金鑰和工作階段權杖。
- 所擔任角色的角色 ID 和 ARN。
- SubjectFromWebIdentityToken 值，其中包含唯一的使用者 ID。

2. 然後，您的應用程式可能會使用回應中傳回的臨時安全登入資料來進行 AWS API 呼叫。這與使用長期安全登入資料進行 AWS API 呼叫的程序相同。差別在於您必須包含工作階段字符，讓 驗證 AWS 臨時安全登入資料是否有效。

您的應用程式應快取 傳回的登入資料 AWS STS ，並視需要重新整理。如果您的應用程式是使用 AWS SDK 建置，則 SDK 具有登入資料提供者，可在登入資料過期之前處理呼叫AssumeRoleWithWebIdentity和重新整理 AWS 登入資料。如需詳細資訊，請參閱 AWS SDKs and Tools Reference Guide 中的 AWS SDKs and Tools standardized credential providers。

透過 SAML 2.0 身分提供者請求憑證

AssumeRoleWithSAML API 操作會傳回 SAML 聯合身分主體的一組臨時安全憑證，這些主體透過您組織現有的身分系統進行驗證。使用者還必須使用 SAML 2.0 (安全聲明標記語言) 來將身分驗證和授權資訊傳遞至 AWS。這個 API 操作適用於整合他們自己的身分系統的組織 (例如，Windows Active Directory 或 OpenLDAP)，而軟體可以產生 SAML 聲明。這種整合會提供有關使用者身分和許可 (例如 Active Directory Federation Services 或 Shibboleth) 的資訊。如需詳細資訊，請參閱SAML 2.0 聯合身分。

1. 呼叫 AssumeRoleWithSAML 操作。

   這是未簽署的呼叫，這表示您不需要在提出請求之前驗證 AWS 安全登入資料。

   > **Note**
   >
   > AssumeRoleWithSAML 呼叫未簽署 (加密)。因此，如果請求是透過信任的媒介傳輸，您應該只包含選用的工作階段政策。在這種情況下，有人可以更改政策以移除限制。

2. 當您呼叫 時AssumeRoleWithSAML， 會 AWS 驗證 SAML 聲明的真實性。假設身分提供者驗證聲明， 會 AWS 傳回下列資訊給您：

   - 一組暫時性安全憑證。這些包含存取金鑰 ID、私密存取金鑰和工作階段權杖。
   - 所擔任角色的角色 ID 和 ARN。
   - Audience 值，其包含 SAML 聲明的 Recipient 元素的 SubjectConfirmationData 屬性值。

- Issuer 值，其包含 SAML 聲明的 Issuer 元素的值。
- 包含從 SAML 供應商的值、 AWS 帳戶 ID 和易記名稱建置的雜湊Issuer值的NameQualifier元素。當與 Subject 元素結合使用時，這些內容可以唯一識別 SAML 聯合身分主體。
- Subject 元素，其包含 SAML 聲明的 NameID 元素的 Subject 元素值。
- SubjectType 元素指出 Subject 元素的格式。此值可以是 persistent、transient 或在您的 SAML 聲明中使用的 Format 與 Subject 元素的完整 NameID URI。如需 NameID 元素的 Format 屬性的詳細資訊，請參閱 為身分驗證回應設定 SAML 聲明。

3. 使用回應中傳回的臨時安全登入資料來進行 AWS API 呼叫。這與使用長期安全登入資料進行 AWS API 呼叫的程序相同。差別在於您必須包含工作階段權杖，這個權杖可讓 AWS 驗證臨時安全憑證是否有效。

您的應用程式應該快取憑證。預設情況下，憑證會在一小時後到期。如果您不是在 AWS SDK 中使用 AmazonSTSCredentialsProvider 動作，則由您和應用程式AssumeRoleWithSAML再次呼叫。呼叫此操作，或取得一組新的暫時性安全憑證，之後舊的憑證才會過期。

透過自訂身分經紀人請求憑證

GetFederationToken API 操作會傳回 AWS STS 一組聯合身分使用者主體的臨時安全登入資料。此 API 與 AssumeRole 不同，其中的預設過期時段是實質上比較長 (12 小時，而不是 1 小時)。此外，您可以使用 DurationSeconds 參數來指定暫時性安全憑證的持續時間，以保持有效。產生的憑證在指定的持續時間內有效，介於 900 秒 (15 分鐘) 至 129,600 秒 (36 小時) 之間。較長的過期期間有助於減少對 的呼叫數量， AWS 因為您不需要經常取得新的登入資料。

1. 使用特定 IAM 使用者 AWS 的安全登入資料進行驗證。此呼叫必須使用有效的 AWS 安全登入資料進行。
2. 呼叫 GetFederationToken 操作。

GetFederationToken 呼叫會傳回暫時性安全憑證，該憑證由工作階段字符、存取金鑰、私密金鑰和過期時間組成。如果您想要管理您組織內的許可 (例如，使用 Proxy 應用程式以指派許可)，您可以使用 GetFederationToken。

以下範例顯示使用 GetFederationToken 的範例請求及回應。此範例請求會將指定期間的呼叫發起使用者與工作階段政策 ARN 和工作階段標籤聯合在一起。產生的工作階段名為 Jane-session。

Example範例請求

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

上述範例中顯示的政策 ARN 包含以下 URL 編碼 ARN：

```
arn:aws:iam::123456789012:policy/Role1policy
```

此外，請注意範例中的 &AUTHPARAMS 參數表示身分驗證資訊的預留位置。這是簽章，您必須將其包含在 AWS HTTP API 請求中。我們建議使用 AWS 開發套件來建立 API 請求。執行此作業的其中一個好處是開發套件會為您處理請求簽署。如果您必須手動建立和簽署 API 請求，請參閱《》中的使用簽章版本 4 簽署 AWS 請求，以Amazon Web Services 一般參考了解如何簽署請求。

除了暫時性安全憑證外，回應包含聯合身分使用者的 Amazon Resource Name (ARN) 和憑證過期時間。

Example回應範例

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
```

```
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>26</PackedPolicySize>
<SessionTokenSize>1105</SessionTokenSize>
<SessionTokenUtilization>26</SessionTokenUtilization>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> 工作階段政策和工作階段標籤會新增至工作階段字符大小。如果字符超過大小限制，即使您的純文字在其他限制內，您的請求也會失敗。SessionTokenUtilization 回應元素會顯示字符使用的大小限制百分比。PackedPolicySize 回應元素已棄用。請改用 SessionTokenUtilization。

AWS 建議您在資源層級授予許可（例如，您將資源型政策連接到 Amazon S3 儲存貯體），您可以省略 Policy 參數。不過，如果您未包含 AWS STS 聯合身分使用者主體的政策，則暫時安全登入資料將不會授予任何許可。在這種情況下，您「必須」使用資源政策，來授予聯合身分使用者存取您 AWS 資源的許可。

例如，假設您的 AWS 帳戶 號碼是 111122223333，而且您要允許 Susan 存取 Amazon S3 儲存貯體。Susan 的暫時性安全憑證不包含儲存貯體的政策。在這種情況下，您需要確保儲存貯體具有其 ARN 與 Susan 的 ARN 相符的政策，例如 arn:aws:sts::111122223333:federated-user/Susan。

為不信任環境中的使用者請求憑證

GetSessionToken API 操作會傳回現有 IAM 使用者的一組臨時安全憑證。這有助於提供增強的安全性，例如僅在 IAM 使用者啟用 MFA 時允許 AWS 請求。因為憑證為暫時性，它們可在您有透過較不安全環境存取您資源的 IAM 使用者時，提供增強的安全性。較不安全環境的範例包含行動裝置或 Web 瀏覽器。

1. 使用特定 IAM 使用者 AWS 的安全登入資料進行驗證。此呼叫必須使用有效的 AWS 安全登入資料進行。
2. 呼叫 GetSessionToken 操作。
3. GetSessionToken 會傳回暫時性安全憑證，其由工作階段字符、存取金鑰 ID 和私密存取金鑰組成。

在預設情況下，IAM 使用者的暫時性安全憑證的有效期最長為 12 小時。但是，您可以使用 DurationSeconds 參數請求持續時間最短為 15 分鐘最長為 36 小時。基於安全考量， 的字符的持續時間限制 AWS 帳戶根使用者 為一小時。

以下範例顯示使用 GetSessionToken 的範例請求及回應。回應還包括暫時性安全憑證的過期時間。

Example範例請求

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
&AUTHPARAMS
```

範例中的 AUTHPARAMS 參數是簽章的預留位置。簽章是您在 AWS HTTP API 請求中必須包含的身分驗證資訊。我們建議使用 AWS 開發套件來建立 API 請求。執行此作業的其中一個好處是開發套件會為您處理請求簽署。如果您必須手動建立和簽署 API 請求，請參閱《》中的使用簽章版本 4 簽署 AWS 請求，以Amazon Web Services 一般參考了解如何簽署請求。

Example回應範例

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
```

```
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<SessionTokenSize>856</SessionTokenSize>
<SessionTokenUtilization>20</SessionTokenUtilization>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

或者，GetSessionToken請求可以包含 AWS 多重要素驗證 (MFA) 驗證的 SerialNumber和 TokenCode值。如果提供的值有效， AWS STS 會提供包含 MFA 身分驗證狀態的臨時安全登入資料。然後，只要 MFA 身分驗證有效，就可以使用臨時安全登入資料來存取受 MFA 保護的 API 操作或 AWS 網站。

以下範例顯示 GetSessionToken 請求，其中包含 MFA 驗證碼和裝置序號。

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> 對 的呼叫 AWS STS 可以是全域端點或您啟用 的任何區域端點 AWS 帳戶。如需詳細資訊，請參閱區域與端點的AWS STS 一節。
> 範例中的 AUTHPARAMS 參數是簽章的預留位置。簽章是您在 AWS HTTP API 請求中必須包含的身分驗證資訊。我們建議使用 AWS 開發套件來建立 API 請求。執行此作業的其中一個好處是開發套件會為您處理請求簽署。如果您必須手動建立和簽署 API 請求，請參閱《》中的使用簽章版本 4 簽署 AWS 請求，以Amazon Web Services 一般參考了解如何簽署請求。

搭配 AWS 資源使用臨時登入資料

您可以使用臨時安全登入資料，使用 AWS CLI 或 AWS API （使用 AWS SDKs) 對 AWS 資源提出程式設計請求。暫時性憑證提供的許可與長期安全憑證許可相同，例如 IAM 使用者憑證。不過，有幾個差異：

- 當您使用臨時安全登入資料進行呼叫時，呼叫必須包含工作階段字符，其會與這些臨時登入資料一起傳回。 AWS 會使用工作階段字符來驗證臨時安全登入資料。
- 暫時性憑證在指定的間隔時間後到期。暫時性憑證到期後，使用這些憑證進行的任何呼叫都將失敗，所以您必須產生一組新的暫時性憑證。暫時性憑證無法延期或重新整理超過原始指定間隔。
- 當您使用暫時憑證發出請求時，您的主體可能會包含一組標籤。這些標籤來自工作階段標籤和連接到您擔任角色的標籤。如需有關工作階段標籤的詳細資訊，請參閱 在 中傳遞工作階段標籤 AWS STS。

如果您使用的是 AWS SDK、AWS Command Line Interface (AWS CLI) 或 Tools for Windows PowerShell，則取得和使用臨時安全憑證的方式會隨內容而有所不同。如果您在 EC2 執行個體內執行程式碼 AWS CLI或 Tools for Windows PowerShell 命令，您可以利用 Amazon EC2 的角色。否則，您可以呼叫 AWS STS API 以取得暫時憑證，然後明確地使用他們呼叫 AWS 服務。

> **Note**
>
> 您可以使用 AWS Security Token Service (AWS STS) 來建立信任的使用者，並提供暫時安全登入資料，以控制對 AWS 資源的存取。如需 的詳細資訊 AWS STS，請參閱 IAM 中的暫時安全憑證。 AWS STS 是一項全域服務，其在 具有預設端點`https://sts.amazonaws.com`。此端點位在美國東部 (維吉尼亞北部) 區域，但您從此端點和其他端點取得的憑證為全域有效。這些憑證可配合任何區域中的服務和資源使用。您也可以選擇對任何支援區域中的端點進行 AWS STS API 呼叫。從地理位置較靠近您的區域的伺服器提出請求，可以降低延遲發生機率。無論您的憑證來自哪個區域，都能全域使用。如需詳細資訊，請參閱 在 AWS STS 中管理 AWS 區域。

內容

- 在 Amazon EC2 執行個體中使用暫時憑證
- 搭配 AWS SDKs
- 搭配 使用臨時安全登入資料 AWS CLI
- 使用暫時性安全憑證與 API 操作
- 其他資訊

在 Amazon EC2 執行個體中使用暫時憑證

如果您想要在 EC2 執行個體內執行 AWS CLI 命令或程式碼，建議取得登入資料的方式是使用 Amazon EC2 的角色。您建立 IAM 角色，指定要授予在 EC2 執行個體上執行之應用程式的許可。當您啟動執行個體時，會建立角色與執行個體的關聯。

然後 AWS CLI，在執行個體上執行的應用程式和 Tools for Windows PowerShell 命令可以從執行個體中繼資料取得自動臨時安全登入資料。您不必明確取得暫時安全憑證。適用於 Windows PowerShell AWS SDKs AWS CLI和工具會自動從 EC2 執行個體中繼資料服務 (IMDS) 取得憑證並使用它們。暫時性憑證擁有您為與執行個體定義關聯的角色所定義的許可。

如需詳細資訊及範例，請參閱下列：

- 使用 IAM 角色授予 AWS Amazon Elastic Compute Cloud 上資源的存取權： 適用於 Java 的 AWS SDK
- 使用 IAM 角色授予存取權 — 適用於 .NET 的 AWS SDK
- 建立角色 — 適用於 Ruby 的 AWS SDK

搭配 AWS SDKs

若要在程式碼中使用臨時安全登入資料，請以程式設計方式呼叫類似 的 AWS STS API，`AssumeRole`並擷取產生的登入資料和工作階段字符。然後，您可以將這些值用作後續呼叫的登入資料 AWS。下列範例顯示假碼，說明如何在使用 AWS SDK 時使用臨時安全登入資料：

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

如需使用 Python 撰寫的範例 (使用 適用於 Python (Boto) 的 AWS SDK)，請參閱切換到 IAM 角色 (AWS API)。此範例會示範如何呼叫 `AssumeRole` 以取得暫時安全憑證，然後使用這些憑證呼叫 Amazon S3。

如需如何呼叫 `AssumeRole`、`GetFederationToken` 以及其他 API 操作的詳細資訊，請參閱 AWS Security Token Service API 參考。有關從結果取得臨時安全憑證和工作階段權杖的詳細資訊，請參閱您正在使用的軟體開發套件的說明文件。您可以在 SDK SDKs 和工具組區段的主要文件頁面上找到所有 AWS SDKs 的文件。 AWS

您必須在舊的憑證到期之前，確實取得一組新的憑證。在某些軟體開發套件中，您可以使用為您管理重新整理憑證程序的提供者；檢查您正在使用的開發套件的文件。

搭配 使用臨時安全登入資料 AWS CLI

您可以使用暫時性安全憑證與 AWS CLI。這很適合用於測試政策。

若使用 AWS CLI，您便可以呼叫 AWS STS API (例如 `AssumeRole` 或 `GetFederationToken`)，然後擷取產生的輸出。以下範例顯示對 `AssumeRole` 的呼叫，其會傳送輸出至檔案。在此範例中，假設 `profile` 參數是 AWS CLI 組態檔案中的設定檔。同時也假設會參考有權擔任角色之 IAM 使用者的憑證。

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

當命令完成時，您可以從您路由的任何地方擷取存取金鑰 ID、私密存取金鑰和工作階段權杖。您可以手動或使用指令碼執行這項作業。然後，您可以將這些值指派給環境變數。

當您執行 AWS CLI 命令時， 會先在環境變數中 AWS CLI 尋找特定順序的登入資料，然後在組態檔案中尋找登入資料。因此，在您將臨時登入資料放入環境變數之後， 預設 AWS CLI 會使用這些登入資料。（ 如果您在 命令中指定`profile`參數， 會 AWS CLI 略過環境變數。 相反地， 會在組態檔案中 AWS CLI 尋找，這可讓您視需要覆寫環境變數中的登入資料。）

下列範例示範如何設定暫時安全登入資料的環境變數，然後呼叫 AWS CLI 命令。由於 AWS CLI 命令中不包含 `profile` 參數，因此 會先在環境變數中 AWS CLI 尋找登入資料，因此會使用臨時登入資料。

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
```

```
C:\> aws ec2 describe-instances --region us-west-1
```

使用暫時性安全憑證與 API 操作

如果您要直接向 HTTPS API 請求 AWS，您可以使用從 AWS Security Token Service () 取得的臨時安全登入資料簽署這些請求AWS STS。若要這樣做，您可以使用您從中接收的存取金鑰 ID 和私密存取金鑰 AWS STS。使用存取金鑰 ID 和私密存取金鑰簽署請求的方式，就和使用長期憑證簽署請求一樣。您也可以將您收到的工作階段字符新增至您的 API 請求 AWS STS。您將工作階段權杖新增到 HTTP 標頭，或查詢名為 `X-Amz-Security-Token` 的字串參數。您將工作階段權杖新增到 HTTP 標頭「或」查詢字串參數，但不是兩者。如需簽署 HTTPS API 請求的詳細資訊，請參閱《》中的簽署 AWS API 請求AWS 一般參考。

其他資訊

如需將 AWS STS 與其他 AWS 服務搭配使用的詳細資訊，請參閱下列連結：

- Amazon S3 請參閱 《Amazon Simple Storage Service 使用者指南》中的使用 IAM 使用者暫時憑證提出請求或使用聯合身分使用者暫時憑證提出請求。
- Amazon SNS。請參閱《Amazon Simple Queue Service 開發人員指南》中的將身分型政策與 Amazon SNS 搭配使用。
- Amazon SQS 請參閱《Amazon Simple Queue Service 開發人員指南》中的 Amazon SQS 中的 Identity and Access Management。
- Amazon SimpleDB。請參閱使用臨時安全憑證中的Amazon SimpleDB 開發人員指南。

臨時安全憑證的許可

您可以使用 AWS Security Token Service (AWS STS) 來建立信任的使用者，並提供暫時安全登入資料，以控制對 AWS 資源的存取。如需 的詳細資訊 AWS STS，請參閱 IAM 中的暫時安全憑證。在 AWS STS 發出臨時安全憑證後，它們在過期期間有效，且無法撤銷。但是，每次發出使用憑證的請求時，都會評估指派給臨時安全憑證的許可，因此您可以透過在發出憑證後變更其存取權來達到撤銷憑證的效果。

下列主題假設您具備 AWS 許可和政策的工作知識。如需有關這些主題的詳細資訊，請參閱 AWS 資源的存取管理。

主題

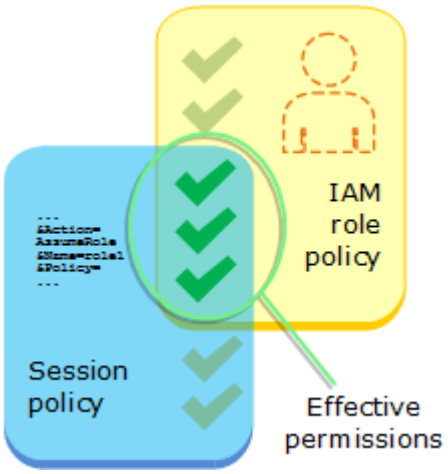
- AssumeRole、AssumeRoleWithSAML 和 AssumeRoleWithWebIdentity 的許可
- 監控並控制使用擔任角色所採取的動作

- GetFederationToken 許可
- GetSessionToken 許可
- 停用臨時安全性憑證的許可
- 授予許可以建立暫時性安全憑證
- 授予許可以使用身分增強型主控台工作階段

AssumeRole、AssumeRoleWithSAML 和 AssumeRoleWithWebIdentity 的許可

所擔任角色的許可政策，將會決定由 AssumeRole、AssumeRoleWithSAML 和 AssumeRoleWithWebIdentity 傳回的臨時安全憑證的許可。當您建立或更新角色時定義這些許可。

或者，您可以傳遞內嵌或受管的工作階段政策作為 AssumeRole、AssumeRoleWithSAML 或 AssumeRoleWithWebIdentity API 操作的參數。工作階段政策會為該角色的臨時憑證工作階段限制許可。所產生工作階段的許可會是角色的身分類型政策和工作階段政策的交集。您可以在後續 AWS API 呼叫中使用角色的臨時登入資料，來存取擁有該角色之帳戶中的資源。您無法使用工作階段政策來授予超出即將擔任角色之以身分為基礎政策所允許的許可。若要進一步了解 AWS 如何決定角色的有效許可，請參閱政策評估邏輯。



進行「允許」或「拒絕」授權決策 AWS 時， AssumeRole 不會評估連接至發出原始呼叫之登入資料的 政策。使用者暫時放棄其原始許可，以支援由擔任角色指派的許可。在 AssumeRoleWithSAML和 AssumeRoleWithWebIdentity API 操作中，由於 API 的發起人不是 AWS 身分，因此沒有要評估的政策。

範例：使用 AssumeRole 指派許可

您可以將 AssumeRole API 操作與不同類型的政策一起使用。以下是幾個範例。

角色許可政策

在此範例中，您在呼叫 AssumeRole API 操作時未使用選用 Policy 參數來指定工作階段。指派給臨時憑證的許可由所擔任角色的許可政策決定。以下範例許可政策授予角色許可，以列出名為 productionapp 的 S3 儲存貯體中包含的所有物件。它還允許角色取得、放置和刪除該儲存貯體中的物件。

Example角色許可政策範例

JSON

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

工作階段政策作為參數傳遞

假設您想要讓使用者擔任與上一個範例相同的角色。但是，此時您希望該角色工作階段擁有的許可，只允許其取得 productionapp S3 儲存貯體中的物件，以及在其中放入物件。您不想讓他們刪除物件。實現此目的一種方法是建立新的角色並在該角色的許可政策中指定所需許可。實現此目的之另一種方法是呼叫 AssumeRole API，並將工作階段政策包含在可選的 Policy 參數做為 API 操作的一部分。所產生工作階段的許可會是角色的以身分為基礎的政策和工作階段政策的交集。工作階段政策不能用來授予超出即將擔任角色之以身分為基礎政策所允許的許可。如需有關角色工作階段許可的詳細資訊，請參閱 工作階段政策。

擷取新的工作階段臨時憑證後，您可以將這些資料傳遞給您希望擁有這些許可的使用者。

例如，假設以下政策做為 API 呼叫的參數傳遞。使用這項工作階段的人員所擁有的許可，將僅能執行下列動作：

- 列出 productionapp 儲存貯體中的所有物件。
- 取得並將物件放入 productionapp 儲存貯體中。

在以下工作階段政策中，s3:DeleteObject 許可會遭篩除，而且擔任的工作階段不會獲得 s3:DeleteObject 許可的授予。此政策設定角色工作階段的最大許可，並供以覆寫該角色的任何現有許可政策。

Example工作階段使用AssumeRole API 呼叫傳遞政策範例

JSON

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

資源型政策

有些 AWS 資源支援以資源為基礎的政策，這些政策提供另一個機制來定義影響臨時安全登入資料的許可。只有少數資源 (例如 Amazon S3 儲存貯體、Amazon SNS 主題和 Amazon SQS 佇列) 支援資源式

政策。以下範例使用名為 productionapp 的 S3 儲存貯體擴展前面的範例。以下政策連接至儲存貯體。

當您將下面以資源為基礎的政策連接到 productionapp 儲存貯體時，將拒絕所有使用者從儲存貯體中刪除物件的許可。(請參閱政策中的 Principal 元素。) 這包括所有擔任角色使用者，即使角色許可政策授予 DeleteObject 許可。明確 Deny 陳述式一律優先於 Allow 陳述式。

Example儲存貯體政策的範例

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

如需如何合併和評估多個政策類型的詳細資訊 AWS，請參閱 政策評估邏輯。

監控並控制使用擔任角色所採取的動作

IAM 角色是 IAM 中受指派許可的物件。當您使用來自 外部的 IAM 身分或身分擔任該角色時 AWS，您會收到具有指派給該角色之許可的工作階段。

當您在 中執行動作時 AWS，您的工作階段相關資訊可以記錄到 AWS CloudTrail ，以供帳戶管理員監控。管理員可以將角色設定為要求身分傳遞自訂字串，以識別在 AWS中執行動作的人員或應用程式。此身分資訊會儲存為 AWS CloudTrail中的來源身分。當管理員查看 CloudTrail 中的活動時，他們可以查看來源身分資訊以判斷曾使用擔任角色工作階段執行動作者。

設定來源身分後，它會出現在角色工作階段期間所採取任何 AWS 動作的請求中。當角色用於透過 AWS CLI 或 AWS API 擔任另一個角色時，設定的 值會持續存在，稱為角色鏈結。設定的值無法在角色工作階段期間變更。

管理員可以根據來源身分的存在或值來設定精細許可，以進一步控制使用共用角色採取 AWS 的動作。您可以決定是否可以使用來源身分屬性、是否需要，以及可以使用的數值。

您使用來源身分的方式與角色工作階段名稱和工作階段標記有很大的不同。來源身分值在設定之後即無法變更，而且對於對角色工作階段採取的任何其他動作，會持續存在。以下說明如何使用工作階段標籤和角色工作階段名稱：

- 工作階段標籤 – 您也可以在擔任角色或與使用者聯合身分時傳遞工作階段標籤。擔任角色時，工作階段標籤即會出現。您可以定義使用標籤條件金鑰的政策，以根據主體的標籤來授與許可。您可以使用 CloudTrail 來檢視為了擔任角色或聯合身分使用者而提出的請求。若要進一步了解工作階段標籤，請參閱 在 中傳遞工作階段標籤 AWS STS。
- 角色工作階段名稱 – 您可以在角色信任政策中使用 `sts:RoleSessionName` 條件金鑰，請求您的使用者在擔任角色時提供特定的工作階段名稱。當不同主體使用角色時，角色工作階段名稱可用來區分角色工作階段。若要進一步了解角色工作階段名稱，請參閱 sts:RoleSessionName。

當您想要控制擔任角色的身分時，建議您使用來源身分。對於挖掘 CloudTrail 日誌以判斷使用角色來執行動作的人員時，來源身分也示非常實用的資訊。

主題

- 設定以使用來源身分
- 來源身分的須知事項
- 設定來源身分所需的許可
- 擔任角色時指定來源身分
- 搭配 AssumeRole 使用來源身分
- 將來源身分與 AssumeRoleWithSAML 搭配使用
- 使用具有 AssumeRoleWithWebIdentity 的來源身分
- 使用來源身分資訊控制存取
- 在 CloudTrail 中檢視來源身分

設定以使用來源身分

您設定為使用來源身分的方式將取決於擔任角色時所使用的方法。例如，您的 IAM 使用者可能會直接使用 `AssumeRole` 操作擔任角色。如果您有企業身分，也稱為人力身分，他們可能會使用 存取您的 AWS 資源`AssumeRoleWithSAML`。如果最終使用者要存取您的行動或 Web 應用程式，他們可能會使用 `AssumeRoleWithWebIdentity` 來存取。以下是高階工作流程概觀，可協助您瞭解如何設定以利用現有環境中的來源身分資訊。

1. 設定測試使用者和角色 – 使用生產前環境，設定測試使用者和角色，並設定其政策以允許設定來源身分。

   如果您使用身分提供者 (IdP) 作為聯合身分，請將 IdP 設定為將您選擇的使用者屬性傳遞至聲明或權杖中的來源身分。
2. 擔任角色 – 測試擔任角色，並使用您設定要進行測試的使用者和角色傳遞來源身分。
3. 檢視 CloudTrail – 檢閱 CloudTrail 日誌中測試角色的來源身分資訊。
4. 培訓您的使用者 – 在生產前環境中進行測試之後，請確保您的使用者知道如何傳入來源身分資訊 (如有必要)。設定您要求使用者在生產環境中提供來源身分的截止日期。
5. 設定生產政策 – 為您的生產環境設定政策，然後將政策新增至您的生產使用者和角色。
6. 監控活動 – 使用 CloudTrail 日誌來監控您的生產角色活動。

來源身分的須知事項

使用來源身分時請記住下列事項。

- 連線至身分提供者 (IdP) 之所有角色的信任政策均須具有 `sts:SetSourceIdentity` 許可。對於角色信任政策中沒有此許可的角色，`AssumeRole*` 操作將會失敗。如果您不想更新每個角色的角色信任政策，則您可以使用個別的 IdP 執行個體傳遞來源身分。然後將 `sts:SetSourceIdentity` 許可僅新增至連線至個別 IdP 的角色。
- 當身分設定來源身分時，`sts:SourceIdentity` 金鑰會存在於請求中。針對在角色工作階段期間採取的後續動作，`aws:SourceIdentity` 金鑰會出現在請求中。 AWS 不會在 `sts:SourceIdentity` 或 `aws:SourceIdentity` 金鑰中控制來源身分的值。如果您選擇要求來源身分，則必須選擇要讓使用者或 IdP 提供的屬性。基於安全考量，您必須確保您可以控制提供這些值的方式。
- 來源身分的值長度必須介於 2 到 256 個字元之間，只能包含英數字元、底線和下列字元：. ， + = @ - （連字號）。您不能使用以文字 **`aws:`** 為開頭的值。此字首保留供 AWS 內部使用。
- 當 AWS 服務或服務連結角色代表聯合身分或人力資源身分執行動作時，CloudTrail 不會擷取來源身分資訊。

> **Important**
>
> 您無法在 中切換到在擔任角色時 AWS 管理主控台 需要設定來源身分的角色。若要擔任這類角色，您可以使用 AWS CLI 或 AWS API 來呼叫 `AssumeRole`操作，並指定來源身分參數。

設定來源身分所需的許可

除了符合 API 操作的動作外，您必須在您的政策中具備下列僅許可動作：

```
sts:SetSourceIdentity
```

- 若要指定來源身分，主體 (IAM 使用者和角色) 必須具有 `sts:SetSourceIdentity` 的許可。身為系統管理員，您可以在角色信任政策和主體的權限政策中設定此選項。
- 當您擔任另一個角色的角色時，稱為角色鏈結，在主體 (擔任角色) 的許可政策和目標角色的角色信任政策中都需要 `sts:SetSourceIdentity` 的許可。否則，擔任角色操作將會失敗。
- 使用來源身分時，連線至 IdP 之所有角色的角色信任政策必須具有 `sts:SetSourceIdentity` 許可。任何連接至 IdP 的角色，在沒有此許可的情況下，`AssumeRole*` 操作將會失敗。如果您不想更新每個角色的角色信任政策，則您可以使用個別的 IdP 執行個體傳遞來源身分，然後將 `sts:SetSourceIdentity` 許可僅新增至連線至個別 IdP 的角色
- 若要跨帳戶界限設定來源身分，您必須在兩個地方均包含 `sts:SetSourceIdentity` 許可。此許可必須存在於原始帳戶中主體的許可政策，以及目標帳戶中角色的角色信任政策。您可能需要執行此操作，例如，當一個角色被用來在另一個具有角色鏈結的帳戶中擔任一個角色時。

身為帳戶管理員，假設您想要允許您帳戶中的 IAM 使用者 `DevUser` 在同一個帳戶中擔任 `Developer_Role`。但是，您希望只有在使用者已將來源身分設定為其 IAM 使用者名稱時，才允許此動作。您可將下列政策連接至 IAM 使用者。

Example連接至 DevUser 的以身分為基礎的政策範例

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
    },
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
```

```
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
}
```

若要強制執行可接受的來源身分值，您可以設定下列角色信任政策。政策會為 IAM 使用者提供DevUser許可來擔任角色並設定來源身分。sts:SourceIdentity 條件索引鍵定義了可接受的來源身分值。

Example來源身分的角色信任政策範例

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
        }
      }
    }
  ]
}
```

使用 IAM 使用者的登入資料DevUser，使用者會嘗試DeveloperRole使用以下 AWS CLI 請求擔任 。

Example AssumeRole CLI 請求範例

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

當 AWS 評估請求時，請求內容會包含 sts:SourceIdentity的 DevUser。

擔任角色時指定來源身分

當您使用其中一個 AWS STS AssumeRole* API 操作來取得角色的臨時安全登入資料時，可以指定來源身分。您使用的 API 操作會視您的使用案例而有所不同。

例如，如果您使用 IAM 角色讓 IAM 使用者存取他們通常無法存取 AWS 的資源，您可以使用 AssumeRole操作。如果您使用企業聯合身分來管理人力使用者，可以使用 AssumeRoleWithSAML 操作。如果您使用 OIDC 聯合身分來允許最終使用者存取您的行動裝置或 Web 應用程式，則可以使用 AssumeRoleWithWebIdentity 操作。以下各節說明如何在每項操作中使用來源身分。若要進一步了解暫時憑證的常見案例，請參閱暫時性憑證的常見案例。

搭配 AssumeRole 使用來源身分

AssumeRole 操作會傳回一組臨時登入資料，您可以用來存取 AWS 資源。您可以使用 IAM 使用者或角色憑證來呼叫 AssumeRole。若要在擔任角色時傳遞來源身分，請使用 --source-identity AWS CLI 選項或 SourceIdentity AWS API 參數。以下範例顯示如何使用 AWS CLI來指定來源身分。

Example AssumeRole CLI 請求範例

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
--role-session-name Audit \
--source-identity Admin \
```

將來源身分與 AssumeRoleWithSAML 搭配使用

主體呼叫 AssumeRoleWithSAML 操作是使用 SAML 類型的聯合進行身分驗證。此操作會傳回一組臨時登入資料，您可以用來存取 AWS 資源。如需使用 SAML 型聯合進行 AWS 管理主控台 存取的詳細

資訊，請參閱 啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。如需 AWS CLI 或 AWS API 存取的詳細資訊，請參閱 SAML 2.0 聯合身分。如需為 Active Directory 使用者設定 SAML 聯合的教學課程，請參閱 AWS 安全部落格中的AWS 使用 Active Directory 聯合服務 (ADFS) 進行聯合身分驗證。

身為管理員，您可以允許公司目錄的成員 AWS 使用 AWS STS `AssumeRoleWithSAML`操作聯合到 。若要執行此操作，您必須完成下列任務：

1. 在組織中設定 SAML 提供者。
2. 在 IAM 中建立 SAML 提供者
3. 在 中 AWS 為您的 SAML 聯合委託人設定角色及其許可。
4. 完成配置 SAML IdP 並為 SAML 身分驗證回應建立聲明。

若要設定來源身分的 SAML 屬性，請包含 `Attribute` 元素與設定為 `https://aws.amazon.com/SAML/Attributes/SourceIdentity` 的 `Name` 屬性。使用 `AttributeValue` 元素指定來源身分的值。例如，假設您希望將下列身分屬性作為來源身分傳遞。

```
SourceIdentity:DiegoRamirez
```

如要傳遞此屬性，請在您的 SAML 聲明中包含下列元素。

Example SAML 聲明的程式碼片段範例

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

使用具有 AssumeRoleWithWebIdentity 的來源身分

主體呼叫 `AssumeRoleWithWebIdentity` 操作是使用與 OpenID Connect (OIDC) 相容的聯合身分進行身分驗證。此操作會傳回一組暫時憑證，讓您用來存取 AWS 資源。如需有關使用 OIDC 聯合身分以存取 AWS 管理主控台 的資訊，請參閱 OIDC 聯合身分。

若要從 OpenID Connect (OIDC) 傳遞來源身分，您必須在 JSON Web 權杖 (JWT) 中包含來源身分。在您提交 `AssumeRoleWithWebIdentity` 請求時，您必須在權杖中的 `https://aws.amazon.com/ source_identity` 命名空間內包含來源身分。若要進一步了解 OIDC 權杖和要求，請參閱《Amazon Cognito 開發人員指南》中的搭配使用者集區使用權杖。

例如，以下解碼後的 JWT 是搭配 `Admin` 來源身分呼叫 `AssumeRoleWithWebIdentity` 的權杖。

Example解碼後的 JSON Web 權杖範例

```
{
    "sub": "john",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

使用來源身分資訊控制存取

初始設定來源身分時，sts:SourceIdentity 金鑰會出現在請求中。設定來源身分之後，aws:SourceIdentity 金鑰會存在於角色工作階段期間所提出的所有後續請求中。身為管理員，您可以撰寫政策，授予條件式授權，以根據來源身分屬性的存在或值執行 AWS 動作。

假設您想要要求開發人員設定來源身分，以擔任具有寫入生產關鍵 AWS 資源許可的關鍵角色。也假設您使用 授予對人力資源身分的 AWS 存取權`AssumeRoleWithSAML`。您只希望資深開發人員 Saanvi 和 Diego 能夠存取角色，因此您為該角色建立下列信任政策。

Example來源身分的角色信任政策範例 (SAML)

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
```

```
          "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

信任政策包含 `sts:SourceIdentity` 的條件，需要 Saanvi 或 Diego 的來源身分，才能擔任關鍵角色。

或者，如果您使用 OIDC 提供者作為聯合身分，且使用者已透過 `AssumeRoleWithWebIdentity` 驗證身分，則您的角色信任政策可能如下所示。

Example來源身分的角色信任政策範例 (OIDC 提供者)

JSON

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

```
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

角色鏈結和跨帳戶需求

假設您想允許已擔任 CriticalRole 的使用者在另一個帳戶中擔任 CriticalRole_2。為擔任 CriticalRole 而取得的角色工作階段憑證用於將角色鏈結至另一個帳戶中的第二個角色 CriticalRole_2。角色是跨帳戶界限擔任。因此，必須在 CriticalRole 的許可政策和 CriticalRole_2 的角色信任政策中授予 sts:SetSourceIdentity 許可。

Example CriticalRole 上的許可政策範例

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
```

```
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

為了確保跨帳戶界限設定來源身分，以下角色信任政策僅信任 CriticalRole 的角色主體來設定來源身分。

Example CriticalRole_2 上的角色信任政策範例

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

使用者使用擔任 CritealRole 所取得的角色工作階段憑證進行下列呼叫。來源身分是在擔任 CrealRole 期間所設定，所以不需要再次明確設定。如果使用者嘗試設定的來源身分與擔任 CriticalRole 時設定的值不同，則擔任角色請求將會遭到拒絕。

Example AssumeRole CLI 請求範例

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

當呼叫主體擔任角色時，請求中的來源身分會從第一個擔任角色工作階段開始一直存在。因此，aws:SourceIdentity 和 sts:SourceIdentity 索引鍵均會出現在請求內容中。

在 CloudTrail 中檢視來源身分

您可以使用 CloudTrail 來檢視為了擔任角色或聯合身分使用者而提出的請求。您也可以檢視角色或使用者請求，以在 AWS中採取行動。CloudTrail 日誌檔案包含經擔任角色或聯合身分使用者工作階段的來源身分設定資訊。如需詳細資訊，請參閱使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail

例如，假設使用者提出 AWS STS AssumeRole請求，並設定來源身分。您可以在您的 CloudTrail 日誌的 requestParameters 索引鍵中找到 sourceIdentity 資訊。

Example AWS CloudTrail 日誌中的範例 requestParameters 區段

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

如果使用者使用擔任的角色工作階段來執行動作，則來源身分資訊會出現在 CloudTrail 日誌的 userIdentity 索引鍵中。

Example AWS CloudTrail 日誌中的範例 userIdentity 金鑰

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```

若要查看 CloudTrail 日誌中的範例 AWS STS API 事件，請參閱 CloudTrail 日誌中的 IAM API 事件範例。如需有關 CloudTrail 日誌檔案中包含資訊的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 CloudTrail 事件參考。

GetFederationToken 許可

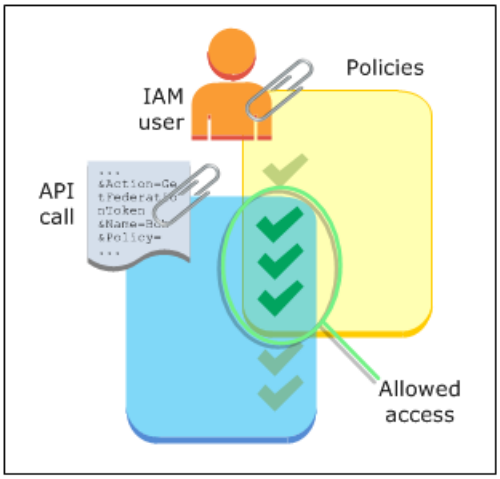
此 `GetFederationToken` 操作是由 IAM 使用者所呼叫並會傳回該使用者的臨時憑證。此操作會聯合使用者。指派給 AWS STS 聯合身分使用者工作階段的許可定義在兩個位置之一：

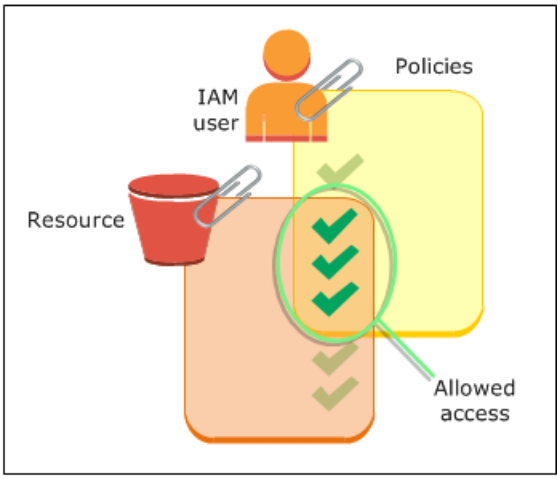
- 該工作階段政策會以 `GetFederationToken` API 呼叫的參數傳遞。(這是最常見的)。
- 以資源為基礎的政策，在政策的 `Principal`元素中明確命名 AWS STS 聯合身分使用者工作階段。(這是最少見的)。

工作階段政策是一種進階政策，且您會在以程式設計方式建立臨時工作階段時，以參數方式傳遞此政策。當您建立 AWS STS 聯合身分使用者工作階段並傳遞工作階段政策時，產生的工作階段許可是使用者身分型政策和工作階段政策的交集。您無法使用工作階段政策來授予超出即將聯合使用者之以身分為基礎政策所允許的許可。

在大部分情況下，如果您未通過 GetFederationToken API 呼叫的政策，則所產生的暫時安全憑證沒有許可。不過，以資源為基礎的政策可提供該工作階段的額外許可。您可以使用以資源為基礎的政策存取資源，該政策會將您的工作階段指定為允許的主體。

下圖顯示政策互動的視覺化呈現，以判斷對 GetFederationToken 呼叫傳回的暫時安全憑證的許可。





範例：使用 GetFederationToken 指派許可

您可以將 GetFederationToken API 動作與不同類型的政策一起使用。以下是幾個範例。

連接至 IAM 使用者的政策

在這個範例中，您有一個以瀏覽器為基礎的用戶端應用程式，它依賴於兩個後端的 Web 服務。一個後端服務是您自己的身分驗證伺服器，使用您自己的身分系統來驗證用戶端應用程式。另一個後端服務是 AWS 服務，可提供一些用戶端應用程式的功能。用戶端應用程式由您的伺服器進行身分驗證，您的伺服器會建立或擷取適當的許可政策。然後，您的伺服器呼叫 GetFederationToken API 以取得暫時安全憑證，並將這些憑證傳回給用戶端應用程式。然後，用戶端應用程式可以使用臨時安全登入資料直接向 AWS 服務提出請求。此架構可讓用戶端應用程式提出 AWS 請求，而無需內嵌長期 AWS 登入資料。

您的身分驗證伺服器使用名為 token-app 之 IAM 使用者的長期安全憑證來呼叫 GetFederationToken API。但長期 IAM 使用者憑證會保持在伺服器上且絕對不會發佈到用戶端。下列範例政策會連接至 token-app IAM 使用者，並定義聯合身分使用者 AWS STS （用戶端）所

需的最廣泛許可集。請注意，您的身分驗證服務需要 sts:GetFederationToken許可，才能取得 AWS STS 聯合身分使用者的臨時安全登入資料。

Example連接到呼叫 GetFederationToken 的 IAM 使用者 token-app 的政策範例

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
    }
  ]
}
```

上述政策會將多個許可授予 IAM 使用者。不過，此政策本身不會將任何許可授予 AWS STS 聯合身分使用者。如果此 IAM 使用者呼叫 GetFederationToken，且 未將政策作為 API AWS STS 呼叫的參數傳遞，則產生的聯合身分使用者沒有有效的許可。

工作階段政策作為參數傳遞

確保 AWS STS 聯合身分使用者獲指派適當許可的最常見方法是在 GetFederationToken API 呼叫中傳遞工作階段政策。擴展前面的範例，假設使用 IAM 使用者 token-app 憑證來呼叫 GetFederationToken。假設以下工作階段政策做為 API 呼叫的參數傳遞。所產生的 AWS STS 聯合身分使用者具有列出名為 productionapp 的 Amazon S3 儲存貯體內容的許可。該使用者無法對 productionapp 儲存貯體中的項目執行 Amazon S3 GetObject、PutObject 和 DeleteObject 動作。

聯合身分使用者會獲派這些許可，因為這些許可是 IAM 使用者政策和您傳遞之工作階段政策的交集。

AWS STS 聯合身分使用者無法在 Amazon SNS、Amazon SQS、Amazon DynamoDB 或任何 S3 儲存貯體中執行動作，但 除外productionapp。即使這些許可已授予給與 GetFederationToken 呼叫相關聯的 IAM 使用者，這些動作仍然會遭到拒絕。

Example工作階段政策作為 GetFederationToken API 呼叫的參數傳遞範例

JSON

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

資源型政策

有些 AWS 資源支援以資源為基礎的政策，而這些政策提供另一種機制，可將許可直接授予 AWS STS 聯合身分使用者。只有部分 AWS 服務支援以資源為基礎的政策。例如，Amazon S3 具有儲存貯體，Amazon SNS 具有主題，並且 Amazon SQS 具有可以連接政策的佇列。如需有關支援以資源為基礎的政策的所有服務的清單，請參閱 AWS 使用 IAM 的 服務 並查看表格中的「以資源為基礎的政策」資料列。您可以使用資源型政策，直接將許可指派給 AWS STS 聯合身分使用者。透過在資源型政策的 `Principal`元素中指定 AWS STS 聯合身分使用者的 Amazon Resource Name (ARN) 來執行此操作。以下範例使用名為 `productionapp` 的 S3 儲存貯體說明此範例並擴展前面的範例。

以下以資源為基礎的政策已連接到儲存貯體。此儲存貯體政策允許名為 Carol AWS STS 的聯合身分使用者存取儲存貯體。當前述的範例政策連接至 `token-app` IAM 使用者時，名為 Carol AWS STS 的聯合身分使用者具有許可，可在名為 的儲存貯體上執行 `s3:GetObject`、 `s3:PutObject`和 `s3:DeleteObject`動作`productionapp`。即使沒有工作階段政策做為 `GetFederationToken` API 呼叫的參數傳遞時，也是如此。這是因為在此案例中，名為 Carol AWS STS 的聯合身分使用者已由下列以資源為基礎的政策明確授予許可。

請記住，只有當 AWS STS 這些許可明確授予 IAM 使用者和聯合身分使用者時， AWS STS 才會授予聯合身分使用者許可。也可以由以資源為基礎的政策授予（在帳戶中），該政策在政策的 `Principal`元素中明確命名 AWS STS 聯合身分使用者，如下列範例所示。

Example允許存取聯合身分使用者的儲存貯體政策範例

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Principal": {
            "AWS": "arn:aws:sts::111122223333:federated-user/Carol"
        },
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3:DeleteObject"
        ],
        "Resource": [
            "arn:aws:s3:::productionapp/*"
        ]
```

```
    }
}
```

如需有關如何評估政策的詳細資訊，請參閱政策評估邏輯。

GetSessionToken 許可

呼叫 GetSessionToken API 操作或 get-session-token CLI 命令的主要時間是，當使用者必須透過多重要素驗證 (MFA) 進行身分驗證時。可以編寫只允許特定操作的一項政策，且只有在這些操作是由經過 MFA 身分驗證的使用者提出的請求時，才允許執行。為成功通過 MFA 身分驗證檢查，使用者必須先呼叫 GetSessionToken，並且包含可選的SerialNumber 和 TokenCode 參數。如果使用者成功通過 MFA 裝置身分驗證，GetSessionToken API 操作傳回的憑證包含 MFA 內容。此內容表示使用者通過 MFA 身分驗證，並且獲得 API 操作授權，其需要 MFA 身分驗證。

GetSessionToken 需要的許可

使用者要取得工作階段權杖並不需要許可。GetSessionToken 操作的目的是使用 MFA 驗證使用者的身分。您不能使用政策來控制驗證操作。

若要授予執行大部分 AWS 操作的許可，請將具有相同名稱的動作新增至政策。例如，若要建立使用者，您必須使用 CreateUser API 操作、create-user CLI 命令或 AWS 管理主控台。若要執行這些操作，您必須擁有一個政策，其可讓您存取 CreateUser 動作。

JSON

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

您可以在您的政策中包含 GetSessionToken 動作，但不會影響使用者執行 GetSessionToken 操作的能力。

GetSessionToken 授予的許可

如果使用 IAM 使用者的憑證呼叫 GetSessionToken，臨時安全性憑證與 IAM 使用者會有相同的許可。同樣地，如果 使用 AWS 帳戶根使用者 登入資料GetSessionToken呼叫 ，則暫時安全登入資料具有根使用者許可。

> **Note**
>
> 我們建議您不要使用根使用者憑證呼叫 GetSessionToken。相反的，請遵循我們的最佳實務，並建立具有所需許可的 IAM 使用者。然後，使用這些 IAM 使用者與 進行日常互動 AWS。

當您呼叫 GetSessionToken 時您獲得的臨時憑證有下列功能和限制：

- 您可以使用登入資料來存取 ， AWS 管理主控台 方法是將登入資料傳遞至位於 的聯合單一登入端點https://signin.aws.amazon.com/federation。如需詳細資訊，請參閱啟用 AWS 主控台的自訂身分代理程式存取。
- 您無法使用憑證來呼叫 IAM 或 AWS STS API 操作。您可以使用它們來呼叫其他服務的 API 操作 AWS 。

將此 API 操作和其限制及功能與在 比較 AWS STS 登入資料 建立臨時安全性憑證的 API 操作相較

如需有關使用 GetSessionToken 的受 MFA 保護之 API 存取的詳細資訊，請參閱 透過 MFA 實現安全的 API 存取。

停用臨時安全性憑證的許可

暫時安全憑證在過期之前持續有效。這些憑證在指定的持續時間內有效，從 900 秒 (15 分鐘) 至最長 129,600 秒 (36 小時)。預設工作階段持續時間為 43,200 秒 (12 小時)。您可以撤銷這些憑證，但也必須變更 IAM 使用者或角色的許可，以停止使用遭洩漏的憑證進行惡意帳戶活動。每次使用指派給臨時安全登入資料的許可來提出 AWS 請求時，都會進行評估。從登入資料中移除所有許可後，使用它們的 AWS 請求會失敗。

政策更新可能需要幾分鐘的時間才會生效。對於 IAM 角色工作階段，您可以撤銷角色的臨時安全憑證，以強制所有擔任該角色的使用者重新驗證並請求新的憑證。如需詳細資訊，請參閱撤銷角色的臨時安全憑證。

您無法變更 的許可 AWS 帳戶根使用者。同樣地，以根使用者身分登入時，您無法更改呼叫 GetFederationToken 或 GetSessionToken 建立的暫時安全憑證許可。因此，我們建議您不要以根使用者身分呼叫 GetFederationToken 或 GetSessionToken。

如需有關如何變更 IAM 使用者的許可的程序，請參閱變更 IAM 使用者的許可。

如需有關如何變更 IAM 角色的許可的程序，請參閱更新角色的許可。

> **Important**
>
> 您無法在 IAM 中編輯從 IAM Identity Center 許可集建立的角色。您必須撤銷 IAM Identity Center 中使用者的作用中許可集工作階段。如需詳細資訊，請參閱 IAM Identity Center User Guide 中的 Revoke active IAM role sessions created by permission sets。

主題

- 拒絕存取與角色關聯的所有 IAM 角色工作階段
- 拒絕存取特定 IAM 角色工作階段
- 拒絕存取具有條件內容索引鍵的臨時安全憑證工作階段
- 使用資源型政策拒絕存取特定主體

拒絕存取與角色關聯的所有 IAM 角色工作階段

此程序拒絕與角色關聯的所有 IAM 角色工作階段的許可。當您擔心被以下身分進行可疑存取時，請使用此方法：

- 使用跨帳戶存取權的其他帳戶中的主體
- 具有存取您帳戶中 AWS 資源許可的外部使用者身分
- 在行動或 Web 應用程式中，已透過 OIDC 提供者驗證的使用者

若要變更或移除透過呼叫 AssumeRole、AssumeRoleWithSAML、AssumeRoleWithWebIdentity、GetFederationToken 或 GetSessionToken 而取得的指派給臨時安全憑證的許可，您可以編輯或刪除為角色定義許可的身分型政策。

> **Important**
>
> 如果存在允許主體存取的以資源為基礎的政策，您還必須為該資源新增明確拒絕。如需詳細資訊，請參閱 使用資源型政策拒絕存取特定主體。

若要拒絕存取與角色關聯的所有 IAM 角色工作階段

1. 登入 AWS 管理主控台 並開啟 IAM 主控台。
2. 在導覽窗格中，選擇角色。
3. 選擇要編輯的角色名稱。您可以使用搜尋方塊來篩選清單。
4. 選擇許可索引標籤。
5. 選取要編輯的相關政策。在編輯客戶管理政策之前，檢閱連接的實體索引標籤，以避免中斷對可能已連接相同政策的其他身分的存取。
6. 選擇 JSON 標籤並更新政策以拒絕所有資源和動作。

> **Note**
>
> 這些許可與 AWS 受管政策 AWSDenyAll 中的許可相同。您可以將此 AWS 受管政策連接至您想要拒絕所有存取的任何 IAM 使用者或角色。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAll",
            "Effect": "Deny",
            "Action": [
                "*"
            ],
            "Resource": "*"
        }
    ]
}
```

7. 在 Review (檢視) 頁面上，查看政策 Summary (摘要)，然後選擇 Save changes (儲存政策) 以儲存您的工作。

在更新政策時，所做出的變更將影響與該角色關聯的所有暫時安全憑證的許可，包括在更改該角色的許可政策之前發行的憑證。

更新政策之後，您可以撤銷角色的暫時安全憑證以立即撤銷角色所發行憑證的所有許可。

拒絕存取特定 IAM 角色工作階段

當您使用全部拒絕政策更新 IAM 角色或完全刪除該角色時，有權存取該角色的所有使用者都會中斷。您可以拒絕存取，而不會影響與該角色關聯的所有其他工作階段的許可。

您可以使用條件內容索引鍵或以資源為基礎的政策拒絕 `Principal` 的許可。

> **Tip**
>
> 您可以使用 AWS CloudTrail 日誌來尋找聯合身分使用者的 ARNs。如需詳細資訊，請參閱如何使用 輕鬆識別您的聯合身分使用者 AWS CloudTrail。

拒絕存取具有條件內容索引鍵的臨時安全憑證工作階段

您可以在想要拒絕存取特定臨時安全憑證工作階段的情況下使用身分型政策中的條件內容索引鍵，而不影響建立憑證的 IAM 使用者或角色的許可。對於 IAM 角色，在更新政策之後，您還可以撤銷角色的臨時安全憑證工作階段，以立即撤銷所有發行的憑證。

如需條件內容索引鍵的詳細資訊，請參閱 AWS 全域條件內容索引鍵。

aws:PrincipalArn

您可以在身分型政策中使用條件內容索引鍵 aws:PrincipalArn，依 Amazon Resource Name (ARN) 拒絕存取特定主體。您可以透過在政策的條件元素中指定暫時安全登入資料與之關聯的 AWS STS IAM 使用者、角色或聯合身分使用者工作階段的 ARN 來執行此操作。

若要依 ARN 拒絕存取特定主體

1. 在 IAM 主控台導覽窗格中，選擇使用者或角色。
2. 選擇要編輯的 IAM 使用者或角色的名稱。您可以使用搜尋方塊來篩選清單。
3. 選擇許可索引標籤。

4. 選取要編輯的相關政策。在編輯客戶管理政策之前，檢閱連接的實體索引標籤，以避免中斷對可能已連接相同政策的其他身分的存取。
5. 選擇 JSON 標籤並為主體 ARN 新增拒絕陳述式，如下列範例所示。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:iam::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
  ]
}
```

6. 在 Review (檢視) 頁面上，查看政策 Summary (摘要)，然後選擇 Save changes (儲存政策) 以儲存您的工作。

aws:SourceIdentity

您可以在身分型政策中使用條件內容索引鍵 aws:SourceIdentity，拒絕存取與 IAM 角色工作階段關聯的特定來源身分。只要角色工作階段是在委託人使用任何 AWS STS `assume-role*` CLI 命令或 AWS STS `AssumeRole*` API 操作擔任角色時，透過設定`SourceIdentity`請求參數發出，就會套用。您可以透過在政策的 `Condition` 元素中指定臨時安全憑證關聯的來源身分來執行此操作。

與內容索引鍵 sts:RoleSessionName 不同，在設定來源身分之後，無法變更值。`aws:SourceIdentity` 索引鍵會出現在由角色所採取的所有動作的請求內容中。當您使用工作階段憑證來擔任另一個角色時，來源身分仍然存在於後續角色工作階段。從另一個角色取得及擔任角色稱為角色鏈結。

下列政策顯示如何使用條件內容索引鍵 aws:SourceIdentity 拒絕存取暫時安全憑證工作階段的範例。如果您指定與角色工作階段關聯的來源身分，則它會拒絕含有具名來源身分的角色工作階段，而不會影響建立憑證的角色的許可。在此範例中，發出角色工作階段時，主體設定的來源身分為 nikki_wolf@example.com。具有來源身分 nikki_wolf@example.com 的角色工作階段發出的任何請求都會遭拒，因為來源身分包含在政策條件中，且政策效果設定為 Deny。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": [
            "nikki_wolf@example.com",
            "<source identity value>"
          ]
        }
      }
    }
  ]
}
```

aws:userid

您可以在身分型政策中使用條件內容索引鍵 aws:userid 拒絕存取與 IAM 使用者或角色關聯的所有或特定臨時安全憑證工作階段。您可以透過在政策的 Condition元素中指定暫時安全登入資料相關聯的 IAM AWS STS 使用者、角色或聯合身分使用者工作階段的唯一識別符 (ID) 來執行此操作。

下列政策顯示如何使用條件內容索引鍵 aws:userid 拒絕存取暫時安全憑證工作階段的範例。

- AIDAXUSER1 代表 IAM 使用者的唯一 ID。指定 IAM 使用者的唯一 ID 作為內容索引鍵 aws:userid 的值將拒絕存取 IAM 使用者。其中包括透過呼叫 GetSessionToken API 建立的任何臨時安全憑證工作階段。

- AROAXROLE1:* 代表與 IAM 角色關聯的所有工作階段的唯一 ID。將 IAM 角色的唯一 ID 和 caller-specified-role-session-name 部分中的萬用字元 (*) 指定為內容索引鍵 aws:userid 的值將拒絕與該角色關聯的所有工作階段。
- AROAXROLE2:<caller-specified-role-session-name> 代表擔任角色工作階段的唯一 ID。在擔任角色唯一 ID 的 caller-specified-role-session-name 部分中，如果使用 StringLike 條件運算子，您可以指定角色工作階段名稱或萬用字元。如果您指定角色工作階段名稱，則它會拒絕具名角色工作階段，而不會影響建立憑證的角色的許可。如果您為角色工作階段名稱指定萬用字元，則它會拒絕與角色關聯的所有工作階段。

> **Note**
>
> 呼叫者指定的角色工作階段名稱是擔任角色工作階段的唯一識別碼的一部分，可能會在角色鏈結期間變更。當一個角色擔任另一個角色時，會發生角色鏈結。當委託人使用 API 操作擔任角色時，會使用 AWS STS AssumeRole RoleSessionName請求參數設定角色工作階段名稱。

- account-id:<federated-user-caller-specified-name> 代表 AWS STS 聯合身分使用者工作階段的唯一 ID。IAM 使用者透過呼叫 GetFederationToken API 來建立此工作階段。為 AWS STS 聯合身分使用者工作階段指定唯一 ID 會拒絕具名聯合身分工作階段，而不會影響建立憑證的 IAM 使用者的許可。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1:*",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "123456789012:<federated-user-caller-specified-name>"
          ]
```

```
        }
      }
    }
  ]
}
```

如需主體索引鍵值的特定範例，請參閱 主體索引鍵值。如需有關 IAM 唯一識別碼以及如何取得它們的資訊，請參閱唯一識別碼。

使用資源型政策拒絕存取特定主體

若要使用資源型政策限制對特定主體的存取，您可以使用 Condition 元素中的條件內容索引鍵 aws:PrincipalArn 或 aws:SourceIdentity。資源型政策是連接至資源的許可政策，可控制誰可以存取資源，以及他們可以對其執行哪些動作。

當您使用aws:PrincipalARN內容金鑰時，請在政策的條件元素中指定與暫時安全登入資料相關聯的 IAM AWS STS 使用者、角色或聯合身分使用者工作階段的 ARN。下列範例政策示範如何在資源型政策中使用 aws:PrincipalARN 內容索引鍵：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "ArnEquals": {
        "aws:PrincipalArn": [
          "arn:aws:iam::222222222222:role/ROLENAME",
          "arn:aws:iam::222222222222:user/USERNAME",
          "arn:aws:sts::222222222222:federated-user/USERNAME"
        ]
      }
    }
  }
}
```

當您使用 aws:SourceIdentity 內容索引鍵時，在政策的 Condition 元素中指定與角色的臨時安全憑證關聯的來源身分值。只要角色工作階段是在委託人使用任何 AWS STS assume-role* CLI 命令或 AWS STS AssumeRole* API 操作擔任角色時，透過設定SourceIdentity請求參數發出，就會套用。下列範例示範如何在資源型政策中使用 aws:SourceIdentity 內容索引鍵：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "StringLike": {
        "aws:SourceIdentity": [
          "nikki_wolf@example.com",
          "<source identity value>"
        ]
      }
    }
  }
}
```

如果您僅更新主體的身分型政策，其仍然可以執行資源型政策中允許的動作，除非在身分型政策中明確拒絕這些動作。

若要在資源型政策中拒絕存取特定主體

1. 請參閱 AWS 使用 IAM 的 服務 以查看服務是否支援以資源為基礎的政策。
2. 登入 AWS 管理主控台 並開啟 服務的 主控台。每個服務在主控台中都有不同的位置用於附加政策。
3. 編輯資源型政策。新增拒絕政策陳述式以指定憑證的識別資訊：
   a. 在 Principal 元素中，輸入萬用字元 (*)。在 Condition 元素中，主體將受到限制。
   b. 在 Effect 元素中，輸入 "Deny"。

c. 在 Action 中，輸入服務命名空間和要拒絕的動作名稱。若要拒絕所有動作，請使用萬用字元 (*)。例如："s3:*"。

d. 在 Resource 元素中，輸入目標資源的 ARN。例如："arn:aws:s3:::amzn-s3-demo-bucket"。

e. 在 Condition 元素中，指定 aws:PrincipalARN 或 aws:SourceIdentity 內容索引鍵。

如果您使用 aws:PrincipalARN 內容索引鍵，請輸入要拒絕其存取之主體的 ARN。

如果您使用 aws:SourceIdentity 內容索引鍵，請在角色工作階段中輸入要拒絕存取的來源身分值。

4. 儲存您的工作。

授予許可以建立暫時性安全憑證

依預設，IAM 使用者沒有為 AWS STS 聯合身分使用者工作階段和角色建立臨時安全憑證的許可。您必須使用政策提供您的使用者這些許可。雖然您可以直接授予許可給使用者，我們強烈建議您將許可授予群組。這可讓許可的管理更輕鬆。當有人不再需要執行與許可相關的任務時，您只需從群組中移除許可。如果其別人需要執行該任務，將它們新增到群組以授予許可。

若要授予 IAM 群組為 AWS STS 聯合身分使用者工作階段或角色建立臨時安全憑證的許可，您可以連接能夠授予以下一個或兩個權限的政策：

- 若要讓 OIDC 和 SAML 聯合主體存取 IAM 角色，請將存取權授予 AWS STS AssumeRole。
- 對於不需要角色的 AWS STS 聯合身分使用者，將存取權授予 AWS STS GetFederationToken。

如需有關 AssumeRole 和 GetFederationToken API 操作間差異的詳細資訊，請參閱 請求臨時安全憑證。

IAM 使用者也可以呼叫 GetSessionToken 建立臨時安全憑證。使用者呼叫 GetSessionToken 不需要許可。此操作的目的是使用 MFA 驗證使用者的身分。您不能使用政策來控制身分驗證。這表示您無法避免 IAM 使用者呼叫 GetSessionToken 來建立臨時憑證。

Example授予許可以擔任角色的政策範例

下列範例政策授予 呼叫 中UpdateApp角色AssumeRole的許可 AWS 帳戶 123123123123。當使用 AssumeRole 時，代表聯合身分使用者建立安全憑證的使用者 (或應用程式)，無法委派角色許可政策中未指定的許可。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example授予許可以建立聯合身分使用者的暫時性安全憑證的政策範例

下列範例政策會授予許可以存取 `GetFederationToken`。

JSON

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

Important

當您授予 IAM 使用者許可，讓他們能夠透過 `GetFederationToken` 為 AWS STS 聯合身分使用者建立臨時安全憑證時，請留意，這會允許這些使用者委派自己的許可。如需在 IAM 使用者和 之間委派許可的詳細資訊 AWS 帳戶，請參閱 委派存取權限的政策範例。如需有關如何控制暫時性安全憑證的許可的詳細資訊，請參閱臨時安全憑證的許可。

Example授予使用者有限許可以建立聯合身分使用者的暫時性安全憑證的政策範例

當您讓 IAM 使用者呼叫 GetFederationToken，最佳實務是限制 IAM 使用者可以委派的許可。例如，下列政策顯示如何讓 IAM 使用者僅為名稱開頭為 Manager AWS STS 的聯合身分使用者建立臨時安全登入資料。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

授予許可以使用身分增強型主控台工作階段

身分增強主控台工作階段可在 AWS IAM Identity Center 使用者登入時將使用者和工作階段 IDs AWS 包含在使用者的主控台工作階段中。例如，Amazon Q Developer Pro 利用身分增強型主控台工作階段來個人化服務體驗。如需有關身分增強型主控台工作階段的詳細資訊，請參閱 AWS IAM Identity Center User Guide 中的 Enabling identity-enhanced console sessions。如需有關 Amazon Q Developer 設定的資訊，請參閱 Amazon Q Developer User Guide 中的 Setting up Amazon Q Developer。

為了讓身分增強型主控台工作階段可供使用者使用，您必須使用身分型政策向 IAM 主體授予代表其自己的主控台工作階段的資源的 sts:SetContext 許可。

> **Important**
>
> 依預設，使用者無權為身分增強型主控台工作階段設定內容。為此，您必須在身分型政策中向 IAM 主體授予 sts:SetContext 許可，如下面的政策範例所示。

下列以身分為基礎的政策範例會將 sts:SetContext許可授予 IAM 主體，允許主體為自己的主控台工作階段設定身分增強型 AWS 主控台工作階段內容。政策資源 arn:aws:sts::*account-id*:self代表呼叫者的 AWS 工作階段。如果跨多個帳戶部署相同的許可政策 (例如使用 IAM Identity Center 許可集部署此政策)，則可以將 account-id ARN 區段取代為萬用字元 *。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:SetContext",
            "Resource": "arn:aws:sts::111122223333:self"
        }
    ]
}
```

在 AWS STS 中管理 AWS 區域

區域端點是 Web 服務特定區域內進入點的 AWS URL。 AWS 建議使用 Regional AWS Security Token Service (AWS STS) 端點而非全域端點，以減少延遲、建置備援並提高工作階段字符有效性。

雖然全域 （舊版） AWS STS 端點`https://sts.amazonaws.com`高度可用，但它託管在單一 AWS 區域、美國東部 （維吉尼亞北部） 和其他端點，但它不提供自動容錯移轉到其他區域中的端點。

- 降低延遲 – 透過 AWS STS 呼叫地理位置上更接近服務和應用程式的端點，您可以存取具有更低延遲和更好回應時間 AWS STS 的服務。
- 內建備援 – 您可以將工作負載內失敗的影響限制為有限數量的元件，且具有可預測的影響限制範圍。使用區域 AWS STS 端點可讓您將元件的範圍與工作階段字符的範圍保持一致。如需有關此可靠性支柱的詳細資訊，請參閱 AWS Well-Architected Framework 中的使用故障隔離來保護工作負載。
- 提高工作階段字符有效性 – 區域 AWS STS 端點的工作階段字符在所有 中都有效 AWS 區域。來自全域 STS 端點的工作階段權杖僅在預設啟用 AWS 區域 的 中有效。如果您想要為帳戶啟用新的區域，您可以從區域 AWS STS 端點使用工作階段字符。如果您選擇使用全域端點，則必須變更全域端點之 AWS STS 工作階段字符的區域相容性。這樣做可確保字符在所有 中都有效 AWS 區域。

如需 AWS STS 區域及其端點的清單，請參閱 AWS STS 區域和端點。

> **Note**
>
> AWS 已對依預設啟用的區域中的 AWS Security Token Service (AWS STS) 全域端點 (`https://sts.amazonaws.com`) 進行變更，以增強其彈性和效能。對全域端點的 AWS STS 請求會在與您的工作負載 AWS 區域 相同的 中自動提供。這些變更不會部署至選擇加入區域。我們建議您使用適當的 AWS STS 區域端點。如需詳細資訊，請參閱AWS STS 全域端點變更。

主題

- 在 AWS STS 中啟用和停用 AWS 區域
- 編寫程式碼以使用 AWS STS 區域
- 管理全域端點的工作階段權杖

在 AWS STS 中啟用和停用 AWS 區域

當您為區域啟用 AWS STS 端點時， AWS STS 可以向帳戶中提出 AWS STS 請求的使用者和角色發出臨時登入資料。然後，您就可以在預設啟用或手動啟用的任何區域中使用這些憑證。

對於預設啟用的區域，您必須在產生臨時憑證的帳戶中啟用區域 AWS STS 端點。提出請求時使用者登入的是相同帳戶或不同帳戶並不重要。

AWS 帳戶 使用手動啟用的區域請求另一個 中角色的臨時登入資料時，目標帳戶 （包含該角色的帳戶） 必須啟用該區域以進行 AWS STS 操作。這樣可以確保正確地產生臨時安全憑證。

例如，假設帳戶 A 中的使用者想要向 AWS STS 區域端點 `https://sts.ap-southeast-3.amazonaws.com` 傳送 `sts:AssumeRole` API 請求。該請求的目的是為帳戶 B 中名為 `Developer` 的角色取得臨時憑證。由於請求的目的是為帳戶 B 中的實體建立憑證，因此帳戶 B 必須啟用 `ap-southeast-3` 區域。來自帳戶A (或任何其他帳戶) 的使用者可以呼叫 `ap-southeast-3` AWS STS 端點以請求帳戶 B 的憑證，無論該帳戶中是否已啟用該區域。若要進一步了解，請參閱AWS 區域 在您的帳戶中啟用或停用 。

> **Note**
>
> 在該帳戶中使用臨時性憑證的每個人都可使用作用中的區域。若要控制哪些 IAM 使用者或角色可以存取區域，請在許可政策中使用 `aws:RequestedRegion` 條件金鑰。

在預設啟用 AWS STS 的區域中啟用或停用 （主控台）

1. 以具有 IAM 管理任務執行許可的根使用者或使用者身分來登入。
2. 開啟 IAM 主控台，然後在導覽窗格中選擇 Account settings (帳戶設定)。
3. 在 Security Token Service (STS) 的端點區段中，找到要設定的區域，然後在 STS 狀態資料欄中選擇作用中或非作用中。
4. 在開啟的對話方塊中，選擇 Activate (啟用) 或 Deactivate (停用)。

對於必須啟用的區域，我們會在您啟用區域時 AWS STS 自動啟用 。啟用區域之後， AWS STS 一律對該區域處於作用中狀態，且您無法將其停用。若要了解如何啟用預設停用的區域，請參閱《 AWS 帳戶管理 參考指南》中的指定 AWS 區域 您的帳戶可以使用哪些區域。

編寫程式碼以使用 AWS STS 區域

啟用區域之後，您可以將 AWS STS API 呼叫導向該區域。以下 Java 程式碼片段示範如何設定 `AWSSecurityTokenService` 物件，以對歐洲 (米蘭) (eu-south-1) 區域提出請求。

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
south-1.amazonaws.com", "eu-south-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STS 建議您呼叫區域端點。若要了解如何手動啟用區域，請參閱 AWS 帳戶管理 Reference Guide 中的Specify which AWS 區域 your account can use。

在這個範例中，第一行會執行個體化稱為 `regionEndpointConfig` 的 `EndpointConfiguration` 物件，並將端點和 AWS 區域 的 URL 做為參數傳遞。

若要了解如何使用 SDKs AWS 的環境變數設定 AWS STS 區域端點，請參閱 AWS SDKs和工具參考指南中的AWS STS 區域化端點。

對於所有其他語言和程式設計環境組合，請參閱相關軟體開發套件的文件。

管理全域端點的工作階段權杖

根據 AWS 服務 預設 AWS 區域 ，大多數都已啟用所有 中的操作。這些區域會自動啟用以搭配 使用 AWS STS。亞太區域 (香港) 之類的某些區域必須手動啟用。若要了解如何啟用和停用 AWS 區域，請參閱 AWS 帳戶管理 Reference Guide 中的 Specify which AWS 區域 your account can use。

當您啟用這些 AWS 區域時，它們會自動啟用以搭配 使用 AWS STS。您無法為已停用的區域啟用 AWS STS 端點。在所有 中有效的工作階段字符 AWS 區域 包含的字元多於在預設啟用的區域中有效的字符。變更此設定可能會影響暫時存放權杖的現有系統。

您可以使用 AWS 管理主控台 AWS CLI或 AWS API 變更此設定。

變更全域端點工作階段權杖的區域相容性 (主控台)

1. 以具有 IAM 管理任務執行許可的根使用者或使用者身分來登入。若要變更工作階段權杖的相容性，您必須擁有允許 `iam:SetSecurityTokenServicePreferences` 動作的政策。
2. 開啟 IAM 主控台。在導覽窗格中，選擇 Account settings (帳戶設定)。
3. 在 Security Token Service (STS) (安全權杖服務 (STS)) 區段的 Session Tokens from the STS endpoints (來自 STS 端點的工作階段權杖) 下方。Global endpoint (全域端點) 表示 `Valid only in AWS ## enabled by default`。選擇 Change (變更)。
4. 在變更區域相容性對話方塊中，選取全部 AWS 區域。接著選擇 Save changes (儲存變更)。

> **Note**
>
> 在所有 中有效的工作階段字符 AWS 區域 包含的字元多於在預設啟用的區域中有效的字符。變更此設定可能會影響暫時存放權杖的現有系統。

變更全域端點工作階段權杖的區域相容性 (AWS CLI)

設定工作階段字符版本。第 1 版字符僅在預設 AWS 區域 可用的 中有效。這些權杖不適用於手動啟用的區域，例如亞太區域 (香港)。版本 2 權杖在所有區域都有效。不過，版本 2 權杖包含較多字元且可能會影響暫時存放權杖的系統。

- `aws iam set-security-token-service-preferences`

變更全域端點 (AWS API) 工作階段字符的區域相容性

設定工作階段字符版本。第 1 版字符僅在預設 AWS 區域 可用的 中有效。這些權杖不適用於手動啟用的區域，例如亞太區域 (香港)。版本 2 權杖在所有區域都有效。不過，版本 2 權杖包含較多字元且可能會影響暫時存放權杖的系統。

- SetSecurityTokenServicePreferences

AWS STS 區域和端點

> **Note**
>
> AWS 已對依預設啟用的區域中的 AWS Security Token Service (AWS STS) 全域端點 (`https://sts.amazonaws.com`) 進行變更，以增強其彈性和效能。對全域端點的 AWS STS 請求會在與您的工作負載 AWS 區域 相同的 中自動提供。這些變更不會部署至選擇加入區域。我們建議您使用適當的 AWS STS 區域端點。如需詳細資訊，請參閱AWS STS 全域端點變更。

下表列出區域及其端點。它會指出哪些是預設啟用，以及哪些可以啟用或停用。

區域名稱	端點	預設為作用中	手動啟動/停用
--全球服務--	sts.amazonaws.com	✓ 是	✗ 否
美國東部 (俄亥俄)	sts.us-east-2.amazonaws.com	✓ 是	✓ 是
美國東部 (維吉尼亞北部)	sts.us-east-1.amazonaws.com	✓ 是	✗ 否
美國西部 (加利佛尼亞北部)	sts.us-west-1.amazonaws.com	✓ 是	✓ 是
美國西部 (奧勒岡)	sts.us-west-2.amazonaws.com	✓ 是	✓ 是
非洲 (開普敦)	sts.af-south-1.amazonaws.com	✗ 否[1]	✗ 否
亞太地區 (香港)	sts.ap-east-1.amazonaws.com	✗ 否[1]	✗ 否
亞太地區 (海德拉巴)	sts.ap-south-2.amazonaws.com	✗ 否[1]	✗ 否

區域名稱	端點	預設為作用中	手動啟動/停用
亞太地區 (雅加達)	sts.ap-southeast-3.amazonaws.com	⊗ 否[1]	⊗ 否
亞太地區 (馬來西亞)	sts.ap-southeast-5.amazonaws.com	⊗ 否[1]	⊗ 否
亞太地區 (墨爾本)	sts.ap-southeast-4.amazonaws.com	⊗ 否[1]	⊗ 否
亞太地區 (孟買)	sts.ap-south-1.amazonaws.com	✓ 是	✓ 是
亞太區域 (大阪)	sts.ap-northeast-3.amazonaws.com	✓ 是	✓ 是
亞太地區 (首爾)	sts.ap-northeast-2.amazonaws.com	✓ 是	✓ 是
亞太地區 (新加坡)	sts.ap-southeast-1.amazonaws.com	✓ 是	✓ 是
亞太地區 (悉尼)	sts.ap-southeast-2.amazonaws.com	✓ 是	✓ 是
亞太區域 (泰國)	sts.ap-southeast-7.amazonaws.com	⊗ 否[1]	⊗ 否
亞太地區 (東京)	sts.ap-northeast-1.amazonaws.com	✓ 是	✓ 是
加拿大 (中部)	sts.ca-central-1.amazonaws.com	✓ 是	✓ 是
加拿大西部 (卡加利)	sts.ca-west-1.amazonaws.com	⊗ 否[1]	⊗ 否
中國 (北京)	sts.cn-north-1.amazonaws.com.cn	✓ 是[2]	⊗ 否
中國 (寧夏)	sts.cn-northwest-1.amazonaws.com.cn	✓ 是[2]	✓ 是
歐洲 (法蘭克福)	sts.eu-central-1.amazonaws.com	✓ 是	✓ 是
歐洲 (愛爾蘭)	sts.eu-west-1.amazonaws.com	✓ 是	✓ 是
歐洲 (倫敦)	sts.eu-west-2.amazonaws.com	✓ 是	✓ 是
歐洲 (米蘭)	sts.eu-south-1.amazonaws.com	⊗ 否[1]	⊗ 否
Europe (Paris)	sts.eu-west-3.amazonaws.com	✓ 是	✓ 是
歐洲 (西班牙)	sts.eu-south-2.amazonaws.com	⊗ 否[1]	⊗ 否

區域名稱	端點	預設為作用中	手動啟動/停用
歐洲 (斯德哥爾摩)	sts.eu-north-1.amazonaws.com	✓ 是	✓ 是
歐洲 (蘇黎世)	sts.eu-central-2.amazonaws.com	✗ 否[1]	✗ 否
以色列 (特拉維夫)	sts.il-central-1.amazonaws.com	✗ 否[1]	✗ 否
墨西哥 (中部)	sts.mx-central-1.amazonaws.com	✗ 否[1]	✗ 否
Middle East (Bahrain)	sts.me-south-1.amazonaws.com	✗ 否[1]	✗ 否
中東 (阿拉伯聯合大公國)	sts.me-central-1.amazonaws.com	✗ 否[1]	✗ 否
南美洲 (聖保羅)	sts.sa-east-1.amazonaws.com	✓ 是	✓ 是

[1] 您必須啟用區域才能使用它。這會自動啟動 AWS STS。您無法手動啟動或停用這些區域中的 AWS STS 。

2若要 AWS 在中國使用 ，您需要一個 帳戶和 登入資料，中國 AWS 為 。

AWS STS 全域端點變更

AWS 已對依預設啟用的區域中的 AWS Security Token Service (AWS STS) 全域端點 (`https://sts.amazonaws.com`) 進行變更，以增強其彈性和效能。先前，對 AWS STS 全球端點的所有請求都由單一 AWS 區域美國東部 （維吉尼亞北部） 提供服務。現在，在預設啟用的區域中， AWS STS 全域端點的請求會在請求來源的相同區域中自動提供，而不是美國東部 （維吉尼亞北部） 區域。這些變更不會部署至選擇加入區域。

透過此變更， AWS STS 會根據使用的原始區域和 DNS 解析程式處理您的請求。如果 AWS STS 全域端點的 DNS 請求是由預設啟用區域中的 Amazon DNS 伺服器處理，則 AWS STS 全域端點的請求會在與 AWS 部署工作負載相同的區域中提供。如果請求來自選擇加入區域，或者請求是使用 DNS 解析程式而不是 Amazon DNS 伺服器進行解析，則向 AWS STS 全域端點發出的請求將繼續在美國東部 (維吉尼亞北部) 區域處理。如需有關 Amazon DNS 的詳細資訊，請參閱 Amazon Virtual Private Cloud User Guide 中的 Amazon DNS server。

下表顯示 全域 AWS STS 端點的請求如何根據您的 DNS 供應商路由。

DNS 解析程式	路由到本機 AWS STS 的全域端點請求 AWS 區域？
預設啟用區域中 Amazon VPC 中的 Amazon DNS 解析程式	是
選擇加入區域中 Amazon VPC 中的 Amazon DNS 解析程式	否，請求將路由至美國東部 (維吉尼亞北部) 區域
由 ISP、公有 DNS 供應商或任何其他 DNS 供應商提供的 DNS 解析程式	否，請求將路由至美國東部 (維吉尼亞北部) 區域

為了確保對現有程序的干擾降到最低， AWS 已實作下列措施：

- AWS CloudTrail 向 AWS STS 全域端點發出的請求日誌會傳送至美國東部 （維吉尼亞北部） 區域。AWS STS 區域端點所提供請求的 CloudTrail 日誌將繼續記錄在 CloudTrail 中的其各自區域。
- AWS STS 全域端點和區域端點所執行操作的 CloudTrail 日誌具有額外的欄位`awsServingRegion`，`endpointType`並指出哪個端點和區域提供請求。如需 CloudTrail 日誌範例，請參閱CloudTrail 日誌檔案中使用全域端點的範例 AWS STS API 事件。
- 無論向哪個區域提供請求，對 AWS STS 全域端點提出`us-east-1`的請求對於`aws:RequestedRegion`條件索引鍵的值為 。
- AWS STS 全域端點處理的請求不會與區域 AWS STS 端點共用每秒請求配額。

如果您在選擇加入區域中有工作負載，但仍使用 AWS STS 全域端點，建議您遷移至 AWS STS 區域端點，以提高彈性和效能。如需設定區域 AWS STS 端點的詳細資訊，請參閱 AWS SDKs和工具參考指南中的AWS STS 區域端點。

AWS CloudTrail 和區域端點

對區域和全域端點的呼叫會記錄在 AWS CloudTrail 的 `tlsDetails` 欄位中。對區域端點 (例如 `us-east-2.amazonaws.com`) 的呼叫會記錄在 CloudTrail 中適當的區域內。對此全球端點 `sts.amazonaws.com` 的呼叫都會記錄為對全球服務的呼叫。全域 AWS STS 端點的事件會記錄到 us-east-1。

> **Note**
>
> 只有支援此欄位的服務可檢視 `tlsDetails`。請參閱 AWS CloudTrail 使用者指南中的 CloudTrail 中支援 TLS 詳細資訊的服務
> 如需詳細資訊，請參閱使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail。

啟用 AWS 主控台的自訂身分代理程式存取

您可以運用編寫和執行程式碼來建立 URL 以使登入到您組織網路的使用者能夠安全存取 AWS 管理主控台。URL 包含您從 取得 AWS 並驗證使用者的登入字符 AWS。產生的主控台工作階段可能因為聯合而包含不同的 `AccessKeyId`。若要透過相關 CloudTrail 事件追蹤聯合登入的存取金鑰使用情況，請參閱 使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail 和 AWS 管理主控台 登入事件。

> **Note**
>
> 如果您的組織使用與 SAML 相容的身分提供者 (IdP)，則無需編寫程式碼即可設定對主控台的存取許可。這適用於 Microsoft 的 Active Directory 聯合身分驗證服務或開放原始碼 Shibboleth 等提供者。如需詳細資訊，請參閱啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。

若要讓組織的使用者能夠存取 AWS 管理主控台，您可以建立執行下列步驟的自訂身分代理程式：

1. 確認您的本機身分系統已對使用者進行身分驗證。
2. 呼叫 AWS Security Token Service (AWS STS) AssumeRole （建議） 或 GetFederationToken API 操作，以取得使用者的臨時安全登入資料。要了解在擔任角色時使用的各種方法，請參閱 擔任角色的方法。若要了解如何在取得您安全憑證時傳遞選用工作階段標籤，請參閱 在 中傳遞工作階段標籤 AWS STS。
   - 如果使用某個 `AssumeRole*` API 操作取得角色的臨時安全性憑證，您可以在呼叫中包含 `DurationSeconds` 參數。此參數指定 900 秒 (15 分鐘) 到角色的最大工作階段持續時間設定之間的角色工作階段持續時間。當您在 `AssumeRole*` 操作中使用 `DurationSeconds` 時，您必須稱之為具有長期憑證的 IAM 使用者。否則，在步驟 3 中對聯合端點的呼叫將失敗。如要了解如何查看或更改角色的最大值，請參閱 更新角色的最大工作階段持續時間。
   - 如果使用某個 `GetFederationToken` API 操作取得憑證，您可以在呼叫中包含 `DurationSeconds` 參數。此參數指定您的角色工作階段的持續時間。該值的範圍是 900 秒 (15 分鐘) 到 129,600 秒 (36 小時)。您只能使用 IAM 使用者的長期 AWS 安全登入資料來進行此 API 呼叫。您也可以使用 AWS 帳戶根使用者 登入資料進行這些呼叫，但我們不建議這麼做。如果您

以根使用者身分執行呼叫，則工作階段預設將持續 1 小時。或者，您可以指定 900 秒 (15 分鐘) 到 3,600 秒 (1 小時) 之間的工作階段。

3. 呼叫 AWS 聯合端點並提供臨時安全登入資料來請求登入字符。
4. 建構包含該權杖的主控台 URL：
   - 如果在 URL 中使用某個 AssumeRole* API 操作，您可以包含 SessionDuration HTTP 參數。此參數指定主控台工作階段持續時間，範圍是 900 秒 (15 分鐘) 到 43200 秒 (12 小時)。
   - 如果在 URL 中使用 GetFederationToken API 操作，您可以包含 DurationSeconds 參數。此參數指定您的聯合身分主控台工作階段的持續時間。該值的範圍是 900 秒 (15 分鐘) 到 129,600 秒 (36 小時)。

   > **Note**
   > - 您的 SessionDuration 不能大於或等於您擔任之角色的工作階段持續時間上限設定。例如，您已將要擔任之角色的工作階段持續時間上限設定為 5 小時。您的 SessionDuration 參數可以是 16,524 秒或 4 小時 59 秒。
   > - 如果透過 GetFederationToken 取得臨時憑證，請勿使用 SessionDuration HTTP 參數。否則，操作會失敗。
   > - 使用一個角色的憑證來擔任不同的角色稱為角色鏈結。當您使用角色鏈結時，新憑證的最大持續時間限制為一小時。使用角色授予許可給在 EC2 執行個體上執行的應用程式時，那些應用程式不受此限制。
   > - 如果透過角色鏈結取得臨時憑證，請勿使用 SessionDuration HTTP 參數。否則，操作會失敗。
5. 將 URL 分配給使用者或代表使用者呼叫 URL。

聯合端點提供的 URL 在建立後的 15 分鐘內有效。這與 URL 關聯的臨時安全性憑證工作階段的持續時間 (以秒為單位) 不同。這些憑證在建立時指定的持續時間內有效，從建立的時間算起。

> **Important**
> AWS 管理主控台 如果您已在關聯的暫時安全登入資料中啟用許可，URL 會透過 授予 AWS 資源的存取權。因此，您應該將 URL 視為機密。我們建議您透過安全的重新引導傳回 URL，例如，在 SSL 連接上使用 302 HTTP 回應狀態碼。如需 302 HTTP 回應狀態碼的詳細資訊，請參閱 RFC 2616 第 10.3.3 節。

若要完成這些任務，您可以使用適用於 AWS Identity and Access Management (IAM) 的 HTTPS Query API 和 AWS Security Token Service (AWS STS)。或者，您可以使用程式設計語言 (例如 Java、Ruby 或 C#) 與適當的 AWS 開發套件。這些方法皆在下列主題中有詳述。

主題

- 使用 IAM 查詢 API 操作的範例程式碼
- 使用 Python 的範例程式碼
- 使用 Java 的範例程式碼
- 範例顯示建構 URL 的方法 (Ruby)

使用 IAM 查詢 API 操作的範例程式碼

您可以建構一個 URL，向角色和聯合身分主體授予 AWS 管理主控台的直接存取權。此任務使用 IAM 和 AWS STS HTTPS 查詢 API。如需有關提出查詢請求的詳細資訊，請參閱提出查詢請求。

> **Note**
>
> 以下過程包括了文字字串的範例。為增加可讀性，一些較長的範例中加入了分行符號。如果您要建立並使用這些字串，必須省略所有分行符號。

讓角色和聯合委託人從 存取您的資源 AWS 管理主控台

1. 在您的身分驗證系統裡驗證該使用者。
2. 為使用者取得臨時安全性憑證。臨時性憑證由存取金鑰 ID、私密存取金鑰和工作階段字符組成。如需有關建立暫時性憑證檔案的詳細資訊，請參閱 IAM 中的暫時安全憑證。

   若要取得臨時登入資料，您可以呼叫 AWS STS AssumeRole API （建議） 或 GetFederationToken API。如需這些 API 操作之間差異的詳細資訊，請參閱 AWS 安全部落格中的了解安全委派存取您 AWS 帳戶的 API 選項。

   > **Important**
   >
   > 在使用 GetFederationToken API 建立暫時安全憑證時，您必須指定這些憑證授予擔任角色之使用者的許可。對於任何以 `AssumeRole*` 開頭的 API 操作，可使用 IAM 角色來分配許可。對於其他 API 操作，機制會因 API 而異。如需詳細資訊，請參閱 臨時安全憑證

的許可。此外，如果使用 AssumeRole* API 操作，您還必須以具有長期憑證的 IAM 使用者身分來呼叫這些操作。否則，在步驟 3 中對聯合端點的呼叫將失敗。

3. 在取得臨時安全性憑證後，將其建構到 JSON 工作階段字串以將其交換為登入權杖。以下範例顯示如何為憑證編碼。請將預留位置文字替換為上一步驟中接收的憑證中的適用值。

```
{"sessionId":"*** temporary access key ID ***",
"sessionKey":"*** temporary secret access key ***",
"sessionToken":"*** session token ***"}
```

4. URL 編碼前一步驟中的工作階段字串。由於您編碼的資訊是敏感資訊，因此建議您避免對此編碼使用 Web 服務。請改用開發套件中在本機安裝的函數或功能來安全地編碼這些資訊。您可以使用 Python 的 `urllib.quote_plus` 函數、Java 的 `URLEncoder.encode` 函數或 Ruby 的 `CGI.escape` 函數。請參閱本主題後述的範例。

5.

> **Note**
>
> AWS 支援此處的 POST 請求。

將您的請求傳送至 AWS 聯合端點：

`https://region-code.signin.aws.amazon.com/federation`

如需可能 `####`值清單，請參閱AWS 登入端點中的區域欄。您可以選擇使用預設 AWS 的登入聯合端點：

`https://signin.aws.amazon.com/federation`

該請求必須包含 `Action` 與 `Session` 參數；(選擇性) 如果已使用 `AssumeRole*` API 操作，還必須包含 `SessionDuration` HTTP 參數，如下列範例所示。

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> 此步驟中的下列指示僅適用於 GET 請求。

SessionDuration HTTP 參數指定主控台工作階段的持續時間。它不同於使用 DurationSeconds 參數指定的臨時性憑證的持續時間。您可指定 SessionDuration 最高值為 43,200 (12 小時)。如果缺少 SessionDuration 參數，則工作階段預設為您在步驟 2 AWS STS 中從 擷取的憑證持續時間 （預設為一小時）。如需如何使用 DurationSeconds 參數指定持續時間的詳細資訊，請參閱 AssumeRole API 的文件。建立超過 1 小時的主控台工作階段的功能是聯合端點的 getSigninToken 操作所內建的特性。

> **Note**
>
> - 您的 SessionDuration 不能大於或等於您擔任之角色的工作階段持續時間上限設定。例如，您已將要擔任之角色的工作階段持續時間上限設定為 5 小時。您的 SessionDuration 參數可以是 16,524 秒或 4 小時 59 秒。
> - 如果透過 GetFederationToken 取得臨時憑證，請勿使用 SessionDuration HTTP 參數。否則，操作會失敗。
> - 使用一個角色的憑證來擔任不同的角色稱為角色鏈結。當您使用角色鏈結時，新憑證的最大持續時間限制為一小時。使用角色授予許可給在 EC2 執行個體上執行的應用程式時，那些應用程式不受此限制。
> - 如果透過角色鏈結取得臨時憑證，請勿使用 SessionDuration HTTP 參數。否則，操作會失敗。

在啟用具有更長持續時間的主控台工作階段時，您會增加洩露憑證的風險。為了幫助緩解這種風險，您可以在 Role Summary (角色摘要) IAM 主控台頁面上選擇 Revoke Sessions (撤銷工作階段) 來立即停用所有角色的使用中的主控台工作階段。如需詳細資訊，請參閱 撤銷 IAM 角色臨時安全憑證。

以下為請求具體形式的範例。在此處換行以便閱讀，但您應將其做為單行字串提交。

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
```

```
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

來自聯合身分端點的回應是一個具有 SigninToken 值的 JSON 文件。看起來與下列範例類似。

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWS 支援此處的 POST 請求。

最後，建立使用者可用於存取 AWS 管理主控台的 URL。此 URL 與您在 Step 5 中使用的聯合身分 URL 端點相同，外加以下參數：

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> 此步驟中的下列指示僅適用於 GET API。

以下範例顯示 URL 的最終形式。URL 的有效期為 15 分鐘，自建立時算起。在 URL 中嵌入的臨時安全性憑證和主控台工作階段的有效期為在最初請求它們時在 SessionDuration HTTP 參數中指定的持續時間。

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
```

```
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

使用 Python 的範例程式碼

下列範例說明如何使用 Python 以程式設計方式建構 URL，向使用者授予 AWS 管理主控台直接存取權。以下是兩個範例：

- 透過 GET 請求聯合至 AWS
- 透過 POST 請求聯合至 AWS

這兩個範例都使用 適用於 Python (Boto3) 的 AWS SDK和 AssumeRole API 來取得臨時安全登入資料。

如果 AssumeRoleSession 憑證來自角色鏈結，請勿包含 SessionDuration。如果包含 SessionDuration，操作將會失敗。

使用 GET 請求

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your AWS ##,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
```

```
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
```

```
# Send final URL to stdout
print (request_url)
```

使用 POST 請求

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your A AWS ##,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
```

```
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']

jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
        form.appendChild(hiddenField);
    }}
```

```
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)

driver = webdriver.Chrome()
driver.execute_script(jsrequest)
input("Press Enter to close the browser window...")
```

使用 Java 的範例程式碼

下列範例說明如何使用 Java 以程式設計方式建構 URL，向使用者授予 AWS 管理主控台的直接存取權。下列程式碼片段使用 AWS SDK for Java。

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
```

```
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");

// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.

String policy = "{\"Version\":\"2012-10-17\",       \"Statement\":[{\"Action\":\"sns:*
\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
                           "?Action=getSigninToken" +
                           "&DurationSeconds=43200" +
                           "&SessionType=json&Session=" +
```

```
                            URLEncoder.encode(sessionJson,"UTF-8");

URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

範例顯示建構 URL 的方法 (Ruby)

以下範例顯示了如何使用 Ruby 以程式設計方式構建 URL，向使用者授予 AWS 管理主控台的直接存取權。此程式碼片段使用 AWS SDK for Ruby。

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
```

```
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",       \"Statement\":{\"Effect\":\"Allow\",
\"Action\":\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
                       "&SessionType=json&Session=" +
                       CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read
```

```
# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

AWS Identity and Access Management 資源的標籤

標籤是一種自訂屬性標籤，可由您指派給 AWS 資源。每個標籤有兩個部分：

- 標籤鍵 (例如，`CostCenter`、`Environment`、`Project` 或 `Purpose`)。
- 一個名為標籤值 (例如，`111122223333`、`Production` 或團隊名稱) 的選用欄位。忽略標籤值基本上等同於使用空字串。

這些合稱為鍵值組。關於可具有 IAM 資源標籤數量的限制，請參閱 IAM AWS STS 和配額。

> **Note**
>
> 如需有關標籤索引鍵和標籤索引鍵值區分大小寫的詳細資訊，請參閱 Case sensitivity。

標籤可協助您識別和組織 AWS 資源。許多 AWS 服務支援標記，因此您可以將相同的標籤指派給來自不同 服務的資源，以指出資源相關。例如，您可以將相同的標籤指派給您指派給 Amazon S3 儲存貯體的 IAM 角色。如需標記策略的詳細資訊，請參閱 標記 AWS 資源使用者指南。

此外，若要使用標籤來識別、整理和追蹤 IAM 資源，您可以在 IAM 政策中使用標籤，以協助控制誰可以檢視您的資源並與其互動。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

您也可以在 中使用標籤 AWS STS ，在擔任角色或聯合使用者時新增自訂屬性。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。

主題

- 選擇 AWS 標籤命名慣例
- 在 IAM 和 中標記的規則 AWS STS
- 標記 IAM 使用者
- 標記 IAM 角色
- 標記客戶管理政策
- 標記 OpenID Connect (OIDC) 身分提供者
- 標記 IAM SAML 身分提供者
- 標記 Amazon EC2 角色的執行個體設定檔
- 標記伺服器憑證
- 標記虛擬 MFA 裝置
- 標記帳戶存取管理員執行個體
- 在 中傳遞工作階段標籤 AWS STS

選擇 AWS 標籤命名慣例

當您開始將標籤連接至您的 IAM 資源時，請小心選擇您的標籤命名慣例。將相同的慣例套用至所有 AWS 標籤。如果您在政策中使用標籤來控制對 AWS 資源的存取，這尤其重要。如果您在 AWS中已經使用標籤，請檢閱您的命名慣例和加以調整。

> **Note**
>
> 如果您的帳戶是 的成員 AWS Organizations，請參閱《 使用者指南》中的 AWS Organizations 標籤政策，進一步了解如何在其中使用標籤 AWS Organizations。

標籤命名的最佳實務

以下是標籤的一些最佳實務和命名慣例。

請確保以一致的方式使用標籤名稱。例如，標籤 `CostCenter` 和 `costcenter` 是不同的，因此可能會將其中一個設定為用於財務分析與報表的成本配置標籤，而另一個則可能不會如此設定。同樣地，許多資源的`Name`標籤會出現在 AWS 主控台中，但`name`標籤不會。如需有關標籤索引鍵和標籤索引鍵值區分大小寫的詳細資訊，請參閱 Case sensitivity。

許多標籤由 預先定義， AWS 或由各種 AWS 服務自動建立。許多 AWS定義的標籤名稱使用所有小寫，連字號分隔名稱中的單字，以及用於識別標籤來源服務的字首。例如：

- `aws:ec2spot:fleet-request-id` 識別啟動執行個體的 Amazon EC2 Spot 執行個體請求。
- `aws:cloudformation:stack-name` 會識別建立資源的 CloudFormation 堆疊。
- `elasticbeanstalk:environment-name` 識別建立資源的應用程式。

請考慮使用全部小寫字母來命名標籤，以連字號分隔文字，並且採用識別組織名稱或縮寫名稱的字首。例如，對於一家名為 AnyCompany 的虛構公司，您可以定義如下的標籤：

- `anycompany:cost-center` 可識別內部成本中心代碼
- `anycompany:environment-type` 可識別環境是否為開發、測試或生產
- `anycompany:application-id` 可識別建立資源的應用程式

字首可確保標籤明確識別為由您的組織所定義，而不是由您或您可能正在使用 AWS 的第三方工具所定義。將所有小寫字母和連字號 (作為分隔符號) 搭配使用可避免對如何大寫標籤名稱造成混淆。例如：`anycompany:project-id` 比 `ANYCOMPANY:ProjectID`、`anycompany:projectID` 或 `Anycompany:ProjectId` 更容易記住。

在 IAM 和 中標記的規則 AWS STS

多種慣例會掌管在 IAM 和 AWS STS中標籤的建立和應用。

命名標籤

為 IAM 資源、 AWS STS assume-role 工作階段和 AWS STS 聯合身分使用者工作階段制定標籤命名慣例時，請遵循下列慣例：

字元要求 – 標籤鍵和值可以包含任意組合字母、數字、空格和 _ . : / = + - @ 符號。

區分大小寫 – 標籤鍵是否區分大小寫會因標記的 IAM 資源類型而有所不同。IAM 使用者和角色的標籤鍵值不區分大小寫，但會保留大小寫。這表示您無法個別擁有 **`Department`** 和 **`department`** 標籤鍵。如果您使用 **`Department=finance`** 標籤為使用者加上標籤，而且您新增 **`department=hr`** 標籤，它會取代第一個標籤。不會新增第二個標籤。

對於其他 IAM 資源類型，標籤鍵值會區分大小寫。這意味著您可以有單獨的 **`Costcenter`** 和 **`costcenter`** 標籤鍵。例如，如果您已使用 **`Costcenter = 1234`** 標籤標記客戶受管政策，並新增 **`costcenter = 5678`** 標籤，則政策將同時具有 **`Costcenter`** 和 **`costcenter`** 標籤鍵。

最佳作法是，建議您避免使用類似的標籤使用不一致的大小寫處理。建議您決定大寫標籤的策略，並一致地在所有資源類型中實作該策略。若要進一步了解標記的最佳實務，請參閱《》中的標記 AWS 資源 AWS 一般參考。

下列清單顯示連接至 IAM 資源之標籤鍵的區分大小寫差異。

標籤鍵值不區分大小寫：

- IAM 角色
- IAM 使用者

標籤鍵值區分大小寫：

- 客戶受管政策
- 執行個體設定檔
- OpenID Connect 身分提供者
- SAML 身分提供者
- 伺服器憑證
- 虛擬 MFA 裝置

此外，適用下列規則：

- 您不能建立標籤鍵或以文字 **aws:** 為開頭的值。此標籤字首保留供 AWS 內部使用。
- 您可以使用 **phoneNumber =** 之類的空值來建立標籤。您不能建立空的標籤鍵。
- 您可以在單一標籤不指定多個值，但您可以在單一值建立自訂多值結構。例如，假設使用者 Zhang 與工程團隊和 QA 團隊合作。如果您連接 **team = Engineering** 標籤，然後連接 **team = QA** 標籤，則您將標籤的值從 **Engineering** 變更為 **QA**。反之，您可以在單一標籤中使用自訂分隔符號來包含多個值。在這個範例中，您可以將 **team = Engineering:QA** 標籤連接至 Zhang。

> **Note**
>
> 若要在此範例中使用 **team** 標籤控制工程師的存取權，您必須建立一個政策，以允許可能包含 **Engineering** 的每個組態，包括 **Engineering:QA**。若要進一步了解在政策中使用標籤的詳細資訊，請參閱 使用標籤控制對 IAM 使用者和角色的存取。

套用和編輯標籤

將標籤連接至 IAM 資源時，請遵守下列慣例：

- 您可以標記大多數 IAM 資源，但不能標記群組、擔任的角色、存取報告或硬體式 MFA 裝置。
- 您不能使用標籤編輯器來標記 IAM 資源。標籤編輯器不支援 IAM 標籤。如需使用標籤編輯器搭配其他服務的詳細資訊，請參閱 AWS Resource Groups 使用者指南中的使用標籤編輯器。
- 若要標記 IAM 資源，您必須擁有特定的許可。若要標記或取消標記資源，您還必須有列出標籤的許可。如需詳細資訊，請參閱本頁結尾每個 IAM 資源的主題清單。
- AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。
- 您可以將相同標籤套用到多個 IAM 資源。例如，假設您有一個名為 `AWS_Development` 的部門，其中包含 12 名成員。您可以擁有 12 名使用者，以及具備 **`department`** 標籤鍵和 **`awsDevelopment`** 值的角色 (**`department = awsDevelopment`**)。您也可以在其他支援標籤的服務中對資源使用相同標籤。
- IAM 實體 (使用者或角色) 不能擁有相同標籤鍵的多個執行個體。例如，如果您有一個使用者與標籤鍵值對 **`costCenter = 1234`**，您可以接著連接標籤鍵值對 **`costCenter = 5678`**。IAM 會更新 **`costCenter`** 標籤的值至 **`5678`**。
- 若要編輯連接至 IAM 實體 (使用者或角色) 的標籤，請連接含有新值的標籤以覆寫現有標籤。例如，假設您有一個使用者，含有標籤鍵值組 **`department = Engineering`**。如果您需要將使用者新增到 QA 部門，則可以將 **`department = QA`** 標籤鍵值組連接至使用者。這會造成 **`Engineering`** 標籤鍵的 **`department`** 值取代為 **`QA`** 值。

標記 IAM 使用者

您可以使用 IAM 標籤鍵值組向 IAM 使用者新增自訂屬性。例如，若要將位置資訊新增到使用者，您可以新增標籤鍵 **`location`** 和標籤值 **`us_wa_seattle`**。或者，您可以使用三個不同的位置的標籤鍵值組：**`loc-country = us`**、**`loc-state = wa`** 和 **`loc-city = seattle`**。您可以使用標籤來控制使用者對資源的存取權，或控制哪些標籤可以連接到使用者。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

您也可以在 中使用標籤 AWS STS ，在擔任角色或聯合使用者時新增自訂屬性。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。

標記 IAM 使用者所需的許可

您必須設定許可，以允許 IAM 使用者標記其他使用者。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- iam:ListUserTags
- iam:TagUser
- iam:UntagUser

允許 IAM 實體新增、列出，或移除特定使用者的標籤

將下列陳述式新增至需要管理標籤之 IAM 使用者的許可政策。使用您的帳戶並使用需要其標籤受管之使用者的名稱取代 *<username>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

允許 IAM 使用者自行管理標籤

將下列陳述式新增至使用者的許可政策，以讓使用者能管理其自身的標籤。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

允許 IAM 使用者新增特定使用者的標籤

將下列陳述式新增至 IAM 使用者的許可政策，而該使用者需要新增 (非移除) 特定使用者的標籤。

> **Note**
>
> 此 `iam:TagUser` 動作需要您也包含 `iam:ListUserTags` 動作。

若要使用此政策，請使用其標籤需要受管之使用者的名稱取代 *<username>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理 IAM 使用者的標籤 (主控台)

您可以從 AWS 管理主控台管理 IAM 使用者的標籤。

管理使用者的標籤 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在主控台導覽窗格中，選擇 Users (使用者)，然後選擇您要編輯的使用者名稱。
3. 選擇 Tags (標籤) 標籤，然後完成以下其中一個動作：
   - 如果使用者還沒有標籤，請選擇 新增標籤。
   - 選擇 Manage tags (管理標籤) 來管理現有的一組標籤。
4. 新增或移除標籤來完成標籤的設定。接著選擇 Save changes (儲存變更)。

管理 IAM 使用者 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接，或移除 IAM 使用者的標籤。您可以使用 AWS CLI 或 AWS API 來管理 IAM 使用者的標籤。

列出目前連接至 IAM 使用者的標籤 (AWS CLI 或 AWS API)

- AWS CLI: aws iam list-user-tags
- AWS API：ListUserTags

將標籤連接至 IAM 使用者 (AWS CLI 或 AWS API)

- AWS CLI：aws iam tag-user
- AWS API：TagUser

從 IAM 使用者 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI：aws iam untag-user
- AWS API：UntagUser

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記 IAM 角色

您可以使用 IAM 標籤鍵值組將自訂屬性新增至 IAM 角色。例如，若要將位置資訊新增至角色，您可以新增標籤鍵 **`location`** 和標籤值 **`us_wa_seattle`**。或者，您可以使用三個不同的位置的標籤鍵值組：**`loc-country = us`**、**`loc-state = wa`** 和 **`loc-city = seattle`**。您可以使用標籤來控制角色對資源的存取權，或控制哪些標籤可以連接到角色。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

您也可以在 中使用標籤 AWS STS ，在擔任角色或聯合使用者時新增自訂屬性。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。

標記 IAM 角色所需的許可

您必須設定許可，允許 IAM 角色標記其他實體 (使用者或角色)。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- `iam:ListRoleTags`

- iam:TagRole
- iam:UntagRole
- iam:ListUserTags
- iam:TagUser
- iam:UntagUser

允許 IAM 角色新增、列出，或移除特定使用者的標籤

將下列陳述式新增至需要管理標籤之 IAM 角色的許可政策。使用您的帳戶並使用需要其標籤受管之使用者的名稱取代 *<username>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

允許 IAM 角色新增特定使用者的標籤

將下列陳述式新增至 IAM 角色的許可政策，而該角色需要新增 (非移除) 特定使用者的標籤。

若要使用此政策，請使用其標籤需要受管之使用者的名稱取代 *<username>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

允許 IAM 角色新增、列出，或移除特定角色的標籤

將下列陳述式新增至需要管理標籤之 IAM 角色的許可政策。用其標籤需要受管的角色名稱取代 *<rolename>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理 IAM 角色的標籤 (主控台)

您可以從 AWS 管理主控台管理 IAM 角色的標籤。

管理角色的標籤 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在主控台導覽窗格中，選擇 Roles (角色)，然後選擇您要編輯的角色名稱。
3. 選擇 Tags (標籤) 標籤，然後完成以下其中一個動作：
   - 如果角色還沒有標籤，請選擇 Add new tag (新增標籤)。
   - 選擇 Manage tags (管理標籤) 來管理現有的一組標籤。
4. 新增或移除標籤來完成標籤的設定。然後，選擇 Save changes (儲存變更)。

管理 IAM 角色 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接，或移除 IAM 角色的標籤。您可以使用 AWS CLI 或 AWS API 來管理 IAM 角色的標籤。

列出目前連接至 IAM 角色 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: aws iam list-role-tags
- AWS API：ListRoleTags

將標籤連接至 IAM 角色 (AWS CLI 或 AWS API)

- AWS CLI：aws iam tag-role
- AWS API：TagRole

從 IAM 角色 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI：aws iam untag-role
- AWS API：UntagRole

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記客戶管理政策

您可以使用 IAM 標籤鍵值組，將自訂屬性新增至客戶受管政策。例如，若要使用部門資訊來標記政策，您可以新增標籤鍵 **Department** 和標籤值 **eng**。或者，您可能想要標記政策，以指出這些政策適用於特定環境，例如 **Environment = lab**。您可以使用標籤來控制對資源的存取權，或控制哪些標籤可以連接到資源。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

您也可以在 中使用標籤 AWS STS ，在擔任角色或聯合使用者時新增自訂屬性。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。

標記客戶受管政策所需的許可

您必須設定許可，以允許 IAM 實體 (使用者或角色) 標記客戶受管政策。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- `iam:ListPolicyTags`
- `iam:TagPolicy`
- `iam:UntagPolicy`

允許 IAM 實體 (使用者或角色) 新增、列出或移除客戶受管政策的標記

將下列陳述式新增至需要管理標籤之 IAM 實體的許可政策。使用您的帳戶號碼，用需要其標籤受管的政策名稱取代 *<policyname>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

允許 IAM 實體 (使用者或角色) 新增標籤至特定客戶受管政策

將下列陳述式新增至 IAM 實體的許可政策，而該實體需要新增 (非移除) 特定政策的標籤。

> **Note**
>
> 此 `iam:TagPolicy` 動作需要您也包含 `iam:ListPolicyTags` 動作。

若要使用此政策，請用需要其標籤受管的政策名稱取代 *<policyname>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理 IAM 客戶受管政策的標籤 (主控台)

您可以從 AWS 管理主控台管理 IAM 客戶受管政策的標籤。

管理客戶受管政策的標籤 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在主控台導覽窗格中，選擇 Policies (政策)，然後選擇您要編輯的客戶受管政策名稱。
3. 選擇標籤索引標籤，然後選擇管理標籤。
4. 新增或移除標籤來完成標籤的設定。接著選擇 Save changes (儲存變更)。

管理 IAM 客戶受管政策 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接或移除 IAM 客戶受管政策的標籤。您可以使用 AWS CLI 或 AWS API 來管理 IAM 客戶受管政策的標籤。

列出目前連接至 IAM 客戶受管政策 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: aws iam list-policy-tags
- AWS API：ListPolicyTags

將標籤連接至 IAM 客戶受管政策 (AWS CLI 或 AWS API)

- AWS CLI: aws iam tag-policy
- AWS API：TagPolicy

從 IAM 客戶受管政策 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI: aws iam untag-policy
- AWS API：UntagPolicy

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記 OpenID Connect (OIDC) 身分提供者

您可以使用 IAM 標籤鍵值將自訂屬性新增至 IAM OpenID Connect (OIDC) 身分提供者。例如，若要識別 OIDC 身分提供者，您可以新增標籤鍵 **`google`** 和標籤值 **`oidc`**。您可以使用標籤來控制對資源的存

取權，或控制哪些標籤可以連接到物件。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

標記 IAM OIDC 身分提供者所需的許可

您必須設定許可，允許 IAM 實體 (使用者或角色) 標記 IAM OIDC 身分提供者。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

若要允許 IAM 實體新增、列出或移除 IAM OIDC 身分提供者的標籤

將下列陳述式新增至需要管理標籤之 IAM 實體的許可政策。使用您的帳戶號碼，用需要其標籤受管的 OIDC 提供者名稱取代 *<OIDCProviderName>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
        "iam:UntagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

允許 IAM 實體 (使用者或角色) 將標籤新增至特定 IAM OIDC 身分提供者

將下列陳述式新增至 IAM 實體的許可政策，而該實體需要新增 (非移除) 特定身分提供者的標籤。

> **Note**
>
> 此 `iam:TagOpenIDConnectProvider` 動作需要您也包含 `iam:ListOpenIDConnectProviderTags` 動作。

若要使用此政策，請用需要其標籤受管的 OIDC 提供者名稱取代 *<OIDCProviderName>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理 IAM OIDC 身分提供者的標籤 (主控台)

您可以從 AWS 管理主控台管理 IAM OIDC 身分提供者的標籤。

管理 OIDC 身分提供者 (主控台) 的標籤

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在主控台的導覽窗格中，選擇 Identity providers (身分提供者)，然後選擇您所要編輯的身分提供者之名稱。
3. 選擇標籤索引標籤，然後在標籤區段中，選擇管理標籤，然後完成下列其中一個動作：
   - 如果 OIDC 身分提供者尚未有標籤或新增標籤，請選擇 Add tag (新增標籤)。
   - 編輯現有標籤鍵和值。
   - 請選擇 Remove tag (移除標籤) 以移除標籤。
4. 接著選擇 Save changes (儲存變更)。

管理 IAM OIDC 身分提供者 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接或移除 IAM OIDC 身分提供者的標籤。您可以使用 AWS CLI 或 AWS API 來管理 IAM OIDC 身分提供者的標籤。

列出目前連接至 IAM OIDC 身分提供者 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: aws iam list-open-id-connect-provider-tags

- AWS API：ListOpenIDConnectProviderTags

將標籤連接至 IAM OIDC 身分提供者 (AWS CLI 或 AWS API)

- AWS CLI: aws iam tag-open-id-connect-provider
- AWS API：TagOpenIDConnectProvider

從 IAM OIDC 身分提供者 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI: aws iam untag-open-id-connect-provider
- AWS API：UntagOpenIDConnectProvider

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記 IAM SAML 身分提供者

您可以使用 IAM 標籤鍵值組，將自訂屬性新增至 SAML 身分提供者。例如，若要識別提供者，您可以新增標籤鍵 **`okta`** 和標籤值 **`saml`**。您可以使用標籤來控制對資源的存取權，或控制哪些標籤可以連接到物件。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

標記 SAML 身分提供者所需的許可

您必須設定權限，以允許 IAM 實體 (使用者或角色) 標記 SAML 2.0 型身分提供者 (IdP)。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

允許 IAM 實體 (使用者或角色) 新增、列出或移除 SAML 身分提供者的標籤

將下列陳述式新增至需要管理標籤之 IAM 實體的許可政策。使用您的帳戶號碼，用需要其標籤受管的 SAML 提供者名稱取代 *`<SAMLProviderName>`*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

允許 IAM 實體 (使用者或角色) 將標籤新增至特定 SAML 身分提供者

將下列陳述式新增至 IAM 實體的許可政策，而該實體需要新增 (非移除) 特定 SAML 提供者的標籤。

> **Note**
>
> 此 `iam:TagSAMLProvider` 動作需要您也包含 `iam:ListSAMLProviderTags` 動作。

若要使用此政策，請用需要其標籤受管的 SAML 提供者名稱取代 *<SAMLProviderName>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理 IAM SAML 身分提供者的標籤 (主控台)

您可以從 AWS 管理主控台管理 IAM SAML 身分提供者的標籤。

管理 SAML 身分提供者 (主控台) 的標籤

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。

2. 在主控台的導覽窗格中，選擇 Identity providers (身分提供者)，然後選擇您所要編輯的 SAML 身分提供者之名稱。
3. 選擇標籤索引標籤，然後在標籤區段中，選擇管理標籤，然後完成下列其中一個動作：
   - 如果 SAML 身分提供者尚未有標籤或新增標籤，請選擇 Add tag (新增標籤)。
   - 編輯現有標籤鍵和值。
   - 請選擇 Remove tag (移除標籤) 以移除標籤。
4. 新增或移除標籤來完成標籤的設定。接著選擇 Save changes (儲存變更)。

管理 IAM SAML 身分提供者 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接或移除 IAM SAML 身分提供者的標籤。您可以使用 AWS CLI 或 AWS API 來管理 IAM SAML 身分提供者的標籤。

列出目前連接至 SAML 身分提供者 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: aws iam list-saml-provider-tags
- AWS API：ListSAMLProviderTags

將標籤連接至 SAML 身分提供者 (AWS CLI 或 AWS API)

- AWS CLI: aws iam tag-saml-provider
- AWS API：TagSAMLProvider

從 SAML 身分提供者 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI: aws iam untag-saml-provider
- AWS API：UntagSAMLProvider

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記 Amazon EC2 角色的執行個體設定檔

啟動 Amazon EC2 執行個體時，指定要與執行個體相關聯的 IAM 角色。執行個體描述檔是適用於 IAM 角色的容器，可讓您在執行個體啟動時將角色資訊傳遞至 Amazon EC2 執行個體。您可以在使用 AWS CLI 或 AWS API 時標記執行個體描述檔。

您可以使用 IAM 標籤鍵值組將自訂屬性新增至執行個體描述檔。例如，若要將部門資訊新增至執行個體設定檔，您可以新增標籤鍵 **access-team** 和標籤值 **eng**。這樣做可讓具有相符標籤的主體存取具有相同標籤的執行個體設定檔。您可以使用多個標籤鍵值組來指定團隊和專案：**access-team = eng** 和 **project = peg**。您可以使用標籤來控制使用者對資源的存取權，或控制哪些標籤可以連接到使用者。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

您也可以在 中使用標籤 AWS STS ，在擔任角色或聯合使用者時新增自訂屬性。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。

標記執行個體設定檔所需的許可

您必須設定許可，允許 IAM 實體 (使用者或角色) 標記執行個體描述檔。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- `iam:ListInstanceProfileTags`
- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

允許 IAM 實體 (使用者或角色) 新增、列出或移除執行個體描述檔的標籤

將下列陳述式新增至需要管理標籤之 IAM 實體的許可政策。使用您的帳戶號碼，用需要其標籤受管的執行個體描述檔取代 *<InstanceProfileName>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
```

```
}
```

允許 IAM 實體 (使用者或角色) 將標籤新增至特定執行個體描述檔

將下列陳述式新增至 IAM 實體的許可政策，而該實體需要新增 (非移除) 特定執行個體描述檔的標籤。

> **Note**
>
> 此 `iam:TagInstanceProfile` 動作需要您也包含 `iam:ListInstanceProfileTags` 動作。

若要使用此政策，請用需要其標籤受管的執行個體描述檔取代 *<InstanceProfileName>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理執行個體描述檔 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接或移除執行個體設定檔的標籤。您可以使用 AWS CLI 或 AWS API 來管理執行個體描述檔的標籤。

列出目前連接至執行個體描述檔 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: aws iam list-instance-profile-tags
- AWS API：ListInstanceProfileTags

將標籤連接至執行個體描述檔 (AWS CLI 或 AWS API)

- AWS CLI：aws iam tag-instance-profile

- AWS API：TagInstanceProfile

從執行個體描述檔 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI: aws iam untag-instance-profile
- AWS API：UntagInstanceProfile

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記伺服器憑證

如果您使用 IAM 管理 SSL/TLS 憑證，您可以使用 AWS CLI 或 AWS API 在 IAM 中標記伺服器憑證。對於 AWS Certificate Manager (ACM) 支援的區域中的憑證，我們建議您使用 ACM 而非 IAM 來佈建、管理和部署伺服器憑證。在不支援的區域中，您必須使用 IAM 作為憑證管理員。如需了解 ACM 支援哪些區域，請參閱 AWS 一般參考 中的 AWS Certificate Manager 端點和配額。

您可以使用 IAM 標籤鍵值組將自訂屬性新增至伺服器憑證。例如，若要新增伺服器憑證擁有者或系統管理員的相關資訊，請新增標籤鍵 **owner** 和標籤值 **net-eng**。或者，您可以透過新增標籤鍵 **CostCenter** 和標籤值 **1234** 來指定成本中心。您可以使用標籤來控制對資源的存取權，或控制哪些標籤可以連接到資源。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

您也可以在 中使用標籤 AWS STS ，在擔任角色或聯合使用者時新增自訂屬性。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。

標記伺服器憑證所需的許可

您必須設定許可，允許 IAM 實體 (使用者或角色) 標記其他伺服器憑證。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

允許 IAM 實體 (使用者或角色) 新增、列出或移除伺服器憑證的標籤

將下列陳述式新增至需要管理標籤之 IAM 實體的許可政策。使用您的帳戶號碼，用需要其標籤受管的伺服器憑證名稱取代 *<CertificateName>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

允許 IAM 實體 (使用者或角色) 將標籤新增至特定伺服器憑證

將下列陳述式新增至 IAM 實體的許可政策，而該實體需要新增 (非移除) 特定伺服器憑證的標籤。

> **Note**
>
> 此 `iam:TagServerCertificate` 動作需要您也包含 `iam:ListServerCertificateTags` 動作。

若要使用此政策，請用需要其標籤受管的伺服器憑證名稱取代 *<CertificateName>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理伺服器憑證 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接或移除伺服器憑證的標籤。您可以使用 AWS CLI 或 AWS API 來管理伺服器憑證的標籤。

列出目前連接至伺服器憑證 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: aws iam list-server-certificate-tags
- AWS API：ListServerCertificateTags

將標籤連接至伺服器憑證 (AWS CLI 或 AWS API)

- AWS CLI: aws iam tag-server-certificate
- AWS API：TagServerCertificate

從伺服器憑證 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI: aws iam untag-server-certificate
- AWS API：UntagServerCertificate

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記虛擬 MFA 裝置

您可以使用 IAM 標籤鍵值組將自訂屬性新增至虛擬 MFA 裝置。例如，若要新增使用者虛擬 MFA 裝置的成本中心資訊，您可以新增標籤鍵 **`CostCenter`** 和標籤值 **`1234`**。您可以使用標籤來控制對資源的存取權，或控制哪些標籤可以連接到物件。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

您也可以在 中使用標籤 AWS STS ，在擔任角色或聯合使用者時新增自訂屬性。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。

標記虛擬 MFA 裝置所需的許可

您必須設定許可，允許 IAM 實體 (使用者或角色) 標記虛擬 MFA 裝置。您可在 IAM 政策中指定以下一個或所有 IAM 標籤動作：

- iam:ListMFADeviceTags
- iam:TagMFADevice
- iam:UntagMFADevice

允許 IAM 實體 (使用者或角色) 新增、列出或移除虛擬 MFA 裝置的標籤

將下列陳述式新增至需要管理標籤之 IAM 實體的許可政策。使用您的帳戶並使用需要其標籤受管之虛擬 MFA 裝置的名稱取代 *<MFATokenID>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

允許 IAM 實體 (使用者或角色) 將標籤新增至特定虛擬 MFA 裝置

將下列陳述式新增至 IAM 實體的許可政策，而該實體需要新增 (非移除) 特定 MFA 裝置的標籤。

> **Note**
>
> 此 iam:TagMFADevice 動作需要您也包含 iam:ListMFADeviceTags 動作。

若要使用此政策，請用需要其標籤受管之虛擬 MFA 裝置的名稱取代 *<MFATokenID>*。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice"
    ],
```

```
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

或者，您可以使用 受 AWS 管政策，例如 IAMFullAccess 來提供 IAM 的完整存取權。

管理虛擬 MFA 裝置 (AWS CLI 或 AWS API) 上的標籤

您可以列出、連接或移除虛擬 MFA 裝置的標籤。您可以使用 AWS CLI 或 AWS API 來管理虛擬 MFA 裝置的標籤。

列出目前連接至虛擬 MFA 裝置 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: aws iam list-mfa-device-tags
- AWS API：ListMFADeviceTags

將標籤連接至虛擬 MFA 裝置 (AWS CLI 或 AWS API)

- AWS CLI: aws iam tag-mfa-device
- AWS API：TagMFADevice

從虛擬 MFA 裝置 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI: aws iam untag-mfa-device
- AWS API：UntagMFADevice

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 IAM 政策元素：變數與標籤。

標記帳戶存取管理員執行個體

您必須設定許可，以允許 IAM 實體（角色或使用者）標記帳戶存取管理員執行個體。您可以在 IAM 政策中指定下列一或多個帳戶存取標籤動作：

- `account-access:TagResource`
- `account-access:ListTagsForResource`

- account-access:UntagResource

允許 IAM 實體（角色或使用者）新增、列出或移除帳戶存取管理員執行個體的標籤

將下列陳述式新增至需要管理標籤之 IAM 實體的許可政策。使用 IAM Identity Center 的帳號和執行個體 IDs 和帳戶存取管理員執行個體來取代預留位置。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “建立 IAM 政策 (主控台)”。

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource",
        "account-access:UntagResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

允許 IAM 實體（角色或使用者）將標籤新增至特定帳戶存取管理員執行個體

將下列陳述式新增至需要新增，但不移除特定帳戶存取管理員執行個體標籤的 IAM 實體許可政策。

> **Note**
>
> 此 account-access:TagResource 動作需要您也包含 account-access:ListTagsForResource 動作。

使用 IAM Identity Center 的帳號和執行個體 IDs 和帳戶存取管理員執行個體來取代預留位置。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “建立 IAM 政策 (主控台)”。

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

管理帳戶存取管理員上的標籤 （主控台）

您可以從 管理帳戶存取管理員執行個體的標籤 AWS 管理主控台。

管理帳戶存取管理員上的標籤 （主控台）

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在主控台的導覽窗格中，選擇帳戶存取管理員，然後選擇設定索引標籤。
3. 在標籤區段中，您可以看到目前設定的標籤。選擇管理標籤，然後完成下列其中一個動作：
   - 選擇新增標籤以新增標籤。
   - 編輯現有標籤鍵和值。
   - 選擇移除以移除標籤。
4. 新增或移除標籤來完成標籤的設定。接著選擇 Save changes (儲存變更)。

管理帳戶存取管理員 (AWS CLI 或 AWS API) 上的標籤

您可以使用 或 API 列出、連接 AWS CLI 或 AWS 移除帳戶存取管理員執行個體的標籤。

列出目前連接至帳戶存取管理員執行個體 (AWS CLI 或 AWS API) 的標籤

- AWS CLI: `aws account-access list-tags-for-resource`
- AWS API： `account-access:ListTagsForResource`

將標籤連接至帳戶存取管理員執行個體 (AWS CLI 或 AWS API)

- AWS CLI: `aws account-access tag-resource`
- AWS API： `TagResource`

從帳戶存取管理員執行個體 (AWS CLI 或 AWS API) 移除標籤

- AWS CLI: `aws account-access untag-resource`
- AWS API： `UntagResource`

如需將標籤連接至其他 AWS 服務資源的相關資訊，請參閱這些服務的文件。

如需使用標籤來設定多個精密許可搭配 IAM 許可政策的更多資訊，請參閱 the section called “變數和標籤”。

在 中傳遞工作階段標籤 AWS STS

工作階段標籤是您在 AWS STS中擔任 IAM 角色或與使用者聯合身分時傳遞的鍵/值對。您可以透過 AWS CLI 身分提供者 (IdP) 提出 AWS STS 或 AWS API 請求來執行此操作。當您使用 AWS STS 請求臨時安全登入資料時，會產生工作階段。工作階段會過期且包含憑證，例如存取金鑰對和工作階段權杖。當您使用工作階段憑證發出後續請求時，請求內容會包含 `aws:PrincipalTag` 內容鍵。您可以在您政策的 `aws:PrincipalTag` 元素中使用 `Condition` 鍵來根據這些標籤允許或拒絕存取。

當您使用暫時憑證發出請求時，您的主體可能會包含一組標籤。這些標籤來自下列來源：

1. 工作階段標籤 – 當您使用 或 AWS CLI AWS API 擔任角色或聯合使用者時傳遞的標籤。如需有關這些操作的詳細資訊，請參閱 工作階段標記操作。
2. 傳入的可轉移工作階段標籤 – 這些標籤會從角色鏈中的上一個工作階段繼承。如需詳細資訊，請參閱本主題稍後的 使用工作階段標籤鏈結角色。
3. IAM 標籤 – 連接至 IAM 的標籤擔任角色。

主題

- 工作階段標記操作
- 工作階段標籤的須知事項
- 新增工作階段標籤所需的許可
- 使用 AssumeRole 傳遞工作階段標籤
- 使用 AssumeRoleWithSAML 傳遞工作階段標籤
- 使用 AssumeRoleWithWebIdentity 傳遞工作階段標籤
- 使用 GetFederationToken 傳遞工作階段標籤
- 使用工作階段標籤鏈結角色
- 使用 ABAC 的工作階段標籤
- 在 CloudTrail 中檢視工作階段標籤

工作階段標記操作

您可以使用下列 AWS CLI 或 AWS API 操作傳遞工作階段標籤 AWS STS。 AWS 管理主控台 切換角色功能不允許您傳遞工作階段標籤。

您也可以將工作階段標籤設為可轉移。可轉移標籤會在角色鏈結期間保存。如需詳細資訊，請參閱使用工作階段標籤鏈結角色。

下表比較傳遞工作階段索引標籤的方法。

作業	誰可以擔任這個角色	傳遞標籤的方法	設定可轉移標籤的方法
assume-role CLI 或 AssumeRole API 操作	IAM 使用者或工作階段	Tags API 參數或 --tags CLI 選項	TransitiveTagKeys API 參數或 --transitive-tag-keys CLI 選項
assume-role-with-saml CLI 或 AssumeRoleWithSAML API 操作	使用 SAML 身分提供者進行身分驗證的任何使用者	PrincipalTag SAML 屬性	TransitiveTagKeys SAML 屬性
assume-role-with-web-identity CLI 或 AssumeRoleWithWebIdentity API 操作	使用 OIDC 提供者進行身分驗證的任何使用者	PrincipalTag OIDC 權杖	TransitiveTagKeys OIDC 權杖
get-federation-token CLI 或 GetFederationToken API 操作	IAM 使用者或根使用者	Tags API 參數或 --tags CLI 選項	不支援

在以下條件下，則支援工作階段標記的操作便可能會失敗：

- 您傳遞的工作階段標籤數超過 50 個。

- 工作階段標籤索引鍵的純文字超過 128 個字元。
- 工作階段標籤值的純文字超過 256 個字元。
- 工作階段政策純文字的總大小超過 2048 個字元。
- 工作階段政策和工作階段標籤會新增至工作階段字符大小。如果字符超過大小限制，操作會失敗。錯誤訊息會以位元組為單位報告工作階段字符大小和允許的大小上限。在 AWS STS API 回應SessionTokenUtilization中監控 以追蹤您的用量。

工作階段標籤的須知事項

使用工作階段標籤前，請檢閱下列工作階段和標籤的相關詳細資訊。

- 使用工作階段標籤時，所有連線至傳遞標籤之身分提供者 (IdP) 之角色的信任政策必須具有 sts:TagSession 許可。對於在信任政策中沒有此許可的角色，AssumeRole 操作即會失敗。
- 當您要求工作階段時，您可以指定主體標籤作為工作階段標籤。標籤會套用至您使用工作階段的憑證所發出的請求。
- 工作階段標籤使用鍵值對。例如，若要將聯絡資訊新增至工作階段，您可以新增工作階段標籤鍵 email 及標籤值 johndoe@example.com。
- 工作階段標籤必須遵循 IAM 和 中命名標籤的規則 AWS STS。本主題包含適用於您工作階段標籤的區分大小寫及受限制字首資訊。
- 新的工作階段標籤會覆寫具有相同標籤鍵的現有擔任角色或聯合身分使用者工作階段標籤，無論字元大小寫為何。
- 您無法使用 傳遞工作階段標籤 AWS 管理主控台。
- 工作階段標籤僅對目前的工作階段有效。
- 工作階段標籤支援角色鏈結。根據預設， AWS STS 不會將標籤傳遞至後續的角色工作階段。但是，您可以將工作階段標籤設為可轉移。可轉移標籤在角色鏈結期間會持續存在，並在角色信任政策評估期間取代相符的 ResourceTag 值。如需詳細資訊，請參閱使用工作階段標籤鏈結角色。
- 您可以使用工作階段標籤來控制對資源的存取，或是控制可傳遞至後續工作階段的標籤。如需詳細資訊，請參閱 IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記。
- 您可以在 AWS CloudTrail 日誌中檢視您工作階段的主體標籤，包括工作階段標籤。如需詳細資訊，請參閱在 CloudTrail 中檢視工作階段標籤。
- 您必須傳遞每個工作階段標籤的單一值。 AWS STS 不支援多值工作階段標籤。
- 您最多可傳遞 50 個工作階段標籤。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

- 工作階段政策和工作階段標籤會新增至工作階段字符大小。如果字符超過大小限制，AWS CLI 或 AWS API 會傳回錯誤。錯誤訊息會以位元組為單位報告工作階段字符大小和允許的大小上限。SessionTokenUtilization 回應元素會顯示字符使用的大小限制百分比。PackedPolicySize 回應元素已棄用。請改用 SessionTokenUtilization。

新增工作階段標籤所需的許可

除了符合 API 操作的動作外，您必須在您的政策中具備下列僅許可動作：

```
sts:TagSession
```

> **Important**
>
> 使用工作階段標籤時，連線至身分提供者 (IdP) 之所有角色的角色信任政策必須具有 sts:TagSession 許可。任何連接至 IdP 的正在傳遞工作階段標籤之角色，在沒有此許可的情況下，AssumeRole 操作即會失敗。如果您不想更新每個角色的角色信任政策，則您可以使用個別的 IdP 執行個體傳遞工作階段標籤。然後，將 sts:TagSession 許可僅新增至連線至個別 IdP 的角色。

您可以使用 sts:TagSession 動作搭配下列條件金鑰。

- aws:PrincipalTag – 將連接至提出請求主體的標籤，與您在政策中所指定的標籤進行比較。例如，您可以允許主體只有在主體提出具備指定標籤的請求時，才能傳遞工作階段標籤。
- aws:RequestTag – 將請求中傳遞的標籤鍵值對與您在政策中所指定的標籤對進行比較。例如，您可以允許主體傳遞指定工作階段標籤，但僅限指定的值。
- aws:ResourceTag – 將您在政策中所指定的標籤鍵值對與連接到資源的鍵值對進行比較。例如，您可以允許主體僅在其擔任的角色包含指定標籤時，才能傳遞工作階段標籤。
- aws:TagKeys – 將請求中的標籤鍵與您在政策中所指定的鍵進行比較。例如，您可以只允許主體傳遞具備指定標籤鍵的工作階段標籤。此條件索引鍵會限制可傳遞的工作階段標籤組上限。
- sts:TransitiveTagKeys - 將請求中的可轉移工作階段標籤鍵與政策中指定的標籤鍵進行比較。例如，您可以撰寫政策，只允許主體將特定標籤設為可轉移。可轉移標籤會在角色鏈結期間保存。如需詳細資訊，請參閱 使用工作階段標籤鏈結角色。

例如，以下角色信任政策會允許 test-session-tags 使用者擔任連接政策的角色。當該使用者擔任該角色時，他們必須使用 AWS CLI 或 AWS API 來傳遞三個必要的工作階段標籤和所需的外部 ID。此外，使用者可以選擇將 Project 和 Department 標籤設為可轉移。

Example工作階段標籤的角色信任政策範例

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
            }
        },
        {
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
```

```
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
                    ]
                }
            }
        }
    ]
}
```

此政策的功能為何？

- AllowIamUserAssumeRole 陳述式允許 test-session-tags 使用者擔任連接政策的角色。當該使用者擔任角色時，他們必須傳遞必要的工作階段標籤和外部 ID。
  - 此陳述式的第一個條件區塊要求使用者傳遞 Project、CostCenter 和 Department 工作階段標籤。由於標籤值在此陳述式中不重要，因此您可以針對標籤值使用萬用字元 (*)。此區塊會確保使用者至少傳遞這三個工作階段標籤。否則，操作會失敗。使用者可傳遞其他標籤。
  - 第二個條件區塊則會要求使用者傳遞值為 Example987 的外部 ID。
- AllowPassSessionTagsAndTransitive 陳述式允許 sts:TagSession 僅許可動作。必須先允許此動作，使用者才能傳遞工作階段標籤。如果您的政策包含第一個陳述式，但沒有第二個陳述式，使用者便無法擔任角色。
  - 此陳述式的第一個條件區塊允許使用者針對 CostCenter 和 Project 工作階段標籤傳遞任何值。您可以在政策中針對標籤值使用萬用字元 (*) 來執行此作業，該政策會要求您使用 StringLike 條件運算子。
  - 第二個條件區塊則只會允許使用者針對 Engineering 工作階段標籤傳遞 Marketing 或 Department 值。
  - 第三個條件區塊則會列出您可設為可轉移的標籤組上限。使用者可以選擇將一部分，或是不將任何標籤設為可轉移。使用者無法將其他標籤設為可轉移。您可以新增另一個包含 "Null": {"sts:TransitiveTagKeys":"false"} 的條件區塊，來要求使用者將其中至少一個標籤設為可轉移。

使用 AssumeRole 傳遞工作階段標籤

AssumeRole 操作會傳回一組可用來存取 AWS 資源的臨時登入資料。您可以使用 IAM 使用者或角色憑證來呼叫 AssumeRole。若要在擔任角色時傳遞工作階段標籤，請使用 --tags AWS CLI 選項或 Tags AWS API 參數。

若要將標籤設定為可轉移，請使用 --transitive-tag-keys AWS CLI 選項或 TransitiveTagKeys AWS API 參數。可轉移標籤會在角色鏈結期間保存。如需詳細資訊，請參閱使用工作階段標籤鏈結角色。

以下範例顯示使用 AssumeRole 的範例請求。在此範例中，當您擔任 my-role-example 角色時，您可以建立名為 my-session 的工作階段。您可以新增工作階段標籤鍵/值對 Project = Automation、CostCenter = 12345 和 Department = Engineering。您也可以透過指定其鍵，將 Project 和 Department 標籤設為可轉移。您必須傳遞每個工作階段標籤的單一值。 AWS STS 不支援多值工作階段標籤。

Example AssumeRole CLI 請求範例

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

使用 AssumeRoleWithSAML 傳遞工作階段標籤

AssumeRoleWithSAML 操作是使用 SAML 類型的聯合進行身分驗證。此操作會傳回一組可用來存取 AWS 資源的臨時登入資料。如需使用 SAML 型聯合進行 AWS 管理主控台 存取的詳細資訊，請參閱啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台。如需 AWS CLI 或 AWS API 存取的詳細資訊，請參閱 SAML 2.0 聯合身分。如需為 Active Directory 使用者設定 SAML 聯合的教學課程，請參閱 AWS 安全部落格中的AWS 使用 Active Directory 聯合服務 (ADFS) 進行聯合身分驗證。

身為管理員，您可以允許公司目錄的成員 AWS 使用 AWS STS AssumeRoleWithSAML操作聯合到 。若要執行此操作，您必須完成下列任務：

1. 將網路配置為適用於 AWS的 SAML 提供者
2. 在 IAM 中建立 SAML 提供者

3. 為 SAML 2.0 聯合身分建立角色 (主控台)
4. 完成配置 SAML IdP 並為 SAML 身分驗證回應建立聲明

AWS 包含身分提供者，其身分解決方案具有經認證的工作階段標籤end-to-end體驗。如要了解如何使用這些身分提供者來設定工作階段標籤，請參閱 將第三方 SAML 解決方案供應商與 整合 AWS。

如要將 SAML 屬性做為工作階段標籤傳遞，請在其中包含 `Attribute` 元素，並將其 `Name` 屬性設為 `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`。使用 `AttributeValue` 元素指定標籤的值。為每個工作階段標籤包含個別的 `Attribute` 元素。

例如，假設您希望將下列身分屬性做為工作階段標籤傳遞：

- `Project:Automation`
- `CostCenter:12345`
- `Department:Engineering`

如要傳遞這些屬性，請在您的 SAML 聲明中包含下列元素。

Example SAML 聲明的程式碼片段範例

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

如要將前述標籤設為可轉移，請在其中包含另一個 `Attribute` 元素，並將其 `Name` 屬性設為 `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`。可轉移標籤會在角色鏈結期間保存。如需詳細資訊，請參閱 使用工作階段標籤鏈結角色。

如要將 `Project` 和 `Department` 標籤設為可轉移，請使用以下多值屬性：

Example SAML 聲明的程式碼片段範例

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
```

```
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

使用 AssumeRoleWithWebIdentity 傳遞工作階段標籤

使用與 OpenID Connect (OIDC) 相容的聯合身分對 `AssumeRoleWithWebIdentity` 操作進行身分驗證。此操作會傳回一組可用來存取 AWS 資源的臨時登入資料。如需使用 Web 聯合身分進行 AWS 管理主控台 存取的詳細資訊，請參閱 OIDC 聯合身分。

若要從 OpenID Connect (OIDC) 傳遞工作階段標籤，您必須在提交 `AssumeRoleWithWebIdentity` 請求時在 JSON Web 權杖 (JWT) 中包含工作階段標籤。若要進一步了解 OIDC 權杖和要求，請參閱《Amazon Cognito 開發人員指南》中的搭配使用者集區使用權杖。

AWS 支援兩種在 JWT 中包含工作階段標籤的宣告格式：

- 巢狀宣告格式
- 平面化宣告格式

巢狀宣告格式

巢狀宣告格式使用 JWT 中 `https://aws.amazon.com/tags` 命名空間內的結構。在此格式中：

- 主體標籤以巢狀物件的形式呈現在 `principal_tags` 索引鍵下。
- 每個主體標籤都是單一字串值。
- 可轉移標籤索引鍵呈現在 `transitive_tag_keys` 索引鍵下的陣列中。
- `principal_tags` 和 `transitive_tag_keys` 都會巢狀在 `https://aws.amazon.com/tags` 命名空間下。

下列範例示範使用巢狀物件格式解碼的 JWT：

Example使用巢狀宣告格式解碼的 JSON Web 權杖的範例

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
```

```
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
            "CostCenter"
        ]
    }
}
```

平面化宣告格式

平面化宣告格式與不支援 JWT 宣告中巢狀物件的身分提供者 (例如 Microsoft Entra ID) 相容。在此格式中：

- 主體標籤以單獨的宣告呈現，字首為 `https://aws.amazon.com/tags/principal_tags/`。
- 每個主體標籤都是單一字串值。
- 可轉移標籤索引鍵在單一宣告中呈現為字首為 `https://aws.amazon.com/tags/transitive_tag_keys` 的字串陣列。

現在，讓我們看看如何使用平面化宣告格式來表示相同的資訊：

Example使用平面化宣告格式解碼的 JSON Web 權杖範例

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags/principal_tags/Project": "Automation",
    "https://aws.amazon.com/tags/principal_tags/CostCenter": "987654",
    "https://aws.amazon.com/tags/principal_tags/Department": "Engineering",
```

```
    "https://aws.amazon.com/tags/transitive_tag_keys": [
        "Project",
        "CostCenter"
    ]
}
```

兩個解碼的 JWT 範例都顯示對 AssumeRoleWithWebIdentity 的呼叫，其中包含 Project、CostCenter 和 Department 工作階段標籤。這兩個權杖都會將 Project 和 CostCenter 標籤設定為可轉移。可轉移標籤會在角色鏈結期間保存。如需詳細資訊，請參閱使用工作階段標籤鏈結角色。

平面化宣告格式實現與巢狀宣告格式相同的結果，但使用標籤的平面化結構。可讓您在 JWT 宣告不支援巢狀 JSON 物件的環境中包含工作階段標籤。使用任一格式時，請確保您的身分提供者已設定為使用適當的宣告結構發出權杖。 AWS 支援兩種宣告格式，因此您可以選擇最適合您身分提供者特定要求的格式。

使用 GetFederationToken 傳遞工作階段標籤

GetFederationToken 允許您聯合您的使用者。此操作會傳回一組可用來存取 AWS 資源的臨時登入資料。若要將標籤新增至聯合身分使用者工作階段，請使用 --tags AWS CLI 選項或 Tags AWS API 參數。當您使用時，您無法將工作階段標籤設為可轉移GetFederationToken，因為您無法使用臨時憑證來擔任角色。在這種情況下，您無法使用角色鏈結。

以下範例顯示使用 GetFederationToken 的範例請求。在此範例中，當您請求權杖時，您會建立名為 my-fed-user 的工作階段。您可以新增工作階段標籤鍵/值對 Project = Automation 和 Department = Engineering。

Example GetFederationToken CLI 請求範例

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

當您使用 GetFederationToken 操作傳回的暫時憑證時，工作階段的主體標籤會包含使用者的標籤和傳遞的工作階段標籤。

使用工作階段標籤鏈結角色

您可以擔任一個角色，然後使用暫時憑證來擔任另一個角色。您可以從一個工作階段繼續前往另一個工作階段。這稱為角色鏈結。當您在擔任角色期間傳遞工作階段標籤時，您可以將鍵設為可轉移。這可確

保會在角色鏈中，將這些工作階段標籤傳遞到後續的工作階段。您無法將角色標籤設為可轉移。如要將這些標籤傳遞至後續工作階段，請將這些標籤指定為工作階段標籤。

> **Note**
>
> 可轉移標籤在角色鏈結期間會持續存在，並在角色信任政策評估期間取代相符的 `ResourceTag` 值。

下列範例顯示 如何將工作階段標籤、傳輸性標籤和角色標籤 AWS STS 傳遞至角色鏈中的後續工作階段。

在此角色鏈結案例中，您會使用 中的 IAM 使用者存取金鑰 AWS CLI 來擔任名為 的角色`Role1`。您接著會使用產生的工作階段憑證來擔任名為 `Role2` 的第二個角色。您接著可以使用第二個工作階段憑證來擔任名為 `Role3` 的第三個角色。這些請求會以三個不同操作的形式發生。每個角色都已在 IAM 中加上標籤。而在每個請求期間，您會傳遞其他工作階段標籤。
角色鏈結。

鏈結角色時，您可以確保將先前工作階段的標籤保存到後續工作階段。若要執行此操作，請使用 `assume-role` CLI 命令；您必須將標籤作為工作階段標籤傳遞，並將標籤設為可轉移。您將標籤 `Star = 1` 設為工作階段標籤。命令還會將標籤 `Heart = 1` 連接到角色，並在您使用工作階段時作為主體標籤套用。但是，您也希望 `Heart = 1` 標籤能自動傳遞到第二個或第三個工作階段。若要執行此作業，您可以手動將其做為工作階段標籤包含在其中。產生的工作階段主體標籤包含這兩個標籤，並將其設定為可轉移。
擔任角色鏈中的第一個角色。

您可以使用下列 AWS CLI 命令執行此請求：

Example AssumeRole CLI 請求範例

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

然後，您可以使用該工作階段的憑證來擔任 `Role2`。命令會將標籤 `Sun = 2` 連接至第二個角色，並在您使用第二個工作階段時作為主體標籤套用。`Heart` 和 `Star`標籤會繼承第一個工作階段中的傳輸工作階段標籤。第二個工作階段所產生的主體標籤是 `Heart = 1`、`Star = 1` 及 `Sun = 2`。`Heart` 和 `Star`

會繼續處於可轉移狀態。連接至 Role2 的 Sun 標籤並未標記為可轉移，因為其不是工作階段標籤。未來的工作階段不會繼承此標籤。
擔任角色鏈中的第二個角色。

您可以使用下列 AWS CLI 命令執行第二個請求：

Example AssumeRole CLI 請求範例

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

您接著會使用第二個工作階段憑證來擔任 Role3。第三個工作階段的主體標籤是來自任何新的工作階段標籤、繼承的可轉移工作階段標籤，以及角色標籤。第二個工作階段上的 Heart = 1 和 Star = 1 標籤是繼承自第一個工作階段中的可轉移工作階段標籤。如您嘗試傳遞 Sun = 2 工作階段標籤，操作將會失敗。繼承的 Star = 1 工作階段標籤會覆寫角色的 Star = 3 標籤。在角色鏈結過程中，進行角色信任政策評估之後，可轉移標籤的值會覆寫與 ResourceTag 值相符的角色。在此範例中，若 Role3 在角色信任政策中使用 Star 作為 ResourceTag，並將 ResourceTag 值設定為來自呼叫角色工作階段的可轉移標籤值。角色Lightning標籤也適用於第三個工作階段，而且不會設定為可轉移。
擔任角色鏈中的第三個角色。

您可以使用下列 AWS CLI 命令執行第三個請求：

Example AssumeRole CLI 請求範例

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

使用 ABAC 的工作階段標籤

屬性類型存取控制 (ABAC) 會使用一種授權策略，根據標籤屬性定義許可。

如果您的公司使用 OIDC 或 SAML 類型的身分提供者 (IdP) 來管理使用者身分，您可以將您的聲明設為將工作階段標籤傳遞到 AWS。例如，當您的員工聯合到公司使用者身分時 AWS， 會將其屬性 AWS 套用至其產生的委託人。您接著可以使用 ABAC 來根據這些屬性允許或拒絕許可。如需詳細資訊，請參閱IAM 教學課程：針對 ABAC 使用 SAML 工作階段標記。

如需有關 IAM Identity Center 搭配 ABAC 使用的詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的存取控制的屬性。

在 CloudTrail 中檢視工作階段標籤

您可以使用 AWS CloudTrail 來檢視用於擔任角色或聯合使用者的請求。CloudTrail 日誌檔案包含經擔任角色或聯合身分使用者工作階段的主體標籤資訊。如需詳細資訊，請參閱 使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail。

例如，假設您提出 AWS STS `AssumeRoleWithSAML`請求、傳遞工作階段標籤，並將這些標籤設定為可轉移。您可以在您的 CloudTrail 日誌中找到以下資訊。

Example AssumeRoleWithSAML CloudTrail 日誌範例

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

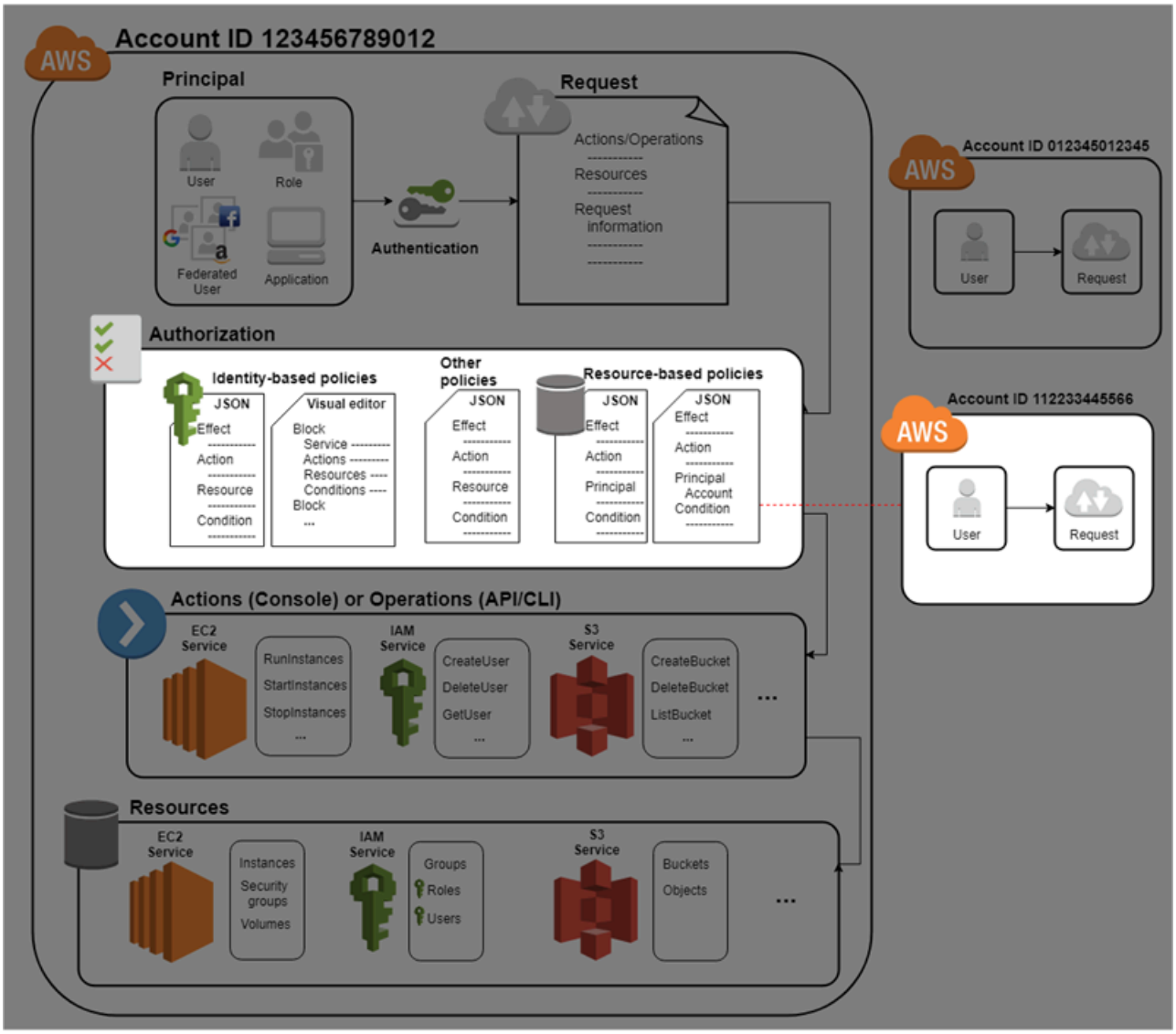
您可以檢視以下範例 CloudTrail 日誌，來檢視使用工作階段標籤的事件。

- CloudTrail 日誌檔案中 AWS STS 的角色鏈結 API 事件範例
- CloudTrail 日誌檔案中的範例 SAML AWS STS API 事件
- CloudTrail 日誌檔案中的範例 OIDC AWS STS API 事件

AWS 資源的存取管理

AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 資源的 AWS 存取。當委託人在 中提出請求時 AWS， AWS 強制執行程式碼會檢查委託人是否經過驗證 （登入） 和授權 （具有許可）。您可以透過建立政策並將其連接到 IAM 身分或 AWS 資源 AWS 來管理 中的存取。政策是 中的 JSON 文件 AWS ，當連接到身分或資源時，定義其許可。如需有關政策類型及其使用的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

如需有關其他身分驗證和授權程序的詳細資訊，請參閱 IAM 的運作方式。



在授權期間， AWS 強制執行程式碼會使用請求內容中的值來檢查相符政策，並決定是否允許或拒絕請求。

AWS 會檢查套用至請求內容的每個政策。如果單一政策拒絕請求， 會 AWS 拒絕整個請求並停止評估政策。此稱為明確拒絕。由於請求是預設拒絕，因此只有在適用的政策允許請求的每個部分時，IAM 才會授權該請求。用於單一帳戶中請求的評估邏輯遵循以下規則：

- 根據預設，所有的請求將以隱含方式拒絕。(此外，在預設情況下， AWS 帳戶根使用者 具有完整存取權限。)
- 若是以身分為基礎或以資源為基礎的政策，當中的明確允許會覆寫此預設值。
- 如果存在許可界限、 AWS Organizations SCP 或工作階段政策，它可能會以隱含拒絕覆寫允許。
- 任何政策中的明確拒絕會覆寫任何允許。

在您的請求經過身分驗證和授權之後， 會 AWS 核准請求。如果您需要運用不同的帳戶來發出請求，則必須透過其他帳戶中的政策來允許您存取資源。此外，您用來發出請求的 IAM 實體必須具備可允許該請求的以身分為基礎政策。

您也可以使用 IAM 暫時委派授予 Amazon 和 AWS 合作夥伴有限、有時間限制的存取權，以設定帳戶中 AWS 的服務。

存取管理資源

如需有關許可和建立政策的詳細資訊，請參閱以下資源：

AWS 安全部落格中的下列項目涵蓋撰寫政策以存取 Amazon S3 儲存貯體和物件的常見方法。

- 撰寫 IAM 政策：如何授予存取 Amazon S3 儲存貯體的許可
- 撰寫 IAM 政策：授予存取 Amazon S3 儲存貯體中使用者特定資料夾的許可
- IAM 政策和儲存貯體政策和 ACL！天啊！(控制 S3 資源的存取)。
- RDS 資源層級許可的入門
- EC2 資源層級許可的說明

中的政策和許可 AWS Identity and Access Management

AWS 透過建立政策並將其連接至 IAM 身分 （使用者、使用者群組或角色） 或 AWS 資源，來管理 中的存取。政策是 中的物件，當與身分或資源相關聯時， AWS 會定義其許可。當 IAM 主體 （使用者或角色） 發出請求時， 會 AWS 評估這些政策。政策中的許可決定是否允許或拒絕請求。大多數政策會以 JSON 文件 AWS 的形式存放在 中。 AWS 支援九種類型的政策：身分型政策、資源型政策、VPC

端點政策、許可界限、 AWS Organizations 服務控制政策 (SCPs)、 AWS Organizations 資源控制政策 (RCPs)、存取控制清單 (ACLs)、資源存取管理員共用 (RAM) 和工作階段政策。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，如果政策允許 GetUser 動作，則具有該政策的使用者可以從 AWS 管理主控台、 AWS CLI或 AWS API 取得使用者資訊。當您建立 IAM 使用者時，您可以選擇允許其主控台存取或程式化存取。如果允許主控台存取，則 IAM 使用者可以利用登入認證登入主控台。若在允許程式化存取情況下，使用者可以使用存取金鑰來操作 CLI 或 API。

政策類型

以下政策類型會以最常使用到較不常使用的順序列出，可以在 AWS中使用。如需更多詳細資訊，請參閱以下各節中的每個政策類型。

- 身分型政策 – 將受管和內嵌政策連接到 IAM 身分 (使用者、使用者所屬的群組或角色)。身分型政策可為身分授予許可。
- 資源型政策 – 將內嵌政策連接到資源。資源型政策的最常見範例是 Amazon S3 儲存貯體政策和 IAM 角色信任政策。資源型政策會將許可授予政策中所指定的主體。主體可以位在與資源相同的帳戶，或位在其他的帳戶中。
- VPC 端點政策 – 連接至 VPC 端點，以控制哪些主體可以使用 VPC 端點政策，以及可透過其存取哪些資源。VPC 端點政策可做為延伸到周遊端點之流量的額外存取界限。
- 許可界限 – 使用受管政策作為 IAM 實體 (使用者或角色) 的許可界限。該政策會定義身分型政策可為實體授予的最大許可，但並不會授予許可。許可界限不會定義資源型政策可以授予實體的許可上限。
- AWS Organizations SCPs – 使用 AWS Organizations 服務控制政策 (SCP) 來定義組織或組織單位 (OU) 中帳戶內 IAM 使用者和 IAM 角色的最大許可。SCP 會限制身分型政策或資源型政策為帳戶中的 IAM 使用者或 IAM 角色授予的許可。SCP 不會授予許可。
- AWS Organizations RCPs – 使用 AWS Organizations 資源控制政策 (RCP) 來定義組織或組織單位 (OU) 中帳戶內資源的最大許可。RCP 會限制身分型和資源型政策可以授予組織內帳戶資源的許可。RCP 不會授予許可。
- 存取控制清單 (ACL) – 使用 ACL，可以控制其他帳戶中的哪些主體可以存取其中已連接 ACL 的資源。ACL 類似資源型政策，雖然是不使用 JSON 政策文件結構的唯一政策類型。ACL 是跨帳戶許可的政策，負責為指定的主體授予許可。ACL 不可以為相同帳戶中的實體授予許可。
- AWS RAM 資源共用 – 使用 AWS Resource Access Manager 跨 AWS 帳戶、組織單位或整個組織共用資源，而無需為每個共用資源撰寫個別資源型政策。

- 工作階段政策 – 當您使用 AWS CLI 或 AWS API 擔任角色或聯合身分使用者時，傳遞進階工作階段政策。工作階段政策限制以角色或使用者身分型政策授予給該工作階段的許可。工作階段政策會限制已建工作階段的許可，但不會限制授予許可。如需詳細資訊，請參閱工作階段政策。

身分型政策

身分型政策是 JSON 許可政策文件，可控制身分 (使用者、使用者群組和角色) 會在何種條件下對哪些資源執行哪些動作。身分型政策可進一步分類：

- 受管政策：您可以連接到 中多個使用者、群組和角色的獨立身分型政策 AWS 帳戶。受管政策有兩種：
  - AWS 受管政策 – 由 建立和管理的受管政策 AWS。
  - 客戶管理政策：您在 AWS 帳戶中建立和管理的受管政策。客戶受管政策提供比 AWS 受管政策更精確的政策控制。
- 內嵌政策 – 您直接新增至單一使用者、群組或角色的政策。內嵌政策可在政策與身分之間維持嚴格的一對一關聯性。當您刪除身分時，它們即會被刪除。

若要了解如何選擇受管政策和內嵌政策的相關資訊，請參閱 在受管政策與內嵌政策之間選擇。

資源型政策

資源型政策是連接到資源 (如 Amazon S3 儲存貯體) 的 JSON 政策文件。這些政策會授予指定的主體許可，允許在該資源上執行特定的動作，並且定義資源所適用的條件。資源型政策是內嵌政策。不存在受管的資源型政策。

若要啟用跨帳戶存取，您可以指定在其他帳戶內的所有帳戶或 IAM 實體，作為資源型政策的主體。新增跨帳戶主體至資源型政策，只是建立信任關係的一半。當委託人和資源是分開的 AWS 帳戶，您還必須使用以身分為基礎的政策來授予委託人對資源的存取權。不過，如果資源型政策會為相同帳戶中的主體授予存取，這時就不需要額外的身分型政策。如需授予跨服務存取權限的逐步指示，請參閱 IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權。

IAM 服務支援一種稱為角色信任政策，且已連接到 IAM 角色的資源型政策。IAM 角色既是一種身分，也是一項資源，可支援資源型政策。因此，您必須同時將信任政策和身分型政策連接到 IAM 角色。信任政策會定義哪些主體實體 (帳戶、使用者、角色和聯合身分使用者) 可擔任該角色。若要了解 IAM 角色與其他以資源為基礎之政策之間的差別，請參閱 IAM 中的跨帳戶資源存取。

若要查看哪些其他服務可以支援資源型政策，請參閱AWS 使用 IAM 的 服務。若要進一步了解資源型政策的詳細資訊，請參閱 以身分為基礎和以資源為基礎的政策。若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？

VPC 端點政策

VPC 端點政策是一種以資源為基礎的政策，您可以連接到 VPC 端點，以控制哪些主體可以使用端點，以及哪些資源可以透過它存取。VPC 端點政策不會覆寫或取代連接到目的地服務的身分型政策或資源型政策，它們充當範圍為周遊端點之流量的額外存取界限。

VPC 端點政策用於確保組織中 AWS Organizations 只有信任的身分才能透過 VPC 端點存取信任的資源。如果您未連接自訂端點政策， 會 AWS 連接允許完整存取的預設政策。如需端點政策的詳細資訊，請參閱《 AWS PrivateLink 指南》中的使用端點政策控制對 VPC 端點的存取。

IAM 許可界限

許可界限是一種進階功能，可供您設定身分型政策可以授予 IAM 實體的最大許可。當您為實體設定許可界限時，實體只能執行由身分型政策和其許可界限同時允許的動作。

如果您在資源型政策的主體元素中指定了角色工作階段或使用者，則在許可界限中不需要明確允許。不過，如果您在資源型政策的主體元素中指定了角色 ARN，則在許可界限中需要明確允許。在這兩種情況下，許可界限中的明確拒絕都有效。任何這些政策中的明確拒絕都會覆寫允許。如需有關許可界限的詳細資訊，請參閱 IAM 實體的許可界限。

AWS Organizations 服務控制政策 SCPs)

若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 是 JSON 政策，負責指定組織或組織單位 (OU) 帳戶內的 IAM 使用者和 IAM 角色的最大許可。SCP 會限制成員帳戶中主體的許可，包括每個主體 AWS 帳戶根使用者。所有這類政策中的明確拒絕都會覆寫其他政策中的允許。

如需 AWS Organizations 和 SCPs的詳細資訊，請參閱AWS Organizations 《 使用者指南》中的服務控制政策 (SCPs)。

AWS Organizations 資源控制政策 RCPs)

如果您啟用組織中的所有功能，則可以使用資源控制政策 (RCP) 集中套用存取控制到多個 AWS 帳戶的資源。RCP 是 JSON 政策，可用來設定您帳戶中資源的可用許可上限，採取這種方式就不需要更新附加至您所擁有的每個資源的 IAM 政策。RCP 會限制成員帳戶中資源的許可，並可能影響身分的有效

許可，包括 AWS 帳戶根使用者，無論它們是否屬於您的組織。任何適用 RCP 中的明確拒絕都會覆寫其他政策中的允許，這些政策可能連接到個別身分或資源。

如需 AWS Organizations 和 RCPs的詳細資訊 AWS 服務 ，包括支援 RCPs的 清單，請參閱AWS Organizations 《 使用者指南》中的資源控制政策 (RCPs)。

存取控制清單 (ACL)

存取控制清單 (ACL) 是一種服務政策，可讓您控制另一個帳戶中的哪些主體可以存取資源。ACL 不能用於控制相同帳戶中主體的存取。ACL 類似資源型政策，雖然是不使用 JSON 政策文件格式的唯一政策類型。Amazon S3 AWS WAF和 Amazon VPC 是支援 ACLs的服務範例。如需進一步瞭解 ACL，請參閱《Amazon Simple Storage Service 開發人員指南》中的存取控制清單 (ACL) 概觀。

AWS 資源存取管理員 (AWS RAM) 資源共用

AWS Resource Access Manager (AWS RAM) 可讓您 AWS 帳戶 與 中的其他 AWS 帳戶、組織單位或整個組織共用您在其中建立的資源 AWS Organizations，而無需為每個共用資源撰寫個別資源型政策。雖然您可以透過直接將資源型政策連接到資源來授予跨帳戶存取權，但 AWS RAM 提供受管、集中的替代方案，無需列舉政策中的帳戶 IDs 或跨帳戶維護相同的政策文件。

使用 RAM 時，取用帳戶會在其服務主控台中原生看到共用資源，資源擁有者會保留對可存取者的完整擁有權和可見性，而受管許可會定義消費者可執行的最大動作，所有動作都由單一資源共用管理，而非依資源政策管理。常見的使用案例包括跨帳戶共用 VPC 子網路、 AWS Transit Gateway 附件、Route 53 Resolver 規則和 License Manager 組態。如需詳細資訊，請參閱什麼是 AWS Resource Access Manager？ AWS RAM 使用者指南中的 。

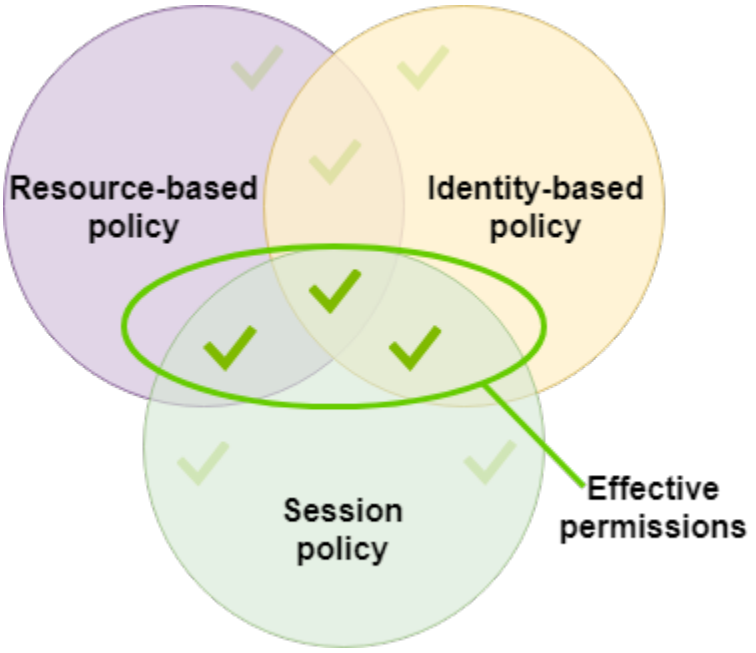
工作階段政策

當您以程式設計方式為角色或 AWS STS 聯合身分使用者主體建立臨時工作階段時，工作階段政策是做為參數傳遞的進階政策。工作階段的許可是用來建立工作階段及工作階段政策 IAM 實體 (使用者或角色) 之身分型政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。

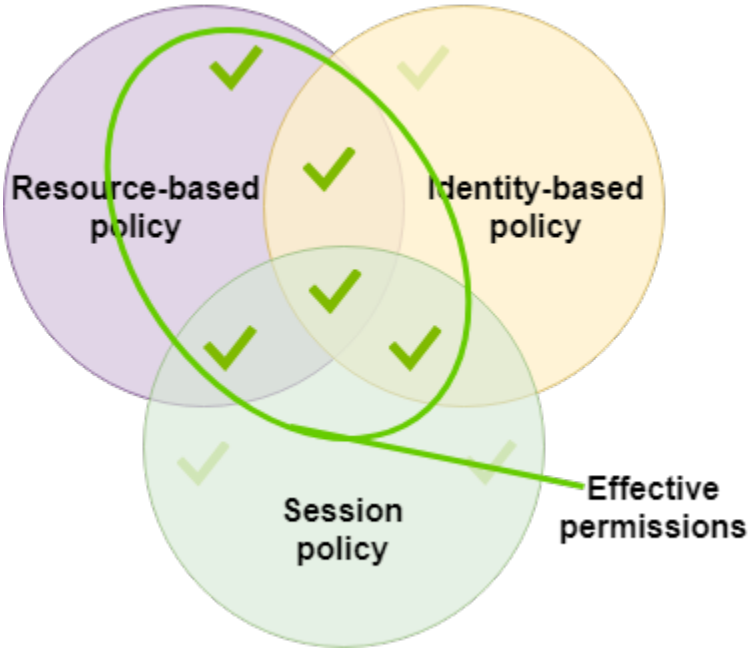
您可以建立角色工作階段，以及透過程式設計方式使用 `AssumeRole`、`AssumeRoleWithSAML` 或 `AssumeRoleWithWebIdentity` API 操作來傳遞工作階段政策。您可以使用 `Policy` 參數傳遞單一 JSON 內嵌工作階段政策文件。您可以使用 `PolicyArns` 參數，指定多達 10 個受管工作階段政策。如需有關建立角色工作階段的詳細資訊，請參閱 臨時安全憑證的許可。

當您建立 AWS STS 聯合身分使用者主體工作階段時，您可以使用 IAM 使用者的存取金鑰，以程式設計方式呼叫 `GetFederationToken` API 操作。您也必須通過工作階段政策。所產生工作階段的許可會是身分類型政策和工作階段政策的交集。如需有關建立聯合身分使用者工作階段的詳細資訊，請參閱透過自訂身分經紀人請求憑證。

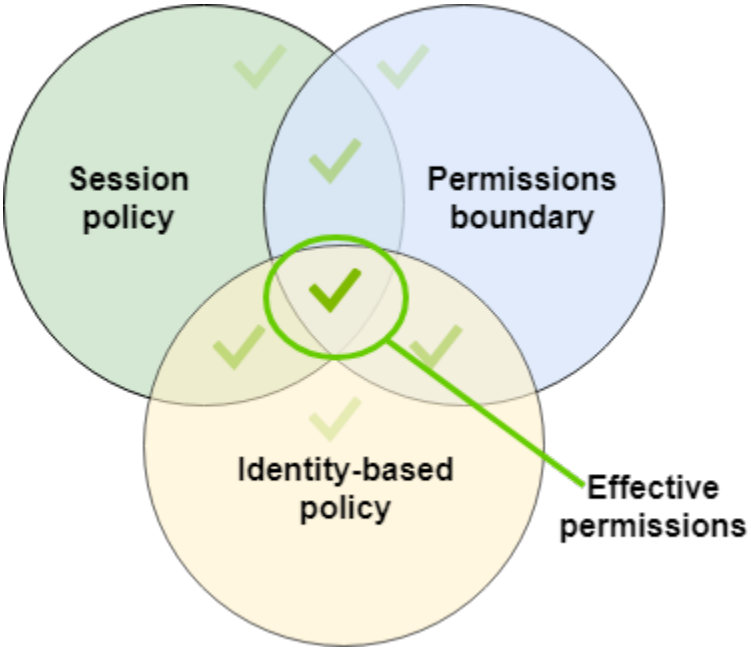
資源型政策可以指定使用者或角色 ARN 作為主體。在該情況下，在工作階段建立之前，依據資源型政策的許可會新增到該角色或使用者的身分型政策。工作階段政策限制由資源型政策、身分型政策所授予的總許可數。產生的工作階段的許可是工作階段政策與資源型政策的交集，加上工作階段政策與身分型政策的交集。



資源型政策可以指定工作階段 ARN 作為主體。在該情況下，會在建立工作階段後，從資源型政策新增許可。資源型政策許可不會受到工作階段政策的限制。產生的工作階段擁有資源型政策的所有許可，加上身分型政策與工作階段政策的交集。



許可界限可以為使用者或角色設定建立工作階段的許可上限。在該情況下，產生的工作階段的許可是工作階段政策、許可界限與身分型政策的交集。不過，許可界限不會限制由資源型政策所授予的許可，指定產生工作階段的 ARN。



政策和根使用者

受某些政策類型影響，但不受其他政策類型 AWS 帳戶根使用者 影響。您無法將身分型政策連接到根使用者，而且您無法為根使用者設定許可界限。不過，您可以在資源型政策或 ACL 中將根使用者指定為主體。根使用者仍是帳戶的成員。如果該帳戶是 中組織的成員 AWS Organizations，根使用者會受到帳戶的 SCPs 和 RCPs影響。

JSON 政策概觀

大多數政策會以 JSON 文件 AWS 形式存放在 中。身分型政策以及用於設定許可界限的政策，都是您連接到使用者或角色的 JSON 政策文件。資源型政策是連接到資源的 JSON 政策文件。SCPs和 RCPs是您連接到 AWS Organizations組織根目錄、組織單位 (OU) 或帳戶的受限語法的 JSON 政策文件。ACL 也會連接到資源，但您必須使用不同的語法。工作階段政策是您在擔任角色或聯合身分使用者工作階段時所需要提供的 JSON 政策。

您不需要了解 JSON 語法。您可以使用 中的視覺化編輯器 AWS 管理主控台 來建立和編輯客戶受管政策，而無需使用 JSON。不過，如果將內嵌政策用於群組或複雜政策，您仍然必須在 JSON 編輯器中利用主控台建立和編輯這些政策。如需有關使用視覺化編輯器的詳細資訊，請參閱 使用客戶管理政策定義自訂 IAM 許可 和 編輯 IAM 政策。

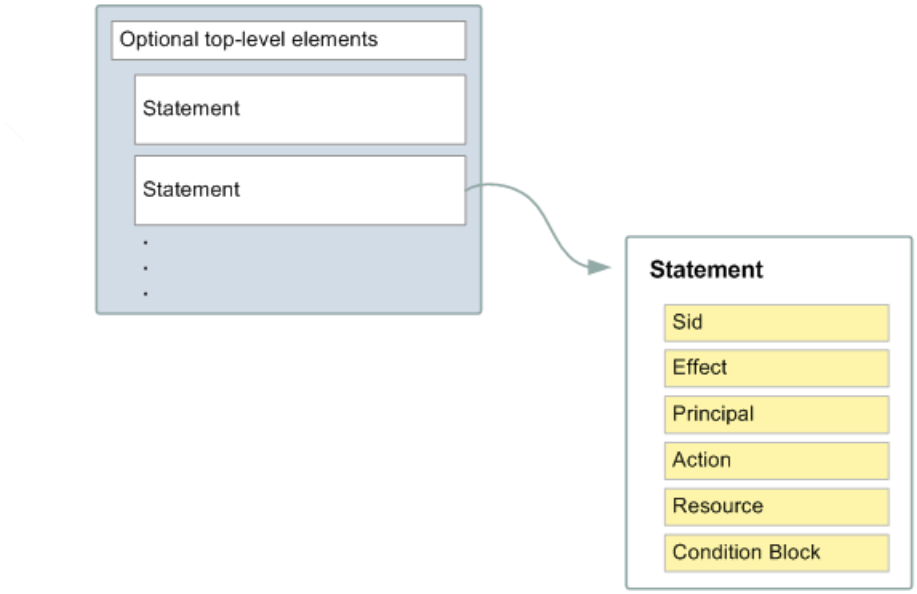
當您建立或編輯 JSON 政策時，IAM 可以執行政策驗證以協助您建立有效的政策。IAM 會識別 JSON 語法錯誤，而 IAM Access Analyzer 會提供額外的政策檢查及建議，協助您進一步改良政策。若要進一步了解政策驗證的資訊，請參閱 IAM 政策驗證。若要進一步了解 IAM Access Analyzer 政策檢查和可動作的建議，請參閱 IAM Access Analyzer 政策驗證。

JSON 政策文件結構

如下圖所示，JSON 政策文件包含這些元素：

- 在文件最上方選用的整體政策資訊
- 一或多個個別的陳述式

每個陳述式包含關於單一許可的資訊。如果政策包含多個陳述式， 會在評估陳述式時OR跨陳述式 AWS 套用邏輯。如果多個政策適用於請求， 會在評估所有OR政策時 AWS 套用邏輯。



陳述式中的資訊包含在一系列的元素內。

- Version – 指定您要使用的政策語言版本。建議您使用最新的 2012-10-17 版本。如需詳細資訊，請參閱IAM JSON 政策元素：Version
- Statement – 使用此主要政策元素作為以下元素的容器。您可以在政策中包含多個陳述式。
- Sid (選用) – 包含選用的陳述式 ID 來區分您的陳述式。
- Effect – 使用 Allow 或 Deny 來表示政策是否允許或拒絕存取。

- Principal （在某些情況下為必要） – 如果您建立以資源為基礎的政策，您必須指定您想要允許或拒絕存取的帳戶、使用者、角色或 AWS STS 聯合身分使用者主體。如果您要建立 IAM 許可政策來連接到使用者或角色，您不能包含此元素。主體意味著該使用者或角色。
- Action – 包含政策允許或拒絕的動作清單。
- Resource (在某些情況下需要) – 如果您建立 IAM 許可政策，則必須指示要套用動作的資源清單。如果您建立資源型政策，則取決於您用來判斷是否需要此元素的資源。
- Condition (選用) – 指定政策授予許可的條件。

若要了解這些內容及其他更進階的政策元素，請參閱IAM JSON 政策元素參考。

多項陳述式和多個政策

如果您想要定義一個以上的實體 (使用者或角色) 許可，您可以在單一政策使用多個陳述式。您也可以連接多個政策。如果您嘗試在單一陳述式定義多個許可，您的政策可能不會授予您預期的存取權。建議您透過資源類型將政策分類。

由於政策的大小限制，可能需要針對更複雜的許可採用多重政策。在不同的客戶管理政策中建立許可的功能群組，也是不錯的想法。例如，建立一個政策給 IAM 使用者管理、一個給自我管理，另一個給 S3 儲存貯體管理。無論多個陳述式和多個政策的組合為何， AWS 都會以相同的方式評估您的政策。

例如，以下政策具有三個陳述式，每一個定義單一帳戶內不同組的許可。陳述式定義下列動作：

- 第一種陳述式具有 `Sid` 的 `FirstStatement` (陳述式 ID)，可讓具有連接政策的使用者變更自己的密碼。此陳述式的 `Resource` 元素是 `*` (其表示「所有資源」)。但事實上 `ChangePassword`API 操作 (或同等 `change-password` CLI 命令) 只對提出請求的使用者的密碼有影響。
- 第二個陳述式可讓使用者在其 AWS 帳戶列出所有的 Amazon S3 儲存貯體。此陳述式的 `Resource` 元素是 `"*"` (其表示「所有資源」)。但是，因為政策不授予其他帳戶的資源存取權，使用者只能列出自己的 AWS 帳戶中的儲存貯體。
- 第三個陳述式可讓使用者列出並擷取儲存貯體中的任何物件，名為 `amzn-s3-demo-bucket-confidential-data`，但前提是使用者必須是以 多重要素驗證 (MFA) 驗證。政策中的 `Condition` 元素強制執行 MFA 身分驗證。

  當政策陳述式包含 `Condition` 元素，陳述式僅於 `Condition` 元素評估為 true 是生效。在這種情況下，當使用者經過 MFA 身分驗證，`Condition` 評估為 true。如果使用者未經過 MFA 身分驗證，這項 `Condition` 評估設為 false。在這種情況下，此政策的第三個陳述式將不適用，而且使用者無法存取 `amzn-s3-demo-bucket-confidential-data` 儲存貯體。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data",
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

JSON 政策語法的範例

以下身分型政策允許暗示的主體列出單一 Amazon S3 儲存貯體，其名稱是 `amzn-s3-demo-bucket`：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
  }
}
```

以下資源型政策可以連接到 Amazon S3 儲存貯體。此政策允許特定 的成員 AWS 帳戶 在名為 的儲存貯體中執行任何 Amazon S3 動作amzn-s3-demo-bucket。它允許可在儲存貯體或其中的物件上執行的任何動作。(由於政策僅對帳戶授予信任，帳戶中的個別使用者仍必須被授予特定 Amazon S3 動作的許可)。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

若要檢視常見案例的範例政策，請參閱以身分為基礎的 IAM 政策範例。

授予最低權限

當您建立 IAM 政策時，請遵循授予最低權限的標準安全建議，或者只授予執行任務所需的許可。決定使用者和角色需要執行哪些任務，然後為他們制定政策，讓使用者只執行這些任務。

從一組最低許可開始，並視需要授予其他許可。這比一開始使用太寬鬆的許可，稍後再嘗試將他們限縮更為安全。

作為最低權限的替代方案，您可以使用 AWS 受管政策或具有萬用字元 * 許可的政策，以開始使用政策。若授與主體的許可超出執行任務所需的許可，需考量其相關的安全性風險。監控這些主體，以了解他們正在使用的許可。然後寫入最低權限政策。

IAM 提供數個選項，協助您精簡您授予的許可。

- 了解存取層級群組 – 您可以使用存取層級的方法進行分組，以了解政策授予的存取層級。政策動作分為 `List`、`Read`、`Write`、`Permissions management` 或 `Tagging`。例如，您可以從 `List` 和 `Read` 存取層級選擇動作以授予唯讀存取權給您的使用者。若要了解如何使用政策摘要，以理解存取層級許可，請參閱政策摘要中的存取層級。
- 驗證您的政策 – 您可以在建立和編輯 JSON 政策時，使用 IAM Access Analyzer 執行政策驗證。我們建議您檢閱並驗證所有現有政策。IAM Access Analyzer 提供超過 100 項的政策檢查，以驗證您的政策。當您政策中的聲明允許我們認為過度寬鬆的存取權時，即會產生安全性警告。在授予最低權限時，您可以使用透過安全性警告提供的可操作建議。若要進一步了解有關 IAM Access Analyzer 所提供的政策檢查，請參閱 IAM Access Analyzer 政策驗證。
- 根據存取活動產生政策 – 為了協助您精簡所授予的許可，您可以根據 IAM 實體 (使用者或角色) 的存取活動產生 IAM 政策。IAM Access Analyzer 會檢閱您的 AWS CloudTrail 日誌，並產生政策範本，其中包含實體在指定時間範圍內已使用的許可。您可以使用範本建立具有精細許可的受管政策，然後將其連接至 IAM 實體。如此一來，您只授予使用者或角色與特定使用案例 AWS 的資源互動所需的許可。若要進一步了解，請參閱 IAM Access Analyzer 政策產生。
- 使用上次存取的資訊 – 另一個有助於實現最低權限的功能是最近存取的資訊。在 IAM 主控台詳細資訊頁面的 Access Advisor (存取 Advisor) 索引標籤上檢視 IAM 使用者、群組、角色或政策的此資料。上次存取的資訊也包括最近存取某些服務之動作的相關資訊，例如 Amazon EC2、IAM、Lambda 和 Amazon S3。如果您使用 AWS Organizations 管理帳戶登入資料登入，您可以在 IAM 主控台的 AWS Organizations區段中檢視服務上次存取的資訊。您也可以使用 AWS CLI 或 AWS API，擷取 IAM 或 中實體或政策上次存取資訊的報告 AWS Organizations。您可以使用此資訊來識別不必要的許可，以便精簡您的 IAM 或 AWS Organizations 政策，以更好地遵守最低權限原則。如需詳細資訊，請參閱AWS 使用上次存取的資訊在 中精簡許可。
- 在 中檢閱帳戶事件 AWS CloudTrail – 若要進一步減少許可，您可以在事件歷史記錄中 AWS CloudTrail 檢視帳戶的事件。 CloudTrail 事件日誌包含可供您用來縮減政策之許可的詳細事件資

訊。此日誌只包含 IAM 實體所需的動作和資源。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的在 CloudTrail 主控台中檢視 CloudTrail 事件。

如需詳細資訊，請參閱下列適用於個別服務的政策主題，其中提供如何為服務特定的資源撰寫政策的範例。

- 《Amazon DynamoDB 開發人員指南》中的 Using resource-based policies for DynamoDB
- 《Amazon Simple Storage Service 使用者指南》中的 Bucket policies for Amazon S3
- 《Amazon Simple Storage Service 使用者指南》中的 Access Control List (ACL) overview

受管政策與內嵌政策

當您在 IAM 中為身分設定許可時，必須決定要使用 AWS 受管政策、客戶管理政策還是內嵌政策。以下幾個主題提供了有關每個類型的身分型政策以及使用時機的更多資訊。

下表概述了這些政策：

政策類型	說明	政策由誰管理？	修改許可？	套用至政策的主體數量？
AWS 受管政策	由 AWS建立和管理的獨立政策。	AWS	否	許多
客戶管理政策	您針對特定使用案例建立的政策，並且您可以隨時更改或更新它們。	您	是	許多
內嵌政策	為單一 IAM 身分 (使用者、群組或角色) 建立的政策，其在政策與身分之間保持嚴格的一對一關係。	您	是	一

如需建立角色和連接這些政策的說明，請參閱 建立角色和連接政策 (主控台)。

主題

- AWS 受管政策
- 客戶管理政策
- 內嵌政策
- 在受管政策與內嵌政策之間選擇
- 將內嵌政策轉換為受管政策
- 已取代的 AWS 受管政策

AWS 受管政策

「AWS 受管政策」為獨立的政策，由 AWS建立並管理。獨立政策表示政策有自己的 Amazon Resource Name (ARN)，其中包含政策名稱。例如， `arn:aws:iam::aws:policy/IAMReadOnlyAccess` 是 AWS 受管政策。如需有關 ARN 的詳細資訊，請參閱 IAM ARN。如需 的 AWS 受管政策清單 AWS 服務，請參閱 AWS 受管政策。

AWS 受管政策可讓您輕鬆地將適當的許可指派給使用者、IAM 群組和角色。這比您自己撰寫政策更快，並且包含常見使用案例的許可。

您無法變更 AWS 受管政策中定義的許可。 AWS occasion 會更新 AWS 受管政策中定義的許可。當 AWS 這樣做時，更新會影響政策連接的所有委託人實體 (IAM 使用者、IAM 群組和 IAM 角色）。當新 AWS 服務啟動或新 API 呼叫可用於現有服務時， AWS 最有可能更新 AWS 受管政策。例如，名為 ReadOnlyAccess 的 AWS 受管政策提供所有 AWS 服務 和 資源的唯讀存取權。 AWS 啟動新服務時，會 AWS 更新 ReadOnlyAccess 政策，為新服務新增唯讀許可。更新的許可會套用於政策連接到的所有主體實體。

完整存取 AWS 受管政策：這些政策透過授予服務的完整存取權來定義服務管理員的許可。範例包括：

- AmazonDynamoDBFullAccess
- IAMFullAccess

進階使用者 AWS 受管政策：這些政策提供對 AWS 服務 和資源的完整存取權，但不允許管理使用者和 IAM 群組。範例包括：

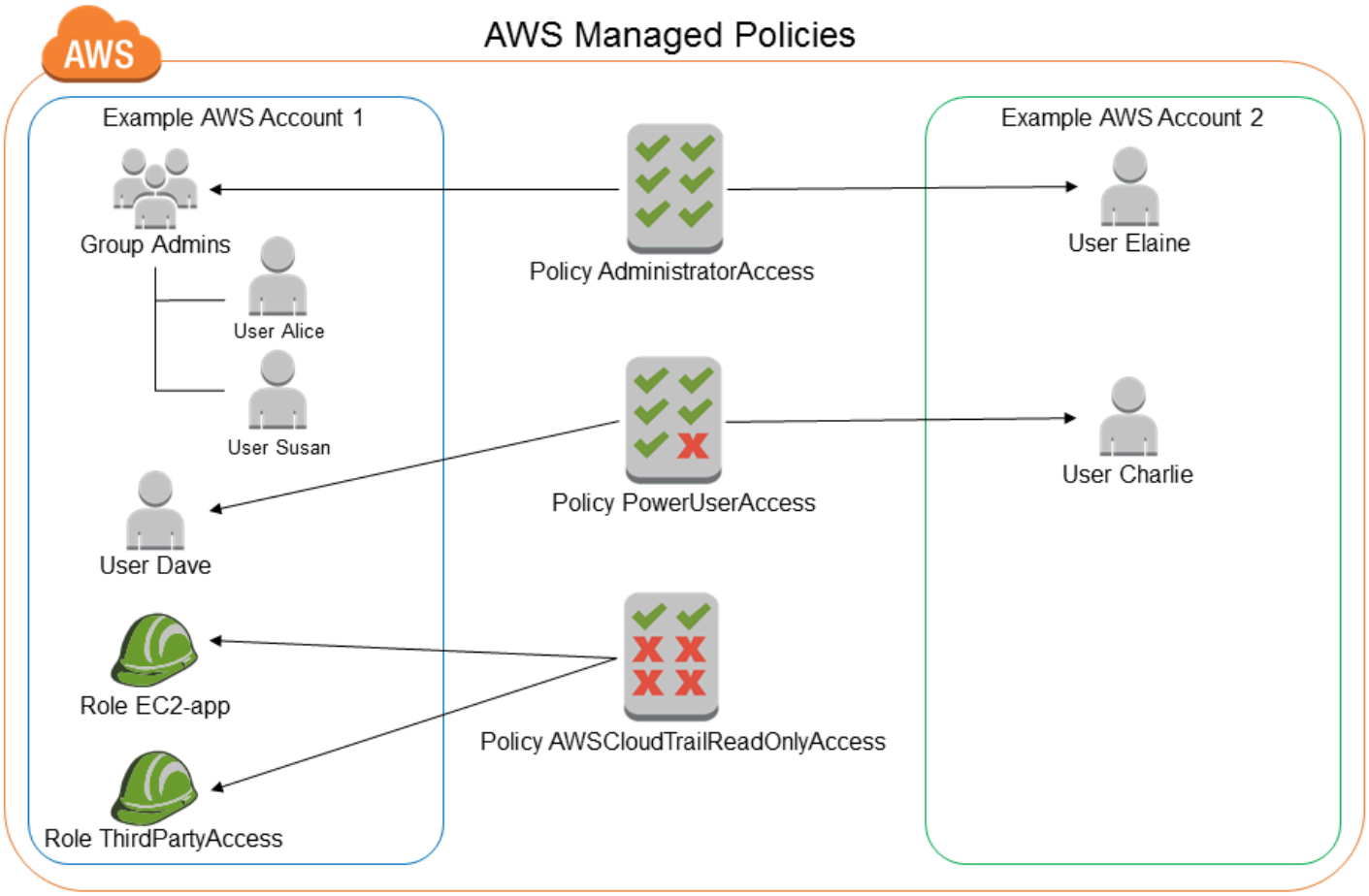
- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

部分存取 AWS 受管政策：這些政策提供對 的特定存取層級， AWS 服務 而不允許許可管理存取層級許可。範例包括：

- AmazonMobileAnalyticsWriteOnlyAccess
- AmazonEC2ReadOnlyAccess

工作職能 AWS 受管政策：這些政策與 IT 產業中常用的工作職能密切一致，並有助於授予這些工作職能的許可。使用任務函數政策的一個主要優點是，隨著新服務和 API 操作的推出 AWS ，它們由 維護和更新。例如，AdministratorAccess 工作職能提供對 AWS中的每個服務和資源的完整存取權和許可委派。我們建議僅將此政策用於帳戶管理員。對於需要完整存取每個服務的進階使用者，但對 IAM 和 的有限存取除外 AWS Organizations，請使用 PowerUserAccess 任務函數。如需有關工作職能政策的清單和說明，請參閱AWS 任務函數的 受管政策。

下圖說明 AWS 受管政策。該圖顯示三個 AWS 受管政策：AdministratorAccess、PowerUserAccess 和 AWS CloudTrail_ReadOnlyAccess。請注意，單一 AWS 受管政策可以連接到不同 中的委託人實體 AWS 帳戶，也可以連接到單一 中的不同委託人實體 AWS 帳戶。



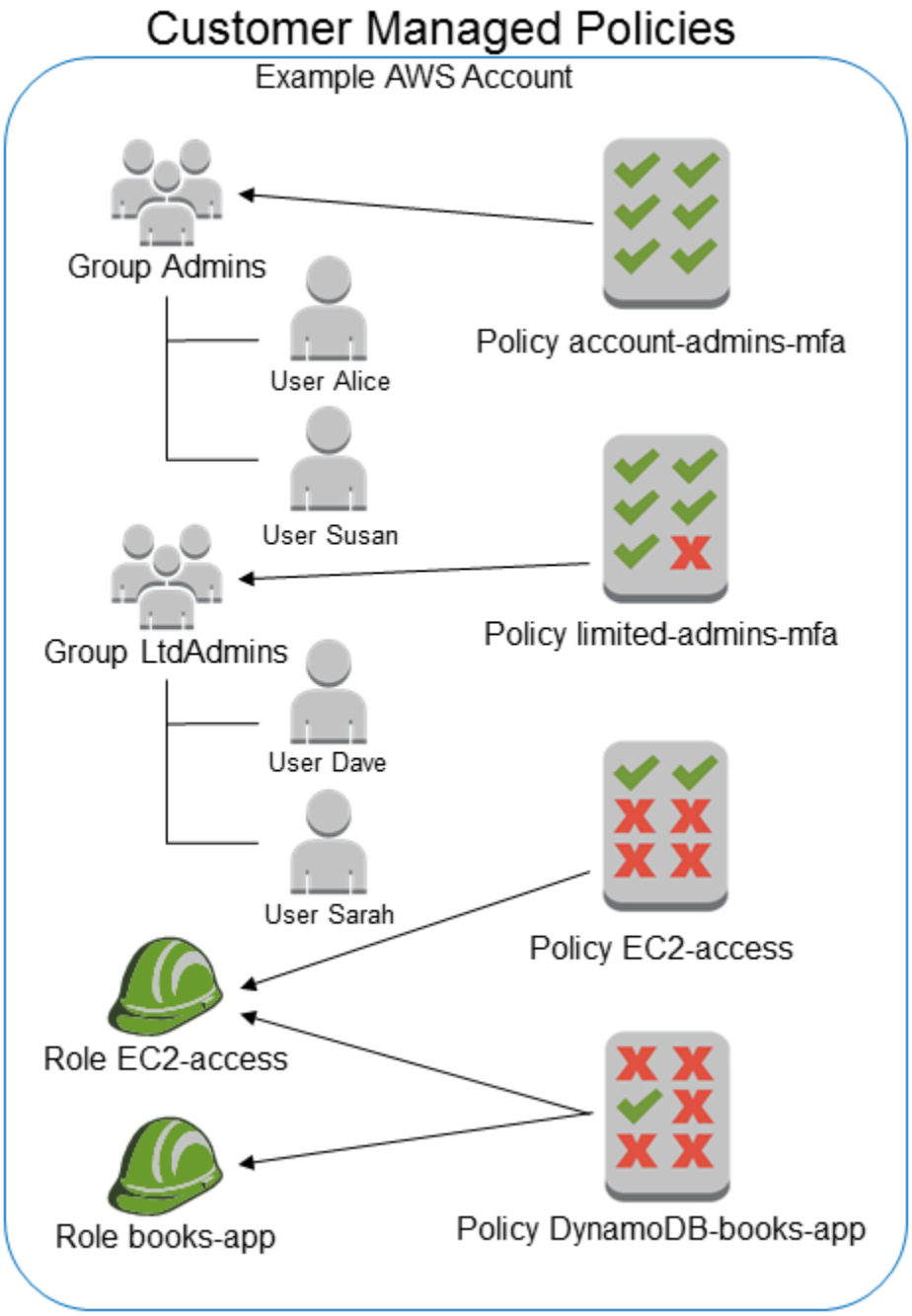
客戶管理政策

您可以在自己的 中建立獨立政策 AWS 帳戶 ，以連接到委託人實體 (IAM 使用者、IAM 群組和 IAM 角色）。您可以針對特定使用案例建立這些客戶管理政策，並且可以隨時進行變更和更新。如同 AWS 受管政策，當您將政策連接至委託人實體時，您會為實體提供政策中定義的許可。更新政策中的許可時，變更會套用於政策連接到的所有主體實體。

建立客戶管理政策的理想方式是從複製一個現有 AWS 受管政策開始。這樣從一開始您就可以確信政策是正確的，只需根據您的環境進行自訂即可。

下圖說明客戶管理政策。每個政策都是 IAM 中的一個實體，有自己的 Amazon Resource Name (ARN)，其中包含政策名稱。請注意，同一政策可以連接到多個主體實體，例如，同一個 DynamoDB-books-app 政策可連接到兩個不同的 IAM 角色。

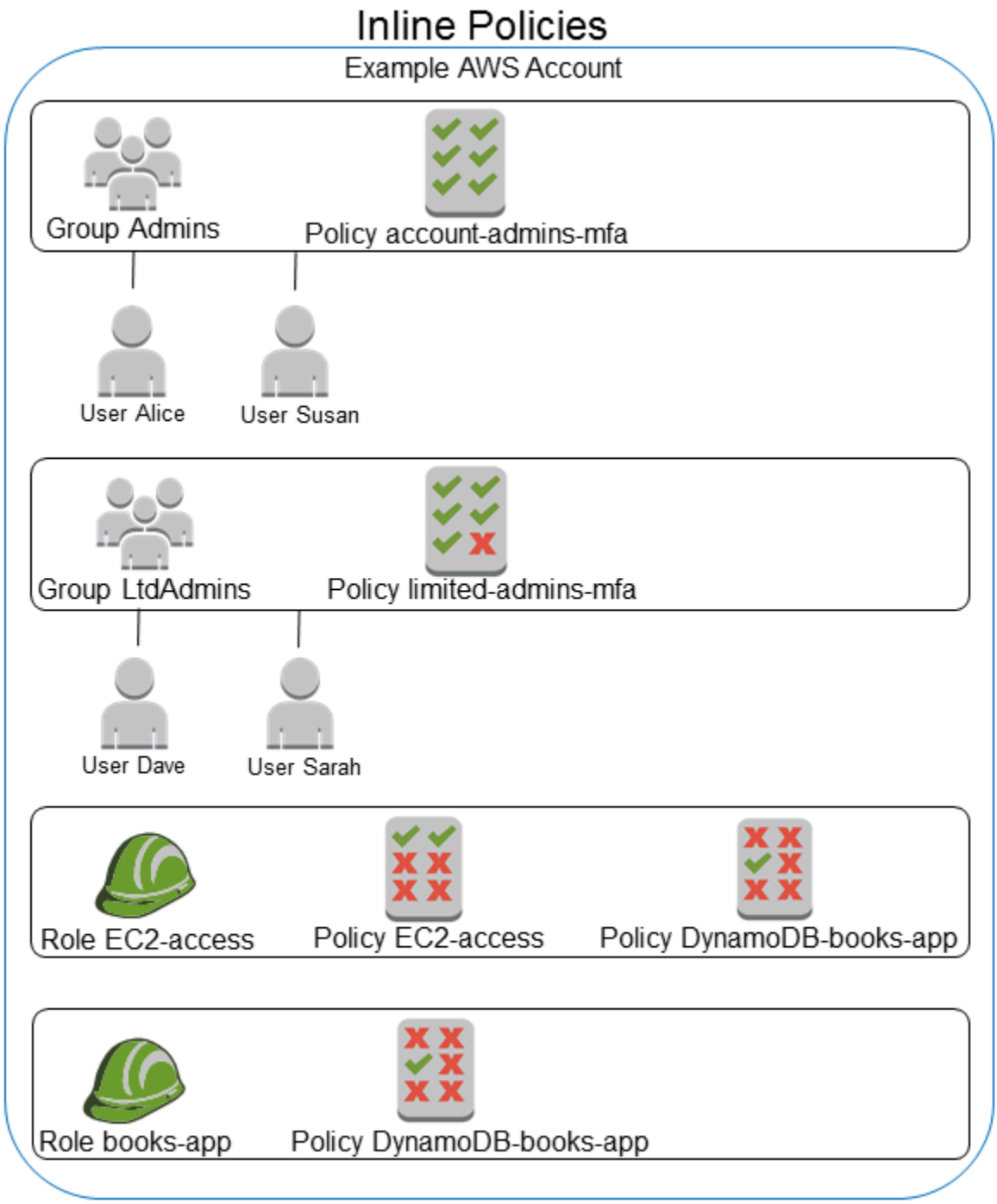
如需詳細資訊，請參閱使用客戶管理政策定義自訂 IAM 許可



內嵌政策

內嵌政策是為單個 IAM 身分 (使用者、使用者群組或角色) 建立的政策。內嵌政策可在政策與身分之間維持嚴格的一對一關聯性。當您刪除身分時，它們即會被刪除。在建立身分時或在建立之後，您可以建立政策並將其嵌入身分。如果政策可套用至多個實體，最好使用受管政策。

下圖說明內嵌政策。每個政策都是使用者、組或角色的固有部分。請注意，兩個角色包含同一政策 (DynamoDB-books-app 政策)，但是它們不共用單一政策。每個角色都有自己的政策複本。



在受管政策與內嵌政策之間選擇

在決定是使用受管政策還是內嵌政策時，考慮您的使用案例。在大多數情況下，我們建議使用受管政策而非內嵌政策。

> **Note**
>
> 您可以同時使用受管政策和內嵌政策，來定義主體實體的一般許可和唯一許可。

受管政策具備以下功能：

可重複使用性

單一受管政策可以連接到多個主體實體 (使用者、群組和角色)。您可以建立政策程式庫，為 定義有用的許可 AWS 帳戶，然後視需要將這些政策連接至委託人實體。

集中變更管理

更改受管政策時，更改會套用於政策連接到的所有主體實體。例如，如果您想要新增新 AWS API 的許可，您可以更新客戶受管政策，或建立 AWS 受管政策的關聯以新增許可。如果您使用的是 AWS 受管政策， 會 AWS 更新政策。更新受管政策時，更改會套用於受管政策連接到的所有主體實體。反之，若要變更內嵌政策，您必須分別編輯包含該內嵌政策的每個身分。例如，如果一個群組和一個角色都包含同一內嵌政策，則您必須分別編輯這兩個主體實體以變更該政策。

版本控制和還原

在更改客戶管理政策時，更改的政策不會覆蓋現有的政策。IAM 反而會建立新版本的受管政策。IAM 最多可以儲存五個版本的客戶管理政策。如果需要，可以使用政策版本將政策還原為較早版本。

> **Note**
>
> 政策版本與 `Version` 政策元素不同。`Version` 政策元素是在政策內使用，並定義政策語言的版本。若要進一步了解政策版本，請參閱 the section called “版本控制 IAM 政策”。若要進一步了解 `Version` 政策元素，請參閱 IAM JSON 政策元素：Version。

委派許可管理

您可以允許 中的使用者 AWS 帳戶 連接和分離政策，同時控制這些政策中定義的許可。因此，您可以將一些使用者指定為完全管理員，也就是建立、更新和刪除政策的管理員。隨後可以將其使用者指定為受限管理員。受限管理員可以將政策 (限於您允許連接的政策) 連接到其他主體實體。

如需有關委派許可管理的詳細資訊，請參閱 控制對政策的存取。

較大政策字元限制

受管政策的字元大小上限大於群組內嵌政策的字元限制。如果達到內嵌政策的字元大小限制，您可以建立更多 IAM 群組，並將受管政策附加到群組。

如需有關配額和限制的詳細資訊，請參閱 IAM AWS STS 和配額。

AWS 受管政策的自動更新

AWS 會維護 AWS 受管政策，並視需要更新這些政策，例如新增新 AWS 服務的許可，而不必進行變更。更新會自動套用至您已連接 AWS 受管政策的主體實體。

受管政策入門

我們建議使用授予最低權限的政策，或僅授予執行任務所需的許可。授予最低權限的最安全方式是撰寫僅具有團隊所需許可的客戶管理政策。您必須建立程序，以允許您的團隊在必要時要求更多許可。建立 IAM 客戶受管政策需要時間和專業知識，而受管政策可為您的團隊提供其所需的許可。

若要開始將許可新增至 IAM 身分（使用者、使用者群組和角色），您可以使用 AWS 受管政策 AWS .managed 政策不會授予最低權限許可。若授予主體的許可超出執行任務所需的許可，您必須考量其相關的安全性風險。

您可以將 AWS 受管政策，包括任務函數，連接到任何 IAM 身分。如需詳細資訊，請參閱新增和移除 IAM 身分許可。

若要切換到最低權限許可，您可以執行 AWS Identity and Access Management 和 Access Analyzer 以使用 AWS 受管政策監控委託人。了解他們正在使用哪些許可後，您可以撰寫或產生僅具有團隊所需許可的客戶管理政策。這較不安全，但在您了解團隊使用 的方式時提供更多彈性 AWS。如需詳細資訊，請參閱產生 IAM Access Analyzer 政策。

AWS 受管政策旨在為許多常用案例提供許可。如需專為特定任務函數設計的 AWS 受管政策的詳細資訊，請參閱 AWS 任務函數的 受管政策。

如需 AWS 受管政策的清單，請參閱 AWS 受管政策參考指南。

使用內嵌政策

如果您要在政策與套用它的身分之間維持嚴格的一對一關聯性，則內嵌政策十分有用。例如，如果您要確保政策中的許可不會無意中分配給預期身分之外的身分。使用內嵌政策時，政策中的許可不能意外分配給錯誤的身分。此外，當您使用 AWS 管理主控台 刪除該身分時，也會刪除內嵌在身分中的政策，因為它們是委託人實體的一部分。

將內嵌政策轉換為受管政策

如果您在您的帳戶中擁有內嵌政策，您可以將它們轉換為受管政策。若要執行此操作，請將該政策複製到新的受管政策。接下來，將新政策連接到具有內嵌政策的身分，然後刪除內嵌政策。

將內嵌政策轉換為受管政策

將內嵌政策轉換為受管政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在服務導覽窗格中，選擇 User groups (使用者群組)、Users (使用者) 或者 Roles (角色)。
3. 在清單中，選擇包含要刪除的政策的使用者群組、使用者、或角色的名稱。
4. 選擇 Permissions (許可) 標籤。
5. 針對 IAM 群組，選取要移除之內嵌政策的名稱。針對使用者和角色，選擇顯示 更多項目，並在必要時展開您想要移除之內嵌政策。
6. 選擇複製，以複製該政策的 JSON 政策文件。
7. 在導覽窗格中，選擇政策。
8. 選擇建立政策，然後選擇 JSON 選項。
9. 將現有的文字取代為您的 JSON 政策文字，然後選擇下一步。
10. 輸入您的政策的名稱和選用描述，然後選擇建立政策。
11. 在導覽窗格中選擇 User groups (使用者群組)、Users (使用者) 或 Roles (角色)，並再次選擇擁有您想要移除之政策的使用者群組、使用者或角色的名稱。
12. 選擇許可索引標籤，然後選擇新增許可。
13. 針對 IAM 群組，選取新政策名稱旁的核取方塊，然後依序選擇新增許可、連接政策。針對使用者或角色，選擇 Add permissions (新增許可)。在下一個頁面上，選擇直接連接現有政策，選取新政策名稱旁的核取方塊，選擇下一步，然後選擇新增許可。

    系統會將您傳回到使用者群組、使用者或角色的 Summary (摘要) 頁面。
14. 選取您要移除之內嵌政策旁的核取方塊，然後選擇移除。

已取代的 AWS 受管政策

為了簡化許可的指派， AWS 提供可連接到 IAM 使用者、群組和角色的預先定義政策。

有時 AWS 需要將新許可新增至現有政策，例如引進新服務時。將新的許可新增至現有政策，並不會中斷或移除任何功能或能力。

不過，如果所需的變更套用到現有政策，則 AWS 可能會選擇建立新的政策，而這些變更可能會影響客戶。例如，從現有政策移除許可可能破壞任何 IAM 實體或應用程式所倚靠的許可，而可能中斷關鍵的操作。

因此，當需要此類變更時， AWS 會建立具有所需變更的全新政策，並將其提供給客戶。然後，舊政策會標示為已作廢。如需詳細資訊，請參閱《 AWS 受管政策參考指南》中的已棄用AWS 受管政策。

使用資料周邊建立許可防護機制

資料周邊護欄旨在做為永遠在線的界限，以協助在廣泛的 AWS 帳戶和資源中保護您的資料。資料周邊遵循 IAM 安全最佳實務，跨多個帳戶建立許可防護機制。這些全組織的許可防護機制不會取代您現有的精細存取控制。相反地，它們用作粗略的存取控制，透過確保使用者、角色和資源遵守一組定義的安全標準，協助改善您的安全策略。

資料周邊是 AWS 環境中的一組許可護欄，可協助確保只有您信任的身分才能從預期的網路存取信任的資源。

- 信任的身分：您 AWS 帳戶中的委託人 (IAM 角色或使用者） 和代表您行事 AWS 的服務。
- 信任的資源： AWS 由您的帳戶或代表您的 AWS 服務所擁有的資源。
- 預期的網路：您的內部部署資料中心和虛擬私有雲端 (VPCs)，或代表您的 AWS 服務網路。

> **Note**
>
> 在某些情況下，您可能需要擴展資料周邊，以納入信任的業務合作夥伴的存取權。當建立特定於您的公司和您使用 AWS 服務的信任的身分、信任的資源和預期的網路之定義時，應當考慮所有預期的資料存取模式。

資料周邊控制應視為資訊安全和風險管理計畫中的任何其他安全控制。這表示您應該執行威脅分析來識別雲端環境中的潛在風險，然後根據您自己的風險接受標準，選取並實作適當的資料周邊控制項。為了更清楚地了解資料周邊實作的反覆風險型方法，您需要了解資料周邊控制以及您的安全優先事項會解決哪些安全風險和威脅向量。

資料周邊控制

資料周邊粗略控制透過實作不同的政策類型和條件索引鍵組合，協助您跨三個資料周邊達成六個不同的安全目標。

周邊	控制目標	使用	套用至	全域條件內容索引鍵
Identity	只有信任的身分可以存取我的資源	RCP	Resources	aws:PrincipalOrgID aws:PrincipalOrgPaths aws:PrincipalAccount aws:PrincipalIsAwsService aws:SourceOrgID aws:SourceOrgPath aws:SourceAccount
	我的網路只允許信任的身分	VPC 端點政策	網路	
Resources	您的身分只能存取信任的資源	SCP	身分	aws:ResourceOrgID aws:ResourceOrgPaths aws:ResourceAccount
	從您的網路只能存取信任的資源	VPC 端點政策	網路	
網路	您的身分只能從預期的網路存取資源	SCP	身分	aws:SourceIp aws:SourceVpc aws:SourceVpce
	您的資源只能從預期的網路存取	RCP	Resources	

周邊	控制目標	使用	套用至	全域條件內容索引鍵
				aws:VpceAccount aws:VpceOrgPaths aws:VpceOrgID aws:ViaAWSService aws:PrincipalIsAwsService

您可以將資料周邊視為在資料周圍建立堅實的界限，以防止意外存取模式。雖然資料周邊可以防止廣泛的意外存取，但您仍需要做出精細的存取控制決策。建立資料周邊並不會減少透過使用 IAM Access Analyzer 等工具作為最低權限旅程的一部分來持續微調許可的需求。

若要對 RCP 目前不支援的資源強制執行資料周邊控制，可以使用直接連接到資源的資源型政策。如需支援 RCP 和資源型政策的服務清單，請參閱 Resource control policies (RCPs) 和 AWS 使用 IAM 的服務。

若要強制執行網路周邊控制，建議您僅在要限制存取的所有服務目前都受支援的情況下才使用 `aws:VpceOrgID`、`aws:VpceOrgPaths` 和 `aws:VpceAccount`。將這些條件索引鍵用於不受支援的服務，可能會導致意外的授權結果。如需支援這些索引鍵的服務清單，請參閱 AWS 全域條件內容索引鍵。如果您需要對更廣泛的服務強制執行這些控制，請考慮改用 `aws:SourceVpc` 和 `aws:SourceVpce`。

身分周邊

身分周邊是一組粗略精細的預防性存取控制，可協助確保只有信任的身分可以存取您的資源，並且只允許從您的網路存取信任的身分。信任的身分通常包含您 AWS 帳戶中的委託人（角色或使用者），以及代表您行事 AWS 的服務。除非授予明確的例外狀況，否則所有其他身分都會被視為不受信任，且會受到身分周邊的阻止。

下列全域條件索引鍵有助於根據您對受信任身分的定義來強制執行身分周邊控制。在資源控制政策中使用這些索引鍵來限制對資源的存取，或在 VPC 端點政策中使用這些索引鍵來限制對網路的存取。

您擁有的身分

您可以使用下列條件索引鍵來定義您在 中建立和管理的 IAM 主體 AWS 帳戶。

- aws:PrincipalOrgID – 您可以使用此條件金鑰，確保提出請求的 IAM 主體屬於其中指定的組織 AWS Organizations。
- aws:PrincipalOrgPaths – 您可以使用此條件金鑰，以確保 IAM 使用者 、IAM 角色、 AWS STS 聯合身分使用者主體、SAML 聯合身分主體、OIDC 聯合身分主體，或 AWS 帳戶根使用者 提出請求屬於指定的組織單位 (OU) AWS Organizations。
- aws:PrincipalAccount – 您可以使用此條件金鑰，確保只有您在政策中指定的委託人帳戶才能存取資源。

代表您執行 AWS 的服務身分

您可以使用下列條件金鑰，允許 AWS 服務在代表您時，使用自己的身分來存取您的 資源。

- aws:PrincipalIsAWSService 和 aws:SourceOrgID (或 aws:SourceOrgPaths和 aws:SourceAccount) – 您可以使用這些條件金鑰，以確保當AWS 服務 委託人存取您的資源時，他們只會代表指定組織、組織單位或 帳戶中的資源執行此操作 AWS Organizations。

如需詳細資訊，請參閱在 上建立資料周邊 AWS：僅允許信任的身分存取公司資料。

資源周邊

資源周邊是一組粗略的預防性存取控制，可協助確保您的身分只能存取信任的資源，而且只能從您的網路存取信任的資源。信任的資源通常包含您 AWS 帳戶或代表您的 AWS 服務所擁有的資源。

下列全域條件索引鍵有助於根據您對受信任資源的定義來強制執行資源周邊控制。在服務控制政策 (SCP) 中使用這些索引鍵來限制您的身分可以存取的資源，或在 VPC 端點政策中使用這些索引鍵來限制可以從您的網路存取的資源。

您擁有的資源

您可以使用下列條件索引鍵來定義您在 中建立和管理 AWS 的資源 AWS 帳戶。

- aws:ResourceOrgID – 您可以使用此條件索引鍵，確保正在存取的資源屬於其中指定的組織 AWS Organizations。
- aws:ResourceOrgPaths – 您可以使用此條件金鑰，以確保正在存取的資源屬於其中指定的組織單位 (OU) AWS Organizations。

- aws:ResourceAccount – 您可以使用此條件金鑰，以確保正在存取的資源屬於指定的 AWS 帳戶。

代表您行事 AWS 的服務資源

在某些情況下，您可能需要允許存取 AWS 擁有的資源、不屬於您組織的資源，以及委託人或代表您的 AWS 服務所存取的資源。如需這些案例的詳細資訊，請參閱在 上建立資料周邊 AWS：僅允許來自組織的信任資源。

網路周邊

網路周邊是一組粗略的預防性存取控制，可協助確保您的身分只能從預期的網路存取資源，而且只能從預期的網路存取您的資源。預期網路通常包括您內部部署的資料中心和虛擬私有雲端 (VPC)，以及代表您執行動作的 AWS 服務的網路。

下列全域條件索引鍵有助於根據您對預期網路的定義來強制執行網路周邊控制。在服務控制政策 (SCP) 中使用這些索引鍵來限制您的身分可以通訊的網路，或在資源控制政策 (RCP) 中使用這些索引鍵來限制對預期網路的資源存取。

您擁有的網路

您可以使用下列條件索引鍵來定義員工和應用程式預期用來存取資源的網路，例如您的公司 IP CIDR 範圍和 VPC。

- aws:SourceIp – 您可以使用此條件金鑰，確保申請者的 IP 地址位於指定的 IP 範圍內。
- aws:SourceVpc – 您可以使用此條件金鑰，以確保請求通過的 VPC 端點屬於指定的 VPC。
- aws:SourceVpce – 您可以使用此條件金鑰，以確保請求通過指定的 VPC 端點。
- aws:VpceAccount – 您可以使用此條件金鑰來確保請求是透過指定 AWS 帳戶所擁有的 VPC 端點。
- aws:VpceOrgPaths – 您可以使用此條件索引鍵來確保請求是透過屬於 中指定組織單位 (OU) 的帳戶所擁有的 VPC 端點 AWS Organizations。
- aws:VpceOrgID – 您可以使用此條件金鑰來確保請求是透過指定組織中帳戶擁有的 VPC 端點 AWS Organizations。

`aws:VpceAccount`、`aws:VpceOrgPaths` 和 `aws:VpceOrgID` 尤其適用於實作隨著 VPC 端點用量自動擴展的網路周邊控制，而無需在建立新端點時更新政策。如需支援這些索引鍵的 AWS 服務 清單，請參閱 AWS 全域條件內容索引鍵。

代表您行事 AWS 的服務網路

您可以使用下列條件金鑰，允許 AWS 服務在代表您時從其網路存取您的資源。

- aws:ViaAWSService – 您可以使用此條件金鑰，以確保 AWS 服務 可以使用 轉送存取工作階段(FAS) 代表您委託人提出請求。
- aws:PrincipalIsAWSService – 您可以使用此條件金鑰，確保 AWS 服務 可以使用 存取您的 資源AWS 服務主體。

在另外一些案例中，您需要允許存取 AWS 服務 ，這些服務從您的網路外部存取您的資源。如需詳細資訊，請參閱在 上建立資料周邊 AWS：僅允許從預期的網路存取公司資料。

進一步了解資料周邊的資源

下列資源可協助您進一步了解跨 AWS的資料周邊。

- 上的資料周邊 AWS – 了解資料周邊及其優點和使用案例。
- 部落格文章系列：在 上建立資料周邊 AWS – 這些部落格文章涵蓋了有關大規模建立資料周邊的規範性指導，包括關鍵安全和實作考量。
- 資料周邊政策範例 – 此 GitHub 儲存庫包含的範例政策涵蓋了一些常見模式，可協助您在 AWS上實作資料周邊。
- 資料周邊協助程式 – 此工具可協助您透過分析 AWS CloudTrail 日誌中的存取活動，來設計和預測資料周邊控制的影響。

IAM 實體的許可界限

AWS 支援 IAM 實體 （使用者或角色） 的許可界限。許可界限是一種進階功能，可供您使用受管政策來設定以身分為基礎的政策可以授予 IAM 實體的最大許可。實體的許可界限可讓其只執行由身分類型政策和其許可界限同時允許的動作。

如需有關政策類型的詳細資訊，請參閱 政策類型。

> **Important**
>
> 對於已附加許可界限政策的 IAM 使用者或角色，請勿使用所含 `NotPrincipal` 政策元素具有 `Deny` 效果的資源型政策陳述式。無論在 `NotPrincipal` 元素中指定何值，具有 `Deny` 效果的 `NotPrincipal` 元素將始終拒絕任何已附加許可界限政策的 IAM 主體。這會造成部分本可存取資源的 IAM 使用者或角色失去存取權。

我們建議您變更資源型政策陳述式，將條件運算子 ArnNotEquals 與 aws:PrincipalArn 內容索引鍵搭配使用來限制存取，而不是使用 NotPrincipal 元素。如需有關 NotPrincipal 元素的資訊，請參閱 AWS JSON 政策元素： NotPrincipal。

您可以使用 AWS 受管政策或客戶受管政策來設定 IAM 實體（使用者或角色）的界限。該政策限制使用者或角色的許可上限。

例如，假設名為 Shirley 的 IAM 使用者應允許僅管理 Amazon S3、Amazon CloudWatch 和 Amazon EC2。為了強行實施這個規則，您可以使用以下政策設定 Shirley 使用者的許可界限：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

當您使用政策來設定使用者的許可界限時，它會限制使用者的許可，但不自行提供許可。在這個範例中，該政策會將 Shirley 的最大許可設為 Amazon S3、CloudWatch 和 Amazon EC2 中的所有操作。Shirley 永遠無法在其他任何服務中執行操作，包括 IAM，即使許可政策允許她這麼做。例如，您可以將以下政策加入到 Shirley 使用者：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

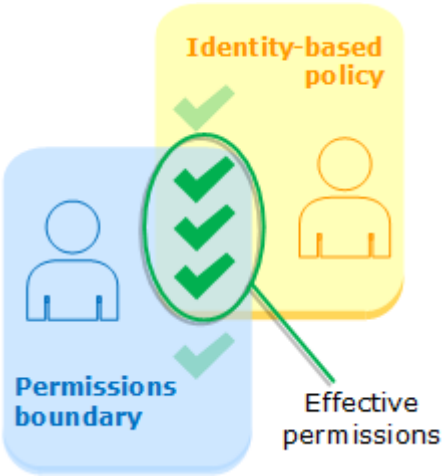
這項政策允許在 IAM 中建立使用者。如果您將此許可政策連接到 Shirley 使用者，當 Shirley 嘗試建立使用者時，操作會失敗。失敗的原因是許可界限不允許 `iam:CreateUser` 操作。因為這兩項政策，Shirley 沒有許可在 AWS中執行任何操作。您必須新增不同的許可政策，才能允許其他服務 (例如 Amazon S3) 中的動作。或者，您可以更新許可界限，允許她在 IAM 中建立使用者。

評估含界限的有效許可

IAM 實體 (使用者或角色) 的許可界限會設定該實體可以擁有的最大許可。這可能改變該使用者或角色的有效許可。實體的有效許可，是指會影響使用者或角色的所有政策所授予的許可。在帳戶範圍內，以身分為基礎的政策、以資源為基礎的政策、許可界限、 AWS Organizations SCP 或工作階段政策，都可能會影響到實體的許可。如需有關不同政策類型的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

如果其中任何一種政策會明確拒絕存取操作，則請求將會遭拒。經由多種許可類型授予的許可將更為複雜。如需 如何 AWS 評估政策的詳細資訊，請參閱 政策評估邏輯。

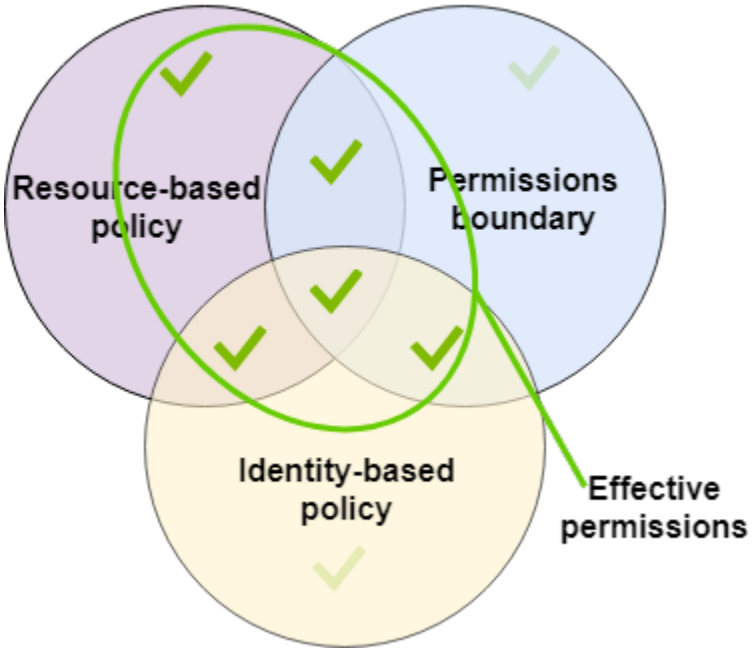
包含界限的以身分為基礎政策 – 以身分為基礎的政策可分為會連接至使用者、使用者群組或角色的內嵌或受管政策。以身分為基礎的政策可為實體授予許可，而許可界限會限制這些許可。有效的許可是兩種政策類型的交集。在這些政策的明確拒絕會覆寫允許。



以資源為基礎的政策 – 以資源為基礎的政策會控制指定主體如何存取資源，即政策連接至其中的資源。

適用於 IAM 使用者的資源型政策

在同一帳戶中，授予許可給 IAM 使用者 ARN （非 AWS STS 聯合身分使用者主體工作階段） 的資源型政策不受身分型政策或許可界限中的隱含拒絕的限制。



適用於 IAM 角色的資源型政策

IAM 角色 – 將許可授予 IAM 角色 ARN 的資源型政策受限於許可界限或工作階段政策中隱含拒絕的限制。

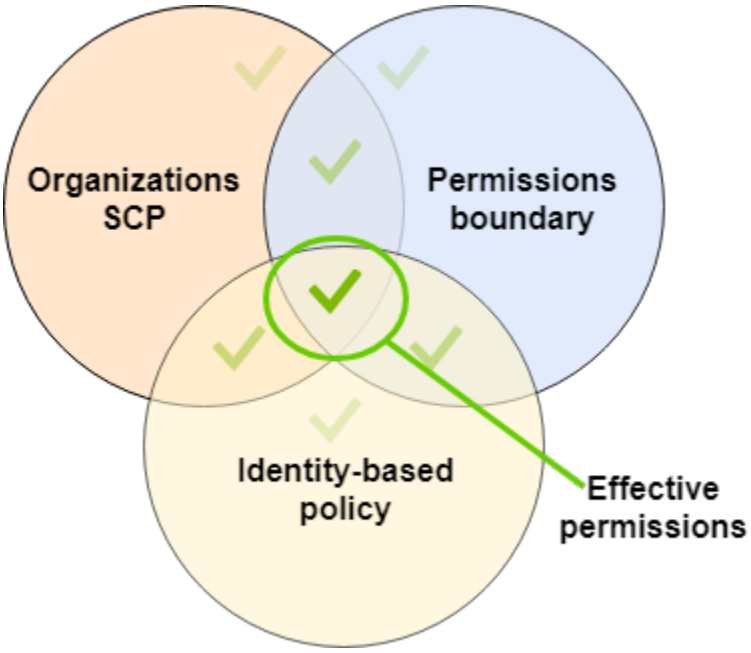
IAM 角色工作階段 – 在同一帳戶內，將許可授予 IAM 角色工作階段 ARN 的資源型政策會直接授予許可給擔任角色工作階段。直接授予工作階段的許可不會受到身分識別型政策、許可界限或工作階段政策中隱含拒絕的限制。當您擔任角色並提出要求時，提出要求的主體是 IAM 角色工作階段 ARN，而不是角色本身的 ARN。

AWS STS 聯合身分使用者主體工作階段的資源型政策

AWS STS 聯合身分使用者主體工作階段 – AWS STS 聯合身分使用者主體工作階段是透過呼叫 建立的工作階段`GetFederationToken`。當聯合身分使用者提出要求時，提出要求的主體是聯合身分使用者 ARN，而不是聯合身分 IAM 使用者的 ARN。在相同的帳戶中，授予許可給聯合身分使用者 ARN 的資源型政策會直接將許可授予工作階段。直接授予工作階段的許可不會受到身分識別型政策、許可界限或工作階段政策中隱含拒絕的限制。

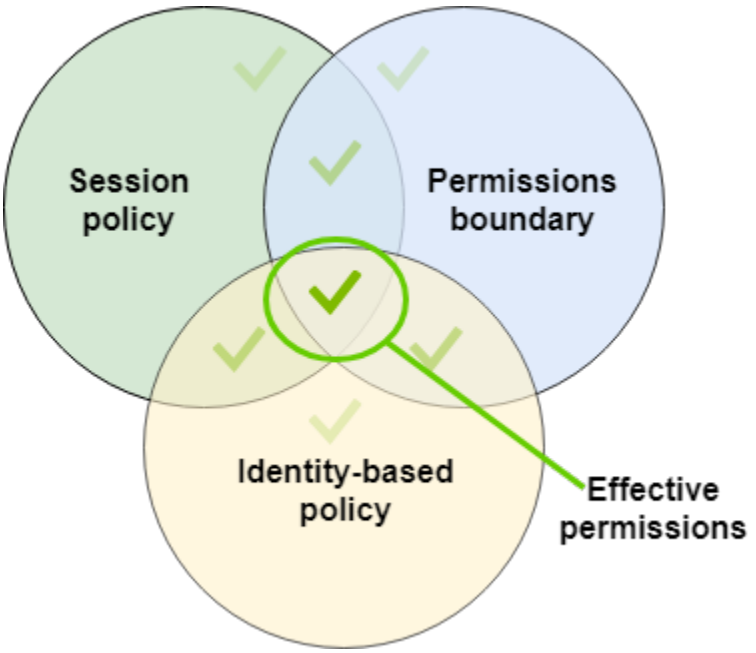
不過，如果資源型政策將許可授予聯合身分 IAM 使用者的 ARN，則 AWS STS 聯合身分使用者主體在工作階段期間提出的請求會受到許可界限或工作階段政策中的隱含拒絕的限制。

AWS Organizations SCPs – SCPs會套用至整個 AWS 帳戶。它們會限制由該帳戶內主體所提出各個請求的許可。IAM 實體 (使用者或角色) 可以提出一個要求，而這個要求會受到 SCP、許可邊界及身分類型政策的影響。在這種情況下，只有當所有三種政策類型允許時才允許該要求。有效許可是所有三種政策類型的交集。任何這些政策中的明確拒絕都會覆寫允許。



您可以了解您的帳戶是否為 AWS Organizations中組織的成員。組織成員可能會受到 SCP 影響。若要使用 AWS CLI 命令或 AWS API 操作檢視此資料，您必須具有 AWS Organizations 實體 `organizations:DescribeOrganization`動作的許可。您必須擁有其他許可，才能在 AWS Organizations 主控台中執行 操作。若要了解 SCP 是否拒絕存取特定請求，或變更您的有效許可，請聯絡您的 AWS Organizations 管理員。

工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合身分使用者的暫時工作階段時，做為參數傳遞。工作階段的許可，該工作階段的來源是建立該工作階段時所使用的 IAM 實體 (使用者或角色) 或工作階段政策。實體的以身分為基礎的政策，會受到工作階段政策及許可界限所限制。這組政策類型的有效許可是所有三種政策類型的交集。所有這類政策中的明確拒絕都會覆寫該允許。如需有關工作階段政策的詳細資訊，請參閱工作階段政策。



使用許可界限委派責任給他人

您可以使用許可界限來委派許可管理任務 (像是使用者建立) 給您的帳戶中的 IAM 使用者。這允許其他人在特定許可界限內代替您執行任務。

例如，假設 María 是 X-Company 的管理員 AWS 帳戶。她想要委派使用者建立責任給 Zhang。不過，她必須確保 Zhang 建立的使用者遵守下列公司規則：

- 使用者不能使用 IAM 建立或管理使用者、群組、角色或政策。
- 使用者被拒絕存取 Amazon S3 `logs` 儲存貯體，且無法存取 `i-1234567890abcdef0` Amazon EC2 執行個體。
- 使用者無法移除自己的界限政策。

為了強制執行這些規則，María 完成以下任務，其包含以下詳細資訊：

1. María 建立 `XCompanyBoundaries` 受管政策以針對帳戶中所有的新的使用者，當成許可界限使用。
2. María 建立 `DelegatedUserBoundary` 受管政策，並將其指派為 Zhang 的許可界限。Maria 記下其管理員 使用者的 ARN，並在防止 Zhang 存取的政策中使用此 ARN。
3. María 建立 `DelegatedUserPermissions` 受管政策，並將其連接為 Zhang 的許可政策。
4. María 告訴 Zhang 有關他的新責任和限制。

任務 1：María 必須先建立受管政策來定義新使用者的界限。María 將允許 Zhang 提供使用者所需的許可政策，但她希望這些使用者受到限制。為了執行此操作，她建立以下客戶受管政策，名稱為 XCompanyBoundaries。此政策會執行下列動作：

- 可讓使用者完整存取數個服務
- 允許在 IAM 主控台中限制自行管理存取。這表示他們可以在登入主控台之後變更密碼。也無法設定其初始密碼。若要允許此操作，請將 "*LoginProfile" 動作加入 AllowManageOwnPasswordAndAccessKeys 陳述式。
- 拒絕對 Amazon S3 日誌儲存貯體或 i-1234567890abcdef0 Amazon EC2 執行個體的存取權限

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
```

```
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

每個陳述式提供不同的用途：

1. 此政策的 ServiceBoundaries陳述式允許完整存取指定的 AWS 服務。這表示新使用者在這些服務的動作，僅受限於連接到使用者的許可政策。
2. AllowIAMConsoleForCredentials 陳述式允許列出所有 IAM 使用者的存取權。這個存取權是導覽 AWS 管理主控台中 Users (使用者) 頁面的必要條件。它也允許檢視帳戶的密碼要求，而這是變更您自身密碼時的必要程序。
3. AllowManageOwnPasswordAndAccessKeys 陳述式允許使用者僅管理自己的主控台密碼和程式設計存取金鑰。如果 Zhang 或另一個管理員為新使用者指派包含完整 IAM 存取的許可政策，這就顯得非常重要。在這種情況下，該使用者可能會變更自己的或其他使用者的許可。此陳述式可避免此種情況發生。
4. DenyS3Logs 陳述式明確拒絕存取 logs 儲存貯體。
5. DenyEC2Production 陳述式明確拒絕存取 i-1234567890abcdef0 執行個體。

任務 2：María 想要允許 Zhang 建立所有 X-Company 使用者，但只允許 XCompanyBoundaries 許可界限。她建立以下客戶受管政策，名稱為 DelegatedUserBoundary。此政策定義 Zhang 可以擁有的許可上限。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
            "Condition": {
               "StringEquals": {
                 "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
```

```
            "iam:GetGroup",
            "iam:GetLoginProfile",
            "iam:GetPolicy",
            "iam:GetPolicyVersion",
            "iam:GetRolePolicy",
            "iam:GetUser",
            "iam:GetUserPolicy",
            "iam:ListAccessKeys",
            "iam:ListAttachedRolePolicies",
            "iam:ListAttachedUserPolicies",
            "iam:ListEntitiesForPolicy",
            "iam:ListGroups",
            "iam:ListGroupsForUser",
            "iam:ListMFADevices",
            "iam:ListPolicies",
            "iam:ListPolicyVersions",
            "iam:ListRolePolicies",
            "iam:ListSSHPublicKeys",
            "iam:ListServiceSpecificCredentials",
            "iam:ListSigningCertificates",
            "iam:ListUserPolicies",
            "iam:ListUsers",
            "iam:SetDefaultPolicyVersion",
            "iam:SimulateCustomPolicy",
            "iam:SimulatePrincipalPolicy",
            "iam:UpdateGroup",
            "iam:UpdateLoginProfile",
            "iam:UpdateUser"
        ],
        "NotResource": "arn:aws:iam::123456789012:user/Maria"
    },
    {
        "Sid": "NoBoundaryPolicyEdit",
        "Effect": "Deny",
        "Action": [
            "iam:CreatePolicyVersion",
            "iam:DeletePolicy",
            "iam:DeletePolicyVersion",
            "iam:SetDefaultPolicyVersion"
        ],
        "Resource": [
            "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
            "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
        ]
```

```
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

每個陳述式提供不同的用途：

1. CreateOrChangeOnlyWithBoundary 陳述式允許 Zhang 建立 IAM 使用者，但前提是他須使用 XCompanyBoundaries 政策來設定許可界限。此陳述式也可讓他為現有的使用者設定許可界限，但只能使用該相同政策。最後，此陳述式讓 Zhang 管理具此許可界限設定之使用者的許可政策。
2. CloudWatchAndOtherIAMTasks 陳述式允許 Zhang 完成其他使用者、群組和政策管理任務。其有權重設密碼，並為 NotResource 政策元素中未列出的任何 IAM 使用者建立存取金鑰。這可讓 Zhang 協助使用者解決登入問題。
3. NoBoundaryPolicyEdit 陳述式拒絕 Zhang 存取更新 XCompanyBoundaries 政策。他不被允許變更任何用於設定他自己或其他使用者的許可界限的政策。
4. NoBoundaryUserDelete 陳述式拒絕讓 Zhang 存取刪除他自己或其他使用者的許可界限。

然後，María 針對 Zhang 使用者將 DelegatedUserBoundary 政策分派許可界限。

任務 3：由於許可界限限制許可上限，但不自行授予存取權，因此 Maria 必須為 Zhang 建立許可政策。她建立以下政策，名稱為 DelegatedUserPermissions。此政策定義在已定義的界限內，Zhang 可以執行的操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
```

```
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3BucketContents",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
    ]
}
```

每個陳述式提供不同的用途：

1. 此政策的 IAM 陳述式允許 Zhang 完整存取 IAM。不過，由於他的許可界限僅允許部分的 IAM 操作，他的有效 IAM 許可僅受限於他的許可界限。
2. `CloudWatchLimited` 陳述式允許 Zhang 在 CloudWatch 中執行五個動作。他的許可界限允許 CloudWatch 中的所有動作，因此他的有效 CloudWatch 許可僅受限於他的許可政策。
3. `S3BucketContents` 陳述式允許 Zhang 列出 `ZhangBucket` Amazon S3 儲存貯體。不過，他的許可界限不允許任何 Amazon S3 動作，因此他無法執行任何 S3 操作，無論他是否有許可政策。

> **Note**
>
> Zhang 的政策允許其建立一個使用者，此使用者可存取 Zhang 無法存取的 Amazon S3 資源。透過委派這些管理動作，Maria 利用 Amazon S3 存取權以有效方式信任了 Zhang。

接著，María 將 `DelegatedUserPermissions` 政策連接為 Zhang 使用者的許可政策。

任務 4：她提供 Zhang 有關建立新使用者的指示。她通知他可以建立具備所需的任何許可的新使用者，但他必須將 XCompanyBoundaries 政策指派給他們做為許可界限。

Zhang 完成下列任務：

1. Zhang 使用 建立使用者 AWS 管理主控台。他輸入使用者名稱 Nikhil，並啟用該使用者的主控台存取權。他清除了 Requires password reset (需要重設密碼) 旁的核取方塊，因為上述政策只允許使用者在登入 IAM 主控台後變更密碼。
2. 在 Set permissions (設定許可) 頁面上，Zhang 選擇 IAMFullAccess 和 AmazonS3ReadOnlyAccess 許可政策，允許 Nikhil 執行他的工作。
3. Zhang 忘了 María 的指示，略過 Set permissions boundary (設定許可界限) 區段。
4. Zhang 檢閱使用者詳細資訊，然後選擇 Create user (建立使用者)。

   操作失敗，存取遭拒。Zhang 的 DelegatedUserBoundary 許可界限要求他所建立的任何使用者都將 XCompanyBoundaries 政策當做許可界限來使用。
5. Zhang 返回到之前的頁面。在 Set permissions boundary (設定許可界限) 區段中，他選擇了 XCompanyBoundaries 政策。
6. Zhang 檢閱使用者詳細資訊，然後選擇 Create user (建立使用者)。

   使用者已建立。

當 Nikhil 登入時，即有權存取 IAM 和 Amazon S3，除了許可界限拒絕的那些操作之外。例如，他可以在 IAM 變更自己的密碼，但無法建立其他使用者或編輯他的政策。Nikhil 只有 Amazon S3 的讀取存取權。

如果有人將資源類型政策新增至 logs 儲存貯體，而讓 Nikhil 在儲存貯體中放入物件，則他仍然無法存取該儲存貯體。原因在於對 logs 儲存貯體執行的任何動作，均會受到其許可界限的明確拒絕。任何政策類型中的明確拒絕，都會導致請求遭到拒絕。

不過，如果 Secrets Manager 秘密連接的以資源為基礎的政策允許 Nikhil 執行 – secretsmanager:GetSecretValue動作，則 Nikhil 可以擷取和解密該秘密。原因在於這項許可界限並未明確拒絕上述 Secrets Manager 操作，而且許可界限的隱含拒絕不會限制資源類型政策。

以身分為基礎和以資源為基礎的政策

政策是 中的物件， AWS 當與身分或資源建立關聯時，會定義其許可。當您建立許可政策來限制對資源的存取時，您可以選擇以身分為基礎的政策或以資源為基礎的政策。

以身分為基礎的政策會連接至 IAM 使用者、群組或角色。這些政策可讓您指定該身分可以執行哪些動作 (其許可)。例如，您可以將政策連接到名為 John 的 IAM 使用者，指出他已獲允許而可執行 Amazon EC2 `RunInstances` 動作。政策可進一步指出 John 已獲允許而可從名為 `MyCompany` 的 Amazon DynamoDB 資料表中取得項目。您也可以允許 John 管理他自己的 IAM 安全憑證。以身分為基礎的政策，可以受管理或內嵌。

以資源為基礎的政策會連接至資源。例如，您可以將資源型政策連接至 Amazon S3 儲存貯體、Amazon SQS 佇列、VPC 端點、 AWS Key Management Service 加密金鑰、Amazon DynamoDB 資料表和串流，以及 AWS 登入資源。如需有關可支援以資源為基礎之政策的服務清單，請參閱AWS 使用 IAM 的 服務。

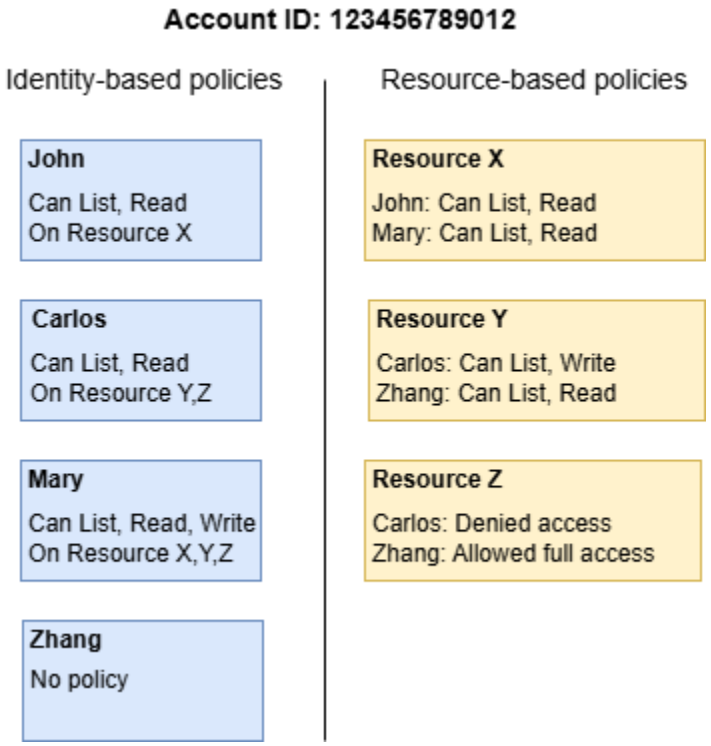
有了以資源為基礎的政策，您可以指定誰可以存取資源，以及他們可以在資源上執行哪些動作。若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？。以資源為基礎的政策僅是內嵌，而不是受管理。

> **Note**
>
> 以資源為基礎的政策不同於資源層級的許可。您可以將以資源為基礎的政策直接連接到本主題中所述的資源。資源層級許可代表有能力使用 ARN 在政策中指定個別的資源。只有部分 AWS 服務支援資源型政策。如需服務支援以資源為基礎的政策和資源層級許可的清單明細，請參閱AWS 使用 IAM 的 服務。

若要了解以身分為基礎的政策和以資源為基礎的政策如何在相同帳戶內進行互動，請參閱 單一帳戶中請求的政策評估。

若要了解政策如何跨帳戶互動，請參閱 跨帳戶政策評估邏輯。

為了更了解這些概念，檢視這些圖表。123456789012 帳戶的管理員會將以身分為基礎的政策連接至`John`、`Carlos` 和 `Mary` 使用者。您可以對特定資源執行在這些政策中的一些動作。例如，使用者 `John` 可以對 `Resource X` 執行一些動作。這是一個在以身分為基礎政策的資源層級許可。管理員也將以資源為基礎的政策新增至 `Resource X`、`Resource Y` 和 `Resource Z`。以資源為基礎的政策允許您指定誰可以存取該資源。例如，在 `Resource X` 上以資源為基礎的政策允許 `John` 和 `Mary` 使用者列出和讀取資源的存取權。



123456789012 帳戶範例允許以下使用者執行所列的動作：

- John – John 可以對 Resource X 執行列出和讀取動作。他是由以其使用者身分為基礎的政策和以資源為基礎的政策 Resource X 獲與此許可。
- Carlos – Carlos 可以對 Resource Y 執行列出、讀取和寫入動作，但存取 Resource Z 時會遭拒。以 Carlos 之身分為基礎的政策允許可執行對 Resource Y 的列出和讀取動作。以 Resource Y 資源為基礎的政策允許擁有寫入許可。不過，雖然他的以身分為基礎的政策可允許存取 Resource Z，以 Resource Z 資源為基礎的政策會拒絕該存取。明確 Deny拒絕會覆寫 Allow，且他對 Resource Z 的存取會遭拒。如需詳細資訊，請參閱政策評估邏輯。
- Mary – Mary 可以對 Resource X、Resource Y 和 Resource Z 執行列出、讀取和寫入操作。她的以身分為基礎的政策允許比以資源為基礎的政策對更多動作執行更多動作，但這些以資源為基礎的政策不會拒絕存取。
- Zhang –Zhang 擁有對 Resource Z 的完整存取權。Zhang 沒有以身分為基礎的政策，但以 Resource Z 資源為基礎的政策可允許完整存取資源。Zhang 也可以對 Resource Y 執行清單和讀取動作。

以身分為基礎的政策和以資源為基礎的政策都是許可政策，且會同時受到評估。對於僅套用許可政策的請求， AWS 請先檢查 的所有政策Deny。如果有的話，則該請求會遭拒。然後， AWS 會檢查各個

Allow。如果至少有一個政策陳述式允許在請求中的動作，則該請求會受到允許。在以身分為基礎的政策中或以資源為基礎的政策中，是否有 Allow 並不重要。

> **Important**
>
> 此邏輯僅在單一 AWS 帳戶內做出該請求時適用。對於從一個帳戶對另一個帳戶所做的請求，在 Account A 中的請求者必須擁有以身分為基礎的政策，該政策能允許對 Account B 中的資源做出請求。此外，在 Account B 中以資源為基礎的政策，必須允許在 Account A 中的請求者存取資源。兩個帳戶中均須有允許操作的政策，否則請求會失敗。如需有關使用跨帳戶存取之以資源為基礎的政策詳細資訊，請參閱 IAM 中的跨帳戶資源存取。

具有特定許可的使用者可能請求資源，而該資源也有連接許可政策。在這種情況下， 會在決定是否授予對資源的存取權時， AWS 評估兩組許可。如需有關如何評估政策的詳細資訊，請參閱政策評估邏輯。

> **Note**
>
> Amazon S3 支援以身分為基礎的政策和以資源為基礎的政策 (稱為儲存貯體政策)。此外，Amazon S3 支援許可機制稱為存取控制清單 (ACL)，其為獨立的 IAM 政策和許可。您可以合併使用 IAM 政策與 Amazon S3 ACL。如需詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的存取控制 。

使用 政策控制對 AWS 資源的存取

您可以使用政策來控制對 IAM 或所有資源的存取 AWS。

若要使用政策來控制 中的存取 AWS，您必須了解 如何 AWS 授予存取。 AWS 由 資源集合組成。IAM 使用者即為資源。Amazon S3 儲存貯體即為資源。當您使用 AWS API AWS CLI、 或 AWS 管理主控台 來執行 操作 （ 例如建立使用者 ） 時，您會傳送該操作的請求。您的請求指定動作、資源、主體實體 (使用者或角色)、主體帳戶，以及所需的任何請求資訊。所有這些資訊提供了上下文。

AWS 然後檢查您 （ 委託人 ） 是否經過身分驗證 （ 登入 ） 和授權 （ 具有許可 ），以對指定的資源執行指定的動作。在授權期間， 會 AWS 檢查適用於請求內容的所有政策。大多數政策會以 JSON 文件 AWS 形式存放在 中，並指定委託人實體的許可。如需有關政策類型及其使用的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

AWS 只有在政策允許請求的每個部分時，才會授權請求。若要檢視此程序的圖表，請參閱 IAM 的運作方式。如需如何 AWS 判斷是否允許請求的詳細資訊，請參閱 政策評估邏輯。

當您建立 IAM 政策時，可以控制對下列項目的存取：

- 主體 – 控制允許發出請求的人員 (主體) 執行哪些操作。
- IAM 身分：控制哪些 IAM 身分 (IAM 群組、使用者與角色) 可被存取以及存取的方法。
- IAM政策 – 控制哪些使用者可以建立、編輯和刪除客戶受管政策，以及哪些使用者可以連接和分開所有受管政策。
- AWS 資源 – 控制哪些使用者有權使用以身分為基礎的政策或以資源為基礎的政策來存取資源。
- AWS 帳戶 – 控制是否僅允許特定帳戶的成員發出請求。

政策可讓您指定誰可以存取 AWS 資源，以及他們可以對這些資源執行哪些動作。每個 IAM 使用者在開始時都沒有許可。換言之，在預設狀態下，使用者無法執行任何動作，甚至不能查看自己的存取金鑰。若要為使用者授予執行某些操作的許可，您可以為使用者新增許可 (也就是將政策連接到使用者)。或者，您可以將使用者加入到具有預期的許可的使用者群組中。

例如，您可能授予使用者列出自己的存取金鑰的許可。您可能擴展該許可，並讓每位使用者建立、更新和刪除自己的索引鍵。

當您授予一個使用者群組許可時，使用者群組內的全部使用者都會獲得那些許可。例如，您可以授予管理員使用者群組許可，以對任何 AWS 帳戶 資源執行任何 IAM 動作。另一個範例：您可以授予 Managers 使用者群組描述 AWS 帳戶的 Amazon EC2 執行個體的許可。

有關如何委託基本許可給使用者、IAM 群組和角色的資訊，請參閱存取 IAM 資源所需的許可。有關說明基本許可的其他政策範例，請參閱 管理 IAM 資源的政策範例。

控制主體的存取許可

您可以使用政策，控制允許發出請求的人員 (主體) 執行哪些操作。若要執行此操作，您必須將以身分為基礎的政策連接到此人的身分 (使用者、使用者群組或角色)。您也可以使用許可界限，設定實體 (使用者或角色) 可擁有的最大許可。

例如，假設您希望使用者 Zhang Wei 能夠完整存取 CloudWatch、Amazon DynamoDB、Amazon EC2 和 Amazon S3。您可以建立兩個不同的政策，以便以後其他使用者需要一組許可時為其分配其中的一個政策。或者，您可以將兩個許可放在單一政策中，然後將該政策連接到名為 Zhang Wei 的 IAM 使用者。您還可以將一個政策連接到 Zhang 所屬的使用者群組，或者連接到 Zhang 可以擔任的角色。因此，在 Zhang 查看 S3 儲存貯體內容時，會允許他的請求。如果他嘗試建立新的 IAM 使用者，他的請求會被拒絕，因為他未獲許可。

您可以在 Zhang 身上使用許可界限，確定他絕對不會被授予 amzn-s3-demo-bucket1 S3 儲存貯體存取權。若要執行此操作，請決定您希望 Zhang 擁有的許可上限。在這種情況下，您可以控制他使用許可政策做哪些事。在這裡，您只在意他不會存取機密儲存貯體。因此，您可以使用下列政策來定義 Zhang 的界限，以允許 Amazon S3 和其他一些服務的所有 AWS 動作，但拒絕存取 amzn-s3-demo-bucket1 S3 儲存貯體。由於許可界限不允許任何 IAM 動作，如此可防止 Zhang IAM 刪除他 (或任何人的) 界限。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
                "ec2:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1",
                "arn:aws:s3:::amzn-s3-demo-bucket1/*"
            ]
        }
    ]
}
```

當您針對使用者指派像這類許可界限的政策時，請記得，並沒有授予任何許可。它負責設定以身分為基礎的政策可為 IAM 實體授予的最大許可。如需有關許可界限的詳細資訊，請參閱 IAM 實體的許可界限。

如需之前所述的程序詳細資訊，請參考以下資源：

- 若要進一步了解有關建立可連接到主體的 IAM 政策的資訊，請參閱 使用客戶管理政策定義自訂 IAM 許可。
- 若要了解將 IAM 政策連接到主體的方法，請參閱 新增和移除 IAM 身分許可。
- 若要查看授予 EC2 完全存取許可的範例政策，請參閱 Amazon EC2：允許在特定區域內，以程式設計方式和在主控台進行 EC2 完全存取。
- 允許對 S3 儲存貯體進行唯讀存取，請使用以下範例政策的前兩個說明：Amazon S3：允許以程式設計方式和在主控台以讀取和寫入存取 S3 儲存貯體中的物件。
- 查看範例政策，以便讓使用者設定其憑證，例如其主控台密碼、程式設計存取金鑰及 MFA 裝置，請參閱 AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。。

控制對身分的存取

您可以透過使用者群組來建立連接到使用者的 IAM 政策，以使用該政策控制您的所有使用者可對某個身分執行哪些操作。若要執行此操作，請建立一個政策以限制可對某個身分執行哪些操作，以及哪些使用者可以存取該身分。

例如，您可以建立一個名為 AllUsers 的使用者群組，然後將該使用者群組連接到所有使用者。在建立該使用者群組時，您可以為所有使用者授予存取權限以設定其憑證，如前一節中所述。然後，您可以建立一個政策，以便拒絕存取以禁止更改該使用者群組，除非在政策條件中包含使用者名稱。但是，該政策部分僅拒絕列出的使用者以外的任何人進行存取。您也必須包含許可權，以允許該使用者群組中的所有使用者可執行所有使用者群組管理動作。最後，將該政策連接到該使用者群組，以便將其套用於所有使用者。因此，在政策中未指定的使用者嘗試更改該使用者群組時，將拒絕請求。

若要使用視覺化編輯器來建立此政策

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。

   如果這是您第一次選擇 Policies (政策)，將會顯示 Welcome to Managed Policies (歡迎使用受管政策) 頁面。選擇 Get Started (開始使用)。
3. 選擇建立政策。
4. 在政策編輯器區段中，選擇視覺化選項。
5. 在選擇服務中，選擇 IAM。
6. 在允許的動作中，在搜尋方塊中輸入 **`group`**。視覺化編輯器將顯示所有包含 `group` 一詞的 IAM 動作。選取所有核取方塊。

7. 選擇 Resources (資源) 來為您的政策指定資源。根據您所選的動作，應會看到群組和使用者資源類型。

   - 群組 – 選擇新增 ARN。針對資源位於，選取任何帳號選項。選取任何具有路徑的群組名稱核取方塊，然後輸入使用者群組名稱 **`AllUsers`**。接著選擇新增 ARN。
   - 使用者：選取此帳戶中的任何旁的核取方塊。

   您選擇的操作之一 (`ListGroups`) 不支援使用特定的資源。您不需要為該動作選擇 All resources (所有資源)。在 JSON 編輯器中儲存或查看您的政策時，您可以看到 IAM 自動建立新的許可區塊，以便為所有資源授予該動作的許可。
8. 若要新增其他許可區塊，請選擇新增更多許可。
9. 選擇選取服務，然後選擇 IAM。
10. 選擇允許的動作，然後選擇切換為拒絕許可。在執行該操作時，將使用整個區塊來拒絕許可。
11. 在搜尋方塊中，輸入 **`group`**。視覺化編輯器將顯示所有包含 `group` 一詞的 IAM 動作。選取下列動作旁的核取方塊：

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup
    - AttachGroupPolicy
    - DeleteGroupPolicy
    - DetachGroupPolicy
    - PutGroupPolicy
    - UpdateGroup
12. 選擇 Resources (資源) 來為您的政策指定資源。根據您所選的動作，應會看到群組資源類型。選擇新增 ARN。針對資源位於，選取任何帳號選項。針對任何具有路徑的群組名稱，輸入使用者群組名稱 **`AllUsers`**。接著選擇新增 ARN。
13. 選擇請求條件 - 選用，然後選擇新增另一個條件。以下列的值來完成表單：

    - 條件金鑰 - 選擇 aws：使用者名稱
    - Qualifier (限定詞) – 選擇 Default (預設)
    - Operator (運算子) – 選擇 StringNotEquals
    - 值 – 輸入 **`srodriguez`** 然後選擇新增，以新增另一個值。輸入 **`mjackson`**，然後選擇新增，以新增另一個值。輸入 **`adesai`**，然後選擇新增條件。

在清單中不包含進行呼叫的使用者時，該條件將確認拒絕存取指定的使用者群組管理動作。由於這會明確拒絕許可權，它將覆寫前面允許這些使用者呼叫這些操作的區塊。不會拒絕清單中的使用者進行存取，並在第一個許可區塊中為他們授予許可，因此，他們可以完全管理該使用者群組。

14. 完成時，選擇 Next (下一步)。

> **Note**
>
> 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器選項中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

15. 在檢視與建立頁面上，針對政策名稱輸入 **`LimitAllUserGroupManagement`**。針對 Description (描述)，輸入 **`Allows all users read-only access to a specific user group, and allows only specific users access to make changes to the user group`**。檢視該政策中定義的許可，來確認您已授予想要的許可。然後選擇 Create policy (建立政策) 來儲存您的新政策。
16. 將政策連接到您的使用者群組。如需詳細資訊，請參閱 新增和移除 IAM 身分許可。

或者，您也可以使用該範例 JSON 政策文件建立相同的政策。若要檢視此 JSON 政策，請參閱 IAM：允許特定 IAM 使用者以程式設計方式在主控台中管理群組。如需關於使用 JSON 文件來建立政策的詳細說明，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

控制對政策的存取

您可以控制使用者如何套用 AWS 受管政策。若要執行此操作，請將該政策連接到您的所有使用者。理想情況下，您可以使用使用者群組執行該操作。

例如，您可以建立一個政策，允許使用者只將 IAMUserChangePassword 和 PowerUserAccess AWS 受管政策連接到新的 IAM 使用者、使用者群組或角色。

對於客戶受管政策，您可以控制哪些使用者可以建立、更新和刪除這些政策。您可以控制哪些使用者可以將政策附加到主體實體 (IAM 群組、使用者和角色) 以及將政策從主體實體中分離。您也可以控制使用者可以對哪些實體連接或分開哪些政策。

例如，您可以為帳戶管理員授予許可以建立、更新和刪除政策。然後，您可以為小組負責人或其他有限管理員授予許可，以便將這些政策連接到該有限管理員管理的主體實體以及將這些政策從這些主體實體中分開。

如需詳細資訊，請參閱下列資源：

- 若要進一步了解有關建立可連接到主體的 IAM 政策的資訊，請參閱 使用客戶管理政策定義自訂 IAM 許可。
- 若要了解將 IAM 政策連接到主體的方法，請參閱 新增和移除 IAM 身分許可。
- 若要查看限制使用受管政策的範例政策，請參閱 IAM：限制可套用至 IAM 使用者、群組或角色的受管政策。

控制建立、更新和刪除客戶受管政策的許可

您可以使用 IAM 政策控制誰有權在您的 AWS 帳戶中建立、更新和刪除客戶管理政策。以下清單包含與建立、更新和刪除政策或政策版本直接相關的 API 操作：

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

上述清單中的 API 操作對應於可使用 IAM 政策允許或拒絕的操作 — 也就是您可以授予的許可。

請考量下列範例政策。允許使用者建立、更新 (即建立新的政策版本)、刪除 AWS 帳戶中的所有客戶管理政策以及設定這些政策的預設版本。該範例政策還允許該使用者列出政策並獲取政策。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

Example允許建立、更新、刪除、列出、獲取所有政策以及設定這些政策的預設版本的政策範例

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
```

```
      "iam:DeletePolicy",
      "iam:DeletePolicyVersion",
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "*"
  }
}
```

您可以建立限制使用這些 API 操作的政策以僅影響指定的受管政策。例如，您可能希望允許使用者設定預設版本和刪除政策版本，但是僅允許針對特定客戶受管政策執行這些操作。運用在授予這些許可權的政策的 `Resource` 元素中指定政策 ARN，可實現此目的。

以下範例顯示的政策，可讓使用者刪除政策版本，並設定預設版本。但是，這些動作只允許用於客戶受管政策，其包含路徑 /TEAM-A/。客戶受管政策 ARN 是在政策的 `Resource` 元素中指定 (在此範例中，ARN 包含一個路徑和一個萬用字元，因而可配對包含路徑 /TEAM-A/ 的所有客戶受管政策)。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

如需有關如何使用該範例 JSON 政策文件建立政策，請參閱 易用名稱和路徑。

Example僅允許針對特定政策刪除政策版本和設定預設版本的政策範例

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:DeletePolicyVersion",
            "iam:SetDefaultPolicyVersion"
        ],
        "Resource": "arn:aws:iam::111122223333:policy/TEAM-A/*"
    }
}
```

控制連接和分開受管政策的許可

您還可以使用 IAM 政策僅允許使用者使用特定的受管政策。實際上，您可以控制允許使用者為其他主體實體授予哪些許可。

以下清單顯示與將受管政策連接到主體實體以及從中分開直接相關的 API 操作：

- AttachGroupPolicy
- AttachRolePolicy
- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

您可以建立限制使用這些 API 操作的政策以僅影響特定的受管政策與/或您指定的主體實體。例如，您可能希望允許使用者連接受管政策，但是只能連接您指定的受管政策。或者，您可能希望允許使用者連接受管政策，但是只能連接到您指定的主體實體。

以下範例政策僅允許使用者將受管政策附加到包含路徑 /TEAM-A/ 的 IAM 群組和角色。使用者群組和角色 ARN 是在政策的 `Resource` 元素中指定的。(在此範例中，ARN 包含一個路徑和一個萬用字元，因而與包含路徑 /TEAM-A/ 的所有 IAM 群組和角色配對。) 若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

Example僅允許將受管政策連接到特定使用者群組或角色的政策範例

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
```

```
        ]
    }
}
```

您可以進一步限制前面範例中的動作，讓它僅影響特定的政策。也就是，透過在政策中新增條件，您可以控制允許使用者連接到其他主體實體的許可。

以下範例中的條件可確保只有在連接的政策與指定政策之一配對時才允許 AttachGroupPolicy 與 AttachRolePolicy 許可。條件使用iam:PolicyARN 條件金鑰來決定要連接哪個政策或哪些政策。以下範例政策延伸之前的範例。其允許使用者僅將包含路徑 /TEAM-A/ 的受管政策連接到包含路徑 /TEAM-A/ 的 IAM 群組和角色。若要了解如何使用此範例 JSON 政策文件來建立政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
        ],
        "Condition": {
            "ArnLike": {
                "iam:PolicyARN": "arn:aws:iam::111122223333:policy/TEAM-A/*"
            }
        }
    }
}
```

本政策使用 ArnLike 條件運算子，但也可以使用 ArnEquals 條件運算子，因為這兩個條件運算子執行相同的操作。如需有關 ArnLike 與 ArnEquals 的詳細資訊，請參閱《政策元素參考》條件類型章節中的 Amazon Resource Name (ARN) 條件運算子。

例如，您可以限制使用操作以僅涉及指定的受管政策。運用在授予這些許可權的政策的 Condition 元素中指定政策 ARN，可實現此目的。例如，指定客戶受管政策的 ARN：

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

您也可以在政策的 Condition元素中指定 AWS 受管政策的 ARN。 AWS 受管政策的 ARN 使用政策 ARN aws中的特殊別名，而不是帳戶 ID，如本範例所示：

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

控制資源的存取許可

您可以控制哪些使用者有權使用以身分為基礎的政策或以資源為基礎的政策來存取資源。在以身分為基礎的政策中，您將政策連接到一個身分並指定該身分可以存取哪些資源。在以資源為基礎的政策中，您將政策連接到要控制的資源。在該政策中，您指定哪些主體可以存取該資源。如需有關兩種政策類型的詳細資訊，請參閱 以身分為基礎和以資源為基礎的政策。

如需詳細資訊，請參閱下列資源：

- 若要進一步了解有關建立可連接到主體的 IAM 政策的資訊，請參閱 使用客戶管理政策定義自訂 IAM 許可。
- 若要了解將 IAM 政策連接到主體的方法，請參閱 新增和移除 IAM 身分許可。
- Amazon S3 支援在儲存貯體上使用以資源為基礎的政策。如需詳細資訊，請參閱儲存貯體政策範例。

資源建立者不會自動擁有許可

如果您使用 AWS 帳戶根使用者 登入資料登入，則有權對屬於該帳戶的資源執行任何動作。但是，對 IAM 使用者來說並非如此。IAM 使用者可以獲得建立資源的許可，但即使對於該資源，該使用者許可也僅限於明確授予的內容。這表示您不會只因為建立資源 (如 IAM 角色) 就自動具有編輯或刪除該角色的許可。此外，帳戶擁有者或有權管理您的許可的其他使用者可以隨時撤銷您的許可。

控制對特定帳戶中的主體的存取

您可以直接向自己帳戶中的 IAM 使用者授予對您的資源的存取權限。如果來自另一個帳戶的使用者需要存取您的資源，您可以建立 IAM 角色。角色是一個實體，其中包含許可，但不與特定使用者關聯。然後，其他帳戶中的使用者可以擔任該角色，並根據您為該角色分配的許可存取資源。如需詳細資訊，請參閱 在您擁有的另一個 IAM 使用者中 AWS 帳戶 存取。

> **Note**
>
> 部分服務支援以資源為基礎的政策，如 以身分為基礎和以資源為基礎的政策 (例如 Amazon S3、Amazon SNS 和 Amazon SQS) 中所述。對於這些服務，使用角色的替代方法是將政策連接到您要共用的資源 (儲存貯體、主題或佇列)。以資源為基礎的政策可以指定具有資源存取許可 AWS 的帳戶。

使用標籤控制對 IAM 使用者和角色的存取

使用以下區段中的資訊來控制可以存取 IAM 使用者和角色的人員，以及使用者和角色可以存取的資源。如需控制其他 AWS 資源存取的一般資訊和範例，包括其他 IAM 資源，請參閱 AWS Identity and Access Management 資源的標籤。

> **Note**
>
> 如需有關標籤索引鍵和標籤索引鍵值區分大小寫的詳細資訊，請參閱 Case sensitivity。

您可以將標籤連接至 IAM 資源、傳遞至請求，或連接至提出請求的主體。IAM 使用者或角色可以同時是資源和主體。例如，您可以編寫政策，此政策會允許使用者列出使用者的群組。只有在發出請求的使用者 (主體) 擁有與其嘗試檢視的使用者相同的 `project=blue` 標籤時，才允許此操作。在這個範例中，只要使用者處理的是同一個專案，他們就可以檢視任何使用者 (包括自己) 的群組成員資格。

若要根據標籤控制存取，則在政策的條件元素中提供標籤資訊。建立 IAM 政策時，您可以使用 IAM 標籤與關聯的標籤條件索引鍵，以控制下列任何項目的存取權：

- 資源 – 根據使用者或角色資源的標籤控制對這些資源的存取權。若要執行此操作，請使用 aws:ResourceTag/***key-name*** 條件金鑰，以指定必須連接到該資源的標籤鍵/值組。如需詳細資訊，請參閱 控制對 AWS 資源的存取。
- 請求 – 控制您可以在 IAM 請求條件中傳遞哪些標籤。若要執行此操作，請使用 aws:RequestTag/***key-name*** 條件金鑰，來指定可以從 IAM 使用者或角色新增、變更或移除哪些標

籤。此金鑰用於 IAM 資源和其他 AWS 資源的方式相同。如需詳細資訊，請參閱在 AWS 請求期間控制存取。

- 主體 – 根據連接至該人員 IAM 使用者或角色的標籤，控制提出請求的人員 (主體) 可執行的操作。若要執行此操作，請使用 aws:PrincipalTag/***key-name*** 條件金鑰來指定必須將哪些標籤連接至 IAM 使用者或角色，然後該請求才能獲得允許。
- 授權程序的任何部分：使用 aws:TagKeys 條件索引鍵，以控制特定標籤鍵是否可用於請求中，或是否可由主體使用。在此情況下，此鍵值無關緊要。此金鑰對 IAM 和其他 AWS 服務的行為類似。不過，當您在 IAM 中標記使用者時，這也會控制主體是否可以對任何服務發出請求。如需詳細資訊，請參閱 根據標籤索引鍵控制存取權限。

您可以使用視覺編輯器、使用 JSON，或匯入現有的受管政策，來建立 IAM 政策。如需詳細資訊，請參閱使用客戶管理政策定義自訂 IAM 許可。

> **Note**
>
> 您也可以在擔任 IAM 角色或與使用者聯合身分時傳遞工作階段標籤。這些標籤僅在工作階段的時長內有效。

控制 IAM 主體的存取權限

您可以根據在該人員身分中連接的標籤，控制主體可執行的操作。

此範例會示範如何建立身分型政策，允許此帳戶中的任何使用者檢視任何使用者的群組成員資格，包括他們自己，只要他們正在處理同一個專案。只有在使用者的資源標籤和主體的標籤具有相同標籤鍵值 `project` 時，才允許此操作。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
```

```
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
            }
        }]
}
```

根據標籤索引鍵控制存取權限

您可以在 IAM 政策中使用標籤，以控制是否可在請求中使用特定標籤鍵，或是否可由主體使用特定標籤鍵。

此範例會示範如何建立身分型政策，僅允許從使用者中刪除具有 `temporary` 索引鍵的標籤。若要使用此政策，請將範例政策中的*########*取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys":
 ["temporary"]}}
    }]
}
```

使用標籤控制對 AWS 資源的存取

您可以使用標籤來控制對支援標記之 AWS 資源的存取，包括 IAM 資源。您可以標記 IAM 使用者和角色，來控制他們可以存取的內容。若要了解如何標記 IAM 使用者和角色，請參閱 AWS Identity and Access Management 資源的標籤。此外，您可以控制對下列 IAM 資源的存取：客戶管理政策、IAM 身分提供者、執行個體設定檔、伺服器憑證和虛擬 MFA 裝置。若要檢視建立和測試允許具有主體標籤的 IAM 角色政策，以相符標籤存取資源的教學，請參閱 IAM 教學課程：定義根據標籤存取 AWS 資源的許可。使用下一節中的資訊來控制對其他 AWS 資源的存取，包括 IAM 資源，而無需標記 IAM 使用者或角色。

使用標籤控制對 AWS 資源的存取之前，您必須了解 如何 AWS 授予存取。 AWS 由資源集合組成。Amazon EC2 執行個體是一種資源。Amazon S3 儲存貯體即為資源。您可以使用 AWS API AWS CLI、 或 AWS 管理主控台 來執行 操作，例如在 Amazon S3 中建立儲存貯體。當您這麼做時，您會為該操作傳送請求。您的請求指定動作、資源、主體實體 (使用者或角色)、主體帳戶，以及所需的任何請求資訊。所有這些資訊提供了上下文。

AWS 然後檢查您（委託人實體）是否經過身分驗證（登入）和授權（具有許可），以對指定的資源執行指定的動作。在授權期間， 會 AWS 檢查適用於請求內容的所有政策。大多數政策會以 JSON 文件 AWS 形式存放在 中，並指定委託人實體的許可。如需有關政策類型及其使用的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

AWS 只有在政策允許請求的每個部分時， 才會授權請求。若要查看圖表和進一步了解 IAM 基礎設施的詳細資訊，請參閱 IAM 的運作方式。有關 IAM 如何決定是否允許請求的詳細資訊，請參閱 政策評估邏輯。

標籤是這個程序中的另一項考慮因素，因為您可以將標籤連接到資源或在請求中將標籤傳遞至支援標籤的服務。若要根據標籤控制存取，則在政策的條件元素中提供標籤資訊。若要了解 AWS 服務是否支援使用標籤控制存取，請參閱 ABAC 欄中的 AWS 使用 IAM 的 服務並尋找具有是的服務。選擇服務的名稱，以檢視該服務的授權和存取控制文件。

然後，您可以建立 IAM 政策，這類政策可根據該資源的標籤允許或拒絕對某資源的存取。在該政策中，您可以使用標籤條件金鑰，以控制執行下列項目的存取權：

- 資源 – 根據這些資源上的標籤控制對 AWS 服務資源的存取。若要執行此操作，請使用 aws:ResourceTag/***key-name*** 條件索引鍵，以判斷是否允許根據連接到資源的標籤來存取資源。
- 請求 – 控制您可以在請求中傳遞哪些標籤。若要執行此操作，請使用 aws:RequestTag/***key-name*** 條件金鑰，指定可以在請求中傳遞哪些標籤鍵/值組來標記 AWS 資源。
- 授權程序的任何部分：使用 aws:TagKeys 條件鍵，以控制特定標籤鍵是否可用於請求中。

您可以使用 JSON，或匯入現有的受管政策，透過視覺化的方式來建立 IAM 政策。如需詳細資訊，請參閱使用客戶管理政策定義自訂 IAM 許可。

> **Note**
>
> 如果使用者有權使用建立資源的動作，部分服務允許使用者在建立資源時指定標籤。

控制對 AWS 資源的存取

您可以使用 IAM 政策中的條件，根據該 AWS 資源上的標籤來控制對資源的存取。作法是使用全域 `aws:ResourceTag/tag-key` 條件金鑰，或服務特定索引鍵。某些服務僅支援此索引鍵的服務特定版本，而不支援全域版本。

> **Warning**
>
> 請勿嘗試透過標記角色，然後使用政策中的 ResourceTag 條件金鑰搭配 iam:PassRole 動作來控制誰可以傳遞角色。這種方法所產生的結果並不可靠。如需有關傳遞角色至服務之必要許可的詳細資訊，請參閱 授予使用者將角色傳遞至 AWS 服務的許可。

此範例會示範如何建立身分型政策，它允許啟動或停止 Amazon EC2 執行個體。只有在 Owner 執行個體標籤具有使用者名稱值時，才允許這些操作。此政策定義了程式設計和主控台存取的許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

您可以將此政策連接到您帳戶中的 IAM 使用者。如果使用者指定 `richard` 嘗試啟動 Amazon EC2 執行個體，您必須為該執行個體加上標籤 `Owner=richard` 或 `owner=richard`。否則，他將會收到存取遭拒。標籤鍵 `Owner` 同時符合 `Owner` 和 `owner`，因為條件金鑰名稱不區分大小寫。如需詳細資訊，請參閱IAM JSON 政策元素：Condition。

此範例會示範如何建立在資源 ARN 中使用 `team` 主體標籤的身分型政策。該政策會授與刪除 Amazon Simple Queue Service 佇列的許可，但前提是佇列名稱必須以團隊名稱為開頭，並在後面接著 `-queue`。例如，`qa-queue` if `qa` 是 `team` 主體標籤的團隊名稱。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2:111122223333:${aws:PrincipalTag/team}-
queue"
      }
}
```

在 AWS 請求期間控制存取

您可以使用 IAM 政策中的條件來控制哪些標籤鍵值對可以在將標籤套用至 AWS 資源的請求中傳遞。

此範例會示範如何建立身分型政策，它允許使用 Amazon EC2 `CreateTags` 動作以將標籤連接至執行個體。只有在標籤包含 `environment` 金鑰和 `preprod` 或 `production` 值時，您才能連接標籤。您可以視需要將 `ForAllValues` 修飾詞與 `aws:TagKeys` 條件金鑰搭配使用，表示只允許在請求中使用 `environment` 索引鍵。這會使得使用者無法包含其他金鑰，例如意外使用 `Environment` 而非 `environment`。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
```

```
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

根據標籤索引鍵控制存取權限

您可以在 IAM 政策中使用條件，以控制是否可在請求中使用特定標籤鍵。

我們建議您使用政策來控制使用標籤的存取時，請使用 aws:TagKeys條件金鑰。支援標籤的 AWS 服務可能允許您建立多個標籤金鑰名稱，這些名稱僅因案例而異，例如使用 stack=production和 標記 Amazon EC2 執行個體Stack=test。鍵名稱在政策條件中不區分大小寫。這表示如果您在政策的條件元素中指定 "aws:ResourceTag/TagKey1": "Value1"，則該條件會符合名為 TagKey1 或 tagkey1 的資源標籤鍵 (但不會同時符合兩者)。未免出現鍵大小寫差異導致的重複標籤，請使用 aws:TagKeys 條件來定義使用者可以套用的標籤鍵，或使用 AWS Organizations提供的標籤政策。如需詳細資訊，請參閱 AWS Organizations User Guide 中的 Tag Policies。

此範例會示範如何建立身分型政策，它允許建立和標記 Secrets Manager 密碼，但僅能使用標籤索引鍵 environment 或 cost-center。Null 條件可確保如果請求中沒有標籤，條件將評估為 false。

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
```

```
                    "environment",
                    "cost-center"
                ]
            }
        }
}
```

IAM 中的跨帳戶資源存取

對於某些 AWS 服務，您可以使用 IAM 授予 資源的跨帳戶存取權。若要執行此操作，您可將直接將資源政策連接到您要分享的資源，或將角色用作代理。

若要直接分享資源，您要分享的資源必須支援資源型政策。與角色的身分型政策不同，資源型政策指定誰 (主體) 可以存取該資源。

如果想要存取不支援資源型政策之其他帳戶中的資源，則請將角色用作代理。

如需有關這些政策類型之間差異的詳細資訊，請參閱 以身分為基礎和以資源為基礎的政策。

> **Note**
>
> IAM 角色和資源型政策只會在單一分割內跨帳戶委派存取許可。例如，您在標準 aws 分割區的美國西部 (加利佛尼亞北部) 中有一個帳戶。您在 aws-cn 分割區的中國也有一個帳戶。您無法在中國的帳戶中使用資源型政策，以允許標準 AWS 帳戶中的使用者存取 。

使用角色進行跨帳戶存取

並非所有 AWS 服務都支援以資源為基礎的政策。對於這些服務，在向多個服務提供跨帳戶存取權時，可以使用跨帳戶 IAM 角色集中管理許可。跨帳戶 IAM 角色是包含信任政策的 IAM 角色，允許另一個 AWS 帳戶中的 IAM 主體擔任該角色。簡單來說，您可以在一個 AWS 帳戶中建立角色，將特定許可委派給另一個 AWS 帳戶。

如需有關將政策連接至 IAM 身分的資訊，請參閱 管理 IAM 政策。

> **Note**
>
> 當主體切換到某個角色以暫時使用角色的許可時，他們會放棄其原始許可，並接受指派給他們假設之角色的許可。

讓我們看一下整體流程，因為其適用於需要存取客戶帳戶的 APN 合作夥伴軟體。

1. 客戶在自己的帳戶中使用政策建立 IAM 角色，以允許存取 APN 合作夥伴所需的 Amazon S3 資源。在此範例中，角色名稱為 APNPartner。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

2. 然後，客戶透過在角色的信任政策中提供 APN 合作夥伴的 AWS 帳戶 ID，指定APNPartner該角色可由合作夥伴 AWS 的帳戶擔任。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. 客戶將角色的 Amazon Resource Name (ARN) 提供給 APN 合作夥伴。ARN 是角色的完整名稱。

```
arn:aws:iam::Customer-Account-ID:role/APNPartner
```

> **Note**
>
> 建議在多租戶情況下使用外部 ID。如需詳細資訊，請參閱存取第三方 AWS 帳戶 擁有的。

4. 當 APN 合作夥伴的軟體需要存取客戶帳戶時，軟體會使用客戶帳戶中角色的 AWS Security Token Service ARN 呼叫 中的 AssumeRole API。STS 會傳回暫時 AWS 登入資料，允許軟體執行其工作。

如需使用角色授予跨帳户存取權的另一個範例，請參閱 在您擁有的另一個 IAM 使用者中 AWS 帳戶 存取。您也可以參照 IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權。

使用資源型政策的跨帳户存取權

帳戶使用資源型政策透過另一個帳戶存取資源時，主體仍可在受信任帳戶中運作，不需為了接受角色許可而放棄其許可。換言之，主體在存取信任帳戶中的資源時，仍可繼續存取受信任帳戶中的資源。這對於像複製資訊到其他帳戶中的共同資源，或從其複製而來的任務，非常受用。

您可以在資源型政策中指定的委託人包括帳戶、IAM 使用者、 AWS STS 聯合身分使用者委託人、SAML 聯合身分委託人、OIDC 聯合身分委託人、IAM 角色、擔任角色工作階段 AWS 或服務。如需詳細資訊，請參閱指定主體。

若要了解在您信任區域外帳戶 (信任組織或帳戶) 中的主體是否可擔任您的角色，請參閱識別與外部實體共用的資源。

下列清單包含支援資源型政策的一些 AWS 服務。如需支援將許可政策連接至資源而非委託人之不斷增加 AWS 的服務完整清單，請參閱 資源型欄中的 AWS 使用 IAM 的 服務並尋找具有是的服務。

- Amazon S3 儲存貯體 – 政策連接到儲存貯體，但政策控制項同時存取儲存貯體和其中的物件。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 Bucket policies for Amazon S3。在某些情況下，最好使用角色以跨帳戶存取 Amazon S3。如需詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的範例演練。
- Amazon Simple Notification Service (Amazon SNS) 主題 — 如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的 Amazon SNS 存取控制的範例案例。
- Amazon Simple Queue Service (Amazon SQS) 佇列 – 如需詳細資訊，請參閱《Amazon Simple Queue Service 開發人員指南》中的附錄：存取政策語言。

委派 AWS 許可的資源型政策

如果資源將許可授予您帳戶中的主體，您可以將這些許可委派給特定的 IAM 身分。身分是您帳戶中的使用者、使用者群組或角色。您可以將政策連接至身分以委派許可。您授予的許可上限為擁有資源的帳戶所允許的最大許可。

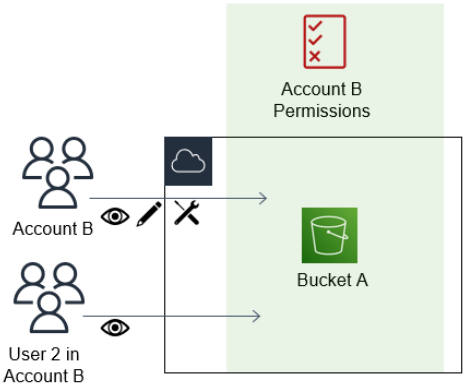
> **Important**
>
> 在跨帳戶存取中，主體在身分型政策和資源型政策中需要 `Allow`。

假設以資源為基礎的政策允許您帳戶中的所有主體具有資源的完整管理存取權。然後，您可以將完整存取權、唯讀存取權或任何其他部分存取權委派給您 AWS 帳戶中的委託人。或者，如果以資源為基礎的政策只允許列出許可，則您只能委派列出存取權。如果您嘗試委派比您的帳戶還多的許可，則您的主體仍然只有列出存取權。

如需有關如何做出這些決策的詳細資訊，請參閱確定帳戶內是否允許或拒絕請求。

> **Note**
>
> IAM 角色和資源型政策只會在單一分割內跨帳戶委派存取許可。例如，您無法在標準 `aws` 分割區的帳戶和 `aws-cn` 分割區的帳戶之間，新增跨帳戶存取權。

例如，假設您管理 `AccountA` 和 `AccountB`。在 AccountA 中，您具有名為 `BucketA` 的 Amazon S3 儲存貯體。



1. 您可以將資源型政策連接至 `BucketA`，以允許 AccountB 中的所有主體完整存取儲存貯體中的物件。他們可以建立、讀取或刪除該儲存貯體中的任何物件。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

透過將 AccountB 命名為資源型政策中的主體，AccountA 允許 AccountB 完整存取 BucketA。因此，AccountB 獲得授權可對 BucketA 執行任何動作，而 AccountB 管理員可以將存取權委派給它在 AccountB 的使用者。

AccountB 根使用者具有授予給帳戶的所有許可。因此，根使用者具有 BucketA 的完整存取權。

2. 在 AccountB 中，將政策連接至名為 User2 的 IAM 使用者。該政策允許使用者唯讀存取 BucketA 中的物件。這表示 User2 可以檢視物件，但無法建立、編輯或刪除物件。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Action" : [
                "s3:Get*",
                "s3:List*" ],
                "Resource" : "arn:aws:s3:::BucketA/*"
        }
    ]
```

```
}
```

AccountB 可委派的最大存取層級是授予帳戶的存取層級。在此情況下，資源型政策將完整存取授予 AccountB，但只將唯讀存取授予 User2。

AccountB 管理員不提供存取權給 User1。依預設，除了明確授予的許可外，使用者不具備任何許可，因此 User1 無權存取 BucketA。

IAM 會在主體提出請求時評估主體的許可。如果您使用萬用字元 (*) 為使用者提供 資源的完整存取權，委託人可以存取您 AWS 帳戶可存取的任何資源。即使對於您在建立使用者政策後新增或取得存取權的資源，也是如此。

在上述範例中，如果 AccountB 已將政策連接至 User2，而此政策允許完整存取所有帳戶中的所有資源，則 User2 可自動存取 AccountB 能夠存取的任何資源。這包括 BucketA 存取，以及 AccountA 中資源型政策所授予對任何其他資源的存取。

如需有關角色的複雜使用詳細資訊，例如授予應用程式和服務存取權，請參閱 IAM 角色的常見案例。

> **Important**
>
> 只將存取權給予您信任的實體，並提供最低必要存取權。每當受信任的實體是另一個 AWS 帳戶時，任何 IAM 主體都可以獲得資源的存取權。信任 AWS 的帳戶只能在授予存取權的範圍內委派存取權；其不能委派超過授予帳戶本身的存取權。

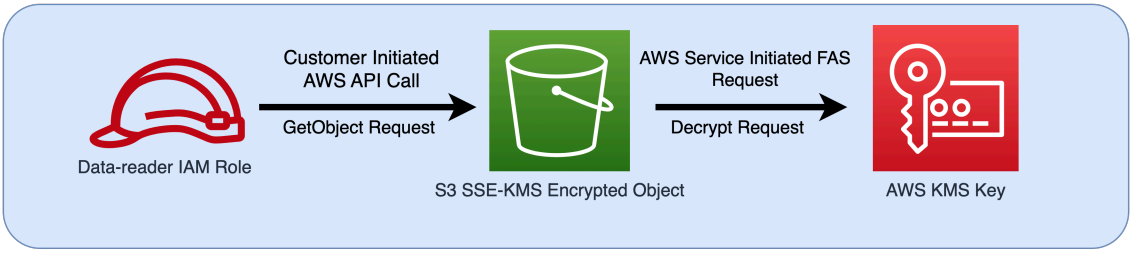
如需有關許可、政策以及用於撰寫政策的許可政策語言的詳細資訊，請參閱AWS 資源的存取管理。

轉送存取工作階段

轉送存取工作階段 (FAS) 是一種 AWS IAM 技術，服務會在 AWS 服務代表您提出請求時，用來傳遞您的身分、許可和工作階段屬性。FAS 使用呼叫 AWS 服務的身分許可，結合 AWS 服務的身分，向下游服務提出請求。FAS 請求只有在服務收到需要與其他 AWS 服務或資源互動才能完成的請求後，才會代表 IAM 主體對 AWS 服務提出。發出 FAS 請求時：

- 從 IAM 主體接收初始請求的服務會檢查 IAM 主體的許可。
- 接收後續 FAS 請求的服務也會檢查相同 IAM 主體的許可。

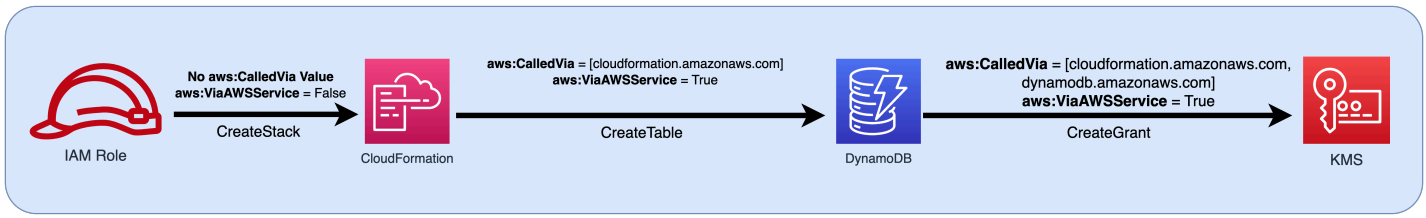
例如，當 SSE-KMS 用於加密物件時，Amazon S3 會使用 FAS 呼叫 來 AWS Key Management Service 解密物件。下載 SSE-KMS 加密物件時，名為資料讀取器的角色會針對 Amazon S3 呼叫物件上的 GetObject，而不會 AWS KMS 直接呼叫 。收到 GetObject 請求並授權資料讀取器後，Amazon S3 會向 發出 FAS 請求， AWS KMS 以解密 Amazon S3 物件。當 KMS 收到 FAS 請求時，其會檢查角色的許可，並且只有在 data-reader 具有 KMS 金鑰的正確許可時才授權解密請求。對 Amazon S3 和 的請求會使用角色的許可進行 AWS KMS 授權，而且只有在資料讀取器同時具有 Amazon S3 物件和 AWS KMS 金鑰的許可時，才會成功。



> **Note**
>
> 收到 FAS 請求的服務可以發出其他 FAS 請求。在這種情況下，請求主體必須具有 FAS 呼叫之所有服務的許可。

FAS 請求和 IAM 政策條件

發出 FAS 請求時，aws:CalledVia、aws:CalledViaFirst 和 aws:CalledViaLast 條件索引鍵會填入啟動 FAS 呼叫之服務的服務主體。每當發出 FAS 請求時，都會將 aws:ViaAWSService 條件索引鍵值設定為 `true`。在下圖中，直接向 CloudFormation 發出的請求沒有設定任何 `aws:CalledVia` 或 `aws:ViaAWSService` 條件索引鍵。當 CloudFormation 和 DynamoDB 代表角色發出下游 FAS 請求時，便會填入這些條件索引鍵的值。



若要允許發出 FAS 請求，以免遭具有測試來源 IP 地址或來源 VPC 之條件索引鍵的拒絕政策陳述式拒絕，您必須在您的拒絕政策中使用條件索引鍵，為 FAS 請求提供例外狀況。這可以透過使用 `aws:ViaAWSService` 條件索引鍵，為所有 FAS 請求完成此操作。若要僅允許特定 AWS 服務提出 FAS 請求，請使用 `aws:CalledVia`。

> **Important**
>
> 當透過 VPC 端點發出初始請求後發出 FAS 請求時，不會在 FAS 請求中使用來自初始請求中 aws:SourceVpce、aws:SourceVpc 和 aws:VpcSourceIp 的條件索引鍵值。當寫入政策使用 aws:VpcSourceIp 或 aws:SourceVpce 有條件地授予存取權時，您也必須使用 aws:ViaAWSService 或 aws:CalledVia 以允許 FAS 請求。當公有 AWS 服務端點收到初始請求後提出 FAS 請求時，後續的 FAS 請求將使用相同的aws:SourceIP條件索引鍵值提出。

範例：允許從 VPC 或使用 FAS 存取 Amazon S3

在以下 IAM 政策範例中，只有當 Amazon S3 GetObject 和 Athena 請求來自連接至 *example_vpc* 的 VPC 端點，或者請求是 Athena 發出的 FAS 請求時，才允許這些請求。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
          "vpc-111bbb22"
          ]
        }
      }
    },
    {
```

```
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

如需使用條件索引鍵允許 FAS 存取的其他範例，請參閱 data perimeter 範例政策儲存庫。

以身分為基礎的 IAM 政策範例

政策是 中的物件，當與身分或資源建立關聯時， AWS 會定義其許可。當 IAM 主體 （使用者或角色） 發出請求時， 會 AWS 評估這些政策。政策中的許可決定是否允許或拒絕請求。大多數政策會以連接至 IAM 身分 （使用者、使用者群組或角色） 的 JSON 文件 AWS 形式存放在 中。以身分為基礎的政策包括 AWS 受管政策、客戶受管政策以及內嵌政策。若要了解如何使用這些範例 JSON 政策文件來建立 IAM 政策，請參閱 the section called “正在使用 JSON 編輯器建立政策”。

在預設情況下，所有請求會遭拒絕，所以您必須提供對要存取身分的服務、動作和資源的存取。如果您還要允許存取在 IAM 主控台中完成指定的動作，則需要提供額外的許可。

以下政策程式庫可協助您定義 IAM 身分的許可。在您找到所需的政策後，選擇 View this policy (查看此政策) 以查看 JSON 的政策。您可以將 JSON 政策文件用作自己政策的範本。

> **Note**
>
> 如果您想提交要包含在本參考指南中的政策，請使用此頁面底部的 Feedback (意見回饋) 按鈕。

範例政策： AWS

- 在特定日期範圍期間允許存取。查看此政策。

- 允許啟用和停用 AWS 區域。查看此政策。
- 允許經過 MFA 身分驗證的使用者在 Security Credentials (安全憑證) 頁面上管理其憑證。查看此政策。
- 在特定日期範圍內使用 MFA 時允許特定存取。查看此政策。
- 允許使用者在安全憑證頁面上管理其憑證。查看此政策。
- 允許使用者在安全憑證頁面上管理其 MFA 裝置。查看此政策。
- 允許使用者在安全憑證頁面上管理其密碼。查看此政策。
- 允許使用者在安全憑證頁面上管理其密碼、存取金鑰和 SSH 公有金鑰。查看此政策。
- AWS 根據請求的區域拒絕對 的存取。查看此政策。
- AWS 根據來源 IP 地址拒絕對 的存取。查看此政策。

範例政策： AWS Data Exchange

- 拒絕存取您帳戶以外的 Amazon S3 資源，但 AWS Data Exchange除外。查看此政策。

範例政策： AWS Data Pipeline

- 拒絕存取使用者沒有建立的管道 (查看此政策。)

範例政策：Amazon DynamoDB

- 允許存取特定 Amazon DynamoDB 資料表 (查看此政策。)
- 允許存取特定 Amazon DynamoDB 屬性 (查看此政策。)
- 允許根據 Amazon Cognito ID 對 Amazon DynamoDB 進行項目層級存取 (查看此政策。)

範例政策：Amazon EC2

- 允許根據標籤將 Amazon EBS 磁碟區與 Amazon EC2 執行個體連接或分開 (查看此政策。)
- 允許以程式設計方式並在主控台的特定子網路中啟動 Amazon EC2 執行個體 (查看此政策。)
- 允許以程式設計方式及在主控台中管理與特定 VPC 相關聯的 Amazon EC2 安全群組 (查看此政策。)
- 允許以程式設計方式及在主控台中啟動或停用使用者已加上標籤的 Amazon EC2 執行個體 (查看此政策。)

- 允許以程式設計方式及在主控台中根據資源和主體標籤來啟用或停用 Amazon EC2 執行個體 (查看此政策。)
- 當資源與主體標籤相符時，允許啟用或停用 Amazon EC2 執行個體 (查看此政策。)
- 允許在特定區域內，以程式設計方式和在主控台進行 Amazon EC2 完全存取。查看此政策。
- 允許以程式設計方式和在主控台中啟動或停用特定 Amazon EC2 執行個體並修改特定安全群組 (查看此政策。)
- 拒絕在沒有 MFA 的情況下，存取特定 Amazon EC2 操作 (查看此政策。)
- 限制將 Amazon EC2 執行個體終止到特定的 IP 地址範圍 (查看此政策。)

範例政策： AWS Identity and Access Management (IAM)

- 允許存取政策模擬器 API (查看此政策。)
- 允許存取政策模擬器主控台 (查看此政策。)
- 能以程式設計方式和主控台中擔任擁有特定標籤的任何角色 (查看此政策。)
- 能以程式設計方式和主控台中允許和拒絕存取多個服務 (查看此政策。)
- 允許以程式設計方式和在主控台中使用不同的特定標籤，將特定標籤新增到 IAM 使用者 (查看此政策。)
- 允許以程式設計方式和在主控台中將特定標籤新增到任何 IAM 使用者或角色 (查看此政策。)
- 允許只使用特定標籤建立新使用者 (查看此政策。)
- 允許產生和擷取 IAM 憑證報告 (查看此政策。)
- 允許以程式設計方式及在主控台中管理群組的成員資格 (查看此政策。)
- 允許管理特定標籤 (查看此政策。)
- 允許將 IAM 角色傳遞至特定服務 (查看此政策。)
- 允許對 IAM 主控台的唯讀存取，而不需要報告 (查看此政策。)
- 允許對 IAM 主控台的唯讀存取 (查看此政策。)
- 允許特定使用者以程式設計方式及在主控台中管理群組 (查看此政策。)
- 允許以程式設計方式在主控台中設定帳戶密碼要求 (查看此政策。)
- 允許對具有特定路徑的使用者使用政策模擬器 API (查看此政策。)
- 允許對具有特定路徑的使用者使用政策模擬器主控台 (查看此政策。)
- 允許 IAM 使用者自行管理 MFA 裝置。查看此政策。
- 允許 IAM 使用者以程式設計方式和在主控台設定自己的憑證。查看此政策。

- 允許在 IAM 主控台中檢視 AWS Organizations 政策的服務上次存取資訊。查看此政策。
- 限制可以套用到新的 IAM 使用者、群組或角色的受管政策 (查看此政策。)
- 僅允許存取您帳戶中的 IAM 政策 (檢視此政策。)

範例政策： AWS Lambda

- 允許 AWS Lambda 函數存取 Amazon DynamoDB 資料表 (查看此政策。)

範例政策：Amazon RDS

- 允許在特定區域內的完整 Amazon RDS 資料庫存取。查看此政策。
- 允許以程式設計方式和在主控台中復原 Amazon RDS 資料庫 (查看此政策。)
- 允許標籤持有者擁有對已加上標籤的 Amazon RDS 資源的完整存取許可 (查看此政策)

範例政策：Amazon S3

- 允許 Amazon Cognito 使用者存取自己的 Amazon S3 儲存貯體中的物件 (查看此政策。)
- 允許具有臨時憑證的使用者以程式設計方式或在主控台中存取 Amazon S3 中他們自己的主目錄 (檢視此政策。)
- 允許完整的 S3 存取，但如果管理員並未在最後 30 分鐘內使用 MFA 登入，則會明確拒絕存取生產儲存貯體。(查看此政策。)
- 允許 IAM 使用者以程式設計方式或在主控台中存取 Amazon S3 中自己的主目錄 (查看此政策。)
- 允許使用者管理單一 Amazon S3 儲存貯體，並拒絕所有其他 AWS 動作和資源 (查看此政策。)
- 允許 `Read` 和 `Write` 存取特定的 Amazon S3 儲存貯體 (查看此政策。)
- 允許 `Read` 和`Write` 以程式設計方式和在主控台存取特定的 Amazon S3 儲存貯體 (查看此政策。)

AWS：允許根據日期和時間進行存取

此範例會示範如何建立身分型政策，允許根據日期和時間存取動作。此政策會限制只能在 2020 年 4 月 1 日和 2020 年 6 月 30 日 (UTC) (含) 間存取動作。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的`########`取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

若要了解使用 IAM 政策中 `Condition` 區塊內的多重條件的相關資訊，請參閱 條件中的多個值。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
            }
        }
    ]
}
```

> **Note**
>
> 您不可搭配使用政策變數與 Date 條件運算子。如需進一步了解，請參閱 條件元素

AWS：允許啟用和停用 AWS 區域

此範例會示範如何建立身分型政策，允許管理員啟用和停用亞太區域 (香港) 區域 (ap-east-1)。此政策定義了程式設計和主控台存取的許可。此設定會顯示在 AWS 管理主控台頁面中的 Account settings (帳戶設定)。本頁包含只有帳戶管理員能夠檢視和管理的敏感帳戶層級資訊。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

> **Important**
>
> 您無法啟用或停用預設啟用的區域。您只能包含預設停用的區域。如需詳細資訊，請參閱《AWS 一般參考》中的管理 AWS 區域。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。

此範例會示範如何建立身分型政策，允許使用多重要素驗證 (MFA) 進行身分驗證的 IAM 使用者在安全憑證頁面中管理自己的憑證。此 AWS 管理主控台 頁面會顯示如帳戶 ID 和正式使用者 ID 的帳戶資訊。使用者也可以檢視和編輯其密碼、存取金鑰、MFA 裝置、X.509 憑證及 SSH 金鑰和 Git 憑證。此政策範例包括檢視和編輯頁面上所有資訊所需的許可。它還要求使用者在 中執行任何其他操作之前，使用 MFA 設定和驗證 AWS。若要允許使用者在不使用 MFA 的情況下管理自己的憑證，請參閱AWS：允許 IAM 使用者在「安全憑證」頁面中管理其自己的憑證。

若要了解使用者如何存取安全憑證頁面，請參閱 IAM 使用者如何變更他們自己的密碼 (主控台)。

> **Note**
>
> - 此範例政策不允許使用者在 AWS 管理主控台 第一次登入 時重設密碼。在新使用者登入之前，建議您不要將許可授予給他們。如需詳細資訊，請參閱如何安全地建立 IAM 使用者？。這也可防止密碼到期的使用者在登入期間重設密碼。您可以透過將 `iam:ChangePassword`

和 iam:GetAccountPasswordPolicy 新增至 DenyAllExceptListedIfNoMFA 陳述式，來允許這項作業。不過，我們不建議您這麼做，因為若允許使用者在不使用 MFA 的情況下變更密碼，則會帶來安全性風險。

- 如果您打算使用此政策進行程式設計存取，則必須叫用 GetSessionToken 以使用 MFA 進行驗證。如需詳細資訊，請參閱透過 MFA 實現安全的 API 存取。

此政策的功能為何？

- AllowViewAccountInfo 陳述式允許使用者檢視帳戶層級資訊。這些許可必須位於自己的陳述式中，因為它們不支援或不需要指定資源 ARN。而是許可指定 "Resource" : "*"。此陳述式包括下列動作，可讓使用者檢視特定的資訊：
  - GetAccountPasswordPolicy – 檢視帳戶密碼需求，同時變更自己的 IAM 使用者密碼。
  - ListVirtualMFADevices – 檢視為使用者啟用之虛擬 MFA 裝置的詳細資訊。
- AllowManageOwnPasswords 陳述式可讓使用者變更自己的密碼。此陳述式也包括檢視 My Security Credentials (我的安全憑證) 頁面上大部分資訊所需的 GetUser 動作。
- AllowManageOwnAccessKeys 陳述式可讓使用者建立、更新及刪除自己的存取金鑰。使用者也能擷取關於指定之存取金鑰最後一次使用時間的資訊。
- AllowManageOwnSigningCertificates 陳述式可讓使用者上傳、更新及刪除自己的簽章憑證。
- AllowManageOwnSSHPublicKeys 陳述式允許使用者上傳、更新及刪除自己用於 CodeCommit 的 SSH 公有金鑰。
- AllowManageOwnGitCredentials 陳述式可讓使用者建立、更新及刪除自己用於 CodeCommit 的 Git 憑證。
- AllowManageOwnVirtualMFADevice 陳述式可讓使用者建立自己的虛擬 MFA 裝置。此陳述式中的資源 ARN 允許使用者以任何名稱建立 MFA 裝置，但政策中的其他陳述式僅允許使用者連接裝置到目前登入的使用者。
- AllowManageOwnUserMFA 陳述式可讓使用者檢視或管理其使用者的虛擬、U2F 或硬體 MFA 裝置。此陳述式中的資源 ARN 僅允許存取使用者自己的 IAM 使用者。使用者不能檢視或管理其他使用者的 MFA 裝置。
- DenyAllExceptListedIfNoMFA 陳述式拒絕存取所有 AWS 服務中的每個動作，但少數列出的動作除外，但前提是使用者未使用 MFA 登入。陳述式使用 "Deny" 和 "NotAction" 的組合，來明確拒絕存取未列出的每項動作。此陳述式不會拒絕或允許列出的項目。然而，政策中的其他陳述式會允許多項動作。如需有關此陳述式邏輯的詳細資訊，請參閱使用拒絕 (Deny) 的 NotAction。如果使用

者使用 MFA 登入，則不會通過 Condition 測試，且此陳述式不會拒絕任何動作。在此情況下，使用者的其他政策或陳述式決定使用者的許可。

此陳述式確保在使用者未使用 MFA 登入時，他們只能執行列出的動作。此外，只有在另一條陳述式或政策允許存取這些動作時，他們才能執行列出的動作。這不允許使用者在登入時建立密碼，因為在未進行 MFA 身分驗證的情況下，並不允許 iam:ChangePassword 動作。

...IfExists 運算子的 Bool 版本會確認，如果 aws:MultiFactorAuthPresent 索引鍵遺失，條件將返回 true。這就表示，拒絕使用長期憑證 (如存取金鑰) 存取 API 的使用者存取為非 IAM API 操作。

此政策不允許使用者檢視 IAM 主控台中的 Users (使用者) 頁面，或使用該頁面存取自己的使用者資訊。若要允許此操作，請將 iam:ListUsers 動作加入到 AllowViewAccountInfo 陳述式和 DenyAllExceptListedIfNoMFA 陳述式。它也不允許使用者在自己的使用者頁面上變更密碼。若要允許此操作，請將 iam:GetLoginProfile 及 iam:UpdateLoginProfile 動作加入 AllowManageOwnPasswords 陳述式。若也要允許使用者在未使用 MFA 登入的情況下，在自己的使用者頁面上變更其密碼，請將 iam:UpdateLoginProfile 動作加入 DenyAllExceptListedIfNoMFA 陳述式。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
```

```
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
```

```
            "iam:ResetServiceSpecificCredential",
            "iam:UpdateServiceSpecificCredential"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnVirtualMFADevice",
        "Effect": "Allow",
        "Action": [
            "iam:CreateVirtualMFADevice"
        ],
        "Resource": "arn:aws:iam::*:mfa/*"
    },
    {
        "Sid": "AllowManageOwnUserMFA",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice",
            "iam:EnableMFADevice",
            "iam:ListMFADevices",
            "iam:ResyncMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "DenyAllExceptListedIfNoMFA",
        "Effect": "Deny",
        "NotAction": [
            "iam:CreateVirtualMFADevice",
            "iam:EnableMFADevice",
            "iam:GetUser",
            "iam:GetMFADevice",
            "iam:ListMFADevices",
            "iam:ListVirtualMFADevices",
            "iam:ResyncMFADevice",
            "sts:GetSessionToken"
        ],
        "Resource": "*",
        "Condition": {
            "BoolIfExists": {
                "aws:MultiFactorAuthPresent": "false"
            }
        }
    }
```

```
    ]
}
```

AWS：允許在特定日期內使用 MFA 進行特定存取

此範例會示範如何建立身分型政策，以使用多個條件，且這些條件使用邏輯 AND 進行評估。它允許完整存取名為 SERVICE-NAME-1 的服務，以及存取名為 ACTION-NAME-A 服務內的 ACTION-NAME-B 和 SERVICE-NAME-2 動作。這些動作只有在使用多重驗證 (MFA) 驗證使用者時才獲允許進行。存取會限制於在 2017 年 7 月 1 日到 2017 年 12 月 31 日 (UTC)(包含在內) 之間發生的動作。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

若要了解使用 IAM 政策中 Condition 區塊內的多重條件的相關資訊，請參閱 條件中的多個值。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
    }
}
```

AWS：允許 IAM 使用者在「安全憑證」頁面中管理其自己的憑證

此範例會示範如何建立身分型政策，允許 IAM 使用者在安全憑證頁面中管理其所有憑證。 AWS 管理主控台 此頁面會顯示帳戶資訊，例如帳戶 ID 和正式使用者 ID。使用者也可以檢視和編輯其密碼、存取金鑰、X.509 憑證、SSH 金鑰和 Git 憑證。除了使用者的 MFA 裝置以外，此政策範例包括檢視和編

輯頁面上所有資訊所需的許可。若要允許使用者使用 MFA 管理自己的憑證，請參閱AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。。

若要了解使用者如何存取安全憑證頁面，請參閱 IAM 使用者如何變更他們自己的密碼 (主控台)。

此政策的功能為何？

- AllowViewAccountInfo 陳述式允許使用者檢視帳戶層級資訊。這些許可必須位於自己的陳述式中，因為它們不支援或不需要指定資源 ARN。而是許可指定 "Resource" : "*"。此陳述式包括下列動作，可讓使用者檢視特定的資訊：
  - GetAccountPasswordPolicy – 檢視帳戶密碼需求，同時變更自己的 IAM 使用者密碼。
  - GetAccountSummary – 檢視帳戶 ID 和帳戶正式使用者 ID。
- AllowManageOwnPasswords 陳述式可讓使用者變更自己的密碼。此陳述式也包括檢視 My Security Credentials (我的安全憑證) 頁面上大部分資訊所需的 GetUser 動作。
- AllowManageOwnAccessKeys 陳述式可讓使用者建立、更新及刪除自己的存取金鑰。使用者也能擷取關於指定之存取金鑰最後一次使用時間的資訊。
- AllowManageOwnSigningCertificates 陳述式可讓使用者上傳、更新及刪除自己的簽章憑證。
- AllowManageOwnSSHPublicKeys 陳述式允許使用者上傳、更新及刪除自己用於 CodeCommit 的 SSH 公有金鑰。
- AllowManageOwnGitCredentials 陳述式可讓使用者建立、更新及刪除自己用於 CodeCommit 的 Git 憑證。

此政策不允許使用者檢視或管理自己的 MFA 裝置。他們也不能檢視 IAM 主控台中的 Users (使用者) 頁面，或使用該頁面存取自己的使用者資訊。若要允許此操作，請將 iam:ListUsers 動作加入 AllowViewAccountInfo 陳述式。它也不允許使用者在自己的使用者頁面上變更密碼。若要允許此操作，請將 iam:CreateLoginProfile、iam:DeleteLoginProfile、iam:GetLoginProfile 及 iam:UpdateLoginProfile 動作加入 AllowManageOwnPasswords 陳述式。

JSON

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

```
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其自己的 MFA 裝置

此範例會示範如何建立身分型政策，允許透過多重要素驗證 (MFA) 進行身分驗證的 IAM 使用者在安全憑證頁面管理自己的 MFA 裝置。 AWS 管理主控台 此頁面會顯示帳戶和使用者資訊，但使用者只能檢視和編輯自己的 MFA 裝置。若要允許使用者使用 MFA 管理自己的所有憑證，請參閱AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。。

> **Note**
>
> 如果具有此政策的 IAM 使用者未經 MFA 驗證，則此政策會拒絕存取所有 AWS 動作，但使用 MFA 驗證所需的動作除外。若要使用 AWS CLI 和 AWS API，IAM 使用者必須先使用 AWS STS GetSessionToken 操作擷取其 MFA 字符，然後使用該字符來驗證所需的操作。其他政策 (例如資源型政策或其他身分型政策) 可允許其他服務中的動作。如果 IAM 使用者並未通過 MFA 身分驗證，此政策將會拒絕該存取。

若要了解使用者如何存取安全憑證頁面，請參閱 IAM 使用者如何變更他們自己的密碼 (主控台)。

此政策的功能為何？

- AllowViewAccountInfo 陳述式可讓使用者檢視為使用者啟用之虛擬 MFA 裝置的詳細資訊。由於此許可不支援指定資源 ARN，因此此許可必須在其陳述式中。您反而必須指定 "Resource" : "*"。
- AllowManageOwnVirtualMFADevice 陳述式可讓使用者建立自己的虛擬 MFA 裝置。此陳述式中的資源 ARN 允許使用者以任何名稱建立 MFA 裝置，但政策中的其他陳述式僅允許使用者連接裝置到目前登入的使用者。
- AllowManageOwnUserMFA 陳述式可讓使用者檢視或管理自己的虛擬、U2F 或硬體 MFA 裝置。此陳述式中的資源 ARN 僅允許存取使用者自己的 IAM 使用者。使用者不能檢視或管理其他使用者的 MFA 裝置。
- DenyAllExceptListedIfNoMFA 陳述式拒絕存取所有 AWS 服務中的每個動作，但少數列出的動作除外，但前提是使用者未使用 MFA 登入。陳述式使用 "Deny" 和 "NotAction" 的組合，來明確拒絕存取未列出的每項動作。此陳述式不會拒絕或允許列出的項目。然而，政策中的其他陳述式會允許多項動作。如需有關此陳述式邏輯的詳細資訊，請參閱使用拒絕 (Deny) 的 NotAction。如果使用者使用 MFA 登入，則不會通過 Condition 測試，且此陳述式不會拒絕任何動作。在此情況下，使用者的其他政策或陳述式決定使用者的許可。

  此陳述式確保在使用者未使用 MFA 登入時，他們只能執行列出的動作。此外，只有在另一條陳述式或政策允許存取這些動作時，他們才能執行列出的動作。

  ...IfExists 運算子的 Bool 版本會確認，如果 aws:MultiFactorAuthPresent 索引鍵遺失，條件將返回 true。這就表示，拒絕使用長期憑證 (如存取金鑰) 存取 API 操作的使用者存取為非 IAM API 操作。

此政策不允許使用者檢視 IAM 主控台中的 Users (使用者) 頁面，或使用該頁面存取自己的使用者資訊。若要允許此操作，請將 iam:ListUsers 動作加入到 AllowViewAccountInfo 陳述式和 DenyAllExceptListedIfNoMFA 陳述式。

> **Warning**
>
> 在沒通過 MFA 身分驗證的情況下不新增刪除 MFA 裝置的許可。使用此政策的使用者可能會嘗試為本身指派虛擬 MFA 裝置，但收到未獲授權執行 iam:DeleteVirtualMFADevice 的錯誤。如果發生這種情況，請勿將該許可新增到 DenyAllExceptListedIfNoMFA 陳述式。未通過 MFA 身分驗證的使用者一律不得刪除其 MFA 裝置。如果使用者先前開始將虛擬 MFA 裝

置指派到其使用者並取消該程序，使用者可能就會看到此錯誤。若要解決此問題，您或其他管理員必須使用 AWS CLI 或 AWS API 刪除使用者的現有虛擬 MFA 裝置。如需詳細資訊，請參閱我未獲得授權，不得執行：iam:DeleteVirtualMFADevice。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
```

```
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS：允許 IAM 使用者在「安全憑證」頁面中變更其主控台密碼

此範例示範如何建立身分型政策，允許 IAM 使用者在安全登入資料頁面上變更自己的 AWS 管理主控台 密碼。 AWS 管理主控台 此頁面會顯示帳戶和使用者資訊，但使用者只能存取自己的密碼。若要允許使用者使用 MFA 管理自己的所有憑證，請參閱AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。。若要允許使用者在不使用 MFA 的情況下管理自己的憑證，請參閱AWS：允許 IAM 使用者在「安全憑證」頁面中管理其自己的憑證。

若要了解使用者如何存取安全憑證頁面，請參閱 IAM 使用者如何變更他們自己的密碼 (主控台)。

此政策的功能為何？

- `ViewAccountPasswordRequirements` 陳述式可讓使用者檢視帳戶密碼需求，同時變更自己的 IAM 使用者密碼。
- `ChangeOwnPassword` 陳述式可讓使用者變更自己的密碼。此陳述式也包括檢視 My Security Credentials (我的安全憑證) 頁面上大部分資訊所需的 `GetUser` 動作。

此政策不允許使用者檢視 IAM 主控台中的 Users (使用者) 頁面，或使用該頁面存取自己的使用者資訊。若要允許此操作，請將 `iam:ListUsers` 動作加入 `ViewAccountPasswordRequirements` 陳述式。它也不允許使用者在自己的使用者頁面上變更密碼。若要允許此操作，請將 `iam:GetLoginProfile` 及 `iam:UpdateLoginProfile` 動作加入 `ChangeOwnPassword` 陳述式。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS：允許 IAM 使用者在「安全憑證」頁面中管理其密碼、存取金鑰和 SSH 公有金鑰

此範例會示範如何建立身分型政策，允許 IAM 使用者在安全憑證頁面中管理其密碼、存取金鑰和 X.509 憑證 。此 AWS 管理主控台 頁面會顯示如帳戶 ID 和正式使用者 ID 的帳戶資訊。使用者也可以檢視和編輯其密碼、存取金鑰、MFA 裝置、X.509 憑證、SSH 金鑰和 Git 憑證。此範例政策包括僅檢視和編輯其密碼、存取金鑰和 X.509 憑證所需的許可。若要允許使用者使用 MFA 管理自己的所有憑證，請參閱AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。。若要允許使用者在不使用 MFA 的情況下管理自己的憑證，請參閱AWS：允許 IAM 使用者在「安全憑證」頁面中管理其自己的憑證。

若要了解使用者如何存取安全憑證頁面，請參閱 IAM 使用者如何變更他們自己的密碼 (主控台)。

此政策的功能為何？

- `AllowViewAccountInfo` 陳述式允許使用者檢視帳戶層級資訊。這些許可必須位於自己的陳述式中，因為它們不支援或不需要指定資源 ARN。而是許可指定 `"Resource" : "*"`。此陳述式包括下列動作，可讓使用者檢視特定的資訊：

  - `GetAccountPasswordPolicy` – 檢視帳戶密碼需求，同時變更自己的 IAM 使用者密碼。
  - `GetAccountSummary` – 檢視帳戶 ID 和帳戶正式使用者 ID。
- `AllowManageOwnPasswords` 陳述式可讓使用者變更自己的密碼。此陳述式也包括檢視 My Security Credentials (我的安全憑證) 頁面上大部分資訊所需的 `GetUser` 動作。
- `AllowManageOwnAccessKeys` 陳述式可讓使用者建立、更新及刪除自己的存取金鑰。使用者也能擷取關於指定之存取金鑰最後一次使用時間的資訊。
- `AllowManageOwnSSHPublicKeys` 陳述式允許使用者上傳、更新及刪除自己用於 CodeCommit 的 SSH 公有金鑰。

此政策不允許使用者檢視或管理自己的 MFA 裝置。他們也不能檢視 IAM 主控台中的 Users (使用者) 頁面，或使用該頁面存取自己的使用者資訊。若要允許此操作，請將 `iam:ListUsers` 動作加入 `AllowViewAccountInfo` 陳述式。它也不允許使用者在自己的使用者頁面上變更密碼。若要允許此操作，請將 `iam:GetLoginProfile` 及 `iam:UpdateLoginProfile` 動作加入 `AllowManageOwnPasswords` 陳述式。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
```

```
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS： AWS 根據請求的區域拒絕對 的存取

此範例會示範如何建立身分型政策，拒絕以 aws:RequestedRegion 條件索引鍵所指定區域以外位置任何操作的存取權，但使用 NotAction 所指定服務的操作除外。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

此政策使用含 Deny 效果的 NotAction 元素，此效果會明確拒絕存取「未」列在陳述式中的所有動作。 支援 不應拒絕 CloudFront、IAM、Route 53 和服務中的動作，因為這些是熱門的 AWS 全域服務，具有實際位於 us-east-1區域的單一端點。這些服務的所有請求是對 us-east-1 區域提出，因此在沒有使用 NotAction 元素的情況下請求會遭拒。編輯此元素來為您使用的其他 AWS 全域服務 (例如，budgets、globalaccelerator、importexport、organizations 或 waf) 包含動作。一些其他全域服務，例如聊天應用程式中的 Amazon Q Developer AWS Device Farm，以及具有實際位於 us-west-2區域的端點的全域服務。要了解具有單一全域端點的所有服務，請參閱 AWS 一般參

考 中的 AWS 區域和端點。如需有關使用含 NotAction 效果之 Deny 元素的詳細資訊，請參閱 IAM JSON 政策元素：NotAction。

> **Important**
>
> 此政策不允許任何動作。將此政策與允許特定動作的其他政策結合使用。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "organizations:*",
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

AWS： AWS 根據來源 IP 拒絕對 的存取

此範例說明如何建立身分型政策，當請求來自指定 IP 範圍之外的主體時，拒絕存取帳戶中的所有 AWS 動作。當您公司的 IP 地址在指定範圍內時，該政策很有用。在此範例中，請求需要源自 CIDR 範圍 192.0.2.0/24 或 203.0.113.0/24 ，否則會遭到拒絕。政策不會拒絕 AWS 服務使用 提出的請求，轉送存取工作階段因為會保留原始請求者的 IP 地址。

請小心使用相同政策陳述式中的負面條件做為 `"Effect": "Deny"`。當您這麼做時，除了指定的委託人之外，政策陳述式中指定的動作在所有條件下都會明確拒絕授予。

> **Important**
>
> 此政策不允許任何動作。將此政策與允許特定動作的其他政策結合使用。

當其他政策允許動作時，主體可以從 IP 地址範圍內提出請求。 AWS 服務也可以使用委託人的登入資料提出請求。當主體從 IP 範圍外提出請求時，即會拒絕該請求。

如需有關使用 `aws:SourceIp` 條件鍵 (包括何時 `aws:SourceIp` 可能無法在政策中運作) 的詳細資訊，請參閱 AWS 全域條件內容索引鍵。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS：拒絕存取您帳戶外部的 Amazon S3 資源，但 AWS Data Exchange

此範例示範如何建立身分型政策，拒絕存取 AWS 中不屬於您帳戶的所有資源，但正常操作 AWS Data Exchange 所需的資源除外。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

您可以建立類似的政策來限制對組織或組織單位內資源的存取，同時使用條件索引鍵 `aws:ResourceOrgPaths`和 來說明 AWS Data Exchange 擁有的資源`aws:ResourceOrgID`。

如果您在 AWS Data Exchange 環境中使用 ，服務會建立資源並與之互動，例如服務帳戶擁有的 Amazon S3 儲存貯體。例如， 代表叫用 API 的 IAM 主體 （使用者或角色），將請求 AWS Data Exchange 傳送至 AWS Data Exchange 服務擁有的 Amazon S3 儲存貯體。 AWS Data Exchange APIs 在這種情況下，在政策`aws:ResourceOrgID`中使用 `aws:ResourceAccount`、`aws:ResourceOrgPaths`或 ，而不考慮 AWS Data Exchange 擁有的資源， 會拒絕存取服務帳戶擁有的儲存貯體。

- 陳述式 `DenyAllAwsResourcesOutsideAccountExceptS3` 會搭配使用 `NotAction` 元素與拒絕效果，明確拒絕存取陳述式中未列出的且不屬於所列帳戶的每個動作。`NotAction` 元素表示此陳述式的例外狀況。這些動作是此陳述式的例外狀況，因為如果對 建立的資源執行動作 AWS Data Exchange，政策會拒絕這些動作。
- `DenyAllS3ResourcesOutsideAccountExceptDataExchange` 陳述式會結合使用 `ResourceAccount` 和 `CalledVia` 條件，拒絕存取先前的陳述式中排除的 Amazon S3 動作。如果資源不屬於列出的帳戶並且如果呼叫服務不是 AWS Data Exchange，陳述式會拒絕執行動作。如果資源屬於列出的帳戶，或者列出的服務主體 `dataexchange.amazonaws.com` 執行操作，則陳述式不會拒絕執行動作。

> **Important**
>
> 此政策不允許任何動作。它使用 `Deny` 效果，明確拒絕存取陳述式中列出的不屬於所列帳戶的所有資源。將此政策與允許存取特定資源的其他政策結合使用。

以下範例示範如何設定政策，以允許存取所需的 Amazon S3 儲存貯體。

JSON

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

```
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWS Data Pipeline：拒絕存取使用者沒有建立的 DataPipeline 管道

此範例會示範如何建立身分型政策，拒絕存取使用者未建立的管道。如果 `PipelineCreator` 欄位的值符合 IAM 使用者名稱，則不會拒絕指定的動作。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。

> **Important**
>
> 此政策不允許任何動作。將此政策與允許特定動作的其他政策結合使用。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator":
 "${aws:userid}"}
            }
```

```
        }
    ]
}
```

Amazon DynamoDB：允許存取特定資料表

此範例會示範如何建立身分型政策，允許完整存取 MyTable DynamoDB 資料表。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

> **Important**
>
> 此政策允許可以在 DynamoDB 資料表上執行的所有動作。若要查看這些動作的詳細資訊，請參閱《Amazon DynamoDB 開發人員指南》中的 DynamoDB API 許可：動作、資源和條件參考。您可以透過列出每個個別的動作來提供相同的許可。不過，如果您在 Action 元素使用萬用字元 (*) (例如 "dynamodb:List*")，則在 DynamoDB 新增新的清單動作時，您就不需要更新您的政策。

此政策僅允許對具有指定名稱的 DynamoDB 資料表執行動作。若要允許使用者Read存取 DynamoDB 中的所有內容，您也可以連接 AmazonDynamoDBReadOnlyAccess AWS 受管政策。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
                "dynamodb:DescribeTable",
                "dynamodb:Get*",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB：允許存取特定屬性

此範例會示範如何建立身身分型政策，允許存取特定 DynamoDB 屬性。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

dynamodb:Select 要求防止 API 動作傳回任何不允許的屬性，例如來自索引投影的屬性。若要進一步了解 DynamoDB 條件金鑰，請參閱《Amazon DynamoDB 開發人員指南》中的指定條件：使用條件金鑰。若要了解使用 IAM 政策中的 Condition 區塊內之多重條件或多重條件索引鍵的相關資訊，請參閱 條件中的多個值。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
```

```
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select":
 "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB：允許根據 Amazon Cognito ID 對 DynamoDB 進行項目層級存取

此範例會示範如何建立身分型政策，允許根據 Amazon Cognito 身分池使用者 ID 對 `MyTable` DynamoDB 資料表進行項目層級存取。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的`########`取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

若要使用此政策，您必須建構 DynamoDB 資料表，使 Amazon Cognito 身分池使用者 ID 是分割區索引鍵。如需詳細資訊，請參閱《Amazon DynamoDB 開發人員指南》中的建立資料表。

若要進一步了解 DynamoDB 條件金鑰，請參閱《Amazon DynamoDB 開發人員指南》中的指定條件：使用條件金鑰。

JSON

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

```
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-
identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
}
```

Amazon EC2：根據標籤將 Amazon EBS 磁碟區與 EC2 執行個體連接或分開

此範例會示範如何建立身分型政策，允許 EBS 磁碟區擁有者將使用標籤 `VolumeUser` 定義的 EBS 磁碟區與標記為開發執行個體的 EC2 執行個體 (`Department=Development`) 連接或分開。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

如需有關建立 IAM 政策以控制對 Amazon EC2 資源的存取的詳細資訊，請參閱 Amazon EC2 User Guide 中的 Controlling Access to Amazon EC2 Resources。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
```

```
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2：允許以程式設計方式在特定子網路中，以及主控台中啟動 EC2 執行個體

此範例會示範如何建立身分型政策，允許列出所有 EC2 物件的資訊並在特定子網路中啟動 EC2 執行個體。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
```

```
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Amazon EC2：允許以程式設計方式和在主控台中管理具備特定標籤鍵值對的 EC2 安全群組

此範例顯示如何建立身分型政策，授予使用者對具有相同標籤的安全群組採取特定動作的許可。此政策會授予許可以在 Amazon EC2 主控台中檢視安全群組，並針對具有 Department=Test 標籤的現有安全群組新增和移除傳入與傳出規則，以及列出和修改規則說明。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
```

```
            "ec2:RevokeSecurityGroupEgress",
            "ec2:ModifySecurityGroupRules",
            "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
            "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
        ],
        "Resource": [
            "arn:aws:ec2:us-east-1:111122223333:security-group/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/Department": "Test"
            }
        }
      },
      {
        "Effect": "Allow",
        "Action": [
            "ec2:ModifySecurityGroupRules"
        ],
        "Resource": [
            "arn:aws:ec2:us-east-1:111122223333:security-group-rule/*"
        ]
      }
    ]
}
```

Amazon EC2：允許以程式設計方式和在主控台中開始或停止使用者已標記的 EC2 執行個體

此範例會示範如何建立身分型政策，允許 IAM 使用者啟動或停止 EC2 執行個體，但前提是執行個體標籤 `Owner` 的值是該使用者的使用者名稱。此政策定義了程式設計和主控台存取的許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
```

```
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

EC2：依據標籤啟動或停止執行個體

此範例會示範如何建立身分型政策，允許使用標籤鍵值對 `Project = DataAnalytics` 來啟動或停止執行個體，但僅可由具有標籤鍵值對 `Department = Data` 的主體進行。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的`#######`取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

如果條件的兩個部分都傳回 true，政策的條件就會傳回 true。執行個體必須擁有 `Project=DataAnalytics` 標籤。此外，發出請求的 IAM 主體 (使用者或角色) 都必須擁有 `Department=Data` 標籤。

> **Note**
>
> 對於某些使用者可能有而某些使用者可能沒有指定的標籤的案例而言，最佳實務是將具有 `aws:PrincipalTag` 條件索引鍵的政策連接至 IAM 群組。

JSON

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

```
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2：依據比對主體和資源的標籤啟動或停止執行個體

此範例會示範如何建立身分型政策，允許主體在執行個體的資源標籤與主體的標籤之標籤鍵 `CostCenter` 值相同時，啟動或停止 Amazon EC2 執行個體。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

> **Note**
>
> 對於某些使用者可能有而某些使用者可能沒有指定的標籤的案例而言，最佳實務是將具有 `aws:PrincipalTag` 條件索引鍵的政策連接至 IAM 群組。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
```

```
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
    }
}
```

Amazon EC2：允許在特定區域內，以程式設計方式和在主控台進行 EC2 完全存取

此範例會示範如何建立身分型政策，允許在特定區域內完整存取 EC2。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。如需區域代碼的清單，請參閱《Amazon EC2 使用者指南》中的可用區域。

或者，您可以使用全域條件金鑰 aws:RequestedRegion，它受所有 Amazon EC2 API 動作支援。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的範例：限制特定區域的存取。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Amazon EC2：允許以程式設計方式和在主控台中啟動或停用特 EC2 執行個體並修改安全群組

此範例會示範如何建立身分型政策，允許啟動或停止特定的 EC2 執行個體並修改特定的安全群組。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Amazon EC2：要求特定 EC2 操作的 MFA (GetSessionToken)

此範例示範如何建立身分型政策，允許完整存取 Amazon EC2 中的所有 AWS API 操作。不過，如果未透過多重要素驗證 (MFA) 來對使用者進行驗證，此政策會明確拒絕對 `StopInstances` 和 `TerminateInstances` API 操作的存取。若要以程式設計的方式執行，使用者必須在呼叫 `SerialNumber` 操作時包含選用 `TokenCode` 和 `GetSessionToken` 值。這個操作會傳回透過使用 MFA 驗證的臨時憑證。要進一步了解 GetSessionToken 的相關資訊，請參閱 為不信任環境中的使用者請求憑證。

此政策的功能為何？

- `AllowAllActionsForEC2` 陳述式會允許所有 Amazon EC2 動作。
- MFA 內容遺失時，`DenyStopAndTerminateWhenMFAIsNotPresent` 陳述式會拒絕 `StopInstances` 和 `TerminateInstances` 動作。這表示當多重驗證內容遺失 (表示未使用 MFA) 時，動作遭拒。拒絕會覆寫允許。

> **Note**
>
> `MultiFactorAuthPresent` 陳述式中 `Deny` 的條件檢查不應該是 `{"Bool": {"aws:MultiFactorAuthPresent":false}}`，因為該索引鍵不存在，並且在不使用 MFA 時無法評估。因此，在檢查值之前，使用 `BoolIfExists` 檢查來查看索引鍵是否存在。如需詳細資訊，請參閱...IfExists 條件運算子。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
```

```
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2：限制終止 EC2 執行個體到 IP 地址範圍

此範例會示範如何建立身分型政策，透過允許動作限制 EC2 執行個體，但當請求來自指定 IP 範圍之外時明確拒絕存取。當您公司的 IP 地址在指定範圍內時，該政策很有用。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

如果此政策與允許 ec2:TerminateInstances動作的其他政策（例如 AmazonEC2FullAccess AWS 受管政策）搭配使用，則會拒絕存取。這是因為明確拒絕陳述式的優先順序會在允許陳述式。如需詳細資訊，請參閱AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求。

> **Important**
>
> aws:SourceIp 條件金鑰拒絕存取代表您進行呼叫 AWS CloudFormation的 AWS 服務，例如 。如需有關使用 aws:SourceIp 條件索引鍵的詳細資訊，請參閱 AWS 全域條件內容索引鍵。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
```

```
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM：存取政策模擬器 API

此範例會示範如何建立身分型政策，允許將政策模擬器 API 用於連接到目前 AWS 帳戶中的使用者、群組或角色的政策。此政策還允許存取模擬做為字串傳送到 API 的不太敏感的政策。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> 若要允許使用者存取政策模擬器主控台，以模擬連接到目前 中使用者、群組或角色的政策 AWS 帳戶，請參閱 IAM：存取政策模擬器主控台。

IAM：存取政策模擬器主控台

此範例會示範如何建立身分型政策，允許將政策模擬器主控台用於連接到目前 AWS 帳戶中的使用者、群組或角色的政策。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "access-analyzer:ValidatePolicy",
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListPolicies",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
```

```
            "Resource": "*"
        }
    ]
}
```

IAM：擔任具有特定標籤的角色

此範例會示範如何建立身分型政策，允許 IAM 使用者擔任具標籤鍵值對 `Project = ExampleCorpABC` 的角色。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的*########*取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

如果具此標籤的角色存在於與使用者相同的帳戶中，則該使用者可擔任該角色。如果具此標籤的角色存在於與使用者不同的帳戶中，則需要額外的許可。跨帳戶角色的信任政策還必須允許使用者或使用者帳戶的所有成員擔任該角色。如需使用跨帳戶存取之角色的詳細資訊，請參閱 在您擁有的另一個 IAM 使用者中 AWS 帳戶 存取。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM：以程式設計方式和在主控台中允許和拒絕對多個服務的存取

此範例會示範如何建立身分型政策，允許完整存取 IAM 中的多項服務和有限的自我管理存取權限。此政策也拒絕對 Amazon S3 `logs` 儲存貯體或 Amazon EC2 `i-1234567890abcdef0` 執行個體的存取

權限。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

> **Warning**
>
> 此政策允許在多個服務中對每個動作和資源的完整存取。此政策應該僅適用於信任的管理員。

您可以將此政策用作為許可界線，來定義以身分為基礎的政策可以授予 IAM 使用者的許可上限。如需詳細資訊，請參閱 使用許可界限委派責任給他人。將此政策用作為使用者的許可界線時，此陳述式會定義以下界線：

- 此 AllowServices 陳述式會允許對指定 AWS 服務的完整存取。這表示使用者在這些服務的動作，僅受限於連接到使用者的許可政策。
- AllowIAMConsoleForCredentials 陳述式允許列出所有 IAM 使用者的存取權。這個存取權是導覽 AWS 管理主控台中 Users (使用者) 頁面的必要條件。它也允許檢視帳戶的密碼要求，對於要變更自身密碼的使用者來說這是必要的程序。
- AllowManageOwnPasswordAndAccessKeys 陳述式允許使用者僅管理自己的主控台密碼和程式設計存取金鑰。這很重要，因為如果另一個政策提供使用者完整的 IAM 存取權，則該使用者便可以變更自己或其他使用者的許可。此陳述式可避免此種情況發生。
- DenyS3Logs 陳述式明確拒絕存取 logs 儲存貯體。此政策會向該使用者強制執行公司限制。
- DenyEC2Production 陳述式明確拒絕存取 i-1234567890abcdef0 執行個體。

此政策不允許存取其他服務或動作。當政策用作使用者的許可界限時，即使連接到使用者的其他政策允許這些動作， 也會 AWS 拒絕請求。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

IAM：將特定標籤新增至具有特定標籤的使用者

此範例會示範如何建立身分型政策，允許對 IAM 使用者新增具有標籤值 Marketing、Development 或 QualityAssurance 的標籤索引鍵 Department。該使用者必須已包含標籤鍵值對 JobFunction = Manager。您可以使用此政策要求經理僅隸屬於下列三個部門之一。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

ListTagsForAllUsers 陳述式可讓您檢視帳戶中的所有使用者。

TagManagerWithSpecificDepartment 陳述式中的第一個條件是使用 StringEquals 條件運算子。如果條件的兩個部分都傳回 true，該條件就會傳回 true。即將標記的使用者必須已經擁有 JobFunction=Manager 標籤。該請求必須包含 Department 標籤鍵，以及其中一個列出的標籤值。

第二個條件會使用 ForAllValues:StringEquals 條件運算子。如果請求中的所有標籤鍵與政策中的鍵相符合，條件會傳回 true。這表示請求中的唯一標籤鍵一定是 Department。如需使用 ForAllValues 的詳細資訊，請參閱 用於多值內容索引鍵的集運算子。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
```

```
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM：新增特定標籤與特定值

此範例會示範如何建立身分型政策，允許對任何 IAM 使用者或角色僅新增標籤索引鍵 CostCenter 以及標籤值 A-123 或標籤值 B-456。您可以使用這個政策，限制標記至特定標籤鍵與一組標籤值。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

ConsoleDisplay 陳述式可讓您檢視帳戶中的所有使用者與角色。

AddTag 陳述式中的第一個條件是使用 StringEquals 條件運算子。如果請求的 CostCenter 標籤鍵包含其中一個列出的標籤值，該條件就會傳回 true。

第二個條件會使用 ForAllValues:StringEquals 條件運算子。如果請求中的所有標籤鍵與政策中的鍵相符合，條件會傳回 true。這表示請求中的唯一標籤鍵一定是 CostCenter。如需使用 ForAllValues 的詳細資訊，請參閱 用於多值內容索引鍵的集運算子。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
```

```
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM：建立僅具有特定標籤的新使用者

此範例會示範如何建立身分型政策，允許建立 IAM 使用者，但僅能使用 Department 和 JobFunction 標籤鍵 (其中之一或兩者)。Department 標籤鍵必須擁有 Development 或 QualityAssurance 標籤值。JobFunction 標籤鍵必須擁有 Employee 標籤值。您可以使用此政策要求新使用者具有特定任務函數與部門。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

陳述式中的第一個條件是使用 StringEqualsIfExists 條件運算子。如果請求中有採用 Department 或 JobFunction 鍵的標籤，則該標籤必須擁有指定值。如果上述鍵都不存在，則這個條件會評估為 true。只有當請求中出現其中一個指定的條件金鑰，且包含的值不同於允許的值時，條件才會評估為 false。如需使用 IfExists 的詳細資訊，請參閱 ...IfExists 條件運算子。

第二個條件會使用 ForAllValues:StringEquals 條件運算子。如果請求中的每個指定的標籤鍵值，至少與政策中的一個值之間相符，條件就會傳回 true。這表示請求中的所有標籤都

一定會屬於這份清單。不過，請求可以在清單中僅包含其中一個標籤。例如，您可以建立僅包含 Department=QualityAssurance 標籤的 IAM 使用者。不過，您無法建立同時包含 JobFunction=Employee 標籤和 Project=core 標籤的 IAM 使用者。如需使用 ForAllValues 的詳細資訊，請參閱 用於多值內容索引鍵的集運算子。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM：產生和擷取 IAM 憑證報告

此範例示範如何建立身分型政策，允許使用者產生和下載報告，列出其中的所有 IAM 使用者 AWS 帳戶。此報告包含使用者憑證的狀態，包括密碼、存取金鑰、MFA 裝置和簽章憑證。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。

如需有關憑證報告的詳細資訊，請參閱 為您的 產生登入資料報告 AWS 帳戶。

JSON

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

IAM：允許以程式設計方式及在主控台中管理群組的成員資格

此範例會示範如何建立身分型政策，允許更新名為 MarketingTeam 的群組成員。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

此政策的功能為何？

- ViewGroups 陳述式可讓使用者在 AWS 管理主控台中列出所有使用者和群組。它還可讓使用者檢視帳戶中使用者的相關基本資訊。這些許可必須位於自己的陳述式中，因為它們不支援或不需要指定資源 ARN。而是許可指定 "Resource" : "*"。
- ViewEditThisGroup 陳述式可讓使用者檢視 MarketingTeam 群組的相關資訊，以及從該群組新增和移除使用者。

此政策不允許使用者檢視或編輯使用者或 MarketingTeam 群組的許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:ListGroupsForUser"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM：管理特定標籤

此範例會示範如何建立身分型政策，允許使用標籤索引鍵 Department 從 IAM 實體 (使用者和角色) 新增和移除 IAM 標籤。此政策不會限制 Department 標籤的值。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

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

```
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM：將 IAM 角色傳遞至特定 AWS 服務

此範例會示範如何建立身分型政策，允許將任何 IAM 服務角色傳遞給 Amazon CloudWatch 服務。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

服務角色是 IAM 角色，可將 AWS 服務指定為可擔任該角色的委託人。此角色可讓服務擔任角色並代表您存取其他服務中的資源。若要允許 Amazon CloudWatch 擔任您傳遞的該角色，您必須將 `cloudwatch.amazonaws.com` 服務主體指定為角色信任政策中的主體。服務主體是由服務定義。若要了解服務的服務主體，請參閱該服務的文件。對於某些服務，請參閱 AWS 使用 IAM 的 服務 並尋找 Service-Linked Role (服務連結角色) 資料欄具有 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。搜尋 `amazonaws.com` 來檢視服務主體。

若要進一步了解將服務角色傳遞到服務的詳細資訊，請參閱 授予使用者將角色傳遞至 AWS 服務的許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
```

```
                "StringEquals": {"iam:PassedToService":
 "cloudwatch.amazonaws.com"}
            }
        }
    ]
}
```

IAM：允許對 IAM 主控台的唯讀存取，而不需要報告

此範例會示範如何建立身分型政策，允許 IAM 使用者執行以字串 Get 或 List 開頭的任何 IAM 動作。當使用者使用主控台時，主控台會向 IAM 發出請求，以列出群組、使用者、角色和政策，並產生與這些資源相關的報告。

星號會做為萬用字元。當您在政策中使用 iam:Get* 時，產生的許可包括以 Get 開頭的所有 IAM 動作，例如 GetUser 和 GetRole。如果在未來將新實體類型新增到 IAM 時萬用字元會很有用。在這種情況下，此政策授予的許可會自動允許使用者列出和取得這些新實體的詳細資訊。

此政策無法用於產生報告或服務上次存取的詳細資訊。如需了解允許此操作的其他政策，請參閱IAM：允許對 IAM 主控台的唯讀存取權。

JSON

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

IAM：允許對 IAM 主控台的唯讀存取權

此範例會示範如何建立身分型政策，允許 IAM 使用者執行以字串 Get、List 或 Generate 開頭的任何 IAM 動作。當使用者使用 IAM 主控台時，該主控台會發出請求，以列出群組、使用者、角色和政策，並產生與這些資源相關的報告。

星號會做為萬用字元。當您在政策中使用 iam:Get* 時，產生的許可包括以 Get 開頭的所有 IAM 動作，例如 GetUser 和 GetRole。使用萬用字元是有幫助的，特別是在未來將新的實體類型新增到 IAM 的情況。在這種情況下，此政策授予的許可會自動允許使用者列出和取得這些新實體的詳細資訊。

使用此政策進行主控台存取，其中包含產生報告或服務上次存取詳細資訊的許可。如需不允許產生動作的其他政策，請參閱 IAM：允許對 IAM 主控台的唯讀存取，而不需要報告。

JSON

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

IAM：允許特定 IAM 使用者以程式設計方式在主控台中管理群組

此範例會示範如何建立身分型政策，允許特定 IAM 使用者管理 AllUsers 群組。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

此政策的功能為何？

- 此 AllowAllUsersToListAllGroups 陳述式允許列出所有群組。主控台存取需要此功能。由於此許可不支援資源 ARN，因此此許可必須在其陳述式中。而是許可指定 "Resource" : "*"。
- 此 AllowAllUsersToViewAndManageThisGroup 陳述式允許可在群組資源類型上執行的所有群組動作。它不允許 ListGroupsForUser 動作，此動作可在使用者資源類型而不是群組資源類型上執行。如需有關可以為 IAM 動作指定的資源類型的詳細資訊，請參閱 AWS Identity and Access Management的動作、資源和條件金鑰。
- LimitGroupManagementAccessToSpecificUsers 陳述式會拒絕具有指定名稱的使用者存取寫入和許可管理群組動作。當政策中指定的使用者嘗試對群組進行變更，此陳述式不會拒絕請

求。AllowAllUsersToViewAndManageThisGroup 陳述式會允許該請求。如果其他使用者嘗試執行這些操作，請求會被拒絕。您可以在 IAM 主控台建立此政策時，檢視透過 Write (寫入) 或 Permissions management (許可管理) 存取層級拒絕的 IAM 動作。若要執行此操作，請從 JSON 索引標籤切換到 Visual editor (視覺編輯器) 索引標籤。如需存取層級的詳細資訊，請參閱 的動作、資源和條件金鑰 AWS Identity and Access Management。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
```

```
                    "srodriguez",
                    "mjackson",
                    "adesai"
                ]
            }
        }
    ]
}
```

IAM：允許以程式設計方式在主控台中設定帳戶密碼要求

此範例會示範如何建立身分型政策，允許使用者檢視及更新其帳戶的密碼要求。密碼要求會指定帳戶成員密碼的複雜性要求和強制輪換期間。此政策定義了程式設計和主控台存取的許可。

若要了解如何為您的帳戶設定帳戶密碼要求政策，請參閱 設定 IAM 使用者的帳戶密碼政策。

JSON

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

IAM：根據使用者路徑存取政策模擬器 API

此範例會示範如何建立身分型政策，允許僅對具有路徑 `Department/Development` 的使用者使用政策模擬器 API。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的`########`取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

> **Note**
>
> 若要建立允許有路徑 Department/Development 的使用者使用政策模擬器主控台的政策，請參閱 IAM：根據使用者路徑存取政策模擬器主控台。

IAM：根據使用者路徑存取政策模擬器主控台

此範例會示範如何建立身分型政策，允許僅對具有路徑 Department/Development 的使用者使用政策模擬器主控台。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
```

```
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

IAM：可讓 IAM 使用者自行管理 MFA 裝置

此範例會示範如何建立身分型政策，允許 IAM 使用者自行管理其多重要素驗證 (MFA) 裝置。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。

> **Note**
>
> 如果具有此政策的 IAM 使用者未經 MFA 驗證，則此政策會拒絕存取所有 AWS 動作，但使用 MFA 驗證所需的動作除外。如果您為登入的使用者新增這些許可 AWS，他們可能需要登出再重新登入，才能查看這些變更。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
```

```
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

IAM：允許 IAM 使用者以程式設計方式和在主控台更新自己的憑證

此範例會示範如何建立身分型政策，允許 IAM 使用者更新自己的存取金鑰、簽署憑證、服務專用憑證和密碼。此政策定義了程式設計和主控台存取的許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
```

```
        }
    ]
}
```

若要了解使用者可在主控台中變更密碼的詳細資訊，請參閱 the section called “IAM 使用者如何變更自己的密碼”。

IAM：檢視 AWS Organizations 政策的服務上次存取資訊

此範例說明如何建立身分型政策，以允許檢視特定 AWS Organizations 政策的上次存取服務資訊。此政策允許使用 p-policy123 ID 來擷取服務控制政策 (SCP) 的資料。產生和檢視報告的人員必須使用 AWS Organizations 管理帳戶登入資料進行身分驗證。此政策允許申請者擷取其組織中任何 AWS Organizations 實體的資料。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

如需上次存取資訊的重要資訊，包含必要的許可、疑難排解及支援區域，請參閱AWS 使用上次存取的資訊在 中精簡許可。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
```

```
            }
        }
    ]
}
```

IAM：限制可套用至 IAM 使用者、群組或角色的受管政策

此範例示範如何建立身分型政策，限制可套用至 IAM 使用者、群組或角色的客戶受管和 AWS 受管政策。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
        }
    }
}
```

AWS：拒絕存取您帳戶外部的資源 AWS ，受管 IAM 政策除外

在身分型政策中使用 aws:ResourceAccount 可能會影響使用者或角色利用某些服務，這些服務需要與某項服務所擁有帳戶中的資源互動。

您可以建立具有例外狀況的政策，以允許 AWS 受管 IAM 政策。您 AWS Organizations 所屬 受管 IAM 政策以外的服務受管帳戶。有四個列出和擷取受管政策的 IAM AWS動作。在陳述式的 NotAction 元素中使用這些動作。政策中的 AllowAccessToResourcesInSpecificAccountsAndSpecificService1。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda：允許 Lambda 函數存取 Amazon DynamoDB 資料表

此範例會示範如何建立身分型政策，允許對特定 Amazon DynamoDB 資料表進行讀寫存取。此政策也可將日誌檔寫入 CloudWatch Logs。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

若要使用此政策，請將政策連接到 Lambda 服務角色。服務角色是您在帳戶中建立的角色，以允許服務來代表您執行動作。該服務角色必須包含 AWS Lambda 作為信任政策中的委託人。如需如何使用此

政策的詳細資訊，請參閱 AWS 安全部落格中的如何建立 AWS IAM 政策以授予 Amazon DynamoDB 資料表的 AWS Lambda 存取權。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
            "Sid": "WriteLogStreamsAndGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
```

```
    ]
}
```

Amazon RDS：允許在特定區域內的完整 RDS 資料庫存取

此範例會示範如何建立身分型政策，允許在特定區域內完整存取 RDS 資料庫。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的*########*取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:us-east-1:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}
```

Amazon RDS：允許以程式設計方式以及在主控台之中還原 RDS 資料庫

此範例會示範如何建立身分型政策，允許還原 RDS 資料庫。此政策定義了程式設計和主控台存取的許可。

JSON

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

```
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon RDS：允許標籤持有者擁有對已標記的 RDS 資源的完整存取權限

此範例會示範如何建立身分型政策，允許標籤擁有者完整存取他們已加上標籤的 RDS 資源。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "rds:DeleteDBInstance",
```

```
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner":
"${aws:username}"}
            }
        },
```

```
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner":
 "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner":
 "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Amazon S3：可讓 Amazon Cognito 使用者存取其儲存貯體中的物件

此範例說明如何建立身分型政策，允許 Amazon Cognito 使用者存取特定 S3 儲存貯體中的物件。此政策僅允許存取名稱包含 `cognito` 的物件、應用程式的名稱和聯合身分主體 ID，以 ${cognito-identity.amazonaws.com:sub} 變數形式呈現。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的*########*取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

> **Note**
>
> 物件索引鍵中使用的 'sub' 值不是使用者集區中的使用者的 sub 值，而是身分集區中與使用者相關聯的身分 ID。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
      }
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
```

```
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

Amazon Cognito 為您的 Web 和行動應用程式提供身分驗證、授權和使用者管理。您的使用者可透過使用者名稱和密碼直接登入，或透過第三方 (例如 Facebook、Amazon 或 Google) 登入。

Amazon Cognito 的兩個主要元件是使用者集區和身分集區。使用者集區是一種使用者目錄，能為應用程式使用者提供註冊和登入的選項。身分集區可讓您授予使用者存取其他 AWS 服務的權限。您可以單獨或一併使用身分集區和使用者集區。

如需有關 Amazon Cognito 的詳細資訊，請參閱 Amazon Cognito 使用者指南。

Amazon S3：允許聯合身分使用者以程式設計方式和在主控台中存取其 Amazon S3 主目錄

此範例說明如何建立身分型政策，允許聯合身分主體在 S3 中存取自己的主目錄儲存貯體物件。主目錄是一個儲存貯體，其中包含 home 資料夾和個別聯合身分主體的資料夾。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

此政策中的 ${aws:userid} 變數會解析成 role-id:specified-name。聯合身分主體 ID 的 role-id 部分是在建立期間指派給聯合身分主體角色的唯一識別碼。如需詳細資訊，請參閱唯一識別碼。specified-name 是當聯合身分主體擔任其角色時傳遞至 AssumeRoleWithWebIdentity 請求的 RoleSessionName 參數。

您可以使用 AWS CLI 命令 檢視角色 IDaws iam get-role --role-name *specified-name*。例如，假設您指定易讀的名稱 John 且 CLI 傳回該角色 ID AROAXXT2NJT7D3SIQN7Z6。在這種情況下，聯合身分主體的使用者 ID 為 AROAXXT2NJT7D3SIQN7Z6:John。之後，此政策會允許聯合身分主體 John 使用字首 AROAXXT2NJT7D3SIQN7Z6:John 存取 Amazon S3 儲存貯體。

JSON

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

```
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3：S3 儲存貯體的存取，但生產儲存貯體在沒有最新 MFA 的情況下遭拒

此範例會示範如何建立身分型政策，允許 Amazon S3 管理員存取任何儲存貯體，包括更新、新增和刪除物件。不過，如果使用者沒有在最近三十分鐘內使用多重要素驗證 (MFA) 登入，則會明確拒絕 `amzn-s3-demo-bucket-production` 儲存貯體的存取權。此政策會授予在主控台中執行此動作或使用 或 AWS CLI AWS API 以程式設計方式執行此動作所需的許可。若要使用此政策，請將範例政策中的`########`取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

此政策永遠不會使用長期使用者存取金鑰來允許以程式設計方式存取 `amzn-s3-demo-bucket` 儲存貯體。這是使用 `aws:MultiFactorAuthAge` 條件索引鍵搭配 `NumericGreaterThanIfExists` 條件運算子一起完成。如果 MFA 不存在或 MFA 的存在時間大於 30 分鐘，則此政策條件會傳回 `true`。在這些情況下會拒絕存取。若要以程式設計方式存取 `amzn-s3-demo-bucket-production` 儲存貯體，S3 管理員必須使用過去 30 分鐘內以 GetSessionToken API 操作產生的臨時憑證。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetObject",
```

```
                "s3:GetObjectAcl",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-production/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3：可讓 IAM 使用者以程式設計方式和在主控台存取其 S3 主目錄

此範例會示範如何建立身分型政策，允許 IAM 使用者在 S3 中存取自己的主目錄儲存貯體物件。主目錄是一個儲存貯體，其中包含 home 資料夾和個別使用者的資料夾。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

使用 IAM 角色時，此政策將無法運作，因為使用 IAM 角色時無法使用 `aws:username` 變數。如需有關主要鍵值的詳細資訊，請參閱 主體索引鍵值。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
```

```
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3：限制管理特定的 S3 儲存貯體

此範例會示範如何建立身分型政策，將 Amazon S3 儲存貯體的管理限制到特定儲存貯體。此政策授予執行所有 Amazon S3 動作的許可，但拒絕存取每個 AWS 服務 ，但 Amazon S3 除外。請參閱以下範例。根據此政策，您只能存取可對 S3 儲存貯體或 S3 物件資源執行的 Amazon S3 動作。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

如果此政策與允許此政策拒絕動作的其他政策（例如 AmazonS3FullAccess 或 AmazonEC2FullAccess AWS 受管政策）搭配使用，則會拒絕存取。這是因為明確拒絕陳述式的優先順序會在允許陳述式。如需詳細資訊，請參閱AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求。

> **Warning**
>
> NotAction 和 NotResource 是必須謹慎使用的進階政策元素。此政策拒絕存取除 Amazon S3 以外的每個 AWS 服務。如果您將此政策連接到使用者，則會忽略向其他服務授予許可的任何其他政策，並拒絕存取。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

對 Amazon S3 儲存貯體物件授予讀取和寫入存取權

此範例會示範如何建立身分型政策，允許 Read 和 Write 存取特定 Amazon S3 儲存貯體中的物件。此政策授予從 AWS API 或 以程式設計方式完成此動作所需的許可 AWS CLI。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

s3:*Object 動作在動作名稱中使用萬用字元。AllObjectActions 陳述式允許 GetObject、DeleteObject、PutObject 以及任何其他以字元「Object」結尾的 Amazon S3 動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

> **Note**
>
> 若要允許 Read 和 Write 存取 Amazon S3 儲存貯體中的物件，並包含對主控台的其他存取權限，請參閱 Amazon S3：允許以程式設計方式和在主控台以讀取和寫入存取 S3 儲存貯體中的物件。

Amazon S3：允許以程式設計方式和在主控台以讀取和寫入存取 S3 儲存貯體中的物件

此範例會示範如何建立身分型政策，允許 Read 和 Write 存取特定 S3 儲存貯體中的物件。此政策定義了程式設計和主控台存取的許可。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

s3:*Object 動作在動作名稱中使用萬用字元。AllObjectActions 陳述式允許 GetObject、DeleteObject、PutObject 以及任何其他以字元「Object」結尾的 Amazon S3 動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
        }
    ]
```

```
}
```

管理 IAM 政策

IAM 提供您建立並管理所有 IAM 政策類型的工具 (受管政策與內嵌政策)。若要新增許可到 IAM 身分 (IAM 使用者、群組或角色)，您可以建立政策、驗證政策，然後將政策連接至身分。您可以將多個政策連接到身分，而每個政策可以包含多個許可。

主題

- 其他資源
- 使用客戶管理政策定義自訂 IAM 許可
- IAM 政策驗證
- 使用 IAM 政策模擬器測試 IAM 政策
- 新增和移除 IAM 身分許可
- 版本控制 IAM 政策
- 編輯 IAM 政策
- 刪除 IAM 政策
- AWS 使用上次存取的資訊在 中精簡許可

其他資源

下列資源可協助您進一步了解 AWS 政策。

- 如需有關不同 IAM 類型的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。
- 如需在 IAM 中使用政策的一般資訊，請參閱 AWS 資源的存取管理。
- 如需如何使用 IAM Access Analyzer 產生以實體存取活動為基礎之 IAM 政策的詳細資訊，請參閱產生 IAM Access Analyzer 政策。
- 如需了解當指定 IAM 身分有多個有效政策時的許可評估方式，請參閱 政策評估邏輯。
- AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

使用客戶管理政策定義自訂 IAM 許可

政策會定義 中身分或資源的許可 AWS。您可以使用 AWS 管理主控台 AWS CLI或 AWS API 在 IAM 中建立客戶受管政策。客戶管理政策是獨立的政策，在您自己的 AWS 帳戶中進行管理。然後，您可以將政策連接到 中的身分（使用者、群組和角色） AWS 帳戶。

身分型政策是 IAM 中連接到身分的政策。身分型政策可以包含 AWS 受管政策、客戶受管政策和內嵌政策。受 AWS 管政策是由 建立和管理 AWS，您可以使用它們，但無法管理它們。內嵌政策是您建立並接內嵌至 IAM 使用者群組、使用者或角色的政策。內嵌政策無法在其他身分上重複使用，或在其存在的身分之外進行管理。如需詳細資訊，請參閱新增和移除 IAM 身分許可。

一般而言，最好使用客戶受管政策，而不是內嵌政策或 AWS 受管政策。 AWS 受管政策通常提供廣泛的管理或唯讀許可。為了達到最高安全性，應授予最低權限，這表示僅授予執行特定任務工作所需的許可。

當您建立或編輯 IAM 政策時， AWS 可以自動執行政策驗證，以協助您建立最低權限的有效政策。在中 AWS 管理主控台，IAM 識別 JSON 語法錯誤，而 IAM Access Analyzer 提供額外的政策檢查與建議，以協助您進一步精簡政策。若要進一步了解政策驗證的資訊，請參閱 IAM 政策驗證。若要進一步了解 IAM Access Analyzer 政策檢查和可動作的建議，請參閱 IAM Access Analyzer 政策驗證。

您可以使用 AWS 管理主控台 AWS CLI或 AWS API 在 IAM 中建立客戶受管政策。如需使用 CloudFormation 範本新增或更新政策的詳細資訊，請參閱CloudFormation 《使用者指南》中的AWS Identity and Access Management 資源類型參考。

主題

- 建立 IAM 政策 (主控台)
- 建立 IAM 政策 (AWS CLI)
- 建立 IAM 政策 (AWS API)

建立 IAM 政策 (主控台)

政策為一個實體，可定義其所連接的身分或資源的許可。您可以使用 AWS 管理主控台 在 IAM 中建立客戶受管政策。客戶管理政策是獨立的政策，在您自己的 AWS 帳戶進行管理。然後，您可以將政策連接到 中的身分（使用者、群組和角色） AWS 帳戶。

AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

主題

- 建立 IAM 政策
- 正在使用 JSON 編輯器建立政策
- 使用視覺化編輯器來建立政策
- 匯入現有的受管政策

建立 IAM 政策

您可以使用下列 AWS 管理主控台 其中一種方法，在 中建立客戶受管政策：

- JSON — 貼上並自訂已發布的以身分為基礎的政策範例。
- 視覺編輯工具 — 在視覺編輯工具中從零開始建構一個新的政策。若您使用視覺化編輯器，您便無需了解 JSON 語法。
- 匯入 — 從帳戶中匯入並自訂受管政策。您可以匯入先前建立的 AWS 受管政策或客戶受管政策。

AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

正在使用 JSON 編輯器建立政策

您可以選擇 JSON 選項以在 JSON 中輸入或貼上政策。此方法對於複製要在帳戶中使用的範例政策很有幫助。或者，您可以在 JSON 編輯器中輸入自己的 JSON 政策文件。您也可以使用 JSON 選項，來在視覺化編輯器與 JSON 之間切換，以比較視圖。

當您建立或編輯 JSON 編輯器中的政策時，IAM 會執行政策驗證以協助您建立有效的政策。IAM 識別 JSON 語法錯誤，而 IAM Access Analyzer 會提供額外的政策檢查及可操作的建議，協助您進一步改良政策。

JSON 政策 文件為包含一或多個陳述式。每個陳述式應包含具有相同效果 (`Allow` 或 `Deny`) 並支援相同資源和條件的所有操作。如果一個動作要求指定所有資源 (`"*"`)，而另一個動作支援特定資源的 Amazon Resource Name (ARN)，則它們必須位於兩個單獨的 JSON 陳述式中。如需關於 ARN 格式的詳細資料，請參閱 AWS 一般參考 指南中的 Amazon Resource Name (ARN)。如需有關 IAM 政策的一般資訊，請參閱 中的政策和許可 AWS Identity and Access Management。如需有關 IAM 政策語言的資訊，請參閱 IAM JSON 政策參考。

若要使用 JSON 政策編輯器來建立政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。

3. 選擇建立政策。
4. 在政策編輯器中，選擇 JSON 選項。
5. 輸入或貼上 JSON 政策文件。如需有關 IAM 政策語言的詳細資訊，請參閱 IAM JSON 政策參考。
6. 解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。

> **Note**
>
> 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

7. （選用）當您在 中建立或編輯政策時 AWS 管理主控台，您可以產生可在 範本中使用的 JSON 或 YAML 政策 CloudFormation 範本。

   若要執行此動作，請在政策編輯器中選擇動作，然後選擇產生 CloudFormation 範本。若要進一步了解， CloudFormation 請參閱 AWS CloudFormation 《 使用者指南》中的AWS Identity and Access Management 資源類型參考。
8. 將許可新增至政策後，請選擇下一步。
9. 在檢視與建立頁面上，為您正在建立的政策輸入政策名稱與描述 (選用)。檢視此政策中定義的許可，來查看您的政策所授予的許可。
10. (選用) 藉由連接標籤作為鍵值組，將中繼資料新增至政策。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
11. 選擇 Create policy (建立政策) 儲存您的新政策。

在建立政策之後，即可將它連接至您的群組、使用者或角色。如需詳細資訊，請參閱 新增和移除 IAM 身分許可。

使用視覺化編輯器來建立政策

IAM 主控台中的視覺化編輯器將引導您建立政策，而無需編寫 JSON 語法。若要檢視使用視覺化編輯器建立政策的範例，請參閱 the section called “控制對身分的存取”。

若要使用視覺化編輯器來建立政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。
3. 選擇建立政策。

4. 在政策編輯器區段中，尋找選取服務區段，然後選擇 AWS 服務。您可用上方的搜尋框來限制服務清單中的結果。您僅可以選擇一項視覺化編輯器許可區塊中的服務。若要授予存取一個以上服務的許可，請選擇新增更多許可，來新增多個許可區塊。
5. 在動作中，選擇要新增至政策的動作。您可採用以下方式來選擇動作：

   - 選取所有動作的核取方塊。
   - 選擇 add actions (新增動作) 來輸入特定動作的名稱。您可以使用萬用字元 (*) 來指定多個動作。
   - 選取其中一個 Access level (存取層級) 群組，以選擇存取層級的所有動作 (例如，Read (讀取)、Write (寫入) 或 List (列出))。
   - 展開各個 Access level (存取級別) 群組來選擇個別動作。

   預設情況下，您建立的政策允許執行選擇的操作。若要拒絕選擇的動作，請選擇 Switch to deny permissions (切換為拒絕許可)。由於 IAM 會根據預設拒絕，作為安全最佳實務，我們建議您僅允許使用者所需的操作和資源的許可。只有在要覆蓋其他語句或政策單獨允許的許可時，才應建立 JSON 陳述式來拒絕許可。我們建議您將拒絕許可數限制為最低，因為它們可能會增加解決許可問題的難度。
6. 對於 Resources (資源)，如果您在先前步驟中選取的服務和動作不支援選擇特定資源，則會允許所有資源，而且您無法編輯此區段。

   如果選擇一或多個支援資源等級許可的動作，視覺化編輯器將列出這些資源。然後，您可以展開 Resources (資源) 來為您的政策指定資源。

   您可採用以下方式來指定資源：

   - 選擇新增 ARN，可根據它們的 Amazon Resource Name (ARN) 來指定資源。您可以使用視覺化 ARN 編輯器或手動列出 ARN。如需 ARN 語法的詳細資訊，請參閱 AWS 一般參考 指南中的 Amazon Resource Name (ARN)。如需使用政策之 `Resource` 元素中 ARN 的詳細資訊，請參閱IAM JSON 政策元素：Resource。
   - 選擇資源旁的此帳戶中的任何，將許可授予該類型的任何資源。
   - 選擇所有，可為服務選擇所有資源。
7. (選用) 選擇請求條件 - (選用)，為您正在建立的政策新增條件。條件可限制 JSON 政策陳述式的效果。例如，您可以指定只有在使用者的請求於特定時間範圍內發生時，使用者才能對資源執行動作。您也可以使用常用的條件，限制使用者必須使用多重要素驗證 (MFA) 裝置進行身分驗證。或者，您可以要求請求必須源自於特定 IP 地址範圍。如需可在政策條件中使用的所有內容金鑰清單，請參閱《服務授權參考》中的 AWS 服務的動作、資源和條件金鑰。

您可採用以下方式來選擇條件：

- 使用核取方塊來選擇常用條件。
- 選擇新增另一個條件，可指定其他條件。選擇條件的 Condition Key (條件金鑰)、Qualifier (函式) 以及 Operator (運算子)，然後輸入一個 Value (值)。若要新增超過一個值，請選擇新增。您可以將這些值視為藉由邏輯「OR」運算子相連。完成時，請選擇新增條件。

若要新增超過一個條件，請再次選擇新增另一個條件。視需要重複執行。每項條件僅適用於這一個視覺化編輯器許可區塊。所有條件的許可區塊皆須為 true 才會被視為符合。換句話說，可以將這些條件視為藉由邏輯「AND」運算子相連。

如需有關 Condition (條件) 元素的詳細資訊，請參閱 IAM JSON 政策參考 中的 IAM JSON 政策元素：Condition。

8. 若要新增更多許可區塊，請選擇新增更多許可。針對每個區塊皆重複步驟 2 到 5。

> **Note**
>
> 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

9. （選用）當您在 中建立或編輯政策時 AWS 管理主控台，您可以產生可在 範本中使用的 JSON 或 YAML 政策 CloudFormation 範本。

   若要執行此動作，請在政策編輯器中選擇動作，然後選擇產生 CloudFormation 範本。若要進一步了解， CloudFormation 請參閱 AWS CloudFormation 《使用者指南》中的AWS Identity and Access Management 資源類型參考。

10. 將許可新增至政策後，請選擇下一步。
11. 在檢視與建立頁面上，為您正在建立的政策輸入政策名稱與描述 (選用)。檢視此政策中定義的許可，可確認您已授予想要的許可。
12. (選用) 藉由連接標籤作為鍵值組，將中繼資料新增至政策。如需有關在 IAM 中使用標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。
13. 選擇 Create policy (建立政策) 儲存您的新政策。

在建立政策之後，即可將它連接至您的群組、使用者或角色。如需詳細資訊，請參閱 新增和移除 IAM 身分許可。

匯入現有的受管政策

若要建立新的政策，有一種簡單的方法是在您的帳戶中導入至少具有一部分所需許可權的現有受管政策。接著便可以自訂該政策，使其符合您的新要求。

您無法匯入內嵌政策。若要了解受管與內嵌政策之間的差異，請參閱 受管政策與內嵌政策。

若要在視覺化編輯器中匯入現有的受管政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。
3. 選擇建立政策。
4. 在政策編輯器中，選擇視覺化，然後在頁面右側選擇動作，再選擇匯入政策。
5. 在匯入政策視窗中，選擇最符合您想要放入新政策之政策內容的受管政策。您可用上方的搜尋框來限制政策清單中的結果。
6. 選擇匯入政策。

   匯入的政策新增於政策底部的新許可區塊中。
7. 使用 Visual editor (視覺編輯工具) 或選擇 JSON 來自訂您的政策。然後選擇下一步。

   > **Note**
   >
   > 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。
8. 在檢視與建立頁面上，為您正在建立的政策輸入政策名稱與描述 (選用)。您之後便無法編輯這些設定。檢視此政策中定義的許可，然後選擇建立政策來儲存您的工作。

若要在 JSON 編輯器中匯入現有的受管政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。
3. 選擇建立政策。
4. 在政策編輯器區段中，選擇 JSON 選項，然後在頁面右側選擇動作，再選擇匯入政策。

5. 在匯入政策視窗中，選擇最符合您想要放入新政策之政策內容的受管政策。您可用上方的搜尋框來限制政策清單中的結果。
6. 選擇匯入政策。

   來自匯入政策的陳述式新增於 JSON 政策底部。
7. 在 JSON 中自訂您的政策。解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。或者在 Visual editor (視覺編輯工具) 中自訂您的政策。然後選擇下一步。

   > **Note**
   >
   > 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。
8. 在檢視與建立頁面上，為您正在建立的政策輸入政策名稱與描述 (選用)。您之後便無法編輯這些內容。檢視政策此政策中定義的許可，然後選擇建立政策來儲存您的工作。

在建立政策之後，即可將它連接至您的群組、使用者或角色。如需詳細資訊，請參閱新增和移除 IAM 身分許可。

建立 IAM 政策 (AWS CLI)

政策為一個實體，可定義其所連接的身分或資源的許可。您可以使用 AWS CLI 在 IAM 中建立客戶受管政策。客戶管理政策是獨立的政策，在您自己的 AWS 帳戶進行管理。作為最佳實務，我們會建議使用您 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作。透過驗證政策，您可以先處理任何錯誤或建議，再將政策連接到您 AWS 帳戶的身分 (使用者、群組及角色)。

AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

建立 IAM 政策 (AWS CLI)

您可以使用 AWS Command Line Interface (AWS CLI) 來建立 IAM 客戶受管政策或者內嵌政策。

若要建立客戶受管政策 (AWS CLI)

使用下列命令：

- : create-policy

為 IAM 身分 (群組、使用者或角色) 建立內嵌政策 (AWS CLI)

請使用以下其中一個命令：

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> 您無法使用 IAM 為服務連結角色嵌入內嵌政策。

驗證客戶受管政策 (AWS CLI)

使用下列 IAM Access Analyzer 指令：

- validate-policy

建立 IAM 政策 (AWS API)

政策為一個實體，可定義其所連接的身分或資源的許可。您可以使用 AWS API 在 IAM 中建立客戶受管政策。客戶管理政策是獨立的政策，在您自己的 AWS 帳戶進行管理。作為最佳實務，我們會建議使用您 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作。透過驗證政策，您可以先處理任何錯誤或建議，再將政策連接到您 AWS 帳戶的身分 (使用者、群組及角色)。

AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

建立 IAM 政策 (AWS API)

您可以使用 AWS API 來建立 IAM 客戶受管政策或者內嵌政策。

建立客戶受管政策 (AWS API)

呼叫以下操作：

- CreatePolicy

為 IAM 身分 （群組、使用者或角色） 建立內嵌政策 (AWS API)

呼叫以下其中一項操作：

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> 您無法使用 IAM 為服務連結角色嵌入內嵌政策。

驗證客戶受管政策 (AWS API)

呼叫下列 IAM Access Analyzer 操作：

- ValidatePolicy

IAM 政策驗證

政策是一種使用 IAM 政策文法的 JSON 文件。當您將政策連接到 IAM 實體 (例如使用者、群組或角色) 時，該政策會對該實體授予許可。

當您使用 建立或編輯 IAM 存取控制政策時 AWS 管理主控台， AWS 會自動檢查它們，以確保它們符合 IAM 政策文法。如果 AWS 判斷政策不符合文法，則會提示您修復政策。

IAM Access Analyzer 會提供額外的政策檢查及建議，協助您進一步改良政策。若要進一步了解 IAM Access Analyzer 政策檢查和可動作的建議，請參閱 IAM Access Analyzer 政策驗證。若要檢視 IAM Access Analyzer 傳回的警告、錯誤和建議清單，請參閱 IAM Access Analyzer 政策檢查參考。

驗證範圍

AWS 會檢查 JSON 政策語法和文法。同時還會驗證您的 ARN 格式是否適當，以及動作名稱和條件索引鍵是否正確。

存取政策驗證

在 AWS 管理主控台中建立 JSON 政策或編輯現有政策時，會自動驗證政策。如果政策語法無效，您將收到通知，並且必須先解決問題，然後才能繼續。 AWS 管理主控台 如果您有 的許可，IAM Access Analyzer 政策驗證的結果會自動在 中傳回`access-analyzer:ValidatePolicy`。您也可以使用 API AWS 或 驗證政策 AWS CLI。

現有政策

您可能有無效的現有政策，因為這些政策是在政策引擎的最新更新之前建立或上次儲存的政策。作為最佳實務，我們會建議使用您 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作。建議您開啟現有政策，並檢閱所產生的政策驗證結果。如果不修復任何政策語法錯誤，則無法編輯和儲存現有政策。

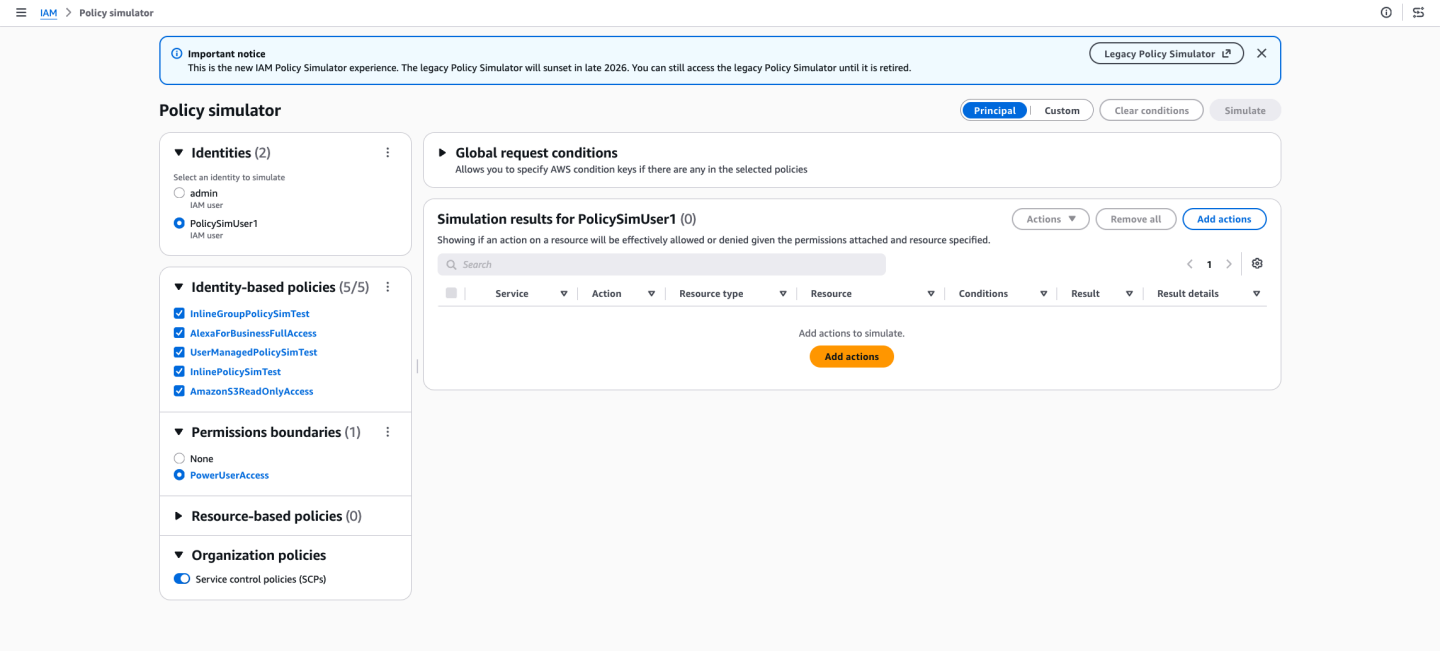
使用 IAM 政策模擬器測試 IAM 政策

使用 IAM 政策模擬器，您可以測試身分型政策、IAM 許可界限、服務控制政策 (SCPs)，以及您提供的資源型政策。您會模擬 AWS 如何根據這些政策評估請求，而不將實際請求傳送至任何 AWS 服務。每個模擬都會針對您測試的每個動作和資源傳回二進位授權結果：允許或拒絕該動作。當結果是允許或明確拒絕時，模擬器也會顯示產生該結果的政策，讓您可以了解並修正原因。

如需有關如何以及為什麼要使用 IAM 政策的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。

> **Important**
>
> 政策模擬器結果可能與您的即時 AWS 環境不同。建議您在使用政策模擬器進行測試後，針對即時 AWS 環境檢查政策，以確認您擁有所需的結果。如需詳細資訊，請參閱IAM 政策模擬器的運作方式。

下列螢幕擷取畫面顯示 IAM 政策模擬器，您可以在其中選取身分及其政策，並在執行模擬之前選擇要測試的動作和資源。



主題

- 政策模擬器模式
- 您可以使用政策模擬器做什麼
- IAM 政策模擬器的運作方式
- IAM 政策模擬器中條件索引鍵評估的運作方式
- 使用政策模擬器的許可
- 如何模擬政策

政策模擬器模式

政策模擬器有兩種模式。選擇符合您想要測試帳戶中已存在的政策，或是您仍在起草的政策的模式。

主體模式

在主體模式中，您會測試已連接至現有 IAM 使用者、角色或群組的政策。您可以為模擬選擇一個身分，而模擬器會評估連接到該身分的政策。您也可以包含或排除模擬的自訂身分政策或許可界限。這些新增項目模擬為連接到委託人，而不是實際連接到 IAM，因此您的帳戶保持不變。

自訂模式

在自訂模式中，您會測試一個或多個您寫入或貼上的政策。這對於在連接政策之前檢查政策非常有用。自訂政策是自行模擬的，不會儲存至您的帳戶。

如需選取模式和執行模擬的詳細step-by-step說明，請參閱 如何模擬政策。

您可以使用政策模擬器做什麼

以下是您可以使用政策模擬器執行的一些常見作業。您可以在 中找到更詳細的step-by-step指南，以及 API 和主控台行為之間的差異如何模擬政策。

- 測試身分型政策，無論它們是否已連接到 IAM 使用者、群組或角色，或是您寫入或貼上的新政策。對於具有多個政策的連接身分，您可以一起測試所有政策或選取個別政策，並查看特定資源允許或拒絕哪些動作。新政策的評估就像是連接到身分一樣，而不會實際連接到您的帳戶；它們僅用於模擬，不會儲存到您的帳戶。
- 測試限制性控制對存取的影響。您可以一次模擬一個許可界限。如果您的 AWS 帳戶 是 中組織的成員AWS Organizations，您也可以測試服務控制政策 (SCPs) 對您身分型政策的影響。模擬器會告訴

您 SCP 是否允許或拒絕動作，但基於安全考量，它不會像其他政策類型一樣從 SCP 浮現相符的陳述式。

- 測試您提供的以資源為基礎的政策的效果。除了 主控台之外，模擬器不會為您擷取資源的政策。若要在模擬中包含資源型政策，您可以提供政策與資源。
- 在實際情況下測試特定服務、動作和資源，並識別決定結果的因素。例如，您可以測試政策是否允許在特定儲存貯體上 Amazon S3 中的 `CreateBucket`、`ListAllMyBuckets`、 和 `DeleteBucket`動作，並提供內容索引鍵，例如正在測試的政策中`Condition`元素所參考的 IP 地址或日期。對於每個結果，模擬器會識別哪些特定陳述式允許或拒絕存取；相符的陳述式可能來自身分型政策和您提供的資源型政策。

IAM 政策模擬器的運作方式

當您執行模擬時，政策模擬器會評估請求範圍內的政策、範圍內的任何 SCPs，以及您提供的輸入，例如動作、資源和內容索引鍵值。在自訂模式中，您可以直接提供政策。在主體模式中，模擬器會使用連接到您所選身分的政策，而且您可以在模擬中包含或排除更多政策。結果是每個動作和資源的二進位結果：允許或拒絕。

政策模擬器與即時 AWS 環境的差別如下：

- 政策模擬器不會提出實際 AWS 的服務請求，因此您可以安全地測試請求，否則可能會對您的即時 AWS 環境進行不必要的變更。它不會使用生產環境中的真實內容索引鍵值，而且因為動作實際上並未執行，所以不會傳回任何服務回應。唯一的結果是是否允許或拒絕每個請求的動作。
- 如果您在政策模擬器中編輯政策，變更只會影響模擬。您 中的對應政策 AWS 帳戶 保持不變。
- 政策模擬器會評估 SCPs，包括拒絕陳述式中的條件索引鍵和資源範圍，但不支援資源控制政策 (RCPs)。模擬結果仍然可能與某些進階組態的即時行為不同，例如 VPC 端點政策、角色鏈結或單一資源上的多個資源型政策。
- 跨帳戶模擬會在 中傳回per-policy-type決策 （身分政策和資源政策）`EvalDecisionDetails`。

IAM 政策模擬器中條件索引鍵評估的運作方式

許多政策使用相依於條件索引鍵的`Condition`元素，例如 `aws:RequestedRegion`或 `aws:MultiFactorAuthPresent`。當您執行模擬時，政策模擬器會從兩個來源取得條件索引鍵的值：模擬器自動填入的值，以及您作為模擬輸入提供的值。如果值不符合條件，則會拒絕動作。

模擬器自動填入的內容

模擬器會填入下列主體和組織內容索引鍵以進行評估：

- 主體內容：aws:PrincipalAccount、aws:PrincipalId、aws:PrincipalType、aws:Type、aws:UserId、aws:UserName
- 組織內容：aws:PrincipalOrgID、aws:PrincipalOrgMasterAccountId、aws:PrincipalOrgPaths

由於這些金鑰會為您填入，因此使用委託人或組織條件SCPs 和其他政策會正確評估。您可以為政策參考的所有其他條件索引鍵提供值。檢閱您正在測試的政策中的Condition元素，並針對不在上述清單中的任何金鑰提供值。

全域條件金鑰和服務支援

對於身分型政策和資源型政策，模擬器不會判斷哪些服務支援指定的全域條件金鑰以進行授權。例如，模擬器不會識別服務不支援 aws:TagKeys。相反地，服務控制政策 (SCPs) 是以服務感知邏輯進行評估，包括全域條件索引鍵，例如 aws:RequestedRegion和 aws:PrincipalAccount。如需條件索引鍵的詳細資訊，請參閱 AWS 全域條件內容索引鍵。

服務控制政策 (SCPs中的條件索引鍵

模擬器會評估 SCPs 中的條件索引鍵和資源範圍。如何提供這些金鑰的值取決於您執行模擬的方式：

- 使用 AWS CLI 或 AWS API，您可以為 SCPs 參考的條件索引鍵提供值。
- 在 IAM 主控台中，您只能為出現在身分型政策、許可界限或資源型政策中的條件金鑰設定值。如果只有 SCP 參考條件索引鍵，您就無法在主控台中為其設定值。不過，如果其中一個其他政策中也顯示相同的條件索引鍵，則您設定的值會用於整個評估，包括 SCP。

基於安全考量，SCP 評估不會傳回缺少的內容值。您仍然會收到請求的允許或拒絕決定。

使用政策模擬器的許可

您需要的許可取決於您使用的是 AWS CLI 或 AWS API，還是 IAM 主控台，以及您使用的模式。

Console

您需要呼叫 的許可iam:GetContextKeysForCustomPolicy，才能在 主控台上使用 IAM 政策模擬器。

您需要的其他許可取決於 模式。在主體模式下，您需要額外的讀取許可，才能列出帳戶中的身分並讀取其連接的政策。若要減少必須授予的項目，如果您只需要測試貼上的政策，請使用自訂模式。

自訂模式

若要使用自訂模式，在您要測試的政策中貼上，您也需要 iam:SimulateCustomPolicy

主體模式

若要使用主體模式，主控台會在其中載入並顯示帳戶中的身分和政策，您需要常見的許可加上下列項目。這是較大的集合，因為主控台必須探索身分並讀取其連接的政策。

模擬：

- iam:SimulatePrincipalPolicy

列出身分：

- iam:ListUsers, iam:ListRoles, iam:ListGroups
- iam:ListGroupsForUser

列出連接的政策：

- iam:ListPolicies, iam:ListUserPolicies, iam:ListRolePolicies, iam:ListGroupPolicies
- iam:ListAttachedUserPolicies, iam:ListAttachedRolePolicies, iam:ListAttachedGroupPolicies

取得政策文件：

- iam:GetPolicyVersion, iam:GetUserPolicy, iam:GetRolePolicy, iam:GetGroupPolicy

取得許可界限：

- iam:GetUser, iam:GetRole

主體模式的其他許可，以模擬以資源為基礎的政策

若要在主體模式模擬中包含資源的政策，請新增該資源類型的讀取許可：

- `s3:GetBucketPolicy`, `sns:GetTopicAttributes`, `sqs:GetQueueUrl`, `sqs:GetQueueAttributes`, `glacier:GetVaultAccessPolicy`

如需授予模擬器存取權的政策範例，請參閱 the section called “範例政策： AWS Identity and Access Management (IAM)”。

AWS CLI and AWS API

對於 AWS CLI 和 AWS API，您只需要 許可，即可針對要測試的政策類型呼叫模擬動作。如果您先使用 `iam:GetContextKeysForCustomPolicy`和 `iam:GetContextKeysForPrincipalPolicy` 來探索內容金鑰，請同時授予這些動作的許可。基於安全考量，動作會分為下列兩種模式。僅授予符合您要模擬之內容的許可。

模擬您以字串形式直接傳遞至 API 的政策 （ 自訂政策 ）

- `iam:SimulateCustomPolicy`
- `iam:GetContextKeysForCustomPolicy` （ 僅當您使用 context-key 探索步驟時 ）

這些動作可讓您測試尚未連接到使用者、使用者群組或角色的政策。您以字串形式傳遞政策，它們僅用於模擬。

模擬連接到 IAM 使用者、使用者群組、角色或資源的政策

- `iam:SimulatePrincipalPolicy`
- `iam:GetContextKeysForPrincipalPolicy` （ 僅當您使用 context-key 探索步驟時 ）

由於這些動作可以顯示指派給其他 IAM 實體的許可，因此請考慮限制誰可以呼叫它們。例如，若要授予名為 Bob 的使用者許可來模擬指派給名為 Alice 的使用者的政策，請授予 Bob 存取 資源的權限`arn:aws:iam::777788889999:user/alice`。

若要檢視允許將政策模擬器 API 用於目前中已連接和未連接政策的範例政策 AWS 帳戶，請參閱 IAM：存取政策模擬器 API。若要檢視範圍限定為具有特定路徑之使用者的範例政策，請參閱 IAM：根據使用者路徑存取政策模擬器 API。

如何模擬政策

您可以在 IAM 主控台或使用 AWS CLI 和 AWS API 執行模擬。如需模擬器評估內容、兩種模式及其如何達到結果的概觀，請參閱 使用 IAM 政策模擬器測試 IAM 政策。如需所需的許可，請參閱 使用政策模擬器的許可。

主題

- 模擬政策 （主控台）
- 模擬政策 (AWS CLI 和 AWS API)

模擬政策 （主控台）

您可以從 IAM 主控台中的政策模擬器執行模擬。在導覽窗格中，選擇政策模擬器。

> **Note**
>
> 位於 https：//https://policysim.aws.amazon.com/ 的獨立政策模擬器主控台不再維護。在 IAM 主控台中使用政策模擬器。

主體模式

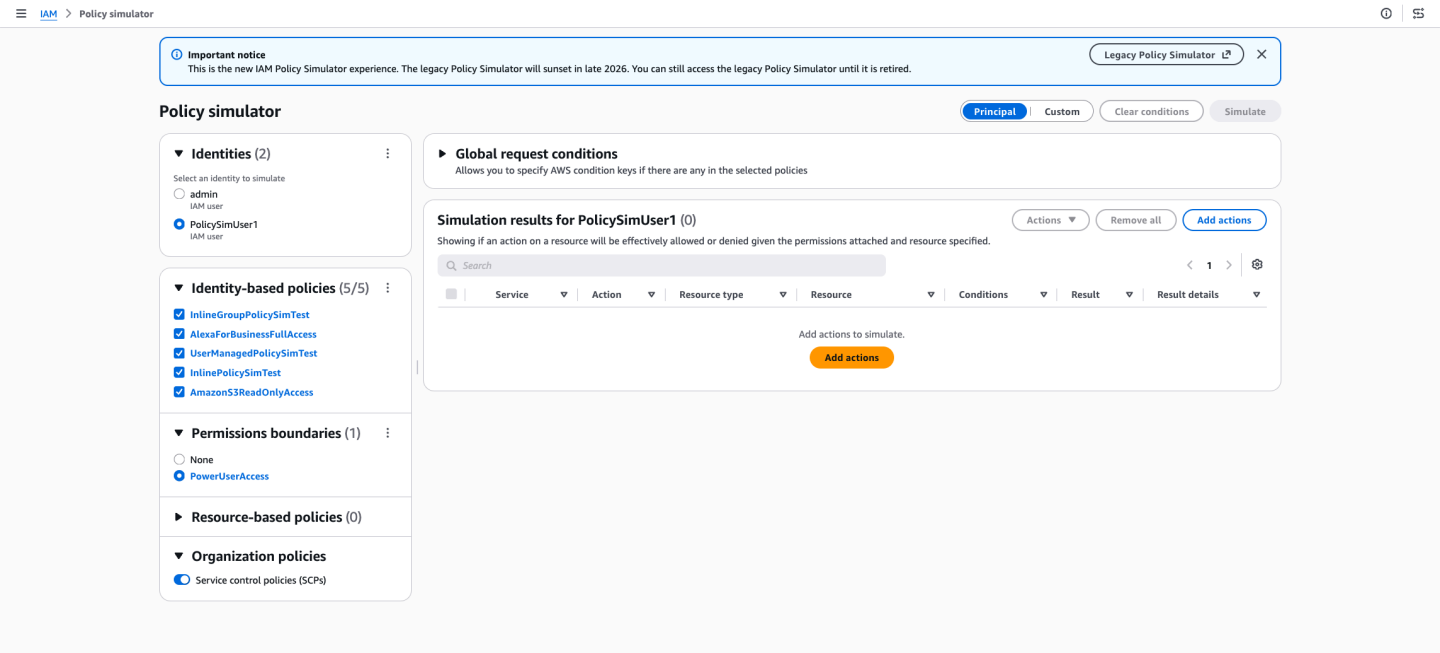
選取身分

首先，請先選取身分。您可以在該帳戶中選取 IAM 使用者、角色或群組。側邊面板會在您選取後填入連接到這些身分的政策。

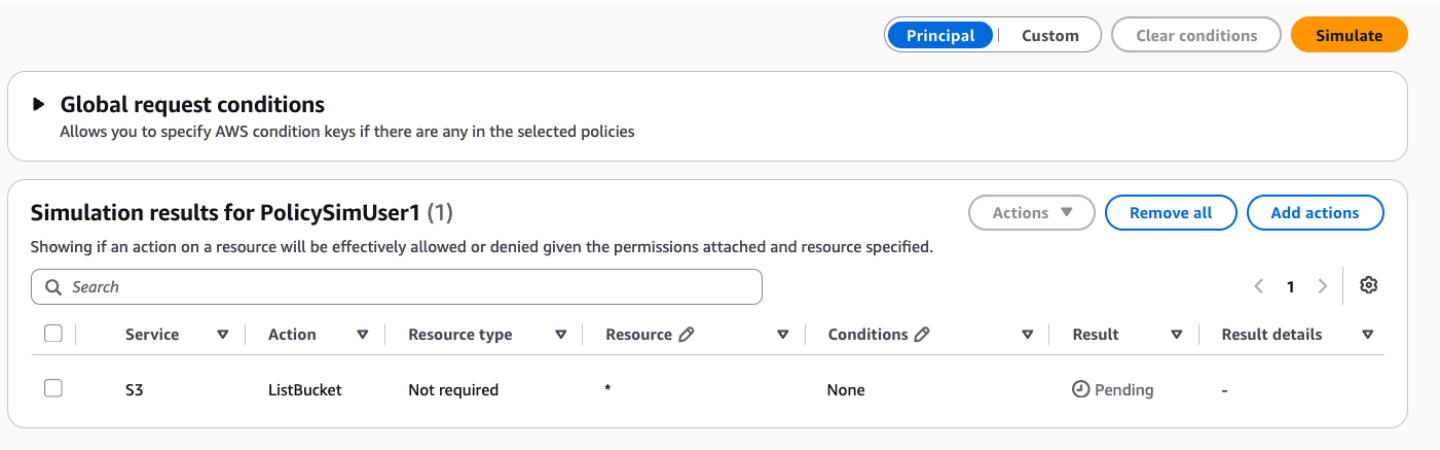
包含或排除政策

側邊面板列出模擬範圍內的政策，包括內嵌 AWS 、受管和客戶受管政策，以及任何許可界限。您可以排除政策，以查看結果在沒有它的情況下如何變更，然後再次包含它。排除授予存取的政策會將結果變更為隱含拒絕。

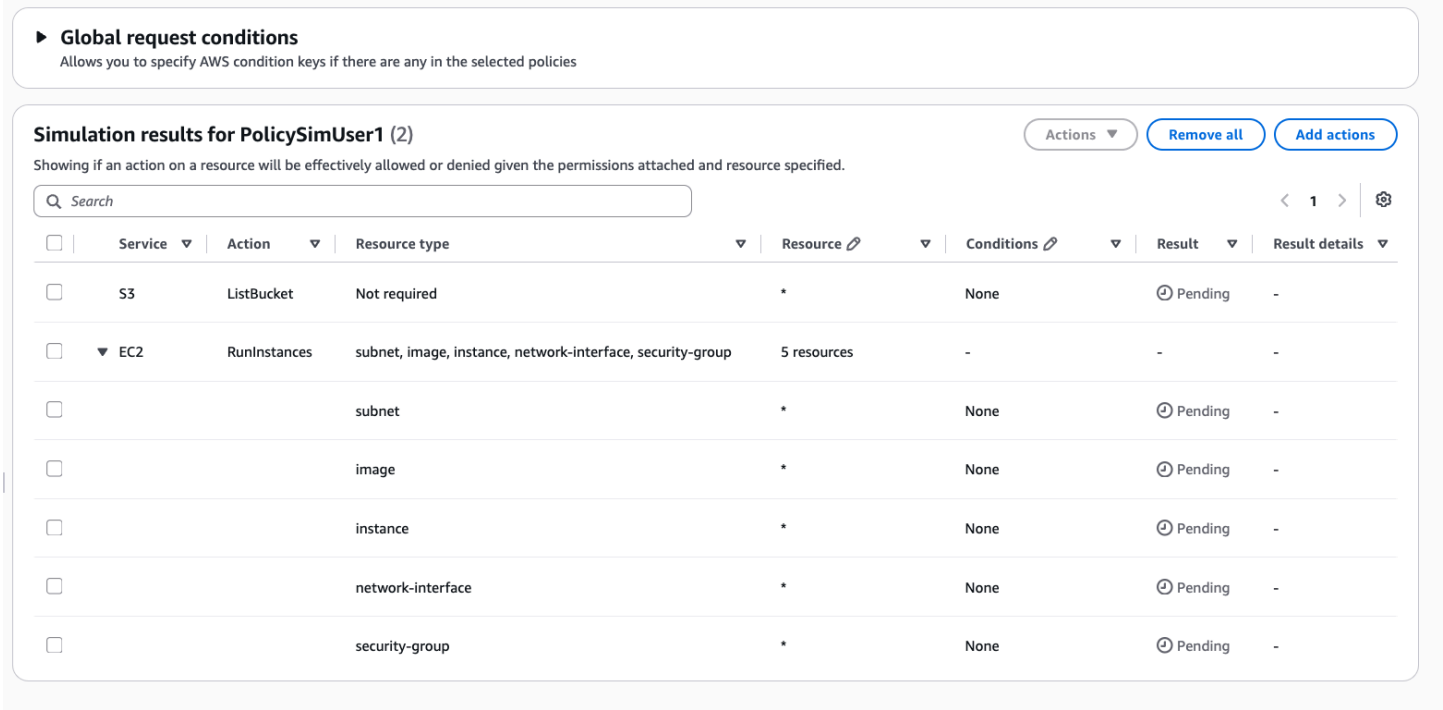
新增動作和資源



在左側面板中選擇身分和政策後，新增您要測試的動作。對於支援資源選擇的動作，您可以針對所有資源進行測試，或輸入一或多個特定資源 ARNs。當多個動作使用相同的資源類型時，您可以一次將一個資源套用至所有資源，或將其範圍限定為單一動作。



有些動作需要一組相關的資源類型，模擬器才能對其進行評估。例如，當您模擬時`ec2:RunInstances`，主控台會提示您根據情況一起指定映像、執行個體和安全群組，以及子網路或磁碟區。



提供條件索引鍵值

如果您測試的政策包含條件索引鍵，模擬器會列出相關索引鍵，並可讓您在執行模擬之前設定值。顯示的金鑰會反映目前選取的政策，因此選取或清除政策會新增或移除其金鑰。如果值不符合條件，則會拒絕動作。

模擬器會在兩個位置提供條件索引鍵的值。全域請求條件適用於整個模擬請求。這些是開頭為 的金鑰`aws:`，例如 `aws:MultiFactorAuthPresent`或 `aws:PrincipalServiceNamesList`，您可以在全域條件區域中設定其值一次。選取或清除政策會新增或移除其全域金鑰，而且在您調整哪些政策在範圍內時，會保留您輸入的值。服務特定條件適用於單一動作。每個動作列都會顯示其參考的條件數目，而您會開啟該列來設定該動作的服務特定金鑰，例如 `cloudwatch:namespace` 的 `cloudwatch:PutMetricData`或 `ram:PermissionResourceType`的 `ram:ListPermissionAssociations`。動作列可以同時包含全域金鑰和服務特定的金鑰，如果任何值不符合條件，則會拒絕該動作。

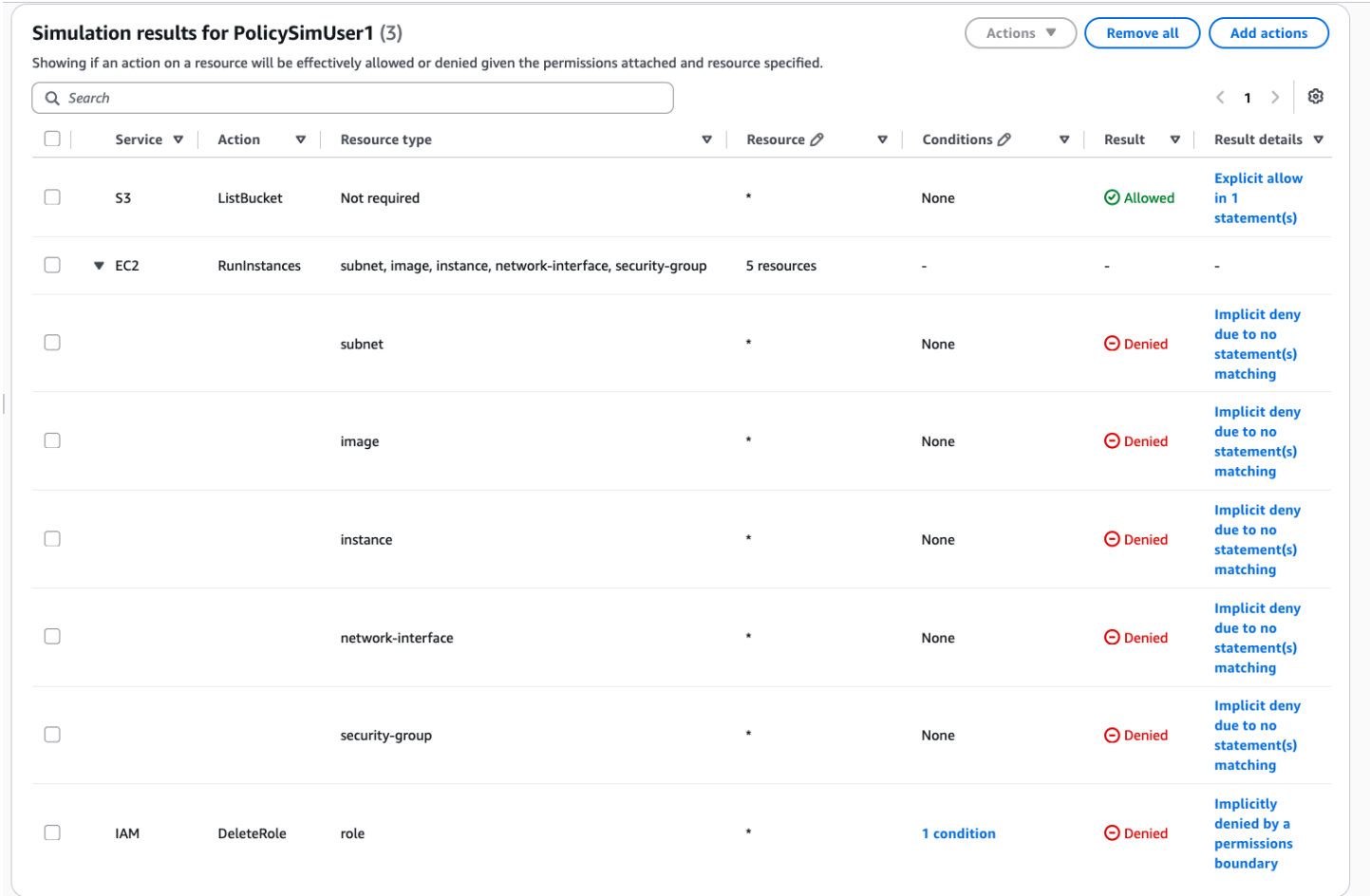
IAM 政策模擬器會為您填入一些主體和組織內容金鑰，並提供其餘的值。如需自動填入的金鑰清單，以及如何評估條件金鑰的詳細資訊，請參閱 IAM 政策模擬器中條件索引鍵評估的運作方式。

若要了解如何使用條件金鑰，請參閱 AWS 全域條件內容索引鍵。

測試許可界限

當身分具有許可界限時，它會出現在側邊面板中，並預設為包含。如果您想要了解界限是否為拒絕動作的原因，您也可以排除界限並再次執行模擬。

例如，假設您`iam:DeleteRole`針對其界限不允許的身分進行模擬。結果是拒絕歸因於許可界限。如果您接著新增允許 動作並排除邊界的政策，則結果會變更為允許。這可確認邊界是唯一封鎖動作的控制項。



包含服務控制政策

如果您的帳戶是組織的成員，您可以在側邊面板的模擬中包含服務控制政策 (SCPs)。模擬器會在決定結果時，評估 SCPs 中的自動填入條件索引鍵和資源範圍。當 SCP 明確拒絕動作時，結果是明確拒絕歸因於 AWS Organizations，而另一個政策中的允許無法覆寫它。關閉控制項並再次執行模擬，以查看沒有組織控制項的結果。如需自動填入條件索引鍵的清單，請參閱 IAM 政策模擬器中條件索引鍵評估的運作方式。

> **Note**
>
> 基於安全考量，模擬器不會顯示 SCP 評估的詳細資訊。僅由 SCPs 參考的條件索引鍵不會列出供您設定值，而且針對結果傳回的相符陳述式不包含來自 SCPs陳述式。金鑰和相符陳述式來自連接到身分的身分型政策，以及您提供的任何政策。

測試以資源為基礎的政策

若要在主控台模擬中包含資源型政策，請輸入特定資源 ARN，然後選擇 選項以包含資源的政策。資源型政策會顯示在側邊面板中，並在結果中進行評估。在 主控台中，這適用於 Amazon S3 儲存貯體、Amazon SQS 佇列、Amazon SNS 主題和 Amazon Glacier 保存庫。

> **Note**
>
> IAM 角色不支援資源型政策的模擬。

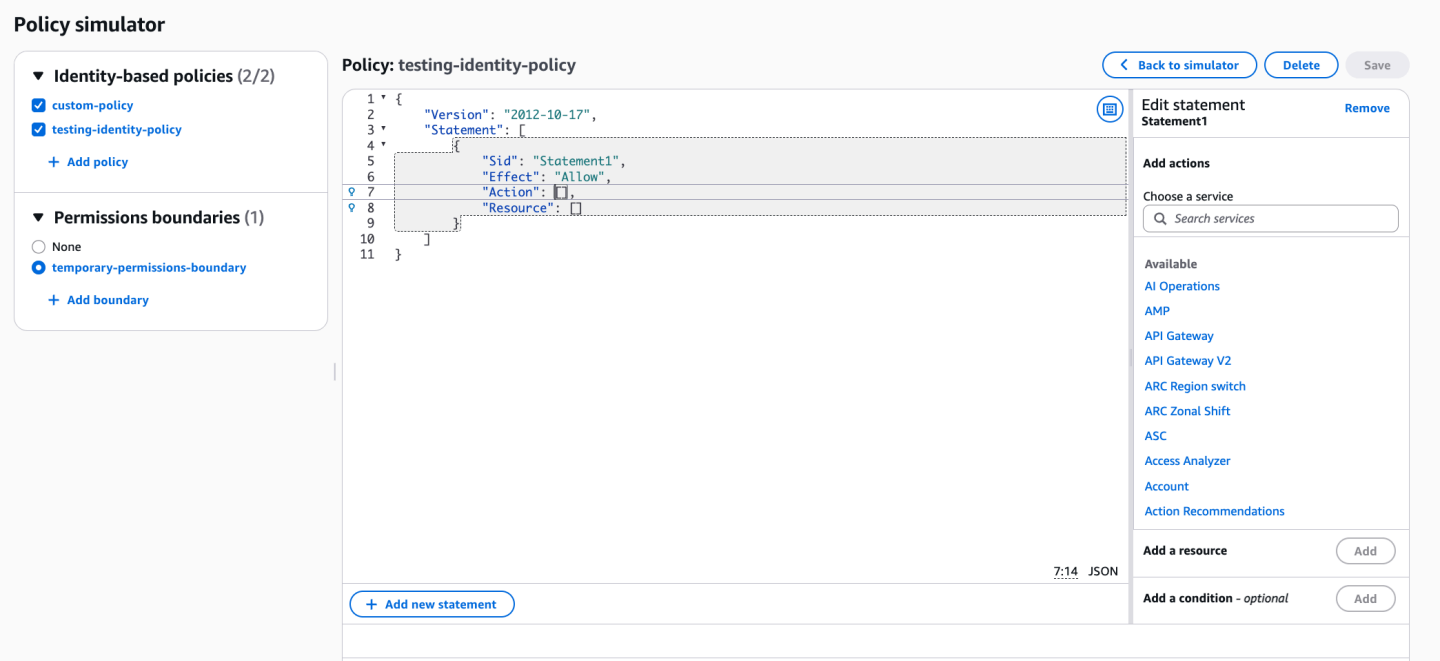
讀取結果

結果表中的每一列都會顯示是否允許、隱含拒絕或明確拒絕動作，以及結果的簡短說明。使用 說明來識別哪些政策或控制項決定結果，例如明確允許、缺少相符陳述式、許可界限或組織控制項。

自訂模式

您可以使用 IAM 政策編輯器，在自訂模式下寫入或貼上身分型政策和許可界限。這些政策不會連接到身分，而且可以在套用之前進行評估。您可以包含多個以身分為基礎的政策，但只能包含一個用於模擬的許可界限。

> **Note**
>
> 在自訂模式中，主控台不支援傳入自訂 SCPs 或模擬以資源為基礎的政策。若要模擬自訂 SCPs或資源型政策，請使用 AWS CLI 或 AWS API。



模擬政策 (AWS CLI 和 AWS API)

使用 AWS CLI 或 AWS API 模擬政策通常需要兩個步驟：

1. 評估政策並傳回其參考的內容索引鍵清單，以便您知道要提供的內容索引鍵值。此為選擇性步驟。
2. 模擬政策，提供要在模擬期間使用的動作、資源和內容索引鍵值。

每個動作都會與每個資源配對，而模擬會決定政策是否允許或拒絕該資源的動作。您可以為政策參考的任何內容索引鍵提供值。如果您未提供內容索引鍵的值，模擬仍會執行並傳回隱含拒絕，而缺少的索引鍵會在 中傳回`missingContextValues`，讓您知道要新增哪些值。如果 SCP 僅參考缺少的內容金鑰，基於安全考量，不會傳回該金鑰，但您仍然會收到決策。

API 動作分為兩個群組：

- 僅模擬以字串形式直接傳遞至 API 的政策的動作：GetContextKeysForCustomPolicy 和 SimulateCustomPolicy。
- 模擬連接至指定 IAM 使用者、使用者群組、角色或資源的政策的動作：GetContextKeysForPrincipalPolicy 和 SimulatePrincipalPolicy。由於這些可以顯示指派給其他 IAM 實體的許可，請考慮限制對這些實體的存取。

這些動作也支援下列項目：

- `OrderedOrganizationPolicyInputList` 在 SimulateCustomPolicy 上，可讓您模擬組織 SCP 階層的效果，而無需使用即時主體。
- `PolicyExclusionList` 在 SimulatePrincipalPolicy 上，可讓您測試「如果我移除此政策該怎麼辦？」 案例，而不變更哪些政策連接到委託人。
- 回應會傳回`EvaluationResult`每個動作的單一 ，並在頂層彙總決策和 中的每個資源詳細資訊`ResourceSpecificResults`。

AWS CLI 和 AWS API 參考

如需模擬政策的範例 AWS CLI 命令，請參閱下列內容：

- AWS CLI：get-context-keys-for-custom-policy、get-context-keys-for-principal-policy、simulate-custom-policy、simulate-principal-policy
- AWS API：GetContextKeysForCustomPolicy、GetContextKeysForPrincipalPolicy、SimulateCustomPolicy、Sir

新增和移除 IAM 身分許可

您可以使用政策來定義身分 (使用者、使用者群組或角色) 的許可。您可以使用 AWS 管理主控台、AWS Command Line Interface (AWS CLI) 或 AWS API 來連接和分離身分的 IAM 政策，以新增和移除許可。您也可以使用相同的方法，利用政策來設定僅適用於實體 (使用者或角色) 的許可界限。許可界限是一項進階 AWS 功能，可控制實體可擁有的最大許可。

主題

- 術語
- 檢視身分活動
- 新增 IAM 身分許可 (主控台)
- 移除 IAM 身分許可 (主控台)
- 新增 IAM 政策 (AWS CLI)
- 移除 IAM 政策 (AWS CLI)
- 新增 IAM 政策 (AWS API)
- 移除 IAM 政策 (AWS API)

術語

當您將許可政策與身分 (IAM 使用者、IAM 群組和 IAM 角色) 相關聯時，術語和程序會有所不同，具體取決於您使用的是受管政策或內嵌政策：

- 連接 – 與受管政策一起使用。您將受管政策連接到身分 (使用者、使用者群組或角色)。將在政策中套用許可的政策連接到身分。
- Detach (分開) – 與受管政策一起使用。您從 IAM 身分 (使用者、使用者群組或角色) 分開受管政策。分開政策會從身分中移除其許可。
- 嵌入 – 與內嵌政策一起使用。在身分中嵌入內嵌政策 (使用者、使用者群組或角色)。將在政策中套用許可的政策嵌入到身分。因為內嵌政策儲存在身分中，所以它是嵌入的而不是連接的，儘管結果是相似的。

> **Note**
>
> 您只能將服務相關角色的內嵌政策嵌入依賴該角色的服務。請參閱服務的 AWS 文件，確認是否支援此功能。

- 刪除 – 與內嵌政策一起使用。您從 IAM 身分 (使用者、使用者群組或角色) 刪除內嵌政策。刪除政策會從身分中移除其許可。

> **Note**
>
> 您只能在依賴於角色的服務中刪除服務連結角色的內嵌政策。請參閱服務的 AWS 文件，確認是否支援此功能。

您可以使用 主控台 AWS CLI或 AWS API 來執行任何這些動作。

其他資訊

- 如需有關受管與內嵌政策之間的差異的詳細資訊，請參閱 受管政策與內嵌政策。
- 如需有關許可界限的詳細資訊，請參閱 IAM 實體的許可界限。
- 如需有關 IAM 政策的一般資訊，請參閱 中的政策和許可 AWS Identity and Access Management。
- 如需有關驗證 IAM 政策的資訊，請參閱 IAM 政策驗證。
- AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

檢視身分活動

變更身分 (使用者、使用者群組或角色) 的許可之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

新增 IAM 身分許可 (主控台)

您可以使用 AWS 管理主控台 將許可新增至身分 （使用者、使用者群組或角色）。若要執行此操作，連接可控制許可的受管政策，或指定可做為許可界限的政策。您也可以嵌入內嵌政策。

將受管政策當做身分的許可政策來使用 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策。
3. 在政策清單中，選取要連接的政策名稱旁的選項按鈕。您可以使用搜尋方塊來篩選政策清單。
4. 選擇 Actions (動作)，然後選擇 Attach (連接)。
5. 選取一或多個身分以將政策連接到。您可使用搜尋方塊來篩選主體實體清單。在選取身分後，選擇 Attach policy (連接政策)。

使用受管政策設定許可界限 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇要設定的政策名稱。您可以使用搜尋方塊來篩選政策清單。
4. 在政策詳細資訊頁面上，選擇連接的實體索引標籤，然後在必要時，開啟作為許可界限連接區段，然後選擇將該政策設定為許可界限。
5. 選擇一或多個要套用許可界限政策的使用者或角色。您可使用搜尋方塊來篩選主體實體清單。選取主體之後，選擇設定許可界限。

為使用者或角色嵌入內嵌政策 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在服務導覽窗格中，選擇 Users (使用者) 或者 Roles (角色)。

3. 在清單中，選擇要內嵌政策的使用者或角色的名稱。
4. 選擇許可索引標籤標籤。
5. 選擇新增許可，然後選擇建立內嵌政策。

> **Note**
>
> 您不能在 IAM 中的服務連結的角色中嵌入內嵌政策。由於連結服務定義您是否可以修改角色的許可，因此您可以從服務主控台、API 或 AWS CLI新增額外的政策。若要查看服務的服務連結角色文件，請參閱 AWS 使用 IAM 的 服務 並在服務的 服務連結角色欄位中選擇 Yes (是)。

6. 選擇下列其中一種方法來檢視建立政策所需的步驟：
   - 匯入現有的受管政策 – 您可以將受管政策匯入帳戶內，然後編輯該政策以依據您的特定要求自訂內容。受管政策可以是您先前建立的 AWS 受管政策或客戶受管政策。
   - 使用視覺化編輯器來建立政策 – 您可以在視覺編輯工具中從零開始建構一個新的政策。若您使用視覺化編輯器，您便無需了解 JSON 語法。
   - 正在使用 JSON 編輯器建立政策 – 在 JSON 編輯器選項中，您可以使用 JSON 語法來建立政策。您可以輸入新的 JSON 政策文件或者貼上範例政策。
7. 在您建立內嵌政策後，它會自動嵌入您的使用者或角色中。

為使用者群組嵌入內嵌政策 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 User groups (使用者群組)。
3. 在清單中，選擇要內嵌政策的使用者群組名稱。
4. 選擇 Permissions (許可) 標籤，選擇 Add permissions (新增許可)，然後選擇 Attach policy (連接政策)。
5. 執行以下任意一項：
   - 選擇視覺化選項，可建立政策。如需詳細資訊，請參閱使用視覺化編輯器來建立政策。
   - 選擇 JSON 選項，可建立政策。如需詳細資訊，請參閱正在使用 JSON 編輯器建立政策。
6. 當您滿意時，選擇 建立政策。

變更一或多個實體的許可界限 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇要設定的政策名稱。您可以使用搜尋方塊來篩選政策清單。
4. 在政策詳細資訊頁面上，選擇連接的實體索引標籤，然後在必要時，開啟作為許可界限連接區段。選取您想要變更其界限的使用者或角色旁的核取方塊，然後選擇變更。
5. 選取用於許可界限新政策。您可以使用搜尋方塊來篩選政策清單。在選取政策後，選擇設定許可界限。

移除 IAM 身分許可 (主控台)

您可以使用 從身分 （使用者、使用者群組或角色） AWS 管理主控台 移除許可。若要執行此操作，請分開可控制許可的受管政策，或移除可做為許可界限的政策。您也可以刪除內嵌政策。

中斷連結當做許可政策來使用的受管政策 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策。
3. 在政策清單中，選取要分開的政策名稱旁的選項按鈕。您可以使用搜尋方塊來篩選政策清單。
4. 選擇 Actions (動作)，然後選擇 Detach (分開)。
5. 選取要從中分開政策的身分。您可以使用搜尋方塊來篩選身分清單。在選取身分後，選擇 Detach policy (分開政策)。

移除許可界限 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇要設定的政策名稱。您可以使用搜尋方塊來篩選政策清單。
4. 在政策摘要頁面上，選擇連接的實體索引標籤，然後在必要時，開啟作為許可界限連接區段，接著選擇要從中移除許可界限的實體。然後選擇移除界限。
5. 確認您想要移除界限，然後選擇移除界限。

刪除內嵌政策 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在服務導覽窗格中，選擇 User groups (使用者群組)、Users (使用者) 或者 Roles (角色)。
3. 在清單中，選擇包含要刪除的政策的使用者群組、使用者、或角色的名稱。
4. 選擇 Permissions (許可) 標籤。
5. 選取政策旁的核取方塊，然後選擇移除。
6. 在確認方塊中，選擇移除。

新增 IAM 政策 (AWS CLI)

您可以使用 AWS CLI 將許可新增至身分 （使用者、使用者群組或角色）。若要執行此操作，連接可控制許可的受管政策，或指定可做為許可界限的政策。您也可以嵌入內嵌政策。

將受管政策當做實體的許可政策來使用 (AWS CLI)

1. (選用) 若要檢視受管政策的相關資訊，請執行下列命令：
   - 列出受管政策：aws iam list-policies
   - 取得關於受管政策的詳細資訊：get-policy
2. 若要將受管政策連接到身分 (使用者、使用者群組或角色)，請使用下列其中一項命令：
   - aws iam attach-user-policy
   - aws iam attach-group-policy
   - aws iam attach-role-policy

使用受管政策設定許可界限 (AWS CLI)

1. (選用) 若要檢視受管政策的相關資訊，請執行下列命令：
   - 列出受管政策：aws iam list-policies
   - 取得關於受管政策的詳細資訊：aws iam get-policy
2. 若要使用受管政策來為實體 (使用者或角色) 設定許可界限，請使用下列其中一個命令：
   - aws iam put-user-permissions-boundary

- aws iam put-role-permissions-boundary

嵌入內嵌政策 (AWS CLI)

若要將內嵌政策嵌入到身分 (非服務連結角色的使用者、使用者群組或角色)，請使用下列命令：

- aws iam put-user-policy
- aws iam put-group-policy
- aws iam put-role-policy

移除 IAM 政策 (AWS CLI)

您可以使用 AWS CLI 分離控制許可的受管政策，或移除做為許可界限的政策。您也可以刪除內嵌政策。

中斷連結當做許可政策來使用的受管政策 (AWS CLI)

1. (選用) 若要檢視關於政策的資訊，請執行下列命令：
   - 列出受管政策：aws iam list-policies
   - 取得關於受管政策的詳細資訊：aws iam get-policy
2. (選用) 如果要了解的政策和身分之間的關係，請執行下列命令：
   - 若要列出受管政策所連接的身分 (IAM 使用者、IAM 群組和 IAM 角色)：
     - aws iam list-entities-for-policy
   - 若要列出連接到身分的受管政策 (使用者、使用者群組或角色)，請使用下列其中一項命令：
     - aws iam list-attached-user-policies
     - aws iam list-attached-group-policies
     - aws iam list-attached-role-policies
3. 若要從身分中分開受管政策 (使用者、使用者群組或角色)，請使用下列其中一項命令：
   - aws iam detach-user-policy
   - aws iam detach-group-policy
   - aws iam detach-role-policy

移除許可界限 (AWS CLI)

1. (選用) 若要檢視目前使用哪個受管政策來為使用者或角色設定許可界限，請執行下列命令：
   - aws iam get-user
   - aws iam get-role
2. (選用) 若要檢視目前在哪些使用者或角色使用受管政策的許可界限，請執行下列命令：
   - aws iam list-entities-for-policy
3. (選用) 若要檢視受管政策的相關資訊，請執行下列命令：
   - 列出受管政策：aws iam list-policies
   - 取得關於受管政策的詳細資訊：aws iam get-policy
4. 若要從使用者或角色移除許可界限，請使用下列其中一個命令：
   - aws iam delete-user-permissions-boundary
   - aws iam delete-role-permissions-boundary

若要刪除內嵌政策 (AWS CLI)

1. (選用) 若要列出連接到身分 (使用者、使用者群組或角色) 的所有內嵌政策，請使用下列其中一項命令：
   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies
2. (選用) 若要擷取嵌入到身分 (使用者、使用者群組或角色) 中的內嵌政策文件，請使用下列其中一項命令：
   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy
3. 若要從身分中刪除內嵌政策 (非服務連結角色的使用者、使用者群組或角色)，請使用下列命令：
   - aws iam delete-user-policy
   - aws iam delete-group-policy

- aws iam delete-role-policy

新增 IAM 政策 (AWS API)

您可以使用 AWS API 連接控制許可的受管政策，或指定做為許可界限的政策。您也可以嵌入內嵌政策。

使用受管政策做為實體的許可政策 (AWS API)

1. (選用) 若要檢視關於政策的資訊，請呼叫下列操作：
   - 列出受管政策：ListPolicies
   - 取得關於受管政策的詳細資訊：GetPolicy
2. 若要將受管政策連接到身分 (使用者、使用者群組或角色)，請呼叫下列其中一項操作：
   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

使用受管政策來設定許可界限 (AWS API)

1. (選用) 若要檢視受管政策的相關資訊，請呼叫下列操作：
   - 列出受管政策：ListPolicies
   - 取得關於受管政策的詳細資訊：GetPolicy
2. 若要使用受管政策來為實體 (使用者或角色) 設定許可界限，請呼叫下列其中一個操作：
   - PutUserPermissionsBoundary
   - PutRolePermissionsBoundary

內嵌內嵌政策 (AWS API)

若要將內嵌政策嵌入在身分 (非服務連結角色的使用者、使用者群組或角色)，請呼叫下列操作：

- PutUserPolicy
- PutGroupPolicy
- PutRolePolicy

移除 IAM 政策 (AWS API)

您可以使用 AWS API 分離控制許可的受管政策，或移除做為許可界限的政策。您也可以刪除內嵌政策。

分離用作許可政策 (AWS API) 的受管政策

1. (選用) 若要檢視關於政策的資訊，請呼叫下列操作：
   - 列出受管政策：ListPolicies
   - 取得關於受管政策的詳細資訊：GetPolicy
2. (選用) 如果要了解的政策和身分之間的關係，請呼叫下列操作：
   - 若要列出受管政策所連接的身分 (IAM 使用者、IAM 群組和 IAM 角色)：
     - ListEntitiesForPolicy
   - 若要列出連接到身分的受管政策 (使用者、使用者群組或角色)，請呼叫下列其中一項操作：
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. 若要從身分中分開受管政策 (使用者、使用者群組或角色)，請呼叫下列其中一項操作：
   - DetachUserPolicy
   - DetachGroupPolicy
   - DetachRolePolicy

移除許可界限 (AWS API)

1. (選用) 若要檢視目前使用哪個受管政策來為使用者或角色設定許可界限，請呼叫下列操作：
   - GetUser
   - GetRole
2. (選用) 若要檢視目前在哪些使用者或角色使用受管政策的許可界限，請呼叫下列操作：
   - ListEntitiesForPolicy
3. (選用) 若要檢視受管政策的相關資訊，請呼叫下列操作：
   - 列出受管政策：ListPolicies

- 取得關於受管政策的詳細資訊：GetPolicy

4. 若要從使用者或角色移除許可界限，請呼叫下列其中一個操作：

   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

刪除內嵌政策 (AWS API)

1. (選用) 若要列出連接到身分 (使用者、使用者群組或角色) 的所有內嵌政策，請呼叫下列其中一項操作：

   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies

2. (選用) 若要擷取嵌入到身分 (使用者、使用者群組或角色) 中的內嵌政策文件，請呼叫下列其中一項操作：

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

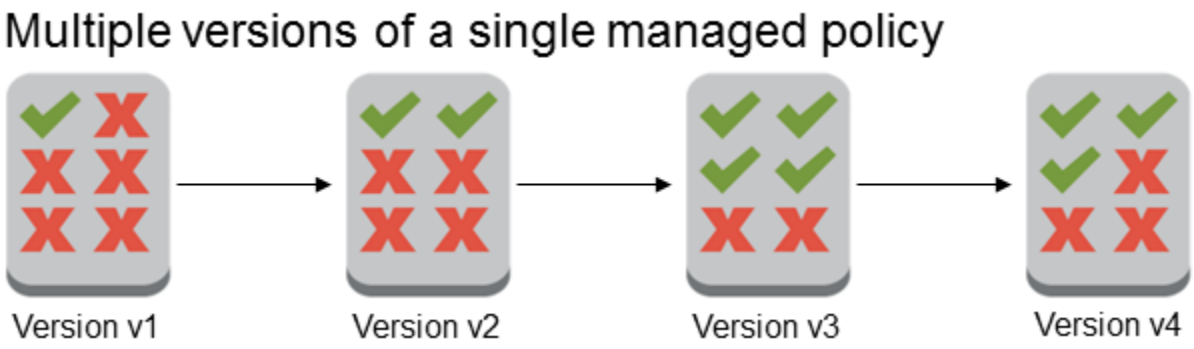
3. 若要從身分中刪除內嵌政策 (非服務連結角色的使用者、使用者群組或角色)，請呼叫下列操作：

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

版本控制 IAM 政策

當您變更 IAM 客戶受管政策，以及 AWS 變更 AWS 受管政策時，變更的政策不會覆寫現有的政策。IAM 反而會建立新版本的受管政策。IAM 最多可以儲存五個版本的客戶受管政策。IAM 不支援內嵌政策版本控制。

下圖說明客戶受管政策的版本控制。在此範例中，版本 1-4 會進行儲存。您最多可以儲存五個受管政策版本到 IAM。當您編輯建立第六個儲存版本的政策時，您可以選擇任一不再儲存的舊版本。您可以隨時恢復到其他四個儲存版本中的任何一個。



政策版本與 Version 政策元素不同。Version 政策元素是在政策內使用，並定義政策語言的版本。若要進一步了解 Version 政策元素，請參閱 IAM JSON 政策元素：Version。

您可以使用版本來追蹤受管政策的變更。例如，您可以變更受管政策，然後發現該變更有各種意外影響。在這種情況下，您可以復原至舊版的受管政策，做法是將舊版本設定為預設版本。

以下各主題說明如何使用受管政策的版本控制。

主題

- 版本限制
- 使用版本復原變更
- 用於設定預設政策版本的許可
- 設定客戶受管政策的預設版本

版本限制

受管政策至多可有 5 個版本。如果您需要從 AWS Command Line Interface或 AWS API 變更超過五個版本的受管政策，您必須先刪除一或多個現有版本。如果您使用 AWS 管理主控台，則不需要在編輯政策之前刪除版本。當您儲存第六個版本，會出現一個對話方塊，提示您刪除一或多個非預設的政策版本。您可以檢視每個版本的 JSON 政策文件，以協助您做決定。如需此對話方塊的詳細資訊，請參閱the section called “編輯 IAM 政策”。

您可以刪除任何想要的受管政策的版本，除了預設版本以外。當您刪除某個版本，剩餘版本的版本識別碼不會變更。因此，版本識別碼可能不會序列化。例如，若刪除受管政策的版本 v2 和 v4，並新增兩個新版本，剩餘的版本識別碼可能為 v1、v3、v5、v6 和 v7。

使用版本復原變更

您可以設定客戶受管政策的預設版本來復原您的變更。例如，考量以下情境：

您建立客戶受管政策，讓使用者能夠使用 AWS 管理主控台管理特定 Amazon S3 儲存貯體。建立後，您的客戶受管政策只有一個版本 (識別為 v1)，所以該版本會自動設定為預設值。該政策按預期運作。

之後，您更新政策來新增許可，以管理第二個 Amazon S3 儲存貯體。IAM 建立新版本的政策 (識別為 v2)，其中包含您的變更。您設定版本 v2 為預設值，不久後您的使用者報告他們沒有獲得使用 Amazon S3 主控台的許可。在這種情況下，您可以復原到版本 v1 的政策，就是您所知按預期運作的版本。為了這樣做，您設定版本 v1 為預設版本。您的使用者現在可以使用 Amazon S3 主控台來管理原始儲存貯體。

之後，在您判斷出政策的版本 v2 錯誤後，您再次更新政策來新增許可，以管理第二個 Amazon S3 儲存貯體。IAM 建立另一個新版本的政策，識別為 v3。您設定版本 v3 為預設值，而此版本按預期運作。此時，您刪除政策的版本 v2。

用於設定預設政策版本的許可

設定預設政策版本所需的許可，對應於任務的 AWS API 操作。您可以使用 `CreatePolicyVersion` 或 `SetDefaultPolicyVersion` API 操作來設定政策的預設版本。若要讓某人設定現有政策的預設政策版本，您可以允許存取 `iam:CreatePolicyVersion` 動作或 `iam:SetDefaultPolicyVersion` 動作。這個 `iam:CreatePolicyVersion` 動作可讓他們建立新版本的政策，並設定該版本為預設值。這個 `iam:SetDefaultPolicyVersion` 動作可讓他們將任何現有的政策版本設定為預設值。

> **Important**
>
> 若要防止使用者變更政策的預設版本，您必須同時拒絕 `iam:CreatePolicyVersion` 和 `iam:SetDefaultPolicyVersion`。

您可以使用以下政策來拒絕使用者存取變更現有客戶的受管政策：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
```

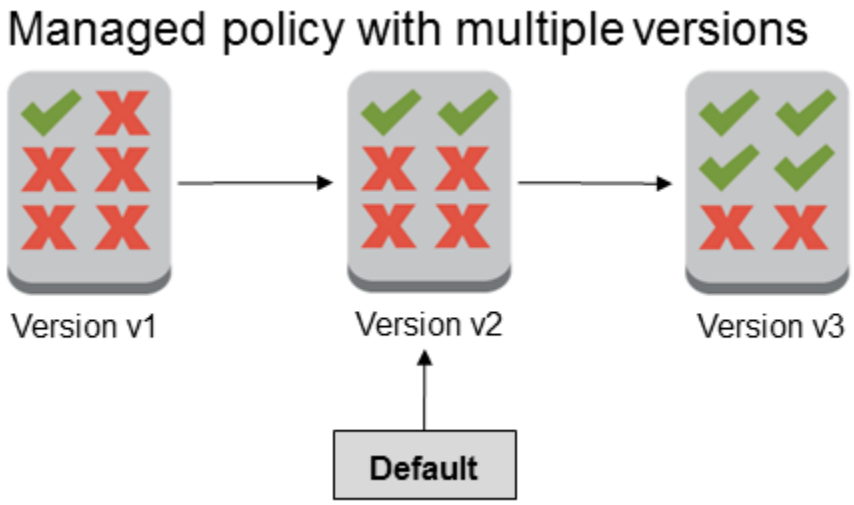
```
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
}
```

設定客戶受管政策的預設版本

其中一個受管政策版本是設定為預設版本。政策的預設版本是生效版本，也就是受管政策所連接的所有主體實體的 (IAM 使用者、IAM 使用者群組和 IAM 角色) 的生效版本。

當您建立客戶受管政策，政策從單一版本 (識別為 v1) 開始。對於只有單一版本的受管政策，該版本會自動設定為預設值。對於具有多個版本的客戶受管政策，您可以選擇將哪個版本設定為預設值。對於 AWS 受管政策，預設版本由 設定 AWS。下圖說明了此概念。





您可以設定預設的客戶受管政策版本，以套用該版本到政策連接的每個 IAM 身分 (使用者、使用者群組和角色)。您無法設定 AWS 受管政策或內嵌政策的預設版本。

設定客戶受管政策的預設版本 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇要設定預設版本的政策的政策名稱。您可以使用搜尋方塊來篩選政策清單。
4. 選擇 Policy versions (政策版本) 標籤。選取要設定為預設版本的版本旁的核取方塊，然後選擇 Set as default (設定為預設)。

若要了解如何從 AWS Command Line Interface 或 AWS API 設定客戶受管政策的預設版本，請參閱編輯 IAM 政策 (AWS CLI)。

編輯 IAM 政策

政策為一個實體，可定義其所連接的身分或資源的許可。政策會以 JSON 文件 AWS 形式存放在 中，並以身分型政策的形式連接到 IAM 中的委託人。您可以將以身分為基礎的政策連接到主體 (或身分)，例如 IAM 使用者群組、使用者或角色。身分型政策包括 AWS 受管政策、客戶受管政策和內嵌政策。您可以在 IAM.managed 政策中編輯客戶 AWS 受管政策和內嵌政策無法編輯。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

一般而言，最好使用客戶受管政策，而不是內嵌政策或 AWS 受管政策。 AWS 受管政策通常提供廣泛的管理或唯讀許可。內嵌政策無法在其他身分上重複使用，或在其存在的身分之外進行管理。為了達到最高安全性，應授予最低權限，這表示僅授予執行特定任務工作所需的許可。

當您建立或編輯 IAM 政策時， AWS 可以自動執行政策驗證，以協助您建立最低權限的有效政策。在中 AWS 管理主控台，IAM 識別 JSON 語法錯誤，而 IAM Access Analyzer 提供額外的政策檢查與建議，以協助您進一步精簡政策。若要進一步了解政策驗證的資訊，請參閱 IAM 政策驗證。若要進一步了解 IAM Access Analyzer 政策檢查和可動作的建議，請參閱 IAM Access Analyzer 政策驗證。

您可以使用 AWS CLI AWS 管理主控台或 AWS API 在 IAM 中編輯客戶受管政策和內嵌政策。如需使用 CloudFormation 範本新增或更新政策的詳細資訊，請參閱CloudFormation 《 使用者指南》中的AWS Identity and Access Management 資源類型參考。

主題

- 編輯 IAM 政策 (主控台)
- 編輯 IAM 政策 (AWS CLI)
- 編輯 IAM 政策 (AWS API)

編輯 IAM 政策 (主控台)

政策為一個實體，可定義其所連接的身分或資源的許可。您可以使用 AWS 管理主控台 編輯客戶受管政策和 IAM. AWS managed 政策中的內嵌政策無法編輯。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

如需有關政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management 和 IAM JSON 政策元素參考。

先決條件

變更政策的許可之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

編輯客戶受管政策 (主控台)

您可以透過 AWS 管理主控台來編輯客戶管理政策，以變更政策中定義的許可。客戶受管政策至多可有 5 個版本。這很重要，因為如果您變更受管政策超過 5 個版本，則 AWS 管理主控台 會提示您決定要刪除的版本。您還可以在編輯之前變更預設版本或或刪除政策版本，以避免出現提示。若要進一步了解版本，請參閱 版本控制 IAM 政策。

Console

編輯客戶管理政策

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇要編輯的政策的政策名稱。您可以使用搜尋方塊來篩選政策清單。
4. 選擇許可索引標籤，然後選擇編輯。
5. 執行以下任意一項：

   - 選擇視覺化選項可變更政策，且無需了解 JSON 語法。您可以變更政策中每個許可區塊的服務、動作、資源或可選條件。您也可以匯入政策以在政策底部新增其他許可。完成變更後，選擇下一步以繼續。
   - 選擇 JSON 選項，可透過在 JSON 文字方塊中輸入或貼上文字來修改政策。您也可以匯入政策以在政策底部新增其他許可。解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。

   > **Note**
   >
   > 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

6. 在檢視與儲存頁面上，檢視此政策中定義的許可，然後選擇儲存變更以儲存工作。
7. 如果受管政策有最多五個版本，選擇儲存變更可顯示一個對話方塊。若要儲存新版本，該政策的最舊非預設版本會遭到移除，並以此新版本取代之。您也可以將新版本設定為預設的政策版本。

   選擇儲存變更，可儲存新的政策版本。

設定客戶管理政策的預設版本 (主控台)

您可以從 設定客戶受管政策的預設版本 AWS 管理主控台。您可以使用此政策為整個組織中的許可建立一致的基準組態。此政策的所有新附件都會使用此標準化許可集。

Console

設定客戶受管政策的預設版本 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇要設定預設版本的政策的政策名稱。您可以使用搜尋方塊來篩選政策清單。
4. 選擇 Policy versions (政策版本) 標籤。選取要設定為預設版本的版本旁的核取方塊，然後選擇 Set as default (設定為預設)。

刪除客戶管理政策的版本 (主控台)

您可能需要刪除客戶管理政策的某個版本，以移除不再需要或存在潛在安全風險的過時或不正確的許可。透過僅維護必要的政策版本，可以協助確保受管政策版本保持在限制的五個以內，為未來的更新和改進留出空間。您可以透過 AWS 管理主控台刪除客戶管理政策的版本。

Console

刪除客戶管理政策的版本

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 選擇具有要刪除的版本的客戶受管政策的名稱。您可以使用搜尋方塊來篩選政策清單。
4. 選擇 Policy versions (政策版本) 標籤。選取要刪除的版本旁的核取方塊。然後選擇 Delete (刪除)。
5. 確認您要刪除該版本，然後選擇 Delete (刪除)。

編輯內嵌政策 (主控台)

您可能需要編輯客戶管理政策來更新或精簡授予的許可，確保這些許可符合組織不斷變化的安全需求和存取控制需求。您可以透過對政策進行編輯來調整政策的 JSON 文件，以及新增、修改或移除特定動作、資源或條件，以維護最低權限原則，並適應環境或程序中的變更。您可以從 AWS 管理主控台主控台中編輯內嵌政策。

Console

編輯使用者、使用者群組或角色的內嵌政策

1. 在服務導覽窗格中，選擇 User (使用者)、Users groups (使用者群組) 或者 Roles (角色)。
2. 選擇您要修改的政策的使用者、 使用者群組或角色的名稱。然後選擇 Permissions (許可) 標籤並展開政策。
3. 若要編輯內嵌政策，請選擇 Edit Policy (編輯政策)。
4. 執行以下任意一項：
   - 選擇視覺化選項可變更政策，且無需了解 JSON 語法。您可以變更政策中每個許可區塊的服務、動作、資源或可選條件。您也可以匯入政策以在政策底部新增其他許可。完成變更後，選擇下一步以繼續。
   - 選擇 JSON 選項，可透過在 JSON 文字方塊中輸入或貼上文字來修改政策。您也可以匯入政策以在政策底部新增其他許可。解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。若要儲存變更而不影響目前連接的實體，請清除 Save as default version (儲存為預設版本) 的核取方塊。

   > **Note**
   >
   > 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。
5. 在檢視頁面上，查看政策摘要，然後選擇儲存變更以儲存您的工作。

編輯 IAM 政策 (AWS CLI)

政策為一個實體，可定義其所連接的身分或資源的許可。您可以使用 AWS Command Line Interface (AWS CLI) 編輯客戶受管政策，且無法編輯 IAM. AWS managed 政策中的內嵌政策。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

如需有關政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management 和 IAM JSON 政策元素參考。

先決條件

變更政策的許可之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

編輯客戶受管政策 (AWS CLI)

您可以從 編輯客戶受管政策 AWS CLI。

> **Note**
>
> 受管政策至多可有 5 個版本。如果您需要變更五個版本以上的客戶受管政策，則必須先刪除一個或多個現有版本。

若要編輯客戶受管政策 (AWS CLI)

1. (選用) 若要檢視關於政策的資訊，請執行下列命令：
   - 列出受管政策：list-policies
   - 取得關於受管政策的詳細資訊：get-policy
2. (選用) 如果要了解的政策和身分之間的關係，請執行下列命令：
   - 若要列出受管政策所連接的身分 (IAM 使用者、IAM 群組和 IAM 角色)：
     - list-entities-for-policy
   - 列出連接到身分的受管政策 (使用者、使用者群組或角色)：
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. 若要編輯客戶受管政策，請執行下列命令：
   - create-policy-version
4. (選用) 若要驗證客戶受管政策，請執行下列 IAM Access Analyzer 命令：
   - validate-policy

設定客戶管理政策的預設版本 (AWS CLI)

您可以從 設定客戶受管政策的預設版本 AWS CLI。

若要設定客戶受管政策的預設版本 (AWS CLI)

1. (選用) 若要列出受管政策，請執行下列命令：
   - : list-policies
2. 若要設定客戶受管政策的預設版本，請執行下列命令：
   - set-default-policy-version

刪除客戶管理政策的版本 (AWS CLI)

您可以透過 AWS CLI刪除客戶管理政策的版本。

若要刪除客戶受管政策的版本 (AWS CLI)

1. (選用) 若要列出受管政策，請執行下列命令：
   - : list-policies
2. 若要刪除客戶受管政策，請執行下列命令：
   - delete-policy-version

編輯內嵌政策 (AWS CLI)

您可以從 AWS CLI主控台中編輯內嵌政策。

若要編輯內嵌政策 (AWS CLI)

1. (選用) 若要檢視關於政策的資訊，請執行下列命令：
   - 若要列出與身分 (使用者、使用者群組或角色) 相關聯的內嵌政策：
     - list-user-policies
     - list-role-policies
     - list-group-policies
   - 若要取得關於內嵌政策的詳細資訊：

- get-user-policy
- get-role-policy
- get-group-policy

2. 若要編輯內嵌政策，請執行下列命令：

- put-user-policy
- put-role-policy
- put-group-policy

3. (選用) 若要驗證內嵌政策，請執行下列 IAM Access Analyzer 命令：

- validate-policy

編輯 IAM 政策 (AWS API)

政策為一個實體，可定義其所連接的身分或資源的許可。您可以使用 AWS API 編輯客戶受管政策，且無法編輯 IAM. AWS managed 政策中的內嵌政策。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

如需有關政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management 和 IAM JSON 政策元素參考。

先決條件

變更政策的許可之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

編輯客戶受管政策 (AWS API)

您可以使用 AWS API 編輯客戶受管政策。

> **Note**
>
> 受管政策至多可有 5 個版本。如果您需要變更五個版本以上的客戶受管政策，則必須先刪除一個或多個現有版本。

編輯客戶受管政策 (AWS API)

1. (選用) 若要檢視關於政策的資訊，請呼叫下列操作：
   - 列出受管政策：ListPolicies
   - 取得關於受管政策的詳細資訊：GetPolicy
2. (選用) 如果要了解的政策和身分之間的關係，請呼叫下列操作：
   - 若要列出受管政策所連接的身分 (IAM 使用者、IAM 群組和 IAM 角色)：
     - ListEntitiesForPolicy
   - 列出連接到身分的受管政策 (使用者、使用者群組或角色)：
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. 若要編輯客戶受管政策，請呼叫下列操作：
   - CreatePolicyVersion
4. (選用) 若要驗證客戶受管政策，請呼叫下列 IAM Access Analyzer 操作：
   - ValidatePolicy

設定客戶受管政策 (AWS API) 的預設版本

您可以從 AWS API 設定客戶受管政策的預設版本。

設定客戶受管政策 (AWS API) 的預設版本

1. (選用) 若要列出受管政策，請呼叫下列操作：
   - ListPolicies
2. 若要設定客戶受管政策的預設版本，請呼叫下列操作：
   - SetDefaultPolicyVersion

刪除客戶受管政策 (AWS API) 的版本

您可以從 AWS API 刪除客戶受管政策的版本。

刪除客戶受管政策 (AWS API) 的版本

1. (選用) 若要列出受管政策，請呼叫下列操作：
   - ListPolicies
2. 若要刪除客戶受管政策，請呼叫下列操作：
   - DeletePolicyVersion

編輯內嵌政策 (AWS API)

您可以從 AWS API 編輯內嵌政策。

編輯內嵌政策 (AWS API)

1. （選用）若要檢視內嵌政策的相關資訊，請呼叫下列操作：
   - 若要列出與身分 (使用者、使用者群組或角色) 相關聯的內嵌政策：
     - ListUserPolicies
     - ListRolePolicies
     - ListGroupPolicies
   - 若要取得關於內嵌政策的詳細資訊：
     - GetUserPolicy
     - GetRolePolicy
     - GetGroupPolicy
2. 若要編輯內嵌政策，請呼叫下列操作：
   - PutUserPolicy
   - PutRolePolicy
   - PutGroupPolicy
3. (選用) 若要驗證內嵌政策，請執行下列 IAM Access Analyzer 操作：
   - ValidatePolicy

刪除 IAM 政策

您可以使用 AWS 管理主控台、 AWS Command Line Interface (AWS CLI) 或 API 刪除 IAM AWS 政策。

> **Note**
>
> 刪除 IAM 政策的動作具有永久性。刪除政策後，將無法復原。

如需有關 IAM 政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management和 IAM JSON 政策元素參考。

如需有關受管與內嵌政策之間的差異的詳細資訊，請參閱 受管政策與內嵌政策。

AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

主題

- 刪除 IAM 政策 (主控台)
- 刪除 IAM 政策 (AWS CLI)
- 刪除 IAM 政策 (AWS API)

刪除 IAM 政策 (主控台)

您可以使用 AWS 管理主控台 刪除 IAM 中的客戶受管政策和內嵌政策。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

> **Note**
>
> 刪除 IAM 政策的動作具有永久性。刪除政策後，將無法復原。

如需有關 IAM 政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management和 IAM JSON 政策元素參考。

如需有關受管與內嵌政策之間的差異的詳細資訊，請參閱 受管政策與內嵌政策。

先決條件

刪除政策之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

刪除 IAM 政策 (主控台)

當客戶管理政策已過時或不再符合組織的安全需求和存取控制需求時，您可能需要將其刪除。刪除不必要的政策可以降低與過時或未使用政策相關聯的潛在安全風險。您可以從 AWS 帳戶移除客戶管理政策。您無法刪除 AWS 受管政策。

Console

刪除客戶管理政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策。
3. 選擇您要刪除的客戶管理政策旁的選項按鈕。您可以使用搜尋方塊來篩選政策清單。
4. 選擇 動作，然後選擇 刪除。
5. 遵循說明以確認您要刪除政策，然後選擇刪除。

刪除內嵌政策 (主控台)

當與某個內嵌政策直接連接的 IAM 使用者、群組或角色不再需要其授予的特定許可時，您可能需要刪除該內嵌政策。刪除不必要的內嵌政策有助於降低意外存取的風險，尤其是出於內嵌政策無法像受管政策那樣重複使用或跨多個身分共用的原因時。您可以刪除內嵌政策，將其從 中移除 AWS 帳戶。您無法刪除 AWS 受管政策。

Console

刪除 IAM 使用者、群組或角色的內嵌政策

1. 在服務導覽窗格中，選擇 User groups (使用者群組)、Users (使用者) 或者 Roles (角色)。
2. 選擇您要透過政策來刪除的使用者群組、使用者、或角色的名稱。然後選擇 許可 標籤。
3. 選取要刪除的政策旁的核取方塊，然後選擇移除。然後，在確認對話方塊中，確認移除和刪除該政策。

- 若要刪除使用者或角色中的內嵌原則，請選擇移除以確認刪除。
- 如果您要刪除 使用者群組 中的單一內嵌政策，請輸入政策的名稱，然後選擇 刪除 。如果您要刪除 使用者群組 中的多項內嵌政策，請輸入您要刪除的政策數量，後接 **inline policies**，然後選擇 刪除 。例如，如果您要刪除三個內嵌政策，請輸入 **3 inline policies**。

刪除 IAM 政策 (AWS CLI)

您可以使用 AWS Command Line Interface (AWS CLI) 在 IAM 中刪除客戶受管政策和內嵌政策。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

> **Note**
>
> 刪除 IAM 政策的動作具有永久性。刪除政策後，將無法復原。

如需有關 IAM 政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management和 IAM JSON 政策元素參考。

如需有關受管與內嵌政策之間的差異的詳細資訊，請參閱 受管政策與內嵌政策。

先決條件

刪除政策之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

刪除客戶管理政策 (AWS CLI)

您可以從 AWS Command Line Interface刪除客戶受管政策。

刪除客戶受管政策 (AWS CLI)

1. (選用) 若要檢視關於政策的資訊，請執行下列命令：
   - 列出受管政策：list-policies
   - 取得關於受管政策的詳細資訊：get-policy
2. (選用) 如果要了解的政策和身分之間的關係，請執行下列命令：

- 若要列出受管政策所連接的身分 (IAM 使用者、IAM 群組和 IAM 角色)，請執行以下命令：
  - list-entities-for-policy
- 若要列出連接到身分 (使用者、使用者群組或角色) 的受管政策，請執行下列其中一項命令：
  - list-attached-user-policies
  - list-attached-group-policies
  - list-attached-role-policies

3. 若要刪除客戶受管政策，請執行下列命令：

   - : delete-policy

刪除內嵌政策 (AWS CLI)

您可以從 AWS CLI刪除內嵌政策。

若要刪除內嵌政策 (AWS CLI)

1. (選用) 若要列出連接到身分 (使用者、使用者群組或角色) 的所有內嵌政策，請使用下列其中一項命令：

   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies

2. (選用) 若要擷取嵌入到身分 (使用者、使用者群組或角色) 中的內嵌政策文件，請使用下列其中一項命令：

   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy

3. 若要從身分中刪除內嵌政策 (非服務連結角色的使用者、使用者群組或角色)，請使用下列命令：

   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

刪除 IAM 政策 (AWS API)

您可以使用 AWS API 在 IAM 中刪除客戶受管政策和內嵌政策。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

> **Note**
>
> 刪除 IAM 政策的動作具有永久性。刪除政策後，將無法復原。

如需有關 IAM 政策結構和語法的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management和 IAM JSON 政策元素參考。

如需有關受管與內嵌政策之間的差異的詳細資訊，請參閱 受管政策與內嵌政策。

先決條件

刪除政策之前，您應該檢閱其最近的服務層級活動。這很重要，因為您不希望從正在使用該許可的主體 (人員或應用程式) 中移除存取。如需有關檢視上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在 中精簡許可。

刪除客戶受管政策 (AWS API)

您可以使用 AWS API 刪除客戶受管政策。

刪除客戶受管政策 (AWS API)

1. (選用) 若要檢視關於政策的資訊，請呼叫下列操作：
   - 列出受管政策：ListPolicies
   - 取得關於受管政策的詳細資訊：GetPolicy
2. (選用) 如果要了解的政策和身分之間的關係，請呼叫下列操作：
   - 若要列出受管政策所連接的身分 (IAM 使用者、IAM 群組和 IAM 角色)，請呼叫以下操作：
     - ListEntitiesForPolicy
   - 若要列出連接到身分的受管政策 (使用者、使用者群組或角色)，請呼叫下列其中一項操作：
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies

3. 若要刪除客戶受管政策，請呼叫下列操作：

   - DeletePolicy

刪除內嵌政策 (AWS API)

您可以使用 AWS API 刪除內嵌政策。

刪除內嵌政策 (AWS API)

1. (選用) 若要列出連接到身分 (使用者、使用者群組或角色) 的所有內嵌政策，請呼叫下列其中一項操作：

   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies

2. (選用) 若要擷取嵌入到身分 (使用者、使用者群組或角色) 中的內嵌政策文件，請呼叫下列其中一項操作：

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

3. 若要從身分中刪除內嵌政策 (非服務連結角色的使用者、使用者群組或角色)，請呼叫下列操作：

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

AWS 使用上次存取的資訊在 中精簡許可

身為管理員，您可能會為 IAM 資源 (角色、使用者、使用者群組或政策) 授予超出其所需範圍的許可。IAM 會提供上次存取的資訊，協助您識別未使用的許可，以便您移除這些許可。您可以使用上次存取的資訊來精細化政策，並僅允許存取 IAM 身分和政策所使用的服務和動作。這有助於您更加符合最低許可的最佳實務。 您可以檢視 IAM 或 AWS Organizations中存在的身分或政策上次存取的資訊。

您可以透過未使用的存取權分析器持續監控上次存取的資訊。如需詳細資訊，請參閱外部和未使用的存取權調查結果。

主題

- IAM 上次存取的資訊類型
- 的上次存取資訊 AWS Organizations
- 關於上次存取資訊的注意事項
- 必要許可
- IAM 和 AWS Organizations 實體的活動疑難排解
- 其中 會 AWS 追蹤上次存取的資訊
- 檢視 IAM 上次存取的資訊
- 檢視 的上次存取資訊 AWS Organizations
- 使用上次存取資訊的範例案例
- IAM 動作最近存取的資訊服務和動作

IAM 上次存取的資訊類型

您可以檢視 IAM 身分上次存取的兩種類型資訊：允許的 AWS 服務資訊和允許的動作資訊。該資訊包含嘗試存取 AWS API 的日期和時間。對於動作，上次存取的資訊會報告服務管理動作。管理動作包括建立、刪除和修改動作。若要進一步了解如何檢視上次存取 IAM 的資訊，請參閱 檢視 IAM 上次存取的資訊。

如需有關使用上次存取的資訊以決定您授予 IAM 身分許可的案例，請參閱 使用上次存取資訊的範例案例。

若要深入瞭解如何提供管理動作資訊，請參閱關於上次存取資訊的注意事項。

的上次存取資訊 AWS Organizations

如果您使用管理帳戶登入資料登入，您可以檢視組織中 AWS Organizations 實體或政策的服務上次存取資訊。 AWS Organizations 實體包括組織根目錄、組織單位 OUs) 或帳戶。的上次存取資訊 AWS Organizations 包含服務控制政策 (SCP) 允許之服務的相關資訊。這些資訊會指出組織或帳戶中哪些主體 (根使用者、IAM 使用者或角色) 上次嘗試存取服務，以及何時存取服務。若要進一步了解報告以及如何檢視 的上次存取資訊 AWS Organizations，請參閱 檢視 的上次存取資訊 AWS Organizations。

如需使用上次存取資訊來決定您授予 AWS Organizations 實體之許可的範例案例，請參閱 使用上次存取資訊的範例案例。

關於上次存取資訊的注意事項

在您使用報告上次存取的資訊來變更 IAM 身分或 AWS Organizations 實體的許可之前，請檢閱下列有關資訊的詳細資訊。

- 追蹤期 – 最近的活動會在四小時內顯示在 IAM 主控台中。服務資訊的追蹤期至少為 400 天，具體視服務何時開始追蹤動作資訊而定。Amazon S3 動作資訊的追蹤期從 2020 年 4 月 12 日開始。Amazon EC2、IAM 和 Lambda 行動追蹤期由 2021 年 4 月 7 日開始。所有其他服務的追蹤期從 2023 年 5 月 23 日開始。如需檢視可使用動作上次存取資訊的服務清單，請參閱 IAM 動作最近存取的資訊服務和動作。如需有關哪些區域提供動作上次存取資訊的更多資訊，請參閱 其中 會 AWS 追蹤上次存取的資訊。
- 報告的嘗試次數 – 服務上次存取的資料包括存取 AWS API 的所有嘗試次數，而不只是成功嘗試次數。這包括使用 AWS 管理主控台、透過任何 SDKs AWS 的 API，或任何命令列工具進行的所有嘗試。在上次存取的服務相關資料中看到未預期的項目並不表示您的帳戶資訊洩露，因為請求可能已遭拒。請參閱您的 CloudTrail 日誌並將其作為有關所有 API 呼叫以及它們是成功還是被拒絕的存取的資訊的權威來源。
- PassRole – 系統不會追蹤 `iam:PassRole` 動作，也不會將其包含在 IAM 動作上次存取的資訊中。
- 動作上次存取的資訊 – 動作上次存取的資訊適用於由 IAM 身分存取的服務管理動作。檢視動作上次存取的報告資訊的服務及其動作清單。

  > **Note**
  >
  > 動作上次存取的資訊無法用於所有資料平面事件。

- 管理事件 – IAM 為 CloudTrail 記錄的服務管理事件提供動作資訊。有時候，CloudTrail 管理事件也被稱為控制平面操作或控制平面事件。管理事件可讓您了解在 資源上執行的管理操作 AWS 帳戶。若要深入了解 CloudTrail 中的管理事件，請參閱《AWS CloudTrail 使用者指南》中的記錄管理事件。
- 報告擁有者 – 只有產生報告的主體才可以檢視報告詳細資訊。這表示當您檢視 中的資訊時 AWS 管理主控台，您可能需要等待它產生和載入。如果您使用 AWS CLI 或 AWS API 取得報告詳細資訊，您的登入資料必須符合產生報告的委託人登入資料。如果您使用角色或 AWS STS 聯合身分使用者主體的臨時登入資料，則必須在相同工作階段期間產生和擷取報告。如需有關擔任角色工作階段主體的詳細資訊，請參閱 AWS JSON 政策元素： Principal。
- IAM 資源 – IAM 上次存取的資訊，包括您的帳戶中的 IAM 資源 (角色、使用者、IAM 群組和政策)。的上次存取資訊 AWS Organizations 包括指定 AWS Organizations 實體中的主體 (IAM 使用者、IAM 角色或 AWS 帳戶根使用者)。上次存取的資訊不包括未驗證的嘗試。

- IAM 政策類型 – IAM 上次存取的資訊包含由 IAM 身分的政策所許可的服務。這些政策連接至角色或者直接或透過群組連接至使用者。您的報告不包含其他政策類型允許的存取。排除的政策類型包括以資源為基礎的政策、存取控制清單、 AWS Organizations SCP、IAM 許可邊界，以及工作階段政策。服務連結角色所提供的許可是由它們連結的服務所定義，而且無法在 IAM 中修改。若要進一步了解服務連結角色，請參閱建立服務連結角色 若要了解如何評估不同原則類型以允許或拒絕存取，請參閱政策評估邏輯。
- AWS Organizations 政策類型 – 的資訊僅 AWS Organizations 包含 AWS Organizations 實體的繼承服務控制政策 (SCPs) 允許的服務。SCP 是連接到根帳戶、OU 或帳戶的政策。您的報告不包含其他政策類型允許的存取。排除的政策類型包含身分類型政策、資源類型政策、存取控制清單、IAM 許可邊界，以及工作階段政策。若要了解如何評估不同的政策類型以允許或拒絕存取，請參閱政策評估邏輯。
- 指定政策 ID – 當您使用 AWS CLI 或 AWS API 為 中上次存取的資訊產生報告時 AWS Organizations，您可以選擇指定政策 ID。產生的報告包含僅由該政策允許的服務資訊。此資訊包含指定 AWS Organizations 實體或實體子項中最新的帳戶活動。如需詳細資訊，請參閱 aws iam generate-organizations-access-report 或 GenerateOrganizationsAccessReport。
- AWS Organizations 管理帳戶 – 您必須登入組織的管理帳戶，才能檢視服務上次存取的資訊。您可以選擇使用 IAM 主控台 AWS CLI、 或 AWS API 檢視管理帳戶的資訊。產生的報告會列出所有 AWS 服務，因為管理帳戶不受 SCPs限制。如果您指定的政策 ID 位於 CLI 或 API 中，便會忽略該政策。對於每個服務，報告包含僅適用於管理帳戶的資訊。不過，其他 AWS Organizations 實體的報告不會傳回管理帳戶中活動的資訊。
- AWS Organizations 設定 – 管理員必須在組織根目錄中啟用 SCPs，然後才能為 產生資料 AWS Organizations。

必要許可

若要在 中檢視上次存取的資訊 AWS 管理主控台，您必須擁有授予必要許可的政策。

IAM 資訊的許可

若要使用 IAM 主控台檢視 IAM 使用者、角色或政策的上次存取資訊，您必須擁有一個包含下列動作的政策：

- `iam:GenerateServiceLastAccessedDetails`
- `iam:Get*`
- `iam:List*`

這些許可允許使用者檢視下列項目：

- 哪些使用者、群組或角色連接至受管政策
- 使用者或角色可以存取哪些服務
- 他們上一次存取該服務的時間
- 他們上次嘗試使用特定的 Amazon EC2、IAM、Lambda 或 Amazon S3 動作的時間

若要使用 AWS CLI 或 AWS API 來檢視 IAM 的上次存取資訊，您必須具有符合您要使用之操作的許可：

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam:ListPoliciesGrantingServiceAccess`

此範例會示範如何建立身分型政策，允許檢視 IAM 上次存取的資訊。此外，還允許對所有 IAM 的唯讀存取。此政策定義了程式設計和主控台存取的許可。

JSON

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

AWS Organizations 資訊的許可

若要使用 IAM 主控台檢視 AWS Organizations中根、組織單位或帳戶實體的報告，您必須具有一個包含下列動作的政策：

- iam:GenerateOrganizationsAccessReport
- iam:GetOrganizationsAccessReport
- organizations:DescribeAccount
- organizations:DescribeOrganization
- organizations:DescribeOrganizationalUnit
- organizations:DescribePolicy
- organizations:ListChildren
- organizations:ListParents
- organizations:ListPoliciesForTarget
- organizations:ListRoots
- organizations:ListTargetsForPolicy

若要使用 AWS CLI 或 AWS API 來檢視 的服務上次存取資訊 AWS Organizations，您必須擁有包含下列動作的政策：

- iam:GenerateOrganizationsAccessReport
- iam:GetOrganizationsAccessReport
- organizations:DescribePolicy
- organizations:ListChildren
- organizations:ListParents
- organizations:ListPoliciesForTarget
- organizations:ListRoots
- organizations:ListTargetsForPolicy

此範例示範如何建立身分型政策，允許檢視 的服務上次存取資訊 AWS Organizations。此外，它允許唯讀存取所有 AWS Organizations。此政策定義了程式設計和主控台存取的許可。

JSON

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

```
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
    }
}
```

您也可以使用 iam：OrganizationsPolicyId 條件金鑰，僅允許針對特定 AWS Organizations 政策產生報告。如需政策範例，請參閱 IAM：檢視 AWS Organizations 政策的服務上次存取資訊。

IAM 和 AWS Organizations 實體的活動疑難排解

在某些情況下，您 AWS 管理主控台 上次存取的資訊表可能空白。或者，您的 AWS CLI 或 AWS API 請求會傳回空的資訊集或 null 欄位。在這些情況下，檢閱下列問題：

- 對於上次存取的動作資訊，清單中可能不會傳回您預期看到的動作。這可能是因為 IAM 身分沒有 動作的許可，或 AWS 尚未追蹤上次存取資訊的動作。
- 對於 IAM 使用者，請確定該使用者至少連接一個內嵌或受管政策，無論是直接或透過群組成員資格連接。
- 對於 IAM 群組，請確認群組至少連接一個內嵌或受管政策。
- 對於 IAM 群組，報告只會傳回使用群組的政策存取服務之成員的服務上次存取資訊。若要了解成員是否使用了其他政策，請檢閱該使用者的上次存取資訊。
- 對於 IAM 角色，請確認角色至少連接一個內嵌或受管政策。
- 對於 IAM 實體 (使用者或角色)，請檢閱可能影響該實體之許可的其他政策類型。其中包括資源型政策、存取控制清單、 AWS Organizations 政策、IAM 許可界限或工作階段政策。如需詳細資訊，請參閱政策類型或單一帳戶中請求的政策評估。
- 對於 IAM 政策，請確定指定的受管政策已連接到至少一個使用者、內含成員的群組，或角色。
- 對於 AWS Organizations 實體（根、OU 或帳戶），請確定您使用 AWS Organizations 管理帳戶登入資料進行簽署。
- 確定已在組織根目錄中啟用 SCP。
- 動作上次存取的資訊僅適用於 IAM 動作最近存取的資訊服務和動作 中列出的動作。

當您進行變更時，請等待至少 4 小時，活動才會顯示在您的 IAM 主控台報告中。如果您使用 AWS CLI 或 AWS API，則必須產生新的報告來檢視更新的資訊。

其中 會 AWS 追蹤上次存取的資訊

AWS 會收集標準 AWS 區域的上次存取資訊。當 AWS 新增其他區域時，這些區域會新增至下表，包括每個區域中 AWS 開始追蹤資訊的日期。

- 服務資訊：服務追蹤期至少為 400 天，如果區域在過去 400 天內開始追蹤此功能，則服務追蹤期會更短。
- 動作資訊 – Amazon S3 管理動作的追蹤期從 2020 年 4 月 12 日開始。Amazon EC2、IAM 和 Lambda 管理行動追蹤期由 2021 年 4 月 7 日開始。所有其他服務的管理動作追蹤期從 2023 年 5 月 23 日開始。如果某區域的追蹤日期遲於 2023 年 5 月 23 日，則該區域的動作上次存取資訊將從較遲的日期開始。

區域名稱	區域	追蹤開始日期
美國東部 (俄亥俄)	us-east-2	2017 年 10 月 27 日
美國東部 (維吉尼亞北部)	us-east-1	2015 年 10 月 1 日
美國西部 (加利佛尼亞北部)	us-west-1	2015 年 10 月 1 日
美國西部 (奧勒岡)	us-west-2	2015 年 10 月 1 日
Africa (Cape Town)	af-south-1	2020 年 4 月 22 日
亞太地區 (香港)	ap-east-1	2019 年 4 月 24 日
亞太地區 (海德拉巴)	ap-south-2	2022 年 11 月 22 日
亞太地區 (雅加達)	ap-southeast-3	2021 年 12 月 13 日
亞太地區 (墨爾本)	ap-southeast-4	2023 年 1 月 23 日
亞太區域 (孟買)	ap-south-1	2016 年 6 月 27 日
亞太地區 (大阪)	ap-northeast-3	2018 年 2 月 11 日
亞太區域 (首爾)	ap-northeast-2	2016 年 1 月 6 日

區域名稱	區域	追蹤開始日期
亞太區域 (新加坡)	ap-southeast-1	2015 年 10 月 1 日
亞太區域 (雪梨)	ap-southeast-2	2015 年 10 月 1 日
亞太區域 (東京)	ap-northeast-1	2015 年 10 月 1 日
加拿大 (中部)	ca-central-1	2017 年 10 月 28 日
歐洲 (法蘭克福)	eu-central-1	2015 年 10 月 1 日
歐洲 (愛爾蘭)	eu-west-1	2015 年 10 月 1 日
歐洲 (倫敦)	eu-west-2	2017 年 10 月 28 日
歐洲 (米蘭)	eu-south-1	2020 年 4 月 28 日
Europe (Paris)	eu-west-3	2017 年 12 月 18 日
歐洲 (西班牙)	eu-south-2	2022 年 11 月 15 日
Europe (Stockholm)	eu-north-1	2018 年 12 月 12 日
歐洲 (蘇黎世)	eu-central-2	2022 年 11 月 8 日
以色列 (特拉維夫)	il-central-1	2023 年 8 月 1 日
Middle East (Bahrain)	me-south-1	2019 年 7 月 29 日
中東 (阿拉伯聯合大公國)	me-central-1	2022 年 8 月 30 日
南美洲 (聖保羅)	sa-east-1	2015 年 12 月 11 日
AWS GovCloud （美國東部）	us-gov-east-1	2023 年 7 月 1 日
AWS GovCloud （美國西部）	us-gov-west-1	2023 年 7 月 1 日

如果某個區域未在上表中列出，則表示此區域尚不提供上次存取的相關資訊。

AWS 區域是地理區域中的 AWS 資源集合。區域會分組成分割區。標準區域是屬於該 aws 分區的區域。如需有關不同分割區的詳細資訊，請參閱 AWS 一般參考中的 Amazon Resource Name (ARN) 格式。如需區域的詳細資訊，請參閱 中的關於 AWS 區域 AWS 一般參考。

檢視 IAM 上次存取的資訊

您可以使用 AWS 管理主控台 AWS CLI、 或 AWS API 檢視 IAM 的上次存取資訊。檢視顯示上次存取資訊的服務及其動作清單。如需有關上次存取的資訊的詳細資訊，請參閱 AWS 使用上次存取的資訊在中精簡許可。

您可以在 IAM 中檢視下列資源類型的資訊。在每個案例中，資訊涵蓋指定報告期間允許的服務：

- 使用者 – 檢視使用者上一次嘗試存取每個允許服務的時間。
- 使用者群組 – 檢視有關上一次使用者群組成員嘗試存取每個允許服務的資訊。此報告也包含曾嘗試存取的成員總數。
- 角色 – 檢視上一次有人使用該角色嘗試存取每個允許服務的時間。
- 政策 – 檢視有關上一次使用者或角色嘗試存取每個允許服務的資訊。此報告也包含曾嘗試存取的實體總數。

> **Note**
>
> 在檢視 IAM 中的資源的存取資訊之前，請確定您了解報告期間、報告的實體，以及您資訊的評估政策類型。如需詳細資訊，請參閱 the section called “關於上次存取資訊的注意事項”。

檢視 IAM 的資訊 (主控台)

您可以在 IAM 主控台的上次存取索引標籤上檢視 IAM 上次存取的資訊。

檢視 IAM 的資訊 (主控台)

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者群組、使用者、角色或政策。
3. 選擇任意使用者、使用者群組、角色或政策名稱以開啟其摘要頁面，然後選擇上次存取索引標籤。根據您選擇的資源，檢視下列資訊：
   - 使用者群組 – 檢視使用者群組成員可存取的服務清單。您也可以檢視成員上次存取服務的時間、他們使用的使用者群組政策，以及提出要求的使用者群組成員。選擇政策的名稱以了解其為

受管政策或內嵌使用者群組政策。選擇使用者群組成員的名稱以檢視該使用者群組的所有成員，以及他們上次存取該服務的時間。

- 使用者 – 檢視使用者可存取的服務清單。您也可以檢視他們上次存取服務的時間，以及目前與該名使用者關聯的政策有哪些。選擇政策的名稱，以了解該政策是受管政策、內嵌使用者政策還是使用者群組的內嵌政策。
- 角色 – 檢視角色可存取的服務清單、角色上次存取該服務的時間，以及他們使用哪些政策。選擇政策的名稱以了解其為受管政策還是內嵌角色政策。
- 政策 – 檢視政策中包含允許動作的服務清單。您也可以檢視上次使用政策來存取服務的時間，以及使用政策的實體 (使用者或角色)。上次存取日期也包含透過其他政策授予此政策存取權的時間。選擇實體的名稱以了解哪些實體有連接政策，以及它們上次存取服務的時間。

4. 在表格的服務欄中，選擇其中一項包含動作上次存取資訊的服務之名稱，以檢視 IAM 實體嘗試存取的管理動作清單。您可以檢視 AWS 區域 和時間戳記，顯示上次有人嘗試執行動作的時間。
5. 針對包含動作上次存取資訊的服務，將為其服務和管理動作顯示上次存取欄。檢閱此欄中傳回的下列可能結果。這些結果取決於是否允許、是否存取服務或動作，以及是否由 追蹤 AWS 上次存取的資訊。

   <number of> 天前

   自追蹤期間內使用服務或動作之後的天數。服務的追蹤期為過去 400 天。Amazon S3 動作的追蹤期由 2020 年 4 月 12 日開始。Amazon EC2、IAM 和 Lambda 行動追蹤期由 2021 年 4 月 7 日開始。所有其他服務的追蹤期從 2023 年 5 月 23 日開始。若要進一步了解每個追蹤開始日期 AWS 區域，請參閱 其中 會 AWS 追蹤上次存取的資訊。

   在追蹤期內未存取

   追蹤的服務或動作尚未由實體在追蹤期間內使用。

   您可以擁有清單中未出現的動作許可。如果 AWS目前不包含動作的追蹤資訊，就可能發生這種情況。您不應該只根據有沒有追蹤資訊來決定許可。相反地，建議您使用此資訊來告知並支援授予最低權限的整體策略。檢查您的政策以確認存取層級是否適當。

檢視 IAM 的資訊 (AWS CLI)

您可以使用 AWS CLI 擷取上次使用 IAM 資源嘗試存取 AWS 服務和 Amazon S3、Amazon EC2、IAM 和 Lambda 動作的相關資訊。IAM 資源可以是使用者、使用者群組、角色或政策。

檢視 IAM 的資訊 (AWS CLI)

1. 產生報告。此請求必須包含您需要報告的 IAM 資源 (使用者、使用者群組、角色或政策) 的 ARN。您可以在報告中指定要產生的資料粒度層級，以檢視任一服務或同時檢視服務和動作的存取詳細資訊。它會傳回 `job-id`，然後您可將它用於 `get-service-last-accessed-details` 和 `get-service-last-accessed-details-with-entities` 操作以監控 `job-status`，直到任務完成。

   - aws iam generate-service-last-accessed-details

2. 使用上個步驟的 `job-id` 參數來擷取報告的詳細資訊。

   - aws iam get-service-last-accessed-details

   根據您在 `generate-service-last-accessed-details` 操作中請求的資源類型與精細程度，此操作會傳回以下資訊：

   - 使用者 – 傳回指定的使用者可存取的服務清單。對於每個服務，此操作會傳回使用者最後一次嘗試的日期與時間，以及該使用者的 ARN。
   - 使用者群組 – 傳回指定使用者群組的成員可使用連接至該使用者群組的政策進行存取的服務清單。對於每個服務，此操作會傳回任何使用者群組成員最後一次嘗試的日期與時間。它也會傳回該使用者的 ARN 以及曾嘗試存取服務的使用者群組成員總數。使用 GetServiceLastAccessedDetailsWithEntities 操作來擷取所有成員的清單。
   - 角色 – 傳回指定的角色可存取的服務清單。對於每個服務，此操作會傳回角色最後一次嘗試的日期與時間，以及該角色的 ARN。
   - 政策 – 傳回指定的政策允許存取的服務清單。對於每個服務，此操作會傳回實體 (使用者或角色) 上次使用政策來嘗試存取服務的日期和時間。它也會傳回實體的 ARN 以及嘗試存取的實體總數。

3. 進一步了解在嘗試存取特定服務時使用使用者群組或政策許可的實體。此操作會傳回實體清單，包括各實體的 ARN、ID、名稱、路徑、類型 (使用者或角色)，以及它們最後一次嘗試存取服務的時間。您也可以針對使用者和角色使用此操作，但只會傳回有關該實體的資訊。

   - aws iam get-service-last-accessed-details-with-entities

4. 進一步了解有關身分 (使用者、使用者群組或角色) 在嘗試存取特定服務時使用之以身分為基礎的政策。當您指定身分和服務時，此操作會傳回實體可用於存取指定服務的許可政策清單。此操作可提供政策的目前狀態，而且不倚賴產生的報告。它也不會傳回其他政策類型，例如以資源為基礎的政策、存取控制清單、 AWS Organizations 政策、IAM 許可邊界，或工作階段政策。如需詳細資訊，請參閱政策類型或單一帳戶中請求的政策評估。

- aws iam list-policies-granting-service-access

檢視 IAM (AWS API) 的資訊

您可以使用 AWS API 擷取上次使用 IAM 資源嘗試存取 AWS 服務和 Amazon S3、Amazon EC2、IAM 和 Lambda 動作的相關資訊。IAM 資源可以是使用者、使用者群組、角色或政策。您可以在報告中指定要產生的資料細微層級，以檢視任一服務或同時檢視服務和動作的詳細資訊。

檢視 IAM (AWS API) 的資訊

1. 產生報告。此請求必須包含您需要報告的 IAM 資源 (使用者、使用者群組、角色或政策) 的 ARN。它會傳回 `JobId`，然後您可將它用於 `GetServiceLastAccessedDetails` 和 `GetServiceLastAccessedDetailsWithEntities` 操作以監控 `JobStatus`，直到任務完成。

   - GenerateServiceLastAccessedDetails

2. 使用上個步驟的 `JobId` 參數來擷取報告的詳細資訊。

   - GetServiceLastAccessedDetails

   根據您在 `GenerateServiceLastAccessedDetails` 操作中請求的資源類型與精細程度，此操作會傳回以下資訊：

   - 使用者 – 傳回指定的使用者可存取的服務清單。對於每個服務，此操作會傳回使用者最後一次嘗試的日期與時間，以及該使用者的 ARN。
   - 使用者群組 – 傳回指定使用者群組的成員可使用連接至該使用者群組的政策進行存取的服務清單。對於每個服務，此操作會傳回任何使用者群組成員最後一次嘗試的日期與時間。它也會傳回該使用者的 ARN 以及曾嘗試存取服務的使用者群組成員總數。使用 GetServiceLastAccessedDetailsWithEntities 操作來擷取所有成員的清單。
   - 角色 – 傳回指定的角色可存取的服務清單。對於每個服務，此操作會傳回角色最後一次嘗試的日期與時間，以及該角色的 ARN。
   - 政策 – 傳回指定的政策允許存取的服務清單。對於每個服務，此操作會傳回實體 (使用者或角色) 上次使用政策來嘗試存取服務的日期和時間。它也會傳回實體的 ARN 以及嘗試存取的實體總數。

3. 進一步了解在嘗試存取特定服務時使用使用者群組或政策許可的實體。此操作會傳回實體清單，包括各實體的 ARN、ID、名稱、路徑、類型 (使用者或角色)，以及它們最後一次嘗試存取服務的時間。您也可以針對使用者和角色使用此操作，但只會傳回有關該實體的資訊。

   - GetServiceLastAccessedDetailsWithEntities

4. 進一步了解有關身分 (使用者、使用者群組或角色) 在嘗試存取特定服務時使用之以身分為基礎的政策。當您指定身分和服務時，此操作會傳回實體可用於存取指定服務的許可政策清單。此操作可提供政策的目前狀態，而且不倚賴產生的報告。它也不會傳回其他政策類型，例如以資源為基礎的政策、存取控制清單、 AWS Organizations 政策、IAM 許可邊界，或工作階段政策。如需詳細資訊，請參閱政策類型或單一帳戶中請求的政策評估。

   - ListPoliciesGrantingServiceAccess

檢視 的上次存取資訊 AWS Organizations

您可以使用 IAM AWS Organizations 主控台 AWS CLI或 AWS API 檢視服務上次存取的資訊。如需關於資料、必要許可、疑難排解及支援區域的重要資訊，請參閱AWS 使用上次存取的資訊在 中精簡許可。

當您使用 AWS Organizations 管理帳戶登入資料登入 IAM 主控台時，您可以檢視組織中任何實體的資訊。 AWS Organizations 實體包括組織根目錄、組織單位 OUs) 和帳戶。您也可以使用 IAM 主控台來檢視組織中任何服務控制政策 (SCP) 的資訊。IAM 會顯示套用至實體的 SCP 所允許的服務清單。對於每個服務，您可以檢視所選 AWS Organizations 實體或實體子系的最新帳戶活動資訊。

當您使用 AWS CLI 或 AWS API 搭配管理帳戶登入資料時，您可以為組織中的任何實體或政策產生報告。實體的程式設計報告包含套用至實體之任何 SCP 所允許的服務清單。對於每個服務，報告包含指定 AWS Organizations 實體或實體子目錄中的最新帳戶活動。

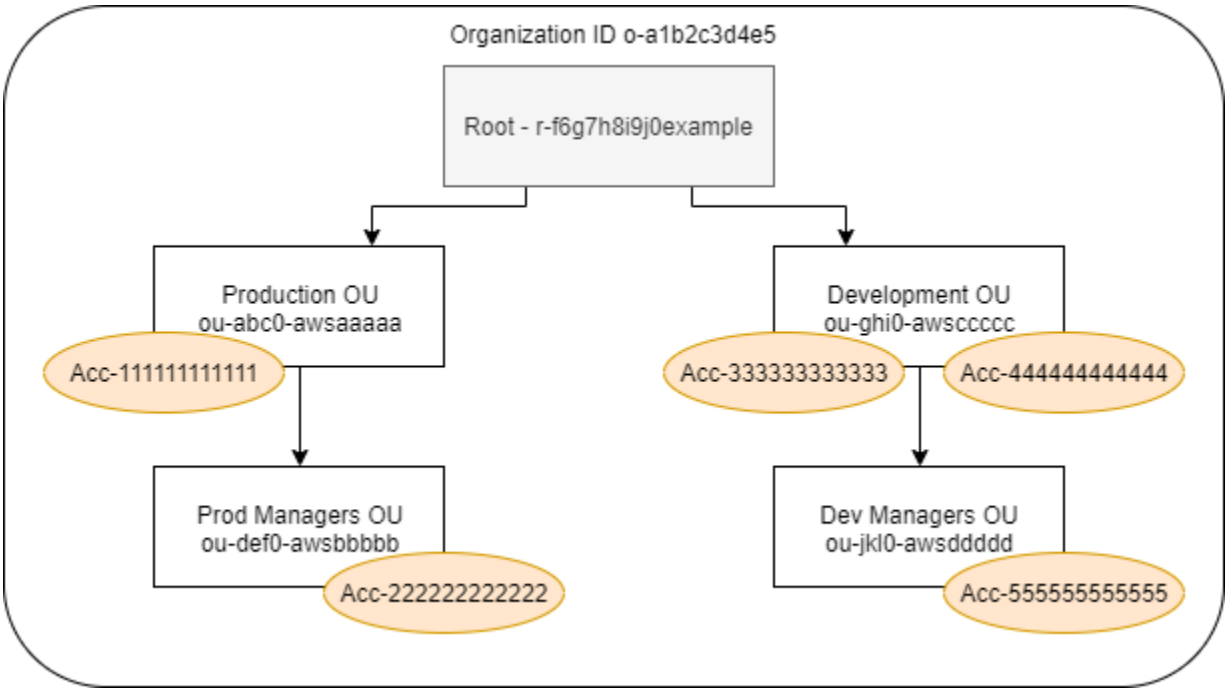
當您為政策產生程式設計報告時，您必須指定 AWS Organizations 實體。這份報告包含指定 SCP 所允許的服務清單。對於每個服務，其包含由政策授予許可之實體或實體子系的最新帳戶活動。如需詳細資訊，請參閱 aws iam generate-organizations-access-report 或 GenerateOrganizationsAccessReport。

檢視報告之前，請確認您了解管理帳戶需求和資訊、報告期間、回報實體和評估的政策類型。如需詳細資訊，請參閱 the section called “關於上次存取資訊的注意事項”。

了解 AWS Organizations 實體路徑

當您使用 AWS CLI 或 AWS API 產生 AWS Organizations 存取報告時，您必須指定實體路徑。路徑是 AWS Organizations 實體結構的文字表示。

您可以使用組織已知的結構來建立實體路徑。例如，假設您在其中具有下列組織結構 AWS Organizations。



開發人員管理員 OU 的路徑是使用組織 ID、根目錄和路徑中所有的 OU 包含 OU。

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsddddd/
```

生產 OU 中帳戶的路徑是使用組織、根、OU 和帳戶號碼的 ID 建立的。

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> 組織 ID 是全域唯一，但 OU ID 和根 ID 只有在組織內是唯一。這表示沒有兩個組織共用相同的組織 ID。不過，另一個組織的 OU 或根可能與您的 ID 相同。我們建議您在指定 OU 或根時，一律包含組織 ID。

檢視 AWS Organizations （主控台）的資訊

您可以使用 IAM 主控台檢視您的根目錄、OU、帳戶或政策的上次存取資訊。

檢視根目錄的資訊 (主控台)

1. AWS 管理主控台 使用 AWS Organizations 管理帳戶登入資料登入 ，然後開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格的 Access reports (存取報告) 區段下方，選擇 Organization activity (組織活動)。
3. 在 Organization activity (組織活動) 頁面，選擇 Root (根)。
4. 在 Details and activity (詳細資訊與活動) 索引標籤上檢視 Service access report (服務存取報告) 部分。該資訊包含直接連接至根目錄之政策所允許的服務清單。該資訊會為您顯示服務上次存取哪個帳戶的服務，以及存取時間。如需關於哪個主體存取服務的詳細資訊，請以該帳戶的管理員身分登入帳戶並檢視 IAM 上次存取資訊。
5. 選擇 Attached SCPs (連接的 SCP) 索引標籤，檢視連接至根目錄的服務控制政策 (SCP) 清單。IAM 會顯示與每個政策連接的目標實體數量。您可以使用此資訊來決定要檢閱哪些 SCP。
6. 選擇 SCP 的名稱，以檢視政策允許的所有服務。對於每個服務，可從中檢視服務上次存取哪個帳戶的服務，以及存取時間。
7. 選擇在 中編輯 AWS Organizations以檢視其他詳細資訊，並在 AWS Organizations 主控台中編輯 SCP。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的更新 SCP。

檢視 OU 或帳戶的資訊 (主控台)

1. AWS 管理主控台 使用 AWS Organizations 管理帳戶登入資料登入 ，然後開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格的 Access reports (存取報告) 區段下方，選擇 Organization activity (組織活動)。
3. 在 Organization activity (組織活動)頁面，展開組織的結構。然後，選擇 OU 或您要檢視任何帳戶的名稱 (管理帳戶除外)。
4. 在 Details and activity (詳細資訊與活動) 索引標籤上檢視 Service access report (服務存取報告) 部分。該資訊包含 SCP 允許的服務清單，而且這些 SCP 都連接至 OU 或帳戶及其所有父系。該資訊會為您顯示服務上次存取哪個帳戶的服務，以及存取時間。如需關於哪個主體存取服務的詳細資訊，請以該帳戶的管理員身分登入帳戶並檢視 IAM 上次存取資訊。
5. 選擇 Attached SCPs (連接的 SCP) 索引標籤，檢視直接連接至 OU 或帳戶的服務控制政策 (SCP) 清單。IAM 會顯示與每個政策連接的目標實體數量。您可以使用此資訊來決定要檢閱哪些 SCP。
6. 選擇 SCP 的名稱，以檢視政策允許的所有服務。對於每個服務，可從中檢視服務上次存取哪個帳戶的服務，以及存取時間。
7. 選擇在 中編輯 AWS Organizations以檢視其他詳細資訊，並在 AWS Organizations 主控台中編輯 SCP。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的更新 SCP。

檢視管理帳戶的資訊 (主控台)

1. AWS 管理主控台 使用 AWS Organizations 管理帳戶登入資料登入 ，然後開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格的 Access reports (存取報告) 區段下方，選擇 Organization activity (組織活動)。
3. 在 Organization activity (組織活動) 頁面，展開組織的結構，然後選擇管理帳戶的名稱。
4. 在 Details and activity (詳細資訊與活動) 索引標籤上檢視 Service access report (服務存取報告) 部分。這些資訊包括所有 AWS 服務的清單。管理帳戶不受 SCP 限制。該資訊會為您顯示帳戶上次是否存取服務，以及存取的時間。如需關於哪個主體存取服務的詳細資訊，請以該帳戶的管理員身分登入帳戶並檢視 IAM 上次存取資訊。
5. 選擇 Attached SCPs (連接的 SCP) 索引標籤，以確認沒有任何連接的 SCP，因為帳戶就是管理帳戶。

檢視政策的資訊 (主控台)

1. AWS 管理主控台 使用 AWS Organizations 管理帳戶登入資料登入 ，然後開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格的 Access reports (存取報告) 區段下方，選擇 Service control policies (SCPs) (服務控制政策 (SCP))。
3. 在Service control policies (SCPs) (服務控制政策 (SCP)) 頁面上，檢視組織的政策清單。您可以檢視每個政策連接的目標實體數量。
4. 選擇 SCP 的名稱，以檢視政策允許的所有服務。對於每個服務，可從中檢視服務上次存取哪個帳戶的服務，以及存取時間。
5. 選擇在 中編輯 AWS Organizations以檢視其他詳細資訊，並在 AWS Organizations 主控台中編輯 SCP。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的更新 SCP。

檢視 AWS Organizations (AWS CLI) 的資訊

您可以使用 AWS CLI 來擷取 AWS Organizations 根、OU、帳戶或政策的服務上次存取資訊。

檢視 AWS Organizations 服務上次存取的資訊 (AWS CLI)

1. 使用您的 AWS Organizations 管理帳戶登入資料搭配必要的 IAM 和 AWS Organizations 許可，並確認已為您的根目錄啟用 SCPs。如需詳細資訊，請參閱關於上次存取資訊的注意事項。
2. 產生報告。請求必須包含您想要報告之 AWS Organizations 實體 （ 根、OU 或帳戶 ） 的路徑。您可以選擇包含 `organization-policy-id` 參數，以檢視特定政策的報告。命令會傳回 `job-`

id，然後您可將它用於 get-organizations-access-report 命令以監控 job-status，直到任務完成為止。

- aws iam generate-organizations-access-report

3. 使用上個步驟的 job-id 參數來擷取報告的詳細資訊。

   - aws iam get-organizations-access-report

   此命令會傳回實體成員可以存取的服務清單。對於每個服務，命令會傳回帳戶成員上次嘗試存取的日期及時間，以及帳戶的實體路徑。它還會傳回可用來存取的服務數量，以及並未存取的服務數量。如果您指定選用的 organizations-policy-id 參數，則可存取的服務就是指定政策允許的服務。

檢視 AWS Organizations (AWS API) 的資訊

您可以使用 AWS API 來擷取根 AWS Organizations 、OU、帳戶或政策的服務上次存取資訊。

檢視 AWS Organizations 服務上次存取資訊 (AWS API)

1. 使用您的 AWS Organizations 管理帳戶登入資料搭配必要的 IAM 和 AWS Organizations 許可，並確認已為您的根目錄啟用 SCPs。如需詳細資訊，請參閱關於上次存取資訊的注意事項。
2. 產生報告。請求必須包含您想要報告之 AWS Organizations 實體（根、OU 或帳戶）的路徑。您可以選擇包含 OrganizationsPolicyId 參數，以檢視特定政策的報告。操作會傳回 JobId，您可將它用於 GetOrganizationsAccessReport 操作以監控 JobStatus，直到任務完成為止。

   - GenerateOrganizationsAccessReport

3. 使用上個步驟的 JobId 參數來擷取報告的詳細資訊。

   - GetOrganizationsAccessReport

   此操作會傳回實體成員可以存取的服務清單。對於每個服務，操作會傳回帳戶成員上次嘗試存取的日期及時間，以及帳戶的實體路徑。它還會傳回可用來存取的服務數量，以及並未存取的服務數量。如果您指定選用的 OrganizationsPolicyId 參數，則可存取的服務就是指定政策允許的服務。

使用上次存取資訊的範例案例

您可以使用上次存取的資訊，對您授予 IAM 實體的許可做出決策 AWS Organizations 。如需詳細資訊，請參閱AWS 使用上次存取的資訊在 中精簡許可。

> **Note**
>
> 在 IAM 或 中檢視實體或政策的存取資訊之前 AWS Organizations，請確定您了解資料的報告期間、報告的實體和評估的政策類型。如需詳細資訊，請參閱 the section called “關於上次存取資訊的注意事項”。

身為管理員的您，可以在可存取性與最低權限取得平衡，以符合您公司的需求。

使用資訊減少 IAM 群組的許可

您可以使用上次存取資訊以減少 IAM 群組許可，使其僅包含您的使用者所需要的服務。此方法在服務等級的授予最低權限中是一個重要的步驟。

例如，Paulo Santos 是負責定義 Example Corp. AWS 使用者許可的管理員。 此公司剛開始使用 AWS，軟體開發團隊尚未定義他們將使用 AWS 的服務。Paulo 打算僅提供該團隊所需服務的存取許可，但由於尚未定義相關服務，因此 Paulo 暫時提供該團隊進階使用者許可。然後，他會使用上次存取的資訊來減少群組的許可。

Paulo 使用以下 JSON 文字建立一個名為 `ExampleDevelopment` 的受管政策。然後，他將其連接至名為 `Development` 的群組，並將所有開發人員新增至該群組。

> **Note**
>
> Paulo 的進階使用者可能需要 `iam:CreateServiceLinkedRole` 許可才能使用某些服務和功能。他了解新增此許可會允許使用者建立任何服務連結的角色。他接受其進階使用者的這種風險。

JSON

```
{

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

```
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Paulo 決定在他檢視上次存取資訊前等待 90 天，然後讓 `Development` 群組使用 AWS 管理主控台。他檢視群組成員曾經存取的服務清單。他得知使用者在過去一週內存取了五個服務： AWS CloudTrail Amazon CloudWatch Logs AWS KMS、Amazon EC2 和 Amazon S3。他們在第一次評估時存取了其他一些服務 AWS，但之後卻無法存取。

Paulo 決定減少政策許可，只包含這五個服務以及必要的 IAM 和 AWS Organizations 動作。他使用以下 JSON 文字編輯 `ExampleDevelopment` 政策。

> **Note**
>
> Paulo 的進階使用者可能需要 `iam:CreateServiceLinkedRole` 許可才能使用某些服務和功能。他了解新增此許可會允許使用者建立任何服務連結的角色。他接受其進階使用者的這種風險。

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

若要進一步減少許可，Paulo 可在 AWS CloudTrail Event history (事件歷程記錄) 中檢視帳戶的事件。他可在此檢視詳細的事件資訊，以用於減少政策的許可，使其僅包含開發人員需要的動作和資源。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的在 CloudTrail 主控台中檢視 CloudTrail 事件。

使用資訊減少 IAM 使用者的許可

您可以使用上次存取資訊，以減少個別 IAM 使用者的許可。

例如，Martha Rivera 是 IT 管理員，負責確保其公司中的人員沒有多餘的 AWS 許可。在定期安全性檢查中，她會檢查所有 IAM 使用者的許可。在這些使用者中，有一位名為 Nikhil Jayashankar 的應用程式開發人員，過去曾經擔任安全工程師。因為任務需求的變更，Nikhil 同時是 `app-dev` 群組和 `security-team` 群組的成員。`app-dev` 群組為其新任務授予多項服務的許可，包括 Amazon EC2、Amazon EBS、Auto Scaling、Amazon S3、Route 53 和 Elastic Transcoder。他的舊任務的 `security-team` 群組授予 IAM 與 CloudTrail 的許可。

作為管理員，Martha 登入 IAM 主控台，依序選擇使用者、名稱 nikhilj，然後選擇上次存取索引標籤。

Martha 會檢閱上次存取的資料欄，並注意到 Nikhil 最近未存取 IAM、CloudTrail、Route 53、Amazon Elastic Transcoder 和許多 AWS 其他服務。Nikhil 已經存取 Amazon S3。Martha 從服務清單中選擇 S3，並得知 Nikhil 在過去兩週內執行了一些 Amazon S3 List 動作。Martha 確認 Nikhil 在她的公司中不再需要存取 IAM 與 CloudTrail，因為他不再是內部安全團隊的成員。

Martha 現在已經準備好對服務採取行動，並採取行動上次存取的資訊。不過，不同於先前範例中的群組，像是 nikhilj 這樣的 IAM 使用者可能會受到多個政策的約束，並且可能是多個群組的成員。Martha 必須小心處理以避免不慎中斷 nikhilj 或其他群組成員的存取。除了了解 Nikhil 應有哪些存取，她也必須判斷 Nikhil 應如何接收這些許可。

Martha 選擇 Permissions (許可) 標籤，她檢視哪些政策直接連接至 nikhilj，以及從群組連接的政策。她展開每個政策並檢視政策摘要，以了解哪個政策允許 Nikhil 存取他沒有在使用的服務：

- IAM – IAMFullAccess AWS 受管政策會直接連接至 nikhilj 群組並連接至 security-team群組。
- CloudTrail – AWS CloudTrailReadOnlyAccess AWS 受管政策會連接到 security-team群組。
- Route 53 – App-Dev-Route53 客戶受管政策連接至 app-dev 群組。
- Elastic Transcoder – App-Dev-ElasticTranscoder 客戶受管政策連接至 app-dev 群組。

Martha 決定移除直接連接到 的IAMFullAccess AWS 受管政策nikhilj。她也移除 Nikhil 的 security-team 群組成員資格。這兩個動作移除了不必要的 IAM 與 CloudTrail 存取。

Nikhil 存取 Route 53 和 Elastic Transcoder 的許可是由 app-dev 群組所授予。雖然 Nikhil 沒有使用這些服務，但群組的其他成員可能會使用。Martha 會檢閱 app-dev 群組上次存取的資訊，並得知多位成員最近存取 Route 53 和 Amazon S3。但在去年沒有任何群組成員存取過 Elastic Transcoder。她從群組移除 App-Dev-ElasticTranscoder 客戶受管政策。

然後，Martha 檢閱了 App-Dev-ElasticTranscoder 客戶受管政策的上次存取資訊。她發現該政策未連接至任何其他 IAM 身分。她在公司內部進行調查以確定未來不需要此政策，然後將此政策刪除。

刪除 IAM 資源前使用資訊

您可以在刪除 IAM 資源之前使用上次存取資訊，以確保在最後一次有人使用該資源之後已經過一段特定的時間。這適用於使用者、群組、角色及政策。若要進一步了解這些動作的詳細資訊，請參閱下列主題：

- IAM 使用者 – 移除或停用 IAM 使用者
- 群組 – 刪除 IAM 群組
- 角色 – 刪除角色或執行個體設定檔
- 政策 – 刪除 IAM 政策 (這也會從身分中分開政策)

編輯 IAM 政策前使用資訊

您可以在編輯會影響該資源的政策之前，檢閱上次存取資訊中的 IAM 身分 (使用者、群組或角色) 或 IAM 政策。這是重要的，因為您不會想要移除使用該政策者的存取權。

例如，Arnav Desai 是 Example Corp 的開發人員和 AWS 管理員。 當他的團隊開始使用 時 AWS，他們會授予所有開發人員進階使用者存取權，讓他們能夠完整存取 IAM 和 以外的所有服務 AWS Organizations。做為授予最低權限的第一步，Arnav 希望使用 AWS CLI 檢閱其帳戶中的受管政策。

因此，Arnav 首先列出其帳戶中連接至身分的客戶受管許可政策，他使用下列命令：

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

他從回應中擷取每個政策的 ARN。然後，Arnav 使用下列命令，為每個政策產生上次存取資訊的報告。

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

從回應中，他從 `JobId` 欄位擷取所產生報告的 ID。然後，Arnav 輪詢下列命令，直到 `JobStatus` 欄位傳回 `COMPLETED` 或 `FAILED` 值。如果任務失敗，他將會擷取錯誤。

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

當任務的狀態為 `COMPLETED` 時，Arnav 剖析 JSON 格式 `ServicesLastAccessed` 陣列的內容。

```
 "ServicesLastAccessed": [
        {
            "TotalAuthenticatedEntities": 1,
            "LastAuthenticated": 2018-11-01T21:24:33.222Z,
            "ServiceNamespace": "dynamodb",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
```

```
            "ServiceName": "Amazon DynamoDB"
        },

        {
            "TotalAuthenticatedEntities": 0,
            "ServiceNamespace": "ec2",
            "ServiceName": "Amazon EC2"
        },

        {
            "TotalAuthenticatedEntities": 3,
            "LastAuthenticated": 2018-08-25T15:29:51.156Z,
            "ServiceNamespace": "s3",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
            "ServiceName": "Amazon S3"
        }
    ]
```

Arnav 透過此資訊發現 `ExamplePolicy1` 政策允許存取三項服務、Amazon DynamoDB、Amazon S3 和 Amazon EC2。名為 `IAMExampleUser` 的 IAM 使用者與 11 月 1 日最後一次嘗試存取 DynamoDB，另有某人於 8 月 25 日使用了 `IAMExampleRole` 角色嘗試存取 Amazon S3。另有兩個實體在過去一年嘗試存取 Amazon S3。不過，過去一年無人嘗試存取 Amazon EC2。

這表示 Arnav 可以安全地從政策中移除 Amazon EC2 動作。Arnav 想要檢閱該政策目前的 JSON 文件。首先，他必須使用以下命令判斷政策的版本號碼。

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Arnav 從回應中的 `Versions` 陣列收集到目前的預設版本號碼。然後，他使用該版本號碼 (v2) 以及以下命令請求 JSON 政策文件。

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav 將傳回的 JSON 政策文件存放於 `Document` 陣列的 `PolicyVersion` 欄位。在政策文件中，Arnav 搜尋 `ec2` 命名空間中的動作。如果政策中沒有來自其他命名空間的動作，他將會分開政策與受影響的身分 (使用者、群組和角色)。而後他會刪除政策。在這種情況下，政策並包含 Amazon DynamoDB 與 Amazon S3 服務。因此，Arnav 會從文件中移除 Amazon EC2 動作，並儲存變更。然後，他使用下列命令來更新使用新文件版本的政策，然後將該版本設定為預設的政策版本。

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

ExamplePolicy1 政策現在已更新，以移除不必要 Amazon EC2 服務的存取權。

其他 IAM 案例

有關 IAM 資源 (使用者、群組、角色或政策) 上次嘗試存取服務時間的資訊，可在您完成下列任一項任務時提供協助：

- 政策 – 編輯現有的客戶受管政策或內嵌政策以移除許可
- 政策 – 將內嵌政策轉換為受管政策，然後刪除它
- 政策 – 將明確拒絕新增至現有的政策
- 政策 – 從身分 (使用者、群組或角色) 分開受管政策
- 實體 – 設定許可界限以控制實體 (使用者或角色) 可擁有的最大許可
- 群組 – 從群組移除使用者

使用資訊來調整組織單位的許可

您可以使用上次存取資訊，以強化 AWS Organizations中組織單位 (OU) 的許可。

例如，John Stiles 是 AWS Organizations 管理員。他負責確保公司中的人員 AWS 帳戶 沒有多餘的許可。在定期安全稽核中，他會檢查其組織的許可。他的 Development OU 包含帳戶，這類帳戶經常用於測試新的 AWS 服務。John 決定要定期檢查超過 180 天未存取的服務報告。然後，他的 OU 成員會移除存取那些服務的許可。

John 使用自己的管理帳戶憑證登入 IAM 主控台。在 IAM 主控台中，他找到 OU Development AWS Organizations 的資料。他檢閱服務存取報告表 AWS ，並看到兩個服務在超過他偏好的 180 天期間內未被存取。他記得為開發團隊新增存取 Amazon Lex 和 的許可 AWS Database Migration Service。John 聯絡開發團隊，並確認他們不再有測試這些服務的商業需求。

Martha 現在已經準備好對上次存取的資訊採取行動。他選擇 Edit in AWS Organizations (在 AWS Organizations中編輯)，而且收到提醒，表示已將 SCP 連接至多個實體。他選擇 Continue (繼續)。在中 AWS Organizations，他會檢閱目標，以了解 SCP 連接到哪些 AWS Organizations 實體。所有實體都位於 Development OU 內。

John 決定拒絕存取 Amazon Lex 和 NewServiceTest SCP 中的 AWS Database Migration Service 動作。這個動作移除了不必要的服務存取。

IAM 動作最近存取的資訊服務和動作

下表列出顯示 IAM 動作上次存取資訊 AWS 的服務。如需每個服務的動作清單，請參閱《服務授權參考》中的AWS 服務的動作、資源和條件索引鍵。

AWS 提供 JSON 格式的動作上次存取資訊，以簡化政策管理工作流程的自動化。使用服務參考資訊，您可以存取動作上次 AWS 服務 從機器可讀取檔案跨 存取的資訊。如需詳細資訊，請參閱 Service Authorization Reference 中的 Simplified AWS 服務 information for programmatic access。

服務	服務前綴
AWS Identity and Access Management 和 Access Analyzer	access-analyzer
AWS 帳戶管理	帳戶
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	mplify
AWS Amplify UI Builder	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS 應用程式成本分析器	application-cost-profiler
Amazon CloudWatch Application Insights	applicationinsights
AWS App Mesh	appmesh
Amazon WorkSpaces 應用程式	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps

服務	服務前綴
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	自動擴展
AWS Marketplace	aws-marketplace
AWS Backup	備份
AWS Batch	批次
Amazon Braket	braket
AWS Budgets	預算
AWS Cloud9	Cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
Amazon CodeGuru Reviewer	codeguru-reviewer
AWS CodePipeline	codepipeline

服務	服務前綴
AWS CodeStar	codestar
AWS CodeStar 通知	codestar-notifications
Amazon Cognito 身分	cognito-identity
Amazon Cognito 使用者集區	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect 客戶	connect
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms

服務	服務前綴
Amazon DocumentDB Elastic Clusters	docdb-elastic
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon EMR on EKS (EMR 容器)	emr-containers
Amazon EMR Serverless	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	events
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms

服務	服務前綴
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift 伺服器	gamelift
Amazon Location Service	geo
Amazon Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
AWS 身分存放區	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents

服務	服務前綴
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Amazon Interactive Video Service Chat	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex V2	lex
AWS License Manager Linux Subscriptions Manager	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	日誌
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2

服務	服務前綴
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	memorydb
AWS Transform MGN	mgn
AWS Migration Hub	mgh
AWS Migration Hub 策略建議	migrationhub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	組織
AWS Performance Insights	pi

服務	服務前綴
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Amazon Connect 客戶設定檔	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS 資源回收筒	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
Amazon Redshift 資料 API	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
AWS 資源總管	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Roles Anywhere	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness

服務	服務前綴
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	s3
Amazon S3 on Outposts	s3-outposts
Amazon SageMaker AI 地理空間功能	sagemaker-geospatial
Savings Plans	savingsplans
Amazon EventBridge Schemas	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub CSPM	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms

服務	服務前綴
Amazon Pinpoint 簡訊和語音服務	sms-voice
AWS Snowball Edge	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
適用於 SAP 的 AWS Systems Manager	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	標籤
Amazon Textract	textract
Amazon Timestream	timestream
AWS 電信網路建置器	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	傳輸
Amazon Translate	translate
連接客戶語音 ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2

服務	服務前綴
AWS Well-Architected Tool	wellarchitected
Amazon Connect Customer Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	工作區
AWS X-Ray	xray

用於動作最近存取資訊的動作

下表會列出可使用動作最近存取資訊的動作。

> **Important**
>
> `iam:UpdateAccountName` 動作將於 2026 年 4 月 22 日棄用。2026 年 4 月 22 日之後，僅 `account:PutAccountName` 許可能夠控制帳戶名稱更新存取權。我們強烈建議您更新所有控制帳戶名稱更新的服務控制政策 (SCP)，以便使用 `account:PutAccountName` 許可。

服務前綴	動作
access-analyzer	access-analyzer:ApplyArchiveRule
	access-analyzer:CancelPolicyGeneration
	access-analyzer:CheckAccessNotGranted
	access-analyzer:CheckNoNewAccess
	access-analyzer:CheckNoPublicAccess
	access-analyzer:CreateAccessPreview
	access-analyzer:CreateAnalyzer
	access-analyzer:CreateArchiveRule

服務前綴	動作
	access-analyzer:DeleteAnalyzer
	access-analyzer:DeleteArchiveRule
	access-analyzer:GenerateFindingRecommendation
	access-analyzer:GetAccessPreview
	access-analyzer:GetAnalyzedResource
	access-analyzer:GetAnalyzer
	access-analyzer:GetArchiveRule
	access-analyzer:GetFinding
	access-analyzer:GetFindingRecommendation
	access-analyzer:GetFindingsStatistics
	access-analyzer:GetGeneratedPolicy
	access-analyzer:ListAccessPreviewFindings
	access-analyzer:ListAccessPreviews
	access-analyzer:ListAnalyzedResources
	access-analyzer:ListAnalyzers
	access-analyzer:ListArchiveRules
	access-analyzer:ListFindings
	access-analyzer:ListPolicyGenerations
	access-analyzer:StartPolicyGeneration
	access-analyzer:StartResourceScan
	access-analyzer:UpdateAnalyzer

服務前綴	動作
	access-analyzer:UpdateArchiveRule access-analyzer:UpdateFindings access-analyzer:ValidatePolicy
帳戶	account:AcceptPrimaryEmailUpdate account:DeleteAlternateContact account:DisableRegion account:EnableRegion account:GetAccountInformation account:GetAlternateContact account:GetContactInformation account：GetGovCloudAccountInformation account:GetPrimaryEmail account:GetRegionOptStatus account:ListRegions account:PutAccountName account:PutAlternateContact account:PutContactInformation account:StartPrimaryEmailUpdate

服務前綴	動作
acm	acm：CreateAcmeEndpoint acm:DeleteCertificate acm:DescribeCertificate acm:ExportCertificate acm:GetAccountConfiguration acm:GetCertificate acm:ImportCertificate acm：ListAcmeEndpoints acm:ListCertificates acm:PutAccountConfiguration acm:RenewCertificate acm:RequestCertificate acm:ResendValidationEmail acm：SearchCertificates acm:UpdateCertificateOptions

服務前綴	動作
airflow	airflow:CreateCliToken airflow:CreateEnvironment airflow:CreateWebLoginToken airflow:DeleteEnvironment airflow:GetEnvironment airflow:ListEnvironments airflow:PublishMetrics airflow:UpdateEnvironment

服務前綴	動作
mplify	amplify:CreateApp
	amplify:CreateBackendEnvironment
	amplify:CreateBranch
	amplify:CreateDeployment
	amplify:CreateDomainAssociation
	amplify:CreateWebHook
	amplify:DeleteApp
	amplify:DeleteBackendEnvironment
	amplify:DeleteBranch
	amplify:DeleteDomainAssociation
	amplify:DeleteJob
	amplify:DeleteWebHook
	amplify:GenerateAccessLogs
	amplify:GetApp
	amplify:GetArtifactUrl
	amplify:GetBackendEnvironment
	amplify:GetBranch
	amplify:GetDomainAssociation
	amplify:GetJob
	amplify:GetWebHook
	amplify:ListApps

服務前綴	動作
	amplify:ListArtifacts amplify:ListBackendEnvironments amplify:ListBranches amplify:ListDomainAssociations amplify:ListJobs amplify:ListWebHooks amplify:StartDeployment amplify:StartJob amplify:StopJob amplify:UpdateApp amplify:UpdateBranch amplify:UpdateDomainAssociation amplify:UpdateWebHook

服務前綴	動作
amplifyuibuilder	amplifyuibuilder:CreateComponent amplifyuibuilder:CreateForm amplifyuibuilder:CreateTheme amplifyuibuilder:DeleteComponent amplifyuibuilder:DeleteForm amplifyuibuilder:DeleteTheme amplifyuibuilder:ExportComponents amplifyuibuilder:ExportThemes amplifyuibuilder:GetCodegenJob amplifyuibuilder:ListCodegenJobs amplifyuibuilder:ListComponents amplifyuibuilder:ListForms amplifyuibuilder:ListThemes amplifyuibuilder:ResetMetadataFlag amplifyuibuilder:StartCodegenJob amplifyuibuilder:UpdateComponent amplifyuibuilder:UpdateForm amplifyuibuilder:UpdateTheme

服務前綴	動作
app-integrations	app-integrations:CreateApplication app-integrations:CreateDataIntegration app-integrations:CreateDataIntegrationAssociation app-integrations:CreateEventIntegration app-integrations:DeleteApplication app-integrations:DeleteDataIntegration app-integrations:DeleteEventIntegration app-integrations:GetApplication app-integrations:GetDataIntegration app-integrations:GetEventIntegration app-integrations:ListApplicationAssociations app-integrations:ListApplications app-integrations:ListDataIntegrationAssociations app-integrations:ListDataIntegrations app-integrations:ListEventIntegrationAssociations app-integrations:ListEventIntegrations app-integrations:UpdateApplication app-integrations:UpdateDataIntegration app-integrations:UpdateDataIntegrationAssociation app-integrations:UpdateEventIntegration

服務前綴	動作
appconfig	appconfig:CreateApplication appconfig:CreateConfigurationProfile appconfig:CreateDeploymentStrategy appconfig:CreateEnvironment appconfig:CreateExtension appconfig:CreateExtensionAssociation appconfig:CreateHostedConfigurationVersion appconfig:DeleteApplication appconfig:DeleteConfigurationProfile appconfig:DeleteDeploymentStrategy appconfig:DeleteEnvironment appconfig:DeleteExtension appconfig:DeleteExtensionAssociation appconfig:DeleteHostedConfigurationVersion appconfig:GetAccountSettings appconfig:GetApplication appconfig:GetConfiguration appconfig:GetConfigurationProfile appconfig:GetDeployment appconfig:GetDeploymentStrategy appconfig:GetEnvironment

服務前綴	動作
	appconfig:GetExtension appconfig:GetExtensionAssociation appconfig:GetHostedConfigurationVersion appconfig:ListApplications appconfig:ListConfigurationProfiles appconfig:ListDeploymentStrategies appconfig:ListDeployments appconfig:ListEnvironments appconfig:ListExtensionAssociations appconfig:ListExtensions appconfig:ListHostedConfigurationVersions appconfig:StartDeployment appconfig:StopDeployment appconfig:UpdateAccountSettings appconfig:UpdateApplication appconfig:UpdateConfigurationProfile appconfig:UpdateDeploymentStrategy appconfig:UpdateEnvironment appconfig:UpdateExtension appconfig:UpdateExtensionAssociation appconfig:ValidateConfiguration

服務前綴	動作
appflow	appflow:CancelFlowExecutions
	appflow:CreateConnectorProfile
	appflow:CreateFlow
	appflow:DeleteConnectorProfile
	appflow:DeleteFlow
	appflow:DescribeConnector
	appflow:DescribeConnectorEntity
	appflow:DescribeConnectorProfiles
	appflow:DescribeConnectors
	appflow:DescribeFlow
	appflow:DescribeFlowExecutionRecords
	appflow:ListConnectorEntities
	appflow:ListConnectors
	appflow:ListFlows
	appflow:RegisterConnector
	appflow:ResetConnectorMetadataCache
	appflow:StartFlow
	appflow:StopFlow
	appflow:UnRegisterConnector
	appflow:UpdateConnectorProfile
	appflow:UpdateConnectorRegistration

服務前綴	動作
	appflow:UpdateFlow

服務前綴	動作
applicationinsights	applicationinsights:AddWorkload applicationinsights:CreateApplication applicationinsights:CreateComponent applicationinsights:CreateLogPattern applicationinsights:DeleteApplication applicationinsights:DeleteComponent applicationinsights:DeleteLogPattern applicationinsights:DescribeApplication applicationinsights:DescribeComponent applicationinsights:DescribeComponentConfiguration applicationinsights:DescribeComponentConfigurationRecommendation applicationinsights:DescribeLogPattern applicationinsights:DescribeObservation applicationinsights:DescribeProblem applicationinsights:DescribeProblemObservations applicationinsights:DescribeWorkload applicationinsights:ListApplications applicationinsights:ListComponents applicationinsights:ListConfigurationHistory applicationinsights:ListLogPatternSets

服務前綴	動作
	applicationinsights:ListLogPatterns applicationinsights:ListProblems applicationinsights:ListWorkloads applicationinsights:RemoveWorkload applicationinsights:UpdateApplication applicationinsights:UpdateComponent applicationinsights:UpdateComponentConfiguration applicationinsights:UpdateLogPattern applicationinsights:UpdateWorkload

服務前綴	動作
appmesh	appmesh:CreateGatewayRoute
	appmesh:CreateMesh
	appmesh:CreateRoute
	appmesh:CreateVirtualGateway
	appmesh:CreateVirtualNode
	appmesh:CreateVirtualRouter
	appmesh:CreateVirtualService
	appmesh:DeleteGatewayRoute
	appmesh:DeleteMesh
	appmesh:DeleteRoute
	appmesh:DeleteVirtualGateway
	appmesh:DeleteVirtualNode
	appmesh:DeleteVirtualRouter
	appmesh:DeleteVirtualService
	appmesh:DescribeGatewayRoute
	appmesh:DescribeMesh
	appmesh:DescribeRoute
	appmesh:DescribeVirtualGateway
	appmesh:DescribeVirtualNode
	appmesh:DescribeVirtualRouter
	appmesh:DescribeVirtualService

服務前綴	動作
	appmesh:ListGatewayRoutes
	appmesh:ListMeshes
	appmesh:ListRoutes
	appmesh:ListVirtualGateways
	appmesh:ListVirtualNodes
	appmesh:ListVirtualRouters
	appmesh:ListVirtualServices
	appmesh:StreamAggregatedResources
	appmesh:UpdateGatewayRoute
	appmesh:UpdateMesh
	appmesh:UpdateRoute
	appmesh:UpdateVirtualGateway
	appmesh:UpdateVirtualNode
	appmesh:UpdateVirtualRouter
	appmesh:UpdateVirtualService

服務前綴	動作
appstream	appstream:AssociateAppBlockBuilderAppBlock appstream:AssociateApplicationFleet appstream:AssociateApplicationToEntitlement appstream:AssociateFleet appstream：AssociateSoftwareToImageBuilder appstream:BatchAssociateUserStack appstream:BatchDisassociateUserStack appstream:CopyImage appstream:CreateAppBlock appstream:CreateAppBlockBuilder appstream:CreateAppBlockBuilderStreamingURL appstream:CreateApplication appstream:CreateDirectoryConfig appstream:CreateEntitlement appstream:CreateFleet appstream:CreateImageBuilder appstream:CreateImageBuilderStreamingURL appstream:CreateStack appstream:CreateStreamingURL appstream:CreateThemeForStack appstream:CreateUpdatedImage

服務前綴	動作
	appstream:CreateUsageReportSubscription appstream:CreateUser appstream:DeleteAppBlock appstream:DeleteAppBlockBuilder appstream:DeleteApplication appstream:DeleteDirectoryConfig appstream:DeleteEntitlement appstream:DeleteFleet appstream:DeleteImage appstream:DeleteImageBuilder appstream:DeleteImagePermissions appstream:DeleteStack appstream:DeleteThemeForStack appstream:DeleteUsageReportSubscription appstream:DeleteUser appstream:DescribeAppBlockBuilderAppBlockAssociations appstream:DescribeAppBlockBuilders appstream:DescribeAppBlocks appstream：DescribeAppLicenseUsage appstream:DescribeApplicationFleetAssociations appstream:DescribeApplications

服務前綴	動作
	appstream:DescribeDirectoryConfigs appstream:DescribeEntitlements appstream:DescribeFleets appstream:DescribeImageBuilders appstream:DescribeImagePermissions appstream:DescribeImages appstream:DescribeSessions appstream:DescribeStacks appstream:DescribeThemeForStack appstream:DescribeUsageReportSubscriptions appstream:DescribeUserStackAssociations appstream:DescribeUsers appstream:DisableUser appstream:DisassociateAppBlockBuilderAppBlock appstream:DisassociateApplicationFleet appstream:DisassociateApplicationFromEntitlement appstream:DisassociateFleet appstream：DisassociateSoftwareFromImageBuilder appstream:EnableUser appstream:ExpireSession appstream：GetExportImageTask

服務前綴	動作
	appstream:ListAssociatedFleets appstream:ListAssociatedStacks appstream:ListEntitledApplications appstream：ListExportImageTasks appstream:StartAppBlockBuilder appstream:StartFleet appstream:StartImageBuilder appstream：StartSoftwareDeploymentToImageBuilder appstream:StopAppBlockBuilder appstream:StopFleet appstream:StopImageBuilder appstream:UpdateAppBlockBuilder appstream:UpdateApplication appstream:UpdateDirectoryConfig appstream:UpdateEntitlement appstream:UpdateFleet appstream:UpdateImagePermissions appstream:UpdateStack appstream:UpdateThemeForStack

服務前綴	動作
appsync	appsync:AssociateApi
	appsync:AssociateMergedGraphqlApi
	appsync:AssociateSourceGraphqlApi
	appsync:CreateApi
	appsync:CreateApiCache
	appsync:CreateApiKey
	appsync:CreateChannelNamespace
	appsync:CreateDataSource
	appsync:CreateDomainName
	appsync:CreateFunction
	appsync:CreateGraphqlApi
	appsync:CreateResolver
	appsync:CreateType
	appsync:DeleteApi
	appsync:DeleteApiCache
	appsync:DeleteApiKey
	appsync:DeleteChannelNamespace
	appsync:DeleteDataSource
	appsync:DeleteDomainName
	appsync:DeleteFunction
	appsync:DeleteGraphqlApi

服務前綴	動作
	appsync:DeleteResolver
	appsync:DeleteType
	appsync:DisassociateApi
	appsync:DisassociateMergedGraphqlApi
	appsync:DisassociateSourceGraphqlApi
	appsync:EvaluateCode
	appsync:EvaluateMappingTemplate
	appsync:FlushApiCache
	appsync:GetApi
	appsync:GetApiAssociation
	appsync:GetApiCache
	appsync:GetChannelNamespace
	appsync:GetDataSource
	appsync:GetDataSourceIntrospection
	appsync:GetDomainName
	appsync:GetFunction
	appsync:GetGraphqlApi
	appsync:GetGraphqlApiEnvironmentVariables
	appsync:GetIntrospectionSchema
	appsync:GetResolver
	appsync:GetSchemaCreationStatus

服務前綴	動作
	appsync:GetSourceApiAssociation
	appsync:GetType
	appsync:ListApiKeys
	appsync:ListApis
	appsync:ListChannelNamespaces
	appsync:ListDataSources
	appsync:ListDomainNames
	appsync:ListFunctions
	appsync:ListGraphqlApis
	appsync:ListResolvers
	appsync:ListResolversByFunction
	appsync:ListSourceApiAssociations
	appsync:ListTypes
	appsync:ListTypesByAssociation
	appsync:PutGraphqlApiEnvironmentVariables
	appsync:StartDataSourceIntrospection
	appsync:StartSchemaCreation
	appsync:StartSchemaMerge
	appsync:UpdateApi
	appsync:UpdateApiCache
	appsync:UpdateApiKey

服務前綴	動作
	appsync:UpdateChannelNamespace appsync:UpdateDataSource appsync:UpdateDomainName appsync:UpdateFunction appsync:UpdateGraphqlApi appsync:UpdateResolver appsync:UpdateSourceApiAssociation appsync:UpdateType

服務前綴	動作
aps	aps:CreateAlertManagerDefinition aps：CreateAnomalyDetector aps:CreateLoggingConfiguration aps:CreateQueryLoggingConfiguration aps:CreateRuleGroupsNamespace aps:CreateWorkspace aps:DeleteAlertManagerDefinition aps：DeleteAnomalyDetector aps:DeleteLoggingConfiguration aps:DeleteQueryLoggingConfiguration aps:DeleteResourcePolicy aps:DeleteRuleGroupsNamespace aps:DeleteScraper aps：DeleteScraperLoggingConfiguration aps:DeleteWorkspace aps:DescribeAlertManagerDefinition aps：DescribeAnomalyDetector aps:DescribeLoggingConfiguration aps:DescribeQueryLoggingConfiguration aps:DescribeResourcePolicy aps:DescribeRuleGroupsNamespace

服務前綴	動作
	aps:DescribeScraper aps：DescribeScraperLoggingConfiguration aps:DescribeWorkspace aps:DescribeWorkspaceConfiguration aps:GetDefaultScraperConfiguration aps：ListAnomalyDetectors aps:ListRuleGroupsNamespaces aps:ListScrapers aps:ListWorkspaces aps:PutAlertManagerDefinition aps：PutAnomalyDetector aps:PutResourcePolicy aps:PutRuleGroupsNamespace aps:UpdateLoggingConfiguration aps:UpdateQueryLoggingConfiguration aps:UpdateScraper aps：UpdateScraperLoggingConfiguration aps:UpdateWorkspaceAlias aps:UpdateWorkspaceConfiguration

服務前綴	動作
athena	athena:BatchGetNamedQuery
	athena:BatchGetPreparedStatement
	athena:BatchGetQueryExecution
	athena:CancelCapacityReservation
	athena:CreateCapacityReservation
	athena:CreateDataCatalog
	athena:CreateNamedQuery
	athena:CreateNotebook
	athena:CreatePreparedStatement
	athena:CreatePresignedNotebookUrl
	athena:CreateWorkGroup
	athena:DeleteCapacityReservation
	athena:DeleteDataCatalog
	athena:DeleteNamedQuery
	athena:DeleteNotebook
	athena:DeletePreparedStatement
	athena:DeleteWorkGroup
	athena:ExportNotebook
	athena:GetCalculationExecution
	athena:GetCalculationExecutionCode
	athena:GetCalculationExecutionStatus

服務前綴	動作
	athena:GetCapacityAssignmentConfiguration
	athena:GetCapacityReservation
	athena:GetDataCatalog
	athena:GetDatabase
	athena:GetNamedQuery
	athena:GetNotebookMetadata
	athena:GetPreparedStatement
	athena:GetQueryExecution
	athena:GetQueryResults
	athena:GetQueryResultsStream
	athena:GetQueryRuntimeStatistics
	athena：GetResourceDashboard
	athena:GetSession
	athena：GetSessionEndpoint
	athena:GetSessionStatus
	athena:GetTableMetadata
	athena:GetWorkGroup
	athena:ImportNotebook
	athena:ListApplicationDPUSizes
	athena:ListCalculationExecutions
	athena:ListCapacityReservations

服務前綴	動作
	athena:ListDataCatalogs
	athena:ListDatabases
	athena:ListEngineVersions
	athena:ListExecutors
	athena:ListNamedQueries
	athena:ListNotebookMetadata
	athena:ListNotebookSessions
	athena:ListPreparedStatements
	athena:ListQueryExecutions
	athena:ListSessions
	athena:ListTableMetadata
	athena:ListWorkGroups
	athena:PutCapacityAssignmentConfiguration
	athena:StartCalculationExecution
	athena:StartQueryExecution
	athena:StartSession
	athena:StopCalculationExecution
	athena:StopQueryExecution
	athena:TerminateSession
	athena:UpdateCapacityReservation
	athena:UpdateDataCatalog

服務前綴	動作
	athena:UpdateNamedQuery
	athena:UpdateNotebook
	athena:UpdateNotebookMetadata
	athena:UpdatePreparedStatement
	athena:UpdateWorkGroup

服務前綴	動作
auditmanager	auditmanager:AssociateAssessmentReportEvidenceFolder auditmanager:BatchAssociateAssessmentReportEvidence auditmanager:BatchCreateDelegationByAssessment auditmanager:BatchDeleteDelegationByAssessment auditmanager:BatchDisassociateAssessmentReportEvidence auditmanager:BatchImportEvidenceToAssessmentControl auditmanager:CreateAssessment auditmanager:CreateAssessmentFramework auditmanager:CreateAssessmentReport auditmanager:CreateControl auditmanager:DeleteAssessment auditmanager:DeleteAssessmentFramework auditmanager:DeleteAssessmentFrameworkShare auditmanager:DeleteAssessmentReport auditmanager:DeleteControl auditmanager:DeregisterAccount auditmanager:DeregisterOrganizationAdminAccount auditmanager:DisassociateAssessmentReportEvidenceFolder auditmanager:GetAccountStatus auditmanager:GetAssessment auditmanager:GetAssessmentFramework

服務前綴	動作
	auditmanager:GetAssessmentReportUrl auditmanager:GetChangeLogs auditmanager:GetControl auditmanager:GetDelegations auditmanager:GetEvidence auditmanager:GetEvidenceByEvidenceFolder auditmanager:GetEvidenceFileUploadUrl auditmanager:GetEvidenceFolder auditmanager:GetEvidenceFoldersByAssessment auditmanager:GetEvidenceFoldersByAssessmentControl auditmanager:GetInsights auditmanager:GetInsightsByAssessment auditmanager:GetOrganizationAdminAccount auditmanager:GetServicesInScope auditmanager:GetSettings auditmanager:ListAssessmentControlInsightsByControlDomain auditmanager:ListAssessmentFrameworkShareRequests auditmanager:ListAssessmentFrameworks auditmanager:ListAssessmentReports auditmanager:ListAssessments auditmanager:ListControlDomainInsights

服務前綴	動作
	auditmanager:ListControlDomainInsightsByAssessment auditmanager:ListControlInsightsByControlDomain auditmanager:ListControls auditmanager:ListKeywordsForDataSource auditmanager:ListNotifications auditmanager:RegisterAccount auditmanager:RegisterOrganizationAdminAccount auditmanager:StartAssessmentFrameworkShare auditmanager:UpdateAssessment auditmanager:UpdateAssessmentControl auditmanager:UpdateAssessmentControlSetStatus auditmanager:UpdateAssessmentFramework auditmanager:UpdateAssessmentFrameworkShare auditmanager:UpdateAssessmentStatus auditmanager:UpdateControl auditmanager:UpdateSettings auditmanager:ValidateAssessmentReportIntegrity

服務前綴	動作
自動擴展	autoscaling:AttachInstances
	autoscaling:AttachLoadBalancerTargetGroups
	autoscaling:AttachLoadBalancers
	autoscaling:AttachTrafficSources
	autoscaling:BatchDeleteScheduledAction
	autoscaling:BatchPutScheduledUpdateGroupAction
	autoscaling:CancelInstanceRefresh
	autoscaling:CompleteLifecycleAction
	autoscaling:CreateAutoScalingGroup
	autoscaling:CreateLaunchConfiguration
	autoscaling:DeleteAutoScalingGroup
	autoscaling:DeleteLaunchConfiguration
	autoscaling:DeleteLifecycleHook
	autoscaling:DeleteNotificationConfiguration
	autoscaling:DeletePolicy
	autoscaling:DeleteScheduledAction
	autoscaling:DeleteWarmPool
	autoscaling:DescribeAccountLimits
	autoscaling:DescribeAdjustmentTypes
	autoscaling:DescribeAutoScalingGroups
	autoscaling:DescribeAutoScalingInstances

服務前綴	動作
	autoscaling:DescribeAutoScalingNotificationTypes
	autoscaling:DescribeInstanceRefreshes
	autoscaling:DescribeLaunchConfigurations
	autoscaling:DescribeLifecycleHookTypes
	autoscaling:DescribeLifecycleHooks
	autoscaling:DescribeLoadBalancerTargetGroups
	autoscaling:DescribeLoadBalancers
	autoscaling:DescribeMetricCollectionTypes
	autoscaling:DescribeNotificationConfigurations
	autoscaling:DescribePolicies
	autoscaling:DescribeScalingActivities
	autoscaling:DescribeScalingProcessTypes
	autoscaling:DescribeScheduledActions
	autoscaling:DescribeTerminationPolicyTypes
	autoscaling:DescribeTrafficSources
	autoscaling:DescribeWarmPool
	autoscaling:DetachInstances
	autoscaling:DetachLoadBalancerTargetGroups
	autoscaling:DetachLoadBalancers
	autoscaling:DetachTrafficSources
	autoscaling:DisableMetricsCollection

服務前綴	動作
	autoscaling:EnableMetricsCollection autoscaling:EnterStandby autoscaling:ExecutePolicy autoscaling:ExitStandby autoscaling:GetPredictiveScalingForecast autoscaling:PutLifecycleHook autoscaling:PutNotificationConfiguration autoscaling:PutScalingPolicy autoscaling:PutScheduledUpdateGroupAction autoscaling:PutWarmPool autoscaling:RecordLifecycleActionHeartbeat autoscaling:ResumeProcesses autoscaling:RollbackInstanceRefresh autoscaling:SetDesiredCapacity autoscaling:SetInstanceHealth autoscaling:SetInstanceProtection autoscaling:StartInstanceRefresh autoscaling:SuspendProcesses autoscaling:TerminateInstanceInAutoScalingGroup autoscaling:UpdateAutoScalingGroup
aws-marketplace	aws-marketplace:GetEntitlements

服務前綴	動作
備份	backup:CancelLegalHold backup:CreateBackupPlan backup:CreateBackupSelection backup:CreateBackupVault backup:CreateFramework backup:CreateLegalHold backup:CreateLogicallyAirGappedBackupVault backup:CreateReportPlan backup:CreateRestoreAccessBackupVault backup:CreateRestoreTestingPlan backup:CreateRestoreTestingSelection 備份：CreateTieringConfiguration backup:DeleteBackupPlan backup:DeleteBackupSelection backup:DeleteBackupVault backup:DeleteBackupVaultAccessPolicy backup:DeleteBackupVaultLockConfiguration backup:DeleteBackupVaultNotifications backup:DeleteFramework backup:DeleteRecoveryPoint backup:DeleteReportPlan

服務前綴	動作
	backup:DeleteRestoreTestingPlan
	backup:DeleteRestoreTestingSelection
	備份：DeleteTieringConfiguration
	backup:DescribeBackupJob
	backup:DescribeBackupVault
	backup:DescribeCopyJob
	backup:DescribeFramework
	backup:DescribeGlobalSettings
	backup:DescribeProtectedResource
	backup:DescribeRecoveryPoint
	backup:DescribeRegionSettings
	backup:DescribeReportJob
	backup:DescribeReportPlan
	backup:DescribeRestoreJob
	備份：DescribeScanJob
	backup:DisassociateRecoveryPoint
	backup:DisassociateRecoveryPointFromParent
	backup:ExportBackupPlanTemplate
	backup:GetBackupPlan
	backup:GetBackupPlanFromJSON
	backup:GetBackupPlanFromTemplate

服務前綴	動作
	backup:GetBackupSelection
	backup:GetBackupVaultAccessPolicy
	backup:GetBackupVaultNotifications
	backup:GetLegalHold
	backup:GetRecoveryPointRestoreMetadata
	backup:GetRestoreJobMetadata
	backup:GetRestoreTestingInferredMetadata
	backup:GetRestoreTestingPlan
	backup:GetRestoreTestingSelection
	backup:GetSupportedResourceTypes
	備份：GetTieringConfiguration
	backup:ListBackupJobSummaries
	backup:ListBackupJobs
	backup:ListBackupPlanTemplates
	backup:ListBackupPlanVersions
	backup:ListBackupPlans
	backup:ListBackupSelections
	backup:ListBackupVaults
	backup:ListCopyJobSummaries
	backup:ListCopyJobs
	backup:ListFrameworks

服務前綴	動作
	backup:ListIndexedRecoveryPoints
	backup:ListLegalHolds
	backup:ListProtectedResources
	backup:ListRecoveryPointsByBackupVault
	backup:ListRecoveryPointsByLegalHold
	backup:ListRecoveryPointsByResource
	backup:ListReportJobs
	backup:ListReportPlans
	backup:ListRestoreAccessBackupVaults
	backup:ListRestoreJobSummaries
	backup:ListRestoreJobs
	backup:ListRestoreJobsByProtectedResource
	backup:ListRestoreTestingPlans
	backup:ListRestoreTestingSelections
	備份：ListScanJobSummaries
	備份：ListScanJobs
	備份：ListTieringConfigurations
	backup:PutBackupVaultAccessPolicy
	backup:PutBackupVaultLockConfiguration
	backup:PutBackupVaultNotifications
	backup:PutRestoreValidationResult

服務前綴	動作
	backup:StartBackupJob
	backup:StartCopyJob
	backup:StartReportJob
	backup:StartRestoreJob
	backup:StopBackupJob
	backup:UpdateBackupPlan
	backup:UpdateFramework
	backup:UpdateGlobalSettings
	backup:UpdateRecoveryPointLifecycle
	backup:UpdateRegionSettings
	backup:UpdateReportPlan
	backup:UpdateRestoreTestingPlan
	backup:UpdateRestoreTestingSelection
	備份：UpdateTieringConfiguration

服務前綴	動作
批次	batch:CancelJob batch:CreateComputeEnvironment batch:CreateConsumableResource batch:CreateJobQueue batch：CreateQuotaShare batch:CreateSchedulingPolicy batch:CreateServiceEnvironment batch:DeleteComputeEnvironment batch:DeleteConsumableResource batch:DeleteJobQueue batch：DeleteQuotaShare batch:DeleteSchedulingPolicy batch:DeleteServiceEnvironment batch:DeregisterJobDefinition batch:DescribeComputeEnvironments batch:DescribeConsumableResource batch:DescribeJobDefinitions batch:DescribeJobQueues batch:DescribeJobs batch：DescribeQuotaShare batch:DescribeSchedulingPolicies

服務前綴	動作
	batch:DescribeServiceEnvironments batch:DescribeServiceJob batch:GetJobQueueSnapshot batch:ListConsumableResources batch:ListJobs batch:ListJobsByConsumableResource batch：ListQuotaShares batch:ListSchedulingPolicies batch:ListServiceJobs batch:RegisterJobDefinition batch:SubmitJob batch:SubmitServiceJob batch:TerminateJob batch:TerminateServiceJob batch:UpdateComputeEnvironment batch:UpdateConsumableResource batch:UpdateJobQueue batch：UpdateQuotaShare batch:UpdateSchedulingPolicy batch:UpdateServiceEnvironment batch：UpdateServiceJob

服務前綴	動作
braket	braket:CancelJob braket:CancelQuantumTask braket:CreateJob braket:CreateQuantumTask braket：CreateSpendingLimit braket:GetDevice braket:GetJob braket:GetQuantumTask braket:SearchDevices braket:SearchJobs braket:SearchQuantumTasks braket：SearchSpendingLimits

服務前綴	動作
預算	budgets:CreateBudgetAction budgets:DeleteBudgetAction budgets:DescribeBudgetAction budgets:DescribeBudgetActionHistories budgets:DescribeBudgetActionsForAccount budgets:DescribeBudgetActionsForBudget budgets:ExecuteBudgetAction budgets:ModifyBudget budgets:UpdateBudgetAction budgets:ViewBudget
Cloud9	cloud9:CreateEnvironmentEC2 cloud9:CreateEnvironmentMembership cloud9:DeleteEnvironment cloud9:DeleteEnvironmentMembership cloud9:DescribeEnvironmentMemberships cloud9:DescribeEnvironmentStatus cloud9:DescribeEnvironments cloud9:ListEnvironments cloud9:UpdateEnvironment cloud9:UpdateEnvironmentMembership

服務前綴	動作
cloudformation	cloudformation:BatchDescribeTypeConfigurations cloudformation:CancelUpdateStack cloudformation:ContinueUpdateRollback cloudformation:CreateChangeSet cloudformation:CreateGeneratedTemplate cloudformation:CreateStack cloudformation:CreateStackInstances cloudformation:CreateStackSet cloudformation:DeactivateType cloudformation:DeleteChangeSet cloudformation:DeleteGeneratedTemplate cloudformation:DeleteStack cloudformation:DeleteStackInstances cloudformation:DeleteStackSet cloudformation:DeregisterType cloudformation:DescribeAccountLimits cloudformation:DescribeChangeSet cloudformation:DescribeChangeSetHooks cloudformation：DescribeEvents cloudformation:DescribeGeneratedTemplate cloudformation:DescribeOrganizationsAccess

服務前綴	動作
	cloudformation:DescribePublisher cloudformation:DescribeResourceScan cloudformation:DescribeStackDriftDetectionStatus cloudformation:DescribeStackEvents cloudformation:DescribeStackInstance cloudformation:DescribeStackResource cloudformation:DescribeStackResourceDrifts cloudformation:DescribeStackResources cloudformation:DescribeStackSet cloudformation:DescribeStackSetOperation cloudformation:DescribeStacks cloudformation:DescribeType cloudformation:DescribeTypeRegistration cloudformation:DetectStackDrift cloudformation:DetectStackResourceDrift cloudformation:DetectStackSetDrift cloudformation:EstimateTemplateCost cloudformation:ExecuteChangeSet cloudformation:GetGeneratedTemplate cloudformation：GetHookResult cloudformation:GetStackPolicy

服務前綴	動作
	cloudformation:GetTemplate cloudformation:GetTemplateSummary cloudformation:ImportStacksToStackSet cloudformation:ListChangeSets cloudformation:ListExports cloudformation:ListGeneratedTemplates cloudformation:ListHookResults cloudformation:ListImports cloudformation:ListResourceScanRelatedResources cloudformation:ListResourceScanResources cloudformation:ListResourceScans cloudformation:ListStackInstanceResourceDrifts cloudformation:ListStackInstances cloudformation:ListStackRefactors cloudformation:ListStackResources cloudformation:ListStackSetAutoDeploymentTargets cloudformation:ListStackSetOperationResults cloudformation:ListStackSetOperations cloudformation:ListStackSets cloudformation:ListTypeRegistrations cloudformation:ListTypeVersions

服務前綴	動作
	cloudformation:ListTypes cloudformation:PublishType cloudformation:RecordHandlerProgress cloudformation:RegisterPublisher cloudformation:RegisterType cloudformation:RollbackStack cloudformation:SetStackPolicy cloudformation:SetTypeConfiguration cloudformation:SetTypeDefaultVersion cloudformation:SignalResource cloudformation:StartResourceScan cloudformation:StopStackSetOperation cloudformation:TestType cloudformation:UpdateGeneratedTemplate cloudformation:UpdateStack cloudformation:UpdateStackInstances cloudformation:UpdateStackSet cloudformation:UpdateTerminationProtection cloudformation:ValidateTemplate

服務前綴	動作
cloudfront	cloudfront:AssociateAlias
	cloudfront:AssociateDistributionTenantWebACL
	cloudfront:AssociateDistributionWebACL
	cloudfront:CreateCachePolicy
	cloudfront:CreateCloudFrontOriginAccessIdentity
	cloudfront：CreateConnectionFunction
	cloudfront:CreateContinuousDeploymentPolicy
	cloudfront:CreateDistributionTenant
	cloudfront:CreateFieldLevelEncryptionConfig
	cloudfront:CreateFieldLevelEncryptionProfile
	cloudfront:CreateFunction
	cloudfront:CreateInvalidation
	cloudfront:CreateKeyGroup
	cloudfront:CreateKeyValueStore
	cloudfront:CreateMonitoringSubscription
	cloudfront:CreateOriginAccessControl
	cloudfront:CreateOriginRequestPolicy
	cloudfront:CreatePublicKey
	cloudfront:CreateRealtimeLogConfig
	cloudfront:CreateResponseHeadersPolicy
	cloudfront：CreateTrustStore

服務前綴	動作
	cloudfront:DeleteAnycastIpList
	cloudfront:DeleteCachePolicy
	cloudfront:DeleteCloudFrontOriginAccessIdentity
	cloudfront：DeleteConnectionFunction
	cloudfront:DeleteConnectionGroup
	cloudfront:DeleteContinuousDeploymentPolicy
	cloudfront:DeleteDistribution
	cloudfront:DeleteDistributionTenant
	cloudfront:DeleteFieldLevelEncryptionConfig
	cloudfront:DeleteFieldLevelEncryptionProfile
	cloudfront:DeleteFunction
	cloudfront:DeleteKeyGroup
	cloudfront:DeleteKeyValueStore
	cloudfront:DeleteMonitoringSubscription
	cloudfront:DeleteOriginAccessControl
	cloudfront:DeleteOriginRequestPolicy
	cloudfront:DeletePublicKey
	cloudfront:DeleteRealtimeLogConfig
	cloudfront:DeleteResponseHeadersPolicy
	cloudfront:DeleteStreamingDistribution
	cloudfront：DeleteTrustStore

服務前綴	動作
	cloudfront:DeleteVpcOrigin
	cloudfront:DescribeFunction
	cloudfront:DescribeKeyValueStore
	cloudfront:DisassociateDistributionTenantWebACL
	cloudfront:DisassociateDistributionWebACL
	cloudfront:GetAnycastIpList
	cloudfront:GetCachePolicy
	cloudfront:GetCachePolicyConfig
	cloudfront:GetCloudFrontOriginAccessIdentity
	cloudfront:GetCloudFrontOriginAccessIdentityConfig
	cloudfront:GetContinuousDeploymentPolicy
	cloudfront:GetContinuousDeploymentPolicyConfig
	cloudfront:GetDistributionConfig
	cloudfront:GetFieldLevelEncryption
	cloudfront:GetFieldLevelEncryptionConfig
	cloudfront:GetFieldLevelEncryptionProfile
	cloudfront:GetFieldLevelEncryptionProfileConfig
	cloudfront:GetFunction
	cloudfront:GetInvalidation
	cloudfront:GetInvalidationForDistributionTenant
	cloudfront:GetKeyGroup

服務前綴	動作
	cloudfront:GetKeyGroupConfig
	cloudfront:GetMonitoringSubscription
	cloudfront:GetOriginAccessControl
	cloudfront:GetOriginAccessControlConfig
	cloudfront:GetOriginRequestPolicy
	cloudfront:GetOriginRequestPolicyConfig
	cloudfront:GetPublicKey
	cloudfront:GetPublicKeyConfig
	cloudfront:GetRealtimeLogConfig
	cloudfront:GetResponseHeadersPolicy
	cloudfront:GetResponseHeadersPolicyConfig
	cloudfront:GetStreamingDistribution
	cloudfront:GetStreamingDistributionConfig
	cloudfront:GetVpcOrigin
	cloudfront:ListAnycastIpLists
	cloudfront:ListCachePolicies
	cloudfront:ListCloudFrontOriginAccessIdentities
	cloudfront:ListConflictingAliases
	cloudfront：ListConnectionFunctions
	cloudfront:ListConnectionGroups
	cloudfront:ListContinuousDeploymentPolicies

服務前綴	動作
	cloudfront:ListDistributionTenants
	cloudfront:ListDistributionTenantsByCustomization
	cloudfront:ListDistributions
	cloudfront:ListDistributionsByAnycastIpListId
	cloudfront:ListDistributionsByCachePolicyId
	cloudfront:ListDistributionsByConnectionMode
	cloudfront:ListDistributionsByKeyGroup
	cloudfront:ListDistributionsByOriginRequestPolicyId
	cloudfront:ListDistributionsByRealtimeLogConfig
	cloudfront:ListDistributionsByResponseHeadersPolicyId
	cloudfront:ListDistributionsByVpcOriginId
	cloudfront:ListDistributionsByWebACLId
	cloudfront:ListFieldLevelEncryptionConfigs
	cloudfront:ListFieldLevelEncryptionProfiles
	cloudfront:ListFunctions
	cloudfront:ListInvalidations
	cloudfront:ListInvalidationsForDistributionTenant
	cloudfront:ListKeyGroups
	cloudfront:ListKeyValueStores
	cloudfront:ListOriginAccessControls
	cloudfront:ListOriginRequestPolicies

服務前綴	動作
	cloudfront:ListPublicKeys
	cloudfront:ListRealtimeLogConfigs
	cloudfront:ListResponseHeadersPolicies
	cloudfront:ListStreamingDistributions
	cloudfront：ListTrustStores
	cloudfront：PublishConnectionFunction
	cloudfront:PublishFunction
	cloudfront：TestConnectionFunction
	cloudfront:TestFunction
	cloudfront：UpdateAnycastIpList
	cloudfront:UpdateCachePolicy
	cloudfront:UpdateCloudFrontOriginAccessIdentity
	cloudfront：UpdateConnectionFunction
	cloudfront:UpdateConnectionGroup
	cloudfront:UpdateContinuousDeploymentPolicy
	cloudfront:UpdateDistribution
	cloudfront:UpdateDistributionTenant
	cloudfront:UpdateFieldLevelEncryptionConfig
	cloudfront:UpdateFieldLevelEncryptionProfile
	cloudfront:UpdateFunction
	cloudfront:UpdateKeyGroup

服務前綴	動作
	cloudfront:UpdateKeyValueStore cloudfront:UpdateOriginAccessControl cloudfront:UpdateOriginRequestPolicy cloudfront:UpdatePublicKey cloudfront:UpdateRealtimeLogConfig cloudfront:UpdateResponseHeadersPolicy cloudfront：UpdateTrustStore
cloudhsm	cloudhsm:CreateHsm cloudhsm:DeleteBackup cloudhsm:DeleteHsm cloudhsm:DeleteResourcePolicy cloudhsm:DescribeBackups cloudhsm:DescribeClusters cloudhsm:GetResourcePolicy cloudhsm:InitializeCluster cloudhsm:ModifyBackupAttributes cloudhsm:ModifyCluster cloudhsm:PutResourcePolicy cloudhsm:RestoreBackup

服務前綴	動作
cloudsearch	cloudsearch:BuildSuggesters
	cloudsearch:CreateDomain
	cloudsearch:DefineAnalysisScheme
	cloudsearch:DefineExpression
	cloudsearch:DefineIndexField
	cloudsearch:DefineSuggester
	cloudsearch:DeleteAnalysisScheme
	cloudsearch:DeleteDomain
	cloudsearch:DeleteExpression
	cloudsearch:DeleteIndexField
	cloudsearch:DeleteSuggester
	cloudsearch:DescribeAnalysisSchemes
	cloudsearch:DescribeAvailabilityOptions
	cloudsearch:DescribeDomainEndpointOptions
	cloudsearch:DescribeDomains
	cloudsearch:DescribeExpressions
	cloudsearch:DescribeIndexFields
	cloudsearch:DescribeScalingParameters
	cloudsearch:DescribeServiceAccessPolicies
	cloudsearch:DescribeSuggesters
	cloudsearch:IndexDocuments

服務前綴	動作
	cloudsearch:ListDomainNames
	cloudsearch:UpdateAvailabilityOptions
	cloudsearch:UpdateDomainEndpointOptions
	cloudsearch:UpdateScalingParameters
	cloudsearch:UpdateServiceAccessPolicies

服務前綴	動作
cloudtrail	cloudtrail:CancelQuery cloudtrail:CreateChannel cloudtrail:CreateDashboard cloudtrail:CreateEventDataStore cloudtrail:CreateTrail cloudtrail:DeleteChannel cloudtrail:DeleteDashboard cloudtrail:DeleteEventDataStore cloudtrail:DeleteResourcePolicy cloudtrail:DeleteTrail cloudtrail:DeregisterOrganizationDelegatedAdmin cloudtrail:DescribeQuery cloudtrail:DescribeTrails cloudtrail:DisableFederation cloudtrail:GenerateQuery cloudtrail:GetChannel cloudtrail:GetDashboard cloudtrail:GetEventConfiguration cloudtrail:GetEventDataStore cloudtrail:GetEventDataStoreData cloudtrail:GetEventSelectors

服務前綴	動作
	cloudtrail:GetImport cloudtrail:GetInsightSelectors cloudtrail:GetResourcePolicy cloudtrail:GetTrail cloudtrail:GetTrailStatus cloudtrail:ListChannels cloudtrail:ListDashboards cloudtrail:ListEventDataStores cloudtrail:ListImportFailures cloudtrail:ListImports cloudtrail：ListInsightsData cloudtrail:ListPublicKeys cloudtrail:ListQueries cloudtrail:ListTrails cloudtrail:LookupEvents cloudtrail:PutEventConfiguration cloudtrail:PutEventSelectors cloudtrail:PutInsightSelectors cloudtrail:PutResourcePolicy cloudtrail:RegisterOrganizationDelegatedAdmin cloudtrail:RestoreEventDataStore

服務前綴	動作
	cloudtrail:SearchSampleQueries cloudtrail:StartEventDataStoreIngestion cloudtrail:StartImport cloudtrail:StartLogging cloudtrail:StartQuery cloudtrail:StopEventDataStoreIngestion cloudtrail:StopImport cloudtrail:StopLogging cloudtrail:UpdateChannel cloudtrail:UpdateDashboard cloudtrail:UpdateEventDataStore cloudtrail:UpdateTrail

服務前綴	動作
cloudwatch	cloudwatch：DeleteAlarmMuteRule
	cloudwatch:DeleteAlarms
	cloudwatch:DeleteAnomalyDetector
	cloudwatch:DeleteDashboards
	cloudwatch:DeleteInsightRules
	cloudwatch:DeleteMetricStream
	cloudwatch:DescribeAlarmHistory
	cloudwatch:DescribeAlarms
	cloudwatch:DescribeAlarmsForMetric
	cloudwatch:DescribeAnomalyDetectors
	cloudwatch:DescribeInsightRules
	cloudwatch:DisableAlarmActions
	cloudwatch:DisableInsightRules
	cloudwatch:EnableAlarmActions
	cloudwatch:EnableInsightRules
	cloudwatch：GetAlarmMuteRule
	cloudwatch:GetDashboard
	cloudwatch：GetDataset
	cloudwatch:GetInsightRuleReport
	cloudwatch:GetMetricStatistics
	cloudwatch:GetMetricStream

服務前綴	動作
	cloudwatch：ListAlarmMuteRules
	cloudwatch:ListDashboards
	cloudwatch:ListManagedInsightRules
	cloudwatch:ListMetricStreams
	cloudwatch：PutAlarmMuteRule
	cloudwatch:PutAnomalyDetector
	cloudwatch:PutCompositeAlarm
	cloudwatch:PutDashboard
	cloudwatch:PutInsightRule
	cloudwatch：PutLogAlarm
	cloudwatch:PutManagedInsightRules
	cloudwatch:PutMetricAlarm
	cloudwatch:PutMetricStream
	cloudwatch:SetAlarmState
	cloudwatch:StartMetricStreams
	cloudwatch:StopMetricStreams

服務前綴	動作
codeartifact	codeartifact:AssociateExternalConnection codeartifact:CopyPackageVersions codeartifact:CreateDomain codeartifact:CreateRepository codeartifact:DeleteDomain codeartifact:DeleteDomainPermissionsPolicy codeartifact:DeletePackage codeartifact:DeletePackageVersions codeartifact:DeleteRepository codeartifact:DeleteRepositoryPermissionsPolicy codeartifact:DescribeDomain codeartifact:DescribePackage codeartifact:DescribePackageVersion codeartifact:DescribeRepository codeartifact:DisassociateExternalConnection codeartifact:DisposePackageVersions codeartifact:GetAssociatedPackageGroup codeartifact:GetAuthorizationToken codeartifact:GetDomainPermissionsPolicy codeartifact:GetPackageVersionAsset codeartifact:GetPackageVersionReadme

服務前綴	動作
	codeartifact:GetRepositoryEndpoint codeartifact:GetRepositoryPermissionsPolicy codeartifact:ListDomains codeartifact:ListPackageGroups codeartifact:ListPackageVersionAssets codeartifact:ListPackageVersionDependencies codeartifact:ListPackageVersions codeartifact:ListPackages codeartifact:ListRepositories codeartifact:ListRepositoriesInDomain codeartifact:PublishPackageVersion codeartifact:PutDomainPermissionsPolicy codeartifact:PutPackageMetadata codeartifact:PutPackageOriginConfiguration codeartifact:PutRepositoryPermissionsPolicy codeartifact:ReadFromRepository codeartifact:UpdatePackageVersionsStatus codeartifact:UpdateRepository

服務前綴	動作
codedeploy	codedeploy:BatchGetApplicationRevisions
	codedeploy:BatchGetApplications
	codedeploy:BatchGetDeploymentGroups
	codedeploy:BatchGetDeploymentInstances
	codedeploy:BatchGetDeploymentTargets
	codedeploy:BatchGetDeployments
	codedeploy:BatchGetOnPremisesInstances
	codedeploy:ContinueDeployment
	codedeploy:CreateApplication
	codedeploy:CreateDeployment
	codedeploy:CreateDeploymentConfig
	codedeploy:CreateDeploymentGroup
	codedeploy:DeleteApplication
	codedeploy:DeleteDeploymentConfig
	codedeploy:DeleteDeploymentGroup
	codedeploy:DeleteGitHubAccountToken
	codedeploy:DeleteResourcesByExternalId
	codedeploy:DeregisterOnPremisesInstance
	codedeploy:GetApplication
	codedeploy:GetApplicationRevision
	codedeploy:GetDeployment

服務前綴	動作
	codedeploy:GetDeploymentConfig
	codedeploy:GetDeploymentGroup
	codedeploy:GetDeploymentInstance
	codedeploy:GetDeploymentTarget
	codedeploy:GetOnPremisesInstance
	codedeploy:ListApplicationRevisions
	codedeploy:ListApplications
	codedeploy:ListDeploymentConfigs
	codedeploy:ListDeploymentGroups
	codedeploy:ListDeploymentInstances
	codedeploy:ListDeploymentTargets
	codedeploy:ListDeployments
	codedeploy:ListGitHubAccountTokenNames
	codedeploy:ListOnPremisesInstances
	codedeploy:PutLifecycleEventHookExecutionStatus
	codedeploy:RegisterApplicationRevision
	codedeploy:RegisterOnPremisesInstance
	codedeploy:SkipWaitTimeForInstanceTermination
	codedeploy:StopDeployment
	codedeploy:UpdateApplication
	codedeploy:UpdateDeploymentGroup

服務前綴	動作
codeguru-profiler	codeguru-profiler:AddNotificationChannels codeguru-profiler:BatchGetFrameMetricData codeguru-profiler:CreateProfilingGroup codeguru-profiler:DeleteProfilingGroup codeguru-profiler:DescribeProfilingGroup codeguru-profiler:GetFindingsReportAccountSummary codeguru-profiler:GetNotificationConfiguration codeguru-profiler:GetPolicy codeguru-profiler:GetProfile codeguru-profiler:GetRecommendations codeguru-profiler:ListFindingsReports codeguru-profiler:ListProfileTimes codeguru-profiler:ListProfilingGroups codeguru-profiler:PutPermission codeguru-profiler:RemoveNotificationChannel codeguru-profiler:RemovePermission codeguru-profiler:SubmitFeedback codeguru-profiler:UpdateProfilingGroup

服務前綴	動作
codeguru-reviewer	codeguru-reviewer:AssociateRepository codeguru-reviewer:CreateCodeReview codeguru-reviewer:DescribeCodeReview codeguru-reviewer:DescribeRecommendationFeedback codeguru-reviewer:DescribeRepositoryAssociation codeguru-reviewer:DisassociateRepository codeguru-reviewer:ListCodeReviews codeguru-reviewer:ListRecommendationFeedback codeguru-reviewer:ListRecommendations codeguru-reviewer:ListRepositoryAssociations codeguru-reviewer:PutRecommendationFeedback

服務前綴	動作
codepipeline	codepipeline:AcknowledgeJob codepipeline:AcknowledgeThirdPartyJob codepipeline:CreateCustomActionType codepipeline:CreatePipeline codepipeline:DeleteCustomActionType codepipeline:DeletePipeline codepipeline:DeleteWebhook codepipeline:DeregisterWebhookWithThirdParty codepipeline:GetActionType codepipeline:GetJobDetails codepipeline:GetPipeline codepipeline:GetPipelineExecution codepipeline:GetPipelineState codepipeline:GetThirdPartyJobDetails codepipeline:ListActionExecutions codepipeline:ListActionTypes codepipeline:ListPipelineExecutions codepipeline:ListPipelines codepipeline:ListRuleExecutions codepipeline:ListRuleTypes codepipeline:ListWebhooks

服務前綴	動作
	codepipeline:OverrideStageCondition codepipeline:PollForJobs codepipeline:PollForThirdPartyJobs codepipeline:PutActionRevision codepipeline:PutApprovalResult codepipeline:PutJobFailureResult codepipeline:PutJobSuccessResult codepipeline:PutThirdPartyJobFailureResult codepipeline:PutThirdPartyJobSuccessResult codepipeline:PutWebhook codepipeline:RegisterWebhookWithThirdParty codepipeline:RollbackStage codepipeline:StartPipelineExecution codepipeline:StopPipelineExecution codepipeline:UpdateActionType codepipeline:UpdatePipeline

服務前綴	動作
codestar	codestar:AssociateTeamMember
	codestar:CreateProject
	codestar:CreateUserProfile
	codestar:DeleteProject
	codestar:DeleteUserProfile
	codestar:DescribeProject
	codestar:DescribeUserProfile
	codestar:DisassociateTeamMember
	codestar:ListProjects
	codestar:ListResources
	codestar:ListTeamMembers
	codestar:ListUserProfiles
	codestar:UpdateProject
	codestar:UpdateTeamMember
	codestar:UpdateUserProfile

服務前綴	動作
codestar-notifications	codestar-notifications:CreateNotificationRule codestar-notifications:DeleteNotificationRule codestar-notifications:DeleteTarget codestar-notifications:DescribeNotificationRule codestar-notifications:ListEventTypes codestar-notifications:ListNotificationRules codestar-notifications:ListTargets codestar-notifications:Subscribe codestar-notifications:Unsubscribe codestar-notifications:UpdateNotificationRule

服務前綴	動作
cognito-identity	cognito-identity:CreateIdentityPool
	cognito-identity:DeleteIdentities
	cognito-identity:DeleteIdentityPool
	cognito-identity:DescribeIdentity
	cognito-identity:DescribeIdentityPool
	cognito-identity:GetIdentityPoolRoles
	cognito-identity:ListIdentities
	cognito-identity:ListIdentityPools
	cognito-identity:LookupDeveloperIdentity
	cognito-identity:MergeDeveloperIdentities
	cognito-identity:SetIdentityPoolRoles
	cognito-identity:UnlinkDeveloperIdentity
	cognito-identity:UpdateIdentityPool

服務前綴	動作
cognito-idp	cognito-idp:AddCustomAttributes cognito-idp：AddUserPoolClientSecret cognito-idp:AdminAddUserToGroup cognito-idp:AdminConfirmSignUp cognito-idp:AdminCreateUser cognito-idp:AdminDeleteUser cognito-idp:AdminDeleteUserAttributes cognito-idp:AdminDisableProviderForUser cognito-idp:AdminDisableUser cognito-idp:AdminEnableUser cognito-idp:AdminForgetDevice cognito-idp:AdminGetDevice cognito-idp:AdminGetUser cognito-idp：AdminGetUserAuthFactors cognito-idp:AdminInitiateAuth cognito-idp:AdminLinkProviderForUser cognito-idp:AdminListDevices cognito-idp:AdminListGroupsForUser cognito-idp:AdminListUserAuthEvents cognito-idp:AdminRemoveUserFromGroup cognito-idp:AdminResetUserPassword

服務前綴	動作
	cognito-idp:AdminRespondToAuthChallenge cognito-idp:AdminSetUserMFAPreference cognito-idp:AdminSetUserPassword cognito-idp:AdminSetUserSettings cognito-idp:AdminUpdateAuthEventFeedback cognito-idp:AdminUpdateDeviceStatus cognito-idp:AdminUpdateUserAttributes cognito-idp:AdminUserGlobalSignOut cognito-idp:AssociateSoftwareToken cognito-idp:ChangePassword cognito-idp:ConfirmDevice cognito-idp:ConfirmForgotPassword cognito-idp:ConfirmSignUp cognito-idp:CreateGroup cognito-idp:CreateIdentityProvider cognito-idp:CreateManagedLoginBranding cognito-idp:CreateResourceServer cognito-idp：CreateTerms cognito-idp:CreateUserImportJob cognito-idp:CreateUserPool cognito-idp:CreateUserPoolClient

服務前綴	動作
	cognito-idp:CreateUserPoolDomain cognito-idp：CreateUserPoolReplica cognito-idp:DeleteGroup cognito-idp:DeleteIdentityProvider cognito-idp:DeleteManagedLoginBranding cognito-idp:DeleteResourceServer cognito-idp：DeleteTerms cognito-idp:DeleteUser cognito-idp:DeleteUserAttributes cognito-idp:DeleteUserPool cognito-idp:DeleteUserPoolClient cognito-idp：DeleteUserPoolClientSecret cognito-idp:DeleteUserPoolDomain cognito-idp：DeleteUserPoolReplica cognito-idp:DescribeIdentityProvider cognito-idp:DescribeManagedLoginBranding cognito-idp:DescribeManagedLoginBrandingByClient cognito-idp:DescribeResourceServer cognito-idp:DescribeRiskConfiguration cognito-idp：DescribeTerms cognito-idp:DescribeUserImportJob

服務前綴	動作
	cognito-idp:DescribeUserPool cognito-idp:DescribeUserPoolClient cognito-idp:DescribeUserPoolDomain cognito-idp:ForgetDevice cognito-idp:ForgotPassword cognito-idp:GetCSVHeader cognito-idp:GetDevice cognito-idp:GetGroup cognito-idp:GetIdentityProviderByIdentifier cognito-idp:GetLogDeliveryConfiguration cognito-idp：GetProvisionedLimit cognito-idp:GetSigningCertificate cognito-idp:GetUICustomization cognito-idp:GetUser cognito-idp:GetUserAttributeVerificationCode cognito-idp:GetUserPoolMfaConfig cognito-idp:GlobalSignOut cognito-idp:InitiateAuth cognito-idp:ListDevices cognito-idp:ListGroups cognito-idp:ListIdentityProviders

服務前綴	動作
	cognito-idp:ListResourceServers cognito-idp：ListTerms cognito-idp:ListUserImportJobs cognito-idp：ListUserPoolClientSecrets cognito-idp:ListUserPoolClients cognito-idp：ListUserPoolReplicas cognito-idp:ListUserPools cognito-idp:ListUsers cognito-idp:ListUsersInGroup cognito-idp:ResendConfirmationCode cognito-idp:RespondToAuthChallenge cognito-idp:RevokeToken cognito-idp:SetLogDeliveryConfiguration cognito-idp:SetRiskConfiguration cognito-idp:SetUICustomization cognito-idp:SetUserMFAPreference cognito-idp:SetUserPoolMfaConfig cognito-idp:SetUserSettings cognito-idp:SignUp cognito-idp:StartUserImportJob cognito-idp:StopUserImportJob

服務前綴	動作
	cognito-idp:UpdateAuthEventFeedback cognito-idp:UpdateDeviceStatus cognito-idp:UpdateGroup cognito-idp:UpdateIdentityProvider cognito-idp：UpdateProvisionedLimit cognito-idp:UpdateResourceServer cognito-idp：UpdateTerms cognito-idp:UpdateUserAttributes cognito-idp:UpdateUserPool cognito-idp:UpdateUserPoolClient cognito-idp:UpdateUserPoolDomain cognito-idp：UpdateUserPoolReplica cognito-idp:VerifySoftwareToken cognito-idp:VerifyUserAttribute

服務前綴	動作
cognito-sync	cognito-sync:BulkPublish
	cognito-sync:DeleteDataset
	cognito-sync:DescribeDataset
	cognito-sync:DescribeIdentityPoolUsage
	cognito-sync:DescribeIdentityUsage
	cognito-sync:GetBulkPublishDetails
	cognito-sync:GetCognitoEvents
	cognito-sync:GetIdentityPoolConfiguration
	cognito-sync:ListDatasets
	cognito-sync:ListIdentityPoolUsage
	cognito-sync:ListRecords
	cognito-sync:RegisterDevice
	cognito-sync:SetCognitoEvents
	cognito-sync:SetIdentityPoolConfiguration
	cognito-sync:SubscribeToDataset
	cognito-sync:UnsubscribeFromDataset
	cognito-sync:UpdateRecords

服務前綴	動作
comprehendmedical	comprehendmedical:DescribeEntitiesDetectionV2Job comprehendmedical:DescribeICD10CMInferenceJob comprehendmedical:DescribePHIDetectionJob comprehendmedical:DescribeRxNormInferenceJob comprehendmedical:DescribeSNOMEDCTInferenceJob comprehendmedical:DetectEntitiesV2 comprehendmedical:DetectPHI comprehendmedical:InferICD10CM comprehendmedical:InferRxNorm comprehendmedical:InferSNOMEDCT comprehendmedical:ListEntitiesDetectionV2Jobs comprehendmedical:ListICD10CMInferenceJobs comprehendmedical:ListPHIDetectionJobs comprehendmedical:ListRxNormInferenceJobs comprehendmedical:ListSNOMEDCTInferenceJobs comprehendmedical:StartEntitiesDetectionV2Job comprehendmedical:StartICD10CMInferenceJob comprehendmedical:StartPHIDetectionJob comprehendmedical:StartRxNormInferenceJob comprehendmedical:StartSNOMEDCTInferenceJob comprehendmedical:StopEntitiesDetectionV2Job

服務前綴	動作
	comprehendmedical:StopICD10CMInferenceJob comprehendmedical:StopPHIDetectionJob comprehendmedical:StopRxNormInferenceJob comprehendmedical:StopSNOMEDCTInferenceJob

服務前綴	動作
compute-optimizer	compute-optimizer:DeleteRecommendationPreferences
	compute-optimizer:DescribeRecommendationExportJobs
	compute-optimizer:ExportAutoScalingGroupRecommendations
	compute-optimizer:ExportEBSVolumeRecommendations
	compute-optimizer:ExportEC2InstanceRecommendations
	compute-optimizer:ExportECSServiceRecommendations
	compute-optimizer:ExportIdleRecommendations
	compute-optimizer:ExportLambdaFunctionRecommendations
	compute-optimizer:ExportLicenseRecommendations
	compute-optimizer:ExportRDSDatabaseRecommendations
	compute-optimizer:GetEC2RecommendationProjectedMetrics
	compute-optimizer:GetECSServiceRecommendationProjectedMetrics
	compute-optimizer:GetEffectiveRecommendationPreferences
	compute-optimizer:GetEnrollmentStatus
	compute-optimizer:GetEnrollmentStatusesForOrganization
	compute-optimizer:GetRDSDatabaseRecommendationProjectedMetrics
	compute-optimizer:GetRecommendationPreferences
	compute-optimizer:GetRecommendationSummaries
	compute-optimizer:PutRecommendationPreferences

服務前綴	動作
	compute-optimizer:UpdateEnrollmentStatus

服務前綴	動作
config	config:BatchGetResourceConfig config:DeleteAggregationAuthorization config:DeleteConfigRule config:DeleteConfigurationAggregator config:DeleteConfigurationRecorder config:DeleteConformancePack config:DeleteDeliveryChannel config:DeleteEvaluationResults config:DeleteOrganizationConfigRule config:DeleteOrganizationConformancePack config:DeletePendingAggregationRequest config:DeleteRemediationConfiguration config:DeleteRemediationExceptions config:DeleteResourceConfig config:DeleteRetentionConfiguration config:DeleteStoredQuery config:DeliverConfigSnapshot config:DescribeAggregateComplianceByConfigRules config:DescribeAggregateComplianceByConformancePacks config:DescribeAggregationAuthorizations config:DescribeComplianceByConfigRule

服務前綴	動作
	config:DescribeComplianceByResource config:DescribeConfigRuleEvaluationStatus config:DescribeConfigRules config:DescribeConfigurationAggregatorSourcesStatus config:DescribeConfigurationAggregators config:DescribeConfigurationRecorderStatus config:DescribeConfigurationRecorders config:DescribeConformancePackCompliance config:DescribeConformancePackStatus config:DescribeConformancePacks config:DescribeDeliveryChannelStatus config:DescribeDeliveryChannels config:DescribeOrganizationConfigRuleStatuses config:DescribeOrganizationConfigRules config:DescribeOrganizationConformancePackStatuses config:DescribeOrganizationConformancePacks config:DescribePendingAggregationRequests config:DescribeRemediationConfigurations config:DescribeRemediationExceptions config:DescribeRemediationExecutionStatus config:DescribeRetentionConfigurations

服務前綴	動作
	config:GetComplianceDetailsByConfigRule config:GetComplianceDetailsByResource config:GetComplianceSummaryByConfigRule config:GetComplianceSummaryByResourceType config:GetConformancePackComplianceDetails config:GetConformancePackComplianceSummary config:GetCustomRulePolicy config:GetDiscoveredResourceCounts config:GetOrganizationConfigRuleDetailedStatus config:GetOrganizationConformancePackDetailedStatus config:GetOrganizationCustomRulePolicy config:GetResourceConfigHistory config:GetResourceEvaluationSummary config:GetStoredQuery config:ListConfigurationRecorders config:ListConformancePackComplianceScores config：ListConnectors config:ListDiscoveredResources config:ListResourceEvaluations config:ListStoredQueries config:PutConfigRule

服務前綴	動作
	config:PutConfigurationAggregator config:PutConfigurationRecorder config:PutConformancePack config：PutConnector config:PutDeliveryChannel config:PutEvaluations config:PutExternalEvaluation config:PutOrganizationConfigRule config:PutOrganizationConformancePack config:PutRemediationConfigurations config:PutRemediationExceptions config:PutResourceConfig config:PutRetentionConfiguration config:PutStoredQuery config:SelectResourceConfig config:StartConfigRulesEvaluation config:StartConfigurationRecorder config:StartRemediationExecution config:StartResourceEvaluation config:StopConfigurationRecorder

服務前綴	動作
connect	connect:ActivateEvaluationForm
	connect:AssociateAnalyticsDataSet
	connect:AssociateApprovedOrigin
	connect:AssociateBot
	connect：AssociateContactWithUser
	connect:AssociateDefaultVocabulary
	connect：AssociateEmailAddressAlias
	connect:AssociateFlow
	connect:AssociateInstanceStorageConfig
	connect:AssociateLambdaFunction
	connect:AssociateLexBot
	connect:AssociatePhoneNumberContactFlow
	connect：AssociateQueueEmailAddresses
	connect:AssociateQueueQuickConnects
	connect:AssociateRoutingProfileQueues
	connect:AssociateSecurityKey
	connect:AssociateUserProficiencies
	connect:BatchAssociateAnalyticsDataSet
	connect：BatchCreateDataTableValue
	connect：BatchDeleteDataTableValue
	connect：BatchDescribeDataTableValue

服務前綴	動作
	connect:BatchDisassociateAnalyticsDataSet
	connect:BatchGetFlowAssociation
	connect:BatchPutContact
	connect：BatchUpdateDataTableValue
	connect:ClaimPhoneNumber
	connect:CreateAgentStatus
	connect：CreateAttachedFile
	connect:CreateContact
	connect:CreateContactFlow
	connect:CreateContactFlowModule
	connect：CreateContactFlowModuleAlias
	connect：CreateContactFlowModuleVersion
	connect:CreateContactFlowVersion
	connect：CreateDataTable
	connect：CreateDataTableAttribute
	connect:CreateEmailAddress
	connect:CreateEvaluationForm
	connect:CreateHoursOfOperation
	connect:CreateInstance
	connect:CreateIntegrationAssociation
	connect:CreateParticipant

服務前綴	動作
	connect:CreatePersistentContactAssociation
	connect:CreatePredefinedAttribute
	connect:CreatePrompt
	connect:CreatePushNotificationRegistration
	connect:CreateQueue
	connect:CreateQuickConnect
	connect:CreateRoutingProfile
	connect:CreateRule
	connect:CreateSecurityProfile
	connect:CreateTaskTemplate
	connect:CreateTrafficDistributionGroup
	connect:CreateUseCase
	connect:CreateUser
	connect:CreateUserHierarchyGroup
	connect:CreateView
	connect:CreateViewVersion
	connect:CreateVocabulary
	connect：CreateWorkspace
	connect:DeactivateEvaluationForm
	connect:DeleteContactEvaluation
	connect:DeleteContactFlow

服務前綴	動作
	connect:DeleteContactFlowModule
	connect：DeleteContactFlowModuleAlias
	connect：DeleteContactFlowModuleVersion
	connect:DeleteContactFlowVersion
	connect：DeleteDataTable
	connect：DeleteDataTableAttribute
	connect:DeleteEmailAddress
	connect:DeleteEvaluationForm
	connect:DeleteHoursOfOperation
	connect:DeleteHoursOfOperationOverride
	connect:DeleteInstance
	connect:DeleteIntegrationAssociation
	connect：DeleteNotification
	connect:DeletePredefinedAttribute
	connect:DeletePrompt
	connect:DeletePushNotificationRegistration
	connect:DeleteQueue
	connect:DeleteQuickConnect
	connect:DeleteRoutingProfile
	connect:DeleteRule
	connect:DeleteSecurityProfile

服務前綴	動作
	connect:DeleteTaskTemplate
	connect:DeleteTrafficDistributionGroup
	connect:DeleteUseCase
	connect:DeleteUser
	connect:DeleteUserHierarchyGroup
	connect:DeleteView
	connect:DeleteVocabulary
	connect：DeleteWorkspace
	connect：DeleteWorkspaceMedia
	connect：DescribeAttachedFilesConfiguration
	connect:DescribeAuthenticationProfile
	connect：DescribeContactFlowModuleAlias
	connect：DescribeDataTableAttribute
	connect:DescribeHoursOfOperationOverride
	connect:DescribeInstanceAttribute
	connect:DescribeInstanceStorageConfig
	connect：DescribeNotification
	connect:DescribePhoneNumber
	connect:DescribeRule
	connect:DescribeTrafficDistributionGroup
	connect:DescribeUserHierarchyStructure

服務前綴	動作
	connect:DescribeVocabulary
	connect:DisassociateAnalyticsDataSet
	connect:DisassociateApprovedOrigin
	connect:DisassociateBot
	connect：DisassociateEmailAddressAlias
	connect:DisassociateFlow
	connect:DisassociateInstanceStorageConfig
	connect:DisassociateLambdaFunction
	connect:DisassociateLexBot
	connect:DisassociatePhoneNumberContactFlow
	connect：DisassociateQueueEmailAddresses
	connect:DisassociateQueueQuickConnects
	connect:DisassociateRoutingProfileQueues
	connect:DisassociateSecurityKey
	connect:DisassociateUserProficiencies
	connect:DismissUserContact
	connect：EvaluateDataTableValues
	connect:GetContactAttributes
	connect:GetContactMetrics
	connect:GetCurrentMetricData
	connect:GetCurrentUserData

服務前綴	動作
	connect:GetEffectiveHoursOfOperations
	connect：GetEvaluationFormValidation
	connect:GetFederationToken
	connect:GetFlowAssociation
	connect:GetMetricData
	connect:GetMetricDataV2
	connect:GetPromptFile
	connect:GetTaskTemplate
	connect:GetTrafficDistribution
	connect:ImportPhoneNumber
	connect：ImportWorkspaceMedia
	connect:ListAnalyticsDataAssociations
	connect:ListAnalyticsDataLakeDataSets
	connect:ListApprovedOrigins
	connect:ListAssociatedContacts
	connect：ListAttachedFilesConfigurations
	connect:ListAuthenticationProfiles
	connect:ListBots
	connect:ListContactEvaluations
	connect：ListContactFlowModuleAliases
	connect：ListContactFlowModuleVersions

服務前綴	動作
	connect:ListContactFlowModules
	connect:ListContactFlowVersions
	connect:ListContactFlows
	connect:ListContactReferences
	connect：ListDataTableAttributes
	connect：ListDataTablePrimaryValues
	connect：ListDataTableValues
	connect：ListDataTables
	connect:ListDefaultVocabularies
	connect:ListEvaluationFormVersions
	connect:ListEvaluationForms
	connect:ListFlowAssociations
	connect:ListHoursOfOperations
	connect:ListInstanceAttributes
	connect:ListInstanceStorageConfigs
	connect:ListIntegrationAssociations
	connect:ListLambdaFunctions
	connect:ListLexBots
	connect：ListNotifications
	connect:ListPhoneNumbers
	connect:ListPhoneNumbersV2

服務前綴	動作
	connect:ListPredefinedAttributes
	connect:ListPrompts
	connect：ListQueueEmailAddresses
	connect:ListQueueQuickConnects
	connect:ListQueues
	connect:ListQuickConnects
	connect:ListRealtimeContactAnalysisSegmentsV2
	connect：ListRoutingProfileManualAssignmentQueues
	connect:ListRoutingProfileQueues
	connect:ListRoutingProfiles
	connect:ListRules
	connect:ListSecurityKeys
	connect:ListSecurityProfileApplications
	connect：ListSecurityProfileFlowModules
	connect:ListSecurityProfilePermissions
	connect:ListSecurityProfiles
	connect:ListTaskTemplates
	connect:ListTrafficDistributionGroups
	connect:ListUseCases
	connect:ListUserHierarchyGroups
	connect：ListUserNotifications

服務前綴	動作
	connect:ListUsers
	connect:ListViewVersions
	connect:ListViews
	connect：ListWorkspaceMedia
	connect：ListWorkspacePages
	connect：ListWorkspaces
	connect:MonitorContact
	connect:PauseContact
	connect:PutUserStatus
	connect:ReleasePhoneNumber
	connect:ReplicateInstance
	connect:ResumeContact
	connect:ResumeContactRecording
	connect:SearchAgentStatuses
	connect:SearchAvailablePhoneNumbers
	connect：SearchContactEvaluations
	connect:SearchContactFlowModules
	connect:SearchContactFlows
	connect:SearchContacts
	connect：SearchDataTables
	connect:SearchEmailAddresses

服務前綴	動作
	connect：SearchEvaluationForms
	connect:SearchHoursOfOperations
	connect:SearchPredefinedAttributes
	connect:SearchPrompts
	connect:SearchQueues
	connect:SearchQuickConnects
	connect:SearchRoutingProfiles
	connect：SearchRules
	connect:SearchSecurityProfiles
	connect:SearchUserHierarchyGroups
	connect：SearchViews
	connect:SearchVocabularies
	connect：SearchWorkspaceAssociations
	connect：SearchWorkspaces
	connect:SendChatIntegrationEvent
	connect:SendOutboundEmail
	connect:StartChatContact
	connect：StartContactConversationalAnalyticsJob
	connect:StartContactEvaluation
	connect：StartContactMediaProcessing
	connect:StartContactRecording

服務前綴	動作
	connect:StartContactStreaming
	connect:StartEmailContact
	connect:StartOutboundChatContact
	connect:StartOutboundEmailContact
	connect:StartOutboundVoiceContact
	connect:StartScreenSharing
	connect:StartTaskContact
	connect:StartWebRTCContact
	connect:StopContact
	connect：StopContactMediaProcessing
	connect:StopContactRecording
	connect:StopContactStreaming
	connect:SubmitContactEvaluation
	connect:SuspendContactRecording
	connect:TransferContact
	connect:UpdateAgentStatus
	connect：UpdateAttachedFilesConfiguration
	connect:UpdateAuthenticationProfile
	connect:UpdateContact
	connect:UpdateContactAttributes
	connect:UpdateContactEvaluation

服務前綴	動作
	connect:UpdateContactFlowContent
	connect:UpdateContactFlowMetadata
	connect：UpdateContactFlowModuleAlias
	connect:UpdateContactFlowModuleContent
	connect:UpdateContactFlowModuleMetadata
	connect:UpdateContactFlowName
	connect:UpdateContactRoutingData
	connect:UpdateContactSchedule
	connect：UpdateDataTableAttribute
	connect：UpdateDataTableMetadata
	connect：UpdateDataTablePrimaryValues
	connect:UpdateEmailAddressMetadata
	connect:UpdateEvaluationForm
	connect:UpdateHoursOfOperation
	connect:UpdateHoursOfOperationOverride
	connect:UpdateInstanceAttribute
	connect:UpdateInstanceStorageConfig
	connect：UpdateNotificationContent
	connect:UpdateParticipantAuthentication
	connect:UpdateParticipantRoleConfig
	connect:UpdatePhoneNumber

服務前綴	動作
	connect:UpdatePhoneNumberMetadata
	connect:UpdatePredefinedAttribute
	connect:UpdatePrompt
	connect:UpdateQueueHoursOfOperation
	connect:UpdateQueueMaxContacts
	connect:UpdateQueueName
	connect:UpdateQueueOutboundCallerConfig
	connect:UpdateQueueOutboundEmailConfig
	connect:UpdateQueueStatus
	connect:UpdateQuickConnectConfig
	connect:UpdateQuickConnectName
	connect:UpdateRoutingProfileAgentAvailabilityTimer
	connect:UpdateRoutingProfileConcurrency
	connect:UpdateRoutingProfileDefaultOutboundQueue
	connect:UpdateRoutingProfileName
	connect:UpdateRoutingProfileQueues
	connect:UpdateRule
	connect:UpdateSecurityProfile
	connect:UpdateTaskTemplate
	connect:UpdateTrafficDistribution
	connect:UpdateUserHierarchy

服務前綴	動作
	connect:UpdateUserHierarchyGroupName connect:UpdateUserHierarchyStructure connect:UpdateUserIdentityInfo connect：UpdateUserNotificationStatus connect:UpdateUserPhoneConfig connect:UpdateUserProficiencies connect:UpdateUserRoutingProfile connect:UpdateUserSecurityProfiles connect:UpdateViewContent connect:UpdateViewMetadata connect：UpdateWorkspaceMetadata connect：UpdateWorkspaceTheme connect：UpdateWorkspaceVisibility
cur	cur:DeleteReportDefinition cur:DescribeReportDefinitions cur:ModifyReportDefinition cur:PutReportDefinition

服務前綴	動作
databrew	databrew:BatchDeleteRecipeVersion
	databrew:CreateDataset
	databrew:CreateProfileJob
	databrew:CreateProject
	databrew:CreateRecipe
	databrew:CreateRecipeJob
	databrew:CreateRuleset
	databrew:CreateSchedule
	databrew:DeleteDataset
	databrew:DeleteJob
	databrew:DeleteProject
	databrew:DeleteRecipeVersion
	databrew:DeleteRuleset
	databrew:DeleteSchedule
	databrew:DescribeDataset
	databrew:DescribeJob
	databrew:DescribeJobRun
	databrew:DescribeProject
	databrew:DescribeRecipe
	databrew:DescribeRuleset
	databrew:DescribeSchedule

服務前綴	動作
	databrew:ListDatasets
	databrew:ListJobRuns
	databrew:ListJobs
	databrew:ListProjects
	databrew:ListRecipeVersions
	databrew:ListRecipes
	databrew:ListRulesets
	databrew:ListSchedules
	databrew:PublishRecipe
	databrew:SendProjectSessionAction
	databrew:StartJobRun
	databrew:StartProjectSession
	databrew:StopJobRun
	databrew:UpdateDataset
	databrew:UpdateProfileJob
	databrew:UpdateProject
	databrew:UpdateRecipe
	databrew:UpdateRecipeJob
	databrew:UpdateRuleset
	databrew:UpdateSchedule

服務前綴	動作
dataexchange	dataexchange:AcceptDataGrant dataexchange:CancelJob dataexchange:CreateDataGrant dataexchange:CreateDataSet dataexchange:CreateEventAction dataexchange:CreateJob dataexchange:CreateRevision dataexchange:DeleteAsset dataexchange:DeleteDataGrant dataexchange:DeleteEventAction dataexchange:DeleteRevision dataexchange:GetDataGrant dataexchange:GetEventAction dataexchange:GetJob dataexchange:GetReceivedDataGrant dataexchange:ListDataGrants dataexchange:ListDataSetRevisions dataexchange:ListDataSets dataexchange:ListEventActions dataexchange:ListJobs dataexchange:ListReceivedDataGrants

服務前綴	動作
	dataexchange:ListRevisionAssets dataexchange:RevokeRevision dataexchange:SendDataSetNotification dataexchange:StartJob dataexchange:UpdateAsset dataexchange:UpdateDataSet dataexchange:UpdateEventAction dataexchange:UpdateRevision

服務前綴	動作
datapipeline	datapipeline:ActivatePipeline datapipeline:CreatePipeline datapipeline:DeactivatePipeline datapipeline:DeletePipeline datapipeline:DescribeObjects datapipeline:DescribePipelines datapipeline:EvaluateExpression datapipeline:GetPipelineDefinition datapipeline:ListPipelines datapipeline:PollForTask datapipeline:PutPipelineDefinition datapipeline:QueryObjects datapipeline:ReportTaskProgress datapipeline:ReportTaskRunnerHeartbeat datapipeline:SetStatus datapipeline:SetTaskStatus datapipeline:ValidatePipelineDefinition

服務前綴	動作
dax	dax:CreateCluster dax:DecreaseReplicationFactor dax:DeleteCluster dax:DeleteParameterGroup dax:DeleteSubnetGroup dax:DescribeClusters dax:DescribeDefaultParameters dax:DescribeEvents dax:DescribeParameterGroups dax:DescribeParameters dax:DescribeSubnetGroups dax:IncreaseReplicationFactor dax:RebootNode dax:UpdateCluster dax:UpdateParameterGroup dax:UpdateSubnetGroup

服務前綴	動作
devicefarm	devicefarm:CreateDevicePool
	devicefarm:CreateInstanceProfile
	devicefarm:CreateNetworkProfile
	devicefarm:CreateProject
	devicefarm:CreateRemoteAccessSession
	devicefarm:CreateTestGridProject
	devicefarm:CreateTestGridUrl
	devicefarm:CreateUpload
	devicefarm:CreateVPCEConfiguration
	devicefarm:DeleteDevicePool
	devicefarm:DeleteInstanceProfile
	devicefarm:DeleteNetworkProfile
	devicefarm:DeleteProject
	devicefarm:DeleteRemoteAccessSession
	devicefarm:DeleteRun
	devicefarm:DeleteTestGridProject
	devicefarm:DeleteUpload
	devicefarm:DeleteVPCEConfiguration
	devicefarm:GetAccountSettings
	devicefarm:GetDevice
	devicefarm:GetDeviceInstance

服務前綴	動作
	devicefarm:GetDevicePool
	devicefarm:GetDevicePoolCompatibility
	devicefarm:GetInstanceProfile
	devicefarm:GetJob
	devicefarm:GetNetworkProfile
	devicefarm:GetOfferingStatus
	devicefarm:GetProject
	devicefarm:GetRemoteAccessSession
	devicefarm:GetRun
	devicefarm:GetSuite
	devicefarm:GetTest
	devicefarm:GetTestGridProject
	devicefarm:GetTestGridSession
	devicefarm:GetUpload
	devicefarm:GetVPCEConfiguration
	devicefarm:ListArtifacts
	devicefarm:ListDeviceInstances
	devicefarm:ListDevicePools
	devicefarm:ListDevices
	devicefarm:ListInstanceProfiles
	devicefarm:ListJobs

服務前綴	動作
	devicefarm:ListNetworkProfiles
	devicefarm:ListOfferingPromotions
	devicefarm:ListOfferingTransactions
	devicefarm:ListOfferings
	devicefarm:ListProjects
	devicefarm:ListRemoteAccessSessions
	devicefarm:ListRuns
	devicefarm:ListSamples
	devicefarm:ListSuites
	devicefarm:ListTestGridProjects
	devicefarm:ListTestGridSessionActions
	devicefarm:ListTestGridSessionArtifacts
	devicefarm:ListTestGridSessions
	devicefarm:ListTests
	devicefarm:ListUniqueProblems
	devicefarm:ListUploads
	devicefarm:ListVPCEConfigurations
	devicefarm:PurchaseOffering
	devicefarm:RenewOffering
	devicefarm:ScheduleRun
	devicefarm:StopJob

服務前綴	動作
	devicefarm:StopRemoteAccessSession
	devicefarm:StopRun
	devicefarm:UpdateDeviceInstance
	devicefarm:UpdateDevicePool
	devicefarm:UpdateInstanceProfile
	devicefarm:UpdateNetworkProfile
	devicefarm:UpdateProject
	devicefarm:UpdateTestGridProject
	devicefarm:UpdateUpload
	devicefarm:UpdateVPCEConfiguration

服務前綴	動作
devops-guru	devops-guru:AddNotificationChannel devops-guru:DeleteInsight devops-guru:DescribeAccountHealth devops-guru:DescribeAccountOverview devops-guru:DescribeAnomaly devops-guru:DescribeEventSourcesConfig devops-guru:DescribeFeedback devops-guru:DescribeInsight devops-guru:DescribeOrganizationHealth devops-guru:DescribeOrganizationOverview devops-guru:DescribeOrganizationResourceCollectionHealth devops-guru:DescribeResourceCollectionHealth devops-guru:DescribeServiceIntegration devops-guru:GetCostEstimation devops-guru:GetResourceCollection devops-guru:ListAnomaliesForInsight devops-guru:ListAnomalousLogGroups devops-guru:ListEvents devops-guru:ListInsights devops-guru:ListMonitoredResources devops-guru:ListNotificationChannels

服務前綴	動作
	devops-guru:ListOrganizationInsights devops-guru:ListRecommendations devops-guru:PutFeedback devops-guru:RemoveNotificationChannel devops-guru:SearchInsights devops-guru:SearchOrganizationInsights devops-guru:StartCostEstimation devops-guru:UpdateEventSourcesConfig devops-guru:UpdateResourceCollection devops-guru:UpdateServiceIntegration

服務前綴	動作
directconnect	directconnect:AcceptDirectConnectGatewayAssociationProposal
	directconnect:AllocateConnectionOnInterconnect
	directconnect:AllocateHostedConnection
	directconnect:AllocatePrivateVirtualInterface
	directconnect:AllocatePublicVirtualInterface
	directconnect:AllocateTransitVirtualInterface
	directconnect:AssociateConnectionWithLag
	directconnect:AssociateHostedConnection
	directconnect:AssociateMacSecKey
	directconnect:AssociateVirtualInterface
	directconnect:ConfirmConnection
	directconnect:ConfirmCustomerAgreement
	directconnect:ConfirmPrivateVirtualInterface
	directconnect:ConfirmPublicVirtualInterface
	directconnect:ConfirmTransitVirtualInterface
	directconnect:CreateBGPPeer
	directconnect:CreateConnection
	directconnect:CreateDirectConnectGateway
	directconnect:CreateDirectConnectGatewayAssociation
	directconnect:CreateDirectConnectGatewayAssociationProposal
	directconnect:CreateInterconnect

服務前綴	動作
	directconnect:CreateLag
	directconnect:CreatePrivateVirtualInterface
	directconnect:CreatePublicVirtualInterface
	directconnect:CreateTransitVirtualInterface
	directconnect:DeleteBGPPeer
	directconnect:DeleteConnection
	directconnect:DeleteDirectConnectGateway
	directconnect:DeleteDirectConnectGatewayAssociation
	directconnect:DeleteDirectConnectGatewayAssociationProposal
	directconnect:DeleteInterconnect
	directconnect:DeleteLag
	directconnect:DeleteVirtualInterface
	directconnect:DescribeConnectionLoa
	directconnect:DescribeConnections
	directconnect:DescribeConnectionsOnInterconnect
	directconnect:DescribeCustomerMetadata
	directconnect:DescribeDirectConnectGatewayAssociationProposals
	directconnect:DescribeDirectConnectGatewayAssociations
	directconnect:DescribeDirectConnectGatewayAttachments
	directconnect:DescribeDirectConnectGateways
	directconnect:DescribeHostedConnections

服務前綴	動作
	directconnect:DescribeInterconnectLoa
	directconnect:DescribeInterconnects
	directconnect:DescribeLags
	directconnect:DescribeLoa
	directconnect:DescribeLocations
	directconnect:DescribeRouterConfiguration
	directconnect:DescribeVirtualGateways
	directconnect:DescribeVirtualInterfaces
	directconnect:DisassociateConnectionFromLag
	directconnect:DisassociateMacSecKey
	directconnect:ListVirtualInterfaceTestHistory
	directconnect:StartBgpFailoverTest
	directconnect:StopBgpFailoverTest
	directconnect:UpdateConnection
	directconnect:UpdateDirectConnectGateway
	directconnect:UpdateDirectConnectGatewayAssociation
	directconnect:UpdateLag
	directconnect:UpdateVirtualInterfaceAttributes

服務前綴	動作
dlm	dlm:CreateLifecyclePolicy dlm:DeleteLifecyclePolicy dlm:GetLifecyclePolicies dlm:GetLifecyclePolicy dlm:UpdateLifecyclePolicy

服務前綴	動作
dms	dms:ApplyPendingMaintenanceAction
	dms:AssociateExtensionPack
	dms:BatchStartRecommendations
	dms：CancelMetadataModelCreation
	dms:CancelReplicationTaskAssessmentRun
	dms:CreateDataProvider
	dms:CreateEndpoint
	dms:CreateEventSubscription
	dms:CreateInstanceProfile
	dms:CreateMigrationProject
	dms:CreateReplicationConfig
	dms:CreateReplicationInstance
	dms:CreateReplicationSubnetGroup
	dms:CreateReplicationTask
	dms:DeleteCertificate
	dms:DeleteConnection
	dms:DeleteDataMigration
	dms:DeleteDataProvider
	dms:DeleteEndpoint
	dms:DeleteEventSubscription
	dms:DeleteFleetAdvisorCollector

服務前綴	動作
	dms:DeleteFleetAdvisorDatabases
	dms:DeleteInstanceProfile
	dms:DeleteMigrationProject
	dms:DeleteReplicationConfig
	dms:DeleteReplicationInstance
	dms:DeleteReplicationSubnetGroup
	dms:DeleteReplicationTask
	dms:DeleteReplicationTaskAssessmentRun
	dms:DescribeAccountAttributes
	dms:DescribeApplicableIndividualAssessments
	dms:DescribeCertificates
	dms:DescribeConnections
	dms:DescribeDataMigrations
	dms:DescribeEndpointSettings
	dms:DescribeEndpointTypes
	dms:DescribeEndpoints
	dms:DescribeEngineVersions
	dms:DescribeEventCategories
	dms:DescribeEventSubscriptions
	dms:DescribeEvents
	dms:DescribeFleetAdvisorCollectors

服務前綴	動作
	dms:DescribeFleetAdvisorDatabases
	dms:DescribeFleetAdvisorLsaAnalysis
	dms:DescribeFleetAdvisorSchemaObjectSummary
	dms:DescribeFleetAdvisorSchemas
	dms：DescribeMetadataModel
	dms：DescribeMetadataModelChildren
	dms：DescribeMetadataModelCreations
	dms:DescribeMetadataModelImports
	dms:DescribeOrderableReplicationInstances
	dms:DescribePendingMaintenanceActions
	dms:DescribeRecommendationLimitations
	dms:DescribeRecommendations
	dms:DescribeRefreshSchemasStatus
	dms:DescribeReplicationConfigs
	dms:DescribeReplicationInstanceTaskLogs
	dms:DescribeReplicationInstances
	dms:DescribeReplicationSubnetGroups
	dms:DescribeReplicationTableStatistics
	dms:DescribeReplicationTaskAssessmentResults
	dms:DescribeReplicationTaskAssessmentRuns
	dms:DescribeReplicationTaskIndividualAssessments

服務前綴	動作
	dms:DescribeReplicationTasks
	dms:DescribeReplications
	dms:DescribeSchemas
	dms:DescribeTableStatistics
	dms:ExportMetadataModelAssessment
	dms:ImportCertificate
	dms:ListDataProviders
	dms:ListExtensionPacks
	dms:ListInstanceProfiles
	dms:ListMetadataModelAssessments
	dms:ListMetadataModelConversions
	dms:ListMetadataModelExports
	dms:ListMigrationProjects
	dms:ModifyDataMigration
	dms:ModifyEndpoint
	dms:ModifyEventSubscription
	dms:ModifyReplicationConfig
	dms:ModifyReplicationInstance
	dms:ModifyReplicationSubnetGroup
	dms:ModifyReplicationTask
	dms:MoveReplicationTask

服務前綴	動作
	dms:RebootReplicationInstance
	dms:RefreshSchemas
	dms:ReloadReplicationTables
	dms:ReloadTables
	dms:RunFleetAdvisorLsaAnalysis
	dms:StartMetadataModelAssessment
	dms:StartMetadataModelConversion
	dms：StartMetadataModelCreation
	dms:StartMetadataModelExportAsScripts
	dms:StartMetadataModelExportToTarget
	dms:StartRecommendations
	dms:StartReplication
	dms:StartReplicationTask
	dms:StartReplicationTaskAssessment
	dms:StopDataMigration
	dms:StopReplicationTask
	dms:TestConnection
	dms:UpdateConversionConfiguration
	dms:UpdateDataProvider
	dms:UpdateInstanceProfile
	dms:UpdateMigrationProject

服務前綴	動作
	dms:UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic:ApplyPendingMaintenanceAction docdb-elastic:CopyClusterSnapshot docdb-elastic:DeleteCluster docdb-elastic:DeleteClusterSnapshot docdb-elastic:GetCluster docdb-elastic:GetClusterSnapshot docdb-elastic:GetPendingMaintenanceAction docdb-elastic:ListClusterSnapshots docdb-elastic:ListClusters docdb-elastic:ListPendingMaintenanceActions docdb-elastic:RestoreClusterFromSnapshot docdb-elastic:StartCluster docdb-elastic:StopCluster docdb-elastic:UpdateCluster

服務前綴	動作
dynamodb	dynamodb:CreateBackup
	dynamodb:CreateGlobalTable
	dynamodb:CreateTable
	dynamodb:DeleteBackup
	dynamodb:DeleteTable
	dynamodb:DescribeBackup
	dynamodb:DescribeContinuousBackups
	dynamodb:DescribeContributorInsights
	dynamodb:DescribeEndpoints
	dynamodb:DescribeExport
	dynamodb:DescribeGlobalTable
	dynamodb:DescribeGlobalTableSettings
	dynamodb:DescribeImport
	dynamodb:DescribeKinesisStreamingDestination
	dynamodb:DescribeLimits
	dynamodb:DescribeStream
	dynamodb:DescribeTable
	dynamodb:DescribeTableReplicaAutoScaling
	dynamodb:DescribeTimeToLive
	dynamodb:DisableKinesisStreamingDestination
	dynamodb:EnableKinesisStreamingDestination

服務前綴	動作
	dynamodb:ExportTableToPointInTime
	dynamodb:GetResourcePolicy
	dynamodb:ImportTable
	dynamodb:ListBackups
	dynamodb:ListContributorInsights
	dynamodb:ListExports
	dynamodb:ListGlobalTables
	dynamodb:ListImports
	dynamodb:ListStreams
	dynamodb:ListTables
	dynamodb:RestoreTableFromBackup
	dynamodb:RestoreTableToPointInTime
	dynamodb:UpdateContinuousBackups
	dynamodb:UpdateContributorInsights
	dynamodb:UpdateGlobalTable
	dynamodb:UpdateGlobalTableSettings
	dynamodb:UpdateKinesisStreamingDestination
	dynamodb:UpdateTable
	dynamodb:UpdateTableReplicaAutoScaling
	dynamodb:UpdateTimeToLive

服務前綴	動作
ebs	ebs:CompleteSnapshot ebs:StartSnapshot

服務前綴	動作
ec2	ec2:AcceptAddressTransfer
	ec2:AcceptCapacityReservationBillingOwnership
	ec2:AcceptReservedInstancesExchangeQuote
	ec2：AcceptTransitGatewayClientVpnAttachment
	ec2:AcceptTransitGatewayMulticastDomainAssociations
	ec2:AcceptTransitGatewayPeeringAttachment
	ec2:AcceptTransitGatewayVpcAttachment
	ec2:AcceptVpcEndpointConnections
	ec2:AcceptVpcPeeringConnection
	ec2:AdvertiseByoipCidr
	ec2:AllocateAddress
	ec2:AllocateHosts
	ec2:AllocateIpamPoolCidr
	ec2:ApplySecurityGroupsToClientVpnTargetNetwork
	ec2:AssignIpv6Addresses
	ec2:AssignPrivateIpAddresses
	ec2:AssignPrivateNatGatewayAddress
	ec2:AssociateAddress
	ec2:AssociateCapacityReservationBillingOwner
	ec2:AssociateClientVpnTargetNetwork
	ec2:AssociateDhcpOptions

服務前綴	動作
	ec2:AssociateEnclaveCertificateIamRole
	ec2:AssociateIamInstanceProfile
	ec2:AssociateInstanceEventWindow
	ec2:AssociateIpamByoasn
	ec2:AssociateIpamResourceDiscovery
	ec2:AssociateNatGatewayAddress
	ec2:AssociateRouteServer
	ec2:AssociateRouteTable
	ec2:AssociateSecurityGroupVpc
	ec2:AssociateSubnetCidrBlock
	ec2:AssociateTransitGatewayMulticastDomain
	ec2:AssociateTransitGatewayPolicyTable
	ec2:AssociateTransitGatewayRouteTable
	ec2:AssociateTrunkInterface
	ec2:AssociateVpcCidrBlock
	ec2:AttachClassicLinkVpc
	ec2：AttachImageWatermark
	ec2:AttachInternetGateway
	ec2:AttachNetworkInterface
	ec2:AttachVerifiedAccessTrustProvider
	ec2:AttachVolume

服務前綴	動作
	ec2:AttachVpnGateway
	ec2:AuthorizeClientVpnIngress
	ec2:AuthorizeSecurityGroupEgress
	ec2:AuthorizeSecurityGroupIngress
	ec2:BundleInstance
	ec2:CancelBundleTask
	ec2:CancelCapacityReservation
	ec2:CancelCapacityReservationFleets
	ec2:CancelConversionTask
	ec2:CancelDeclarativePoliciesReport
	ec2:CancelExportTask
	ec2:CancelImageLaunchPermission
	ec2:CancelImportTask
	ec2:CancelReservedInstancesListing
	ec2:CancelSpotFleetRequests
	ec2:CancelSpotInstanceRequests
	ec2:ConfirmProductInstance
	ec2:CopyFpgaImage
	ec2:CopyImage
	ec2:CopySnapshot
	ec2：CopyVolumes

服務前綴	動作
	ec2：CreateCapacityManagerDataExport
	ec2:CreateCapacityReservation
	ec2:CreateCapacityReservationBySplitting
	ec2：CreateCapacityReservationCancellationQuote
	ec2:CreateCapacityReservationFleet
	ec2:CreateCarrierGateway
	ec2:CreateClientVpnEndpoint
	ec2:CreateClientVpnRoute
	ec2:CreateCoipCidr
	ec2:CreateCoipPool
	ec2:CreateCustomerGateway
	ec2:CreateDefaultSubnet
	ec2:CreateDefaultVpc
	ec2:CreateDelegateMacVolumeOwnershipTask
	ec2:CreateDhcpOptions
	ec2:CreateEgressOnlyInternetGateway
	ec2:CreateFleet
	ec2:CreateFlowLogs
	ec2:CreateFpgaImage
	ec2:CreateImage
	ec2：CreateImageUsageReport

服務前綴	動作
	ec2:CreateInstanceConnectEndpoint
	ec2:CreateInstanceEventWindow
	ec2:CreateInstanceExportTask
	ec2：CreateInternetGateway
	ec2：CreateInterruptibleCapacityReservationAllocation
	ec2:CreateIpam
	ec2:CreateIpamExternalResourceVerificationToken
	ec2：CreateIpamPolicy
	ec2:CreateIpamPool
	ec2：CreateIpamPrefixListResolver
	ec2：CreateIpamPrefixListResolverTarget
	ec2:CreateIpamResourceDiscovery
	ec2:CreateIpamScope
	ec2:CreateKeyPair
	ec2:CreateLaunchTemplateVersion
	ec2:CreateLocalGatewayRoute
	ec2:CreateLocalGatewayRouteTable
	ec2:CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:CreateLocalGatewayRouteTableVpcAssociation
	ec2:CreateLocalGatewayVirtualInterface

服務前綴	動作
	ec2:CreateLocalGatewayVirtualInterfaceGroup
	ec2:CreateMacSystemIntegrityProtectionModificationTask
	ec2:CreateManagedPrefixList
	ec2:CreateNatGateway
	ec2:CreateNetworkAcl
	ec2:CreateNetworkAclEntry
	ec2:CreateNetworkInsightsAccessScope
	ec2:CreateNetworkInsightsPath
	ec2:CreateNetworkInterface
	ec2:CreateNetworkInterfacePermission
	ec2:CreatePlacementGroup
	ec2:CreatePublicIpv4Pool
	ec2:CreateReplaceRootVolumeTask
	ec2:CreateReservedInstancesListing
	ec2:CreateRestoreImageTask
	ec2：CreateRoute
	ec2:CreateRouteServer
	ec2:CreateRouteServerEndpoint
	ec2:CreateRouteServerPeer
	ec2：CreateRouteTable
	ec2：CreateSecondaryNetwork

服務前綴	動作
	ec2：CreateSecondarySubnet
	ec2：CreateSecurityGroup
	ec2:CreateSnapshots
	ec2:CreateSpotDatafeedSubscription
	ec2:CreateStoreImageTask
	ec2：CreateSubnet
	ec2:CreateSubnetCidrReservation
	ec2:CreateTrafficMirrorFilter
	ec2:CreateTrafficMirrorFilterRule
	ec2:CreateTrafficMirrorSession
	ec2:CreateTrafficMirrorTarget
	ec2:CreateTransitGateway
	ec2:CreateTransitGatewayConnect
	ec2:CreateTransitGatewayConnectPeer
	ec2：CreateTransitGatewayMeteringPolicy
	ec2：CreateTransitGatewayMeteringPolicyEntry
	ec2:CreateTransitGatewayMulticastDomain
	ec2:CreateTransitGatewayPeeringAttachment
	ec2:CreateTransitGatewayPolicyTable
	ec2:CreateTransitGatewayPrefixListReference
	ec2:CreateTransitGatewayRoute

服務前綴	動作
	ec2:CreateTransitGatewayRouteTable
	ec2:CreateTransitGatewayRouteTableAnnouncement
	ec2:CreateTransitGatewayVpcAttachment
	ec2:CreateVerifiedAccessEndpoint
	ec2:CreateVerifiedAccessGroup
	ec2:CreateVerifiedAccessInstance
	ec2:CreateVerifiedAccessTrustProvider
	ec2:CreateVolume
	ec2：CreateVpc
	ec2:CreateVpcBlockPublicAccessExclusion
	ec2：CreateVpcEncryptionControl
	ec2:CreateVpcEndpoint
	ec2:CreateVpcEndpointConnectionNotification
	ec2:CreateVpcEndpointServiceConfiguration
	ec2:CreateVpcPeeringConnection
	ec2：CreateVpnConcentrator
	ec2:CreateVpnConnection
	ec2:CreateVpnConnectionRoute
	ec2:CreateVpnGateway
	ec2：DeleteCapacityManagerDataExport
	ec2:DeleteCarrierGateway

服務前綴	動作
	ec2:DeleteClientVpnEndpoint
	ec2:DeleteClientVpnRoute
	ec2:DeleteCoipCidr
	ec2:DeleteCoipPool
	ec2:DeleteCustomerGateway
	ec2:DeleteDhcpOptions
	ec2:DeleteEgressOnlyInternetGateway
	ec2:DeleteFleets
	ec2:DeleteFlowLogs
	ec2:DeleteFpgaImage
	ec2：DeleteImageUsageReport
	ec2:DeleteInstanceConnectEndpoint
	ec2:DeleteInstanceEventWindow
	ec2:DeleteInternetGateway
	ec2:DeleteIpam
	ec2:DeleteIpamExternalResourceVerificationToken
	ec2：DeleteIpamPolicy
	ec2:DeleteIpamPool
	ec2：DeleteIpamPrefixListResolver
	ec2：DeleteIpamPrefixListResolverTarget
	ec2:DeleteIpamResourceDiscovery

服務前綴	動作
	ec2:DeleteIpamScope
	ec2:DeleteKeyPair
	ec2:DeleteLaunchTemplate
	ec2:DeleteLaunchTemplateVersions
	ec2:DeleteLocalGatewayRoute
	ec2:DeleteLocalGatewayRouteTable
	ec2:DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:DeleteLocalGatewayRouteTableVpcAssociation
	ec2:DeleteLocalGatewayVirtualInterface
	ec2:DeleteLocalGatewayVirtualInterfaceGroup
	ec2:DeleteManagedPrefixList
	ec2:DeleteNatGateway
	ec2:DeleteNetworkAcl
	ec2:DeleteNetworkAclEntry
	ec2:DeleteNetworkInsightsAccessScope
	ec2:DeleteNetworkInsightsAccessScopeAnalysis
	ec2:DeleteNetworkInsightsAnalysis
	ec2:DeleteNetworkInsightsPath
	ec2:DeleteNetworkInterface
	ec2:DeleteNetworkInterfacePermission

服務前綴	動作
	ec2:DeletePlacementGroup
	ec2:DeletePublicIpv4Pool
	ec2:DeleteQueuedReservedInstances
	ec2:DeleteRoute
	ec2:DeleteRouteServer
	ec2:DeleteRouteServerEndpoint
	ec2:DeleteRouteServerPeer
	ec2:DeleteRouteTable
	ec2：DeleteSecondaryNetwork
	ec2：DeleteSecondarySubnet
	ec2:DeleteSecurityGroup
	ec2:DeleteSpotDatafeedSubscription
	ec2:DeleteSubnet
	ec2:DeleteSubnetCidrReservation
	ec2:DeleteTrafficMirrorFilter
	ec2:DeleteTrafficMirrorFilterRule
	ec2:DeleteTrafficMirrorSession
	ec2:DeleteTrafficMirrorTarget
	ec2:DeleteTransitGateway
	ec2：DeleteTransitGatewayClientVpnAttachment
	ec2:DeleteTransitGatewayConnect

服務前綴	動作
	ec2:DeleteTransitGatewayConnectPeer
	ec2：DeleteTransitGatewayMeteringPolicy
	ec2：DeleteTransitGatewayMeteringPolicyEntry
	ec2:DeleteTransitGatewayMulticastDomain
	ec2:DeleteTransitGatewayPeeringAttachment
	ec2:DeleteTransitGatewayPolicyTable
	ec2:DeleteTransitGatewayPrefixListReference
	ec2:DeleteTransitGatewayRoute
	ec2:DeleteTransitGatewayRouteTable
	ec2:DeleteTransitGatewayRouteTableAnnouncement
	ec2:DeleteTransitGatewayVpcAttachment
	ec2:DeleteVerifiedAccessEndpoint
	ec2:DeleteVerifiedAccessGroup
	ec2:DeleteVerifiedAccessInstance
	ec2:DeleteVerifiedAccessTrustProvider
	ec2:DeleteVolume
	ec2:DeleteVpc
	ec2:DeleteVpcBlockPublicAccessExclusion
	ec2：DeleteVpcEncryptionControl
	ec2:DeleteVpcEndpointConnectionNotifications
	ec2:DeleteVpcEndpointServiceConfigurations

服務前綴	動作
	ec2:DeleteVpcEndpoints
	ec2:DeleteVpcPeeringConnection
	ec2：DeleteVpnConcentrator
	ec2:DeleteVpnConnection
	ec2:DeleteVpnConnectionRoute
	ec2:DeleteVpnGateway
	ec2:DeprovisionByoipCidr
	ec2:DeprovisionIpamByoasn
	ec2:DeprovisionIpamPoolCidr
	ec2:DeprovisionPublicIpv4PoolCidr
	ec2:DeregisterImage
	ec2:DeregisterInstanceEventNotificationAttributes
	ec2:DeregisterTransitGatewayMulticastGroupMembers
	ec2:DeregisterTransitGatewayMulticastGroupSources
	ec2:DescribeAccountAttributes
	ec2:DescribeAddressTransfers
	ec2:DescribeAddresses
	ec2:DescribeAddressesAttribute
	ec2:DescribeAggregateIdFormat
	ec2：DescribeAvailabilityZones
	ec2:DescribeAwsNetworkPerformanceMetricSubscriptions

服務前綴	動作
	ec2:DescribeBundleTasks
	ec2:DescribeByoipCidrs
	ec2:DescribeCapacityBlockExtensionHistory
	ec2:DescribeCapacityBlockExtensionOfferings
	ec2:DescribeCapacityBlockStatus
	ec2:DescribeCapacityBlocks
	ec2：DescribeCapacityManagerDataExports
	ec2:DescribeCapacityReservationBillingRequests
	ec2：DescribeCapacityReservationCancellationQuotes
	ec2:DescribeCapacityReservationFleets
	ec2：DescribeCapacityReservationTopology
	ec2:DescribeCapacityReservations
	ec2:DescribeCarrierGateways
	ec2:DescribeClassicLinkInstances
	ec2:DescribeClientVpnAuthorizationRules
	ec2:DescribeClientVpnConnections
	ec2:DescribeClientVpnEndpoints
	ec2:DescribeClientVpnRoutes
	ec2:DescribeClientVpnTargetNetworks
	ec2:DescribeCoipPools
	ec2:DescribeConversionTasks

服務前綴	動作
	ec2:DescribeCustomerGateways
	ec2:DescribeDeclarativePoliciesReports
	ec2:DescribeDhcpOptions
	ec2:DescribeEgressOnlyInternetGateways
	ec2:DescribeElasticGpus
	ec2:DescribeExportImageTasks
	ec2:DescribeExportTasks
	ec2:DescribeFastLaunchImages
	ec2:DescribeFastSnapshotRestores
	ec2:DescribeFleetHistory
	ec2:DescribeFleetInstances
	ec2:DescribeFleets
	ec2:DescribeFlowLogs
	ec2:DescribeFpgaImageAttribute
	ec2:DescribeFpgaImages
	ec2:DescribeHostReservationOfferings
	ec2:DescribeHostReservations
	ec2:DescribeHosts
	ec2:DescribeIamInstanceProfileAssociations
	ec2:DescribeIdFormat
	ec2:DescribeIdentityIdFormat

服務前綴	動作
	ec2:DescribeImageAttribute
	ec2：DescribeImageReferences
	ec2：DescribeImageUsageReportEntries
	ec2：DescribeImageUsageReports
	ec2:DescribeImportImageTasks
	ec2:DescribeImportSnapshotTasks
	ec2:DescribeInstanceConnectEndpoints
	ec2:DescribeInstanceCreditSpecifications
	ec2:DescribeInstanceEventNotificationAttributes
	ec2:DescribeInstanceEventWindows
	ec2:DescribeInstanceImageMetadata
	ec2：DescribeInstanceSqlHaHistoryStates
	ec2：DescribeInstanceSqlHaStates
	ec2:DescribeInstanceTopology
	ec2:DescribeInstanceTypes
	ec2：DescribeInternetGateways
	ec2:DescribeIpamByoasn
	ec2:DescribeIpamExternalResourceVerificationTokens
	ec2：DescribeIpamPolicies
	ec2：DescribeIpamPoolAllocations
	ec2:DescribeIpamPools

服務前綴	動作
	ec2：DescribeIpamPrefixListResolverTargets
	ec2：DescribeIpamPrefixListResolvers
	ec2:DescribeIpamResourceDiscoveries
	ec2:DescribeIpamResourceDiscoveryAssociations
	ec2:DescribeIpamScopes
	ec2:DescribeIpams
	ec2:DescribeIpv6Pools
	ec2：DescribeKeyPairs
	ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	ec2:DescribeLocalGatewayRouteTableVpcAssociations
	ec2:DescribeLocalGatewayRouteTables
	ec2:DescribeLocalGatewayVirtualInterfaceGroups
	ec2:DescribeLocalGatewayVirtualInterfaces
	ec2:DescribeLocalGateways
	ec2:DescribeLockedSnapshots
	ec2:DescribeMacHosts
	ec2:DescribeMacModificationTasks
	ec2:DescribeManagedPrefixLists
	ec2:DescribeMovingAddresses
	ec2:DescribeNatGateways

服務前綴	動作
	ec2:DescribeNetworkAcls
	ec2:DescribeNetworkInsightsAccessScopeAnalyses
	ec2:DescribeNetworkInsightsAccessScopes
	ec2:DescribeNetworkInsightsAnalyses
	ec2:DescribeNetworkInsightsPaths
	ec2:DescribeNetworkInterfaceAttribute
	ec2:DescribeNetworkInterfacePermissions
	ec2:DescribeNetworkInterfaces
	ec2:DescribeOutpostLags
	ec2:DescribePlacementGroups
	ec2:DescribePrefixLists
	ec2:DescribePrincipalIdFormat
	ec2:DescribePublicIpv4Pools
	ec2:DescribeRegions
	ec2:DescribeReplaceRootVolumeTasks
	ec2:DescribeReservedInstances
	ec2:DescribeReservedInstancesListings
	ec2:DescribeReservedInstancesModifications
	ec2:DescribeReservedInstancesOfferings
	ec2:DescribeRouteServerEndpoints
	ec2:DescribeRouteServerPeers

服務前綴	動作
	ec2:DescribeRouteServers
	ec2：DescribeRouteTables
	ec2:DescribeScheduledInstanceAvailability
	ec2:DescribeScheduledInstances
	ec2：DescribeSecondaryInterfaces
	ec2：DescribeSecondaryNetworks
	ec2：DescribeSecondarySubnets
	ec2:DescribeSecurityGroupReferences
	ec2:DescribeSecurityGroupRules
	ec2:DescribeSecurityGroupVpcAssociations
	ec2：DescribeSecurityGroups
	ec2:DescribeServiceLinkVirtualInterfaces
	ec2:DescribeSnapshotAttribute
	ec2:DescribeSnapshotTierStatus
	ec2:DescribeSpotDatafeedSubscription
	ec2:DescribeSpotFleetInstances
	ec2:DescribeSpotFleetRequestHistory
	ec2:DescribeSpotFleetRequests
	ec2:DescribeSpotInstanceRequests
	ec2:DescribeSpotPriceHistory
	ec2:DescribeStaleSecurityGroups

服務前綴	動作
	ec2:DescribeStoreImageTasks
	ec2:DescribeTrafficMirrorFilterRules
	ec2:DescribeTrafficMirrorFilters
	ec2:DescribeTrafficMirrorSessions
	ec2:DescribeTrafficMirrorTargets
	ec2:DescribeTransitGatewayAttachments
	ec2:DescribeTransitGatewayConnectPeers
	ec2:DescribeTransitGatewayConnects
	ec2：DescribeTransitGatewayMeteringPolicies
	ec2:DescribeTransitGatewayMulticastDomains
	ec2:DescribeTransitGatewayPeeringAttachments
	ec2:DescribeTransitGatewayPolicyTables
	ec2:DescribeTransitGatewayRouteTableAnnouncements
	ec2:DescribeTransitGatewayRouteTables
	ec2:DescribeTransitGatewayVpcAttachments
	ec2:DescribeTransitGateways
	ec2:DescribeTrunkInterfaceAssociations
	ec2:DescribeVerifiedAccessEndpoints
	ec2:DescribeVerifiedAccessGroups
	ec2:DescribeVerifiedAccessInstanceLoggingConfigurations
	ec2:DescribeVerifiedAccessInstances

服務前綴	動作
	ec2:DescribeVerifiedAccessTrustProviders
	ec2:DescribeVolumeAttribute
	ec2:DescribeVolumeStatus
	ec2:DescribeVolumes
	ec2:DescribeVolumesModifications
	ec2:DescribeVpcAttribute
	ec2:DescribeVpcBlockPublicAccessExclusions
	ec2:DescribeVpcBlockPublicAccessOptions
	ec2:DescribeVpcClassicLink
	ec2:DescribeVpcClassicLinkDnsSupport
	ec2：DescribeVpcEncryptionControls
	ec2:DescribeVpcEndpointAssociations
	ec2:DescribeVpcEndpointConnectionNotifications
	ec2:DescribeVpcEndpointConnections
	ec2:DescribeVpcEndpointServiceConfigurations
	ec2:DescribeVpcEndpointServicePermissions
	ec2:DescribeVpcEndpointServices
	ec2:DescribeVpcEndpoints
	ec2:DescribeVpcPeeringConnections
	ec2：DescribeVpcs
	ec2：DescribeVpnConcentrators

服務前綴	動作
	ec2:DescribeVpnConnections
	ec2:DescribeVpnGateways
	ec2:DetachClassicLinkVpc
	ec2：DetachImageWatermark
	ec2:DetachInternetGateway
	ec2:DetachNetworkInterface
	ec2:DetachVerifiedAccessTrustProvider
	ec2:DetachVolume
	ec2:DetachVpnGateway
	ec2:DisableAddressTransfer
	ec2:DisableAllowedImagesSettings
	ec2:DisableAwsNetworkPerformanceMetricSubscription
	ec2：DisableCapacityManager
	ec2:DisableEbsEncryptionByDefault
	ec2:DisableFastLaunch
	ec2:DisableFastSnapshotRestores
	ec2:DisableImage
	ec2:DisableImageBlockPublicAccess
	ec2:DisableImageDeprecation
	ec2:DisableImageDeregistrationProtection
	ec2：DisableInstanceSqlHaStandbyDetections

服務前綴	動作
	ec2:DisableIpamOrganizationAdminAccount
	ec2：DisableIpamPolicy
	ec2:DisableRouteServerPropagation
	ec2:DisableSerialConsoleAccess
	ec2:DisableSnapshotBlockPublicAccess
	ec2:DisableTransitGatewayRouteTablePropagation
	ec2:DisableVgwRoutePropagation
	ec2:DisableVpcClassicLink
	ec2:DisableVpcClassicLinkDnsSupport
	ec2:DisassociateAddress
	ec2:DisassociateCapacityReservationBillingOwner
	ec2:DisassociateClientVpnTargetNetwork
	ec2:DisassociateEnclaveCertificateIamRole
	ec2:DisassociateIamInstanceProfile
	ec2:DisassociateInstanceEventWindow
	ec2:DisassociateIpamByoasn
	ec2:DisassociateIpamResourceDiscovery
	ec2:DisassociateNatGatewayAddress
	ec2:DisassociateRouteServer
	ec2:DisassociateRouteTable
	ec2:DisassociateSecurityGroupVpc

服務前綴	動作
	ec2:DisassociateSubnetCidrBlock
	ec2:DisassociateTransitGatewayMulticastDomain
	ec2:DisassociateTransitGatewayPolicyTable
	ec2:DisassociateTransitGatewayRouteTable
	ec2:DisassociateTrunkInterface
	ec2:DisassociateVpcCidrBlock
	ec2:EnableAddressTransfer
	ec2:EnableAllowedImagesSettings
	ec2:EnableAwsNetworkPerformanceMetricSubscription
	ec2：EnableCapacityManager
	ec2:EnableEbsEncryptionByDefault
	ec2:EnableFastLaunch
	ec2:EnableFastSnapshotRestores
	ec2:EnableImage
	ec2:EnableImageBlockPublicAccess
	ec2:EnableImageDeprecation
	ec2:EnableImageDeregistrationProtection
	ec2：EnableInstanceSqlHaStandbyDetections
	ec2:EnableIpamOrganizationAdminAccount
	ec2：EnableIpamPolicy
	ec2:EnableReachabilityAnalyzerOrganizationSharing

服務前綴	動作
	ec2:EnableRouteServerPropagation
	ec2:EnableSerialConsoleAccess
	ec2:EnableSnapshotBlockPublicAccess
	ec2:EnableTransitGatewayRouteTablePropagation
	ec2:EnableVgwRoutePropagation
	ec2:EnableVolumeIO
	ec2:EnableVpcClassicLink
	ec2:EnableVpcClassicLinkDnsSupport
	ec2:ExportClientVpnClientCertificateRevocationList
	ec2:ExportClientVpnClientConfiguration
	ec2:ExportImage
	ec2:ExportTransitGatewayRoutes
	ec2:ExportVerifiedAccessInstanceClientConfiguration
	ec2:GetActiveVpnTunnelStatus
	ec2:GetAllowedImagesSettings
	ec2:GetAssociatedEnclaveCertificateIamRoles
	ec2:GetAssociatedIpv6PoolCidrs
	ec2:GetAwsNetworkPerformanceData
	ec2：GetCapacityManagerAttributes
	ec2：GetCapacityManagerMetricData
	ec2：GetCapacityManagerMetricDimensions

服務前綴	動作
	ec2:GetCapacityReservationUsage
	ec2:GetCoipPoolUsage
	ec2:GetConsoleOutput
	ec2:GetConsoleScreenshot
	ec2:GetDeclarativePoliciesReportSummary
	ec2:GetDefaultCreditSpecification
	ec2:GetEbsDefaultKmsKeyId
	ec2:GetEbsEncryptionByDefault
	ec2：GetEnabledIpamPolicy
	ec2:GetFlowLogsIntegrationTemplate
	ec2:GetGroupsForCapacityReservation
	ec2:GetHostReservationPurchasePreview
	ec2：GetImageAncestry
	ec2:GetImageBlockPublicAccessState
	ec2:GetInstanceMetadataDefaults
	ec2:GetInstanceTpmEkPub
	ec2:GetInstanceTypesFromInstanceRequirements
	ec2:GetInstanceUefiData
	ec2:GetIpamAddressHistory
	ec2:GetIpamDiscoveredAccounts
	ec2:GetIpamDiscoveredPublicAddresses

服務前綴	動作
	ec2:GetIpamDiscoveredResourceCidrs
	ec2：GetIpamPolicyAllocationRules
	ec2：GetIpamPolicyOrganizationTargets
	ec2:GetIpamPoolAllocations
	ec2:GetIpamPoolCidrs
	ec2：GetIpamPrefixListResolverRules
	ec2：GetIpamPrefixListResolverVersionEntries
	ec2：GetIpamPrefixListResolverVersions
	ec2:GetIpamResourceCidrs
	ec2:GetLaunchTemplateData
	ec2:GetManagedPrefixListAssociations
	ec2:GetManagedPrefixListEntries
	ec2：GetManagedResourceVisibility
	ec2:GetNetworkInsightsAccessScopeAnalysisFindings
	ec2:GetNetworkInsightsAccessScopeContent
	ec2:GetPasswordData
	ec2:GetReservedInstancesExchangeQuote
	ec2:GetRouteServerAssociations
	ec2:GetRouteServerPropagations
	ec2:GetRouteServerRoutingDatabase
	ec2:GetSecurityGroupsForVpc

服務前綴	動作
	ec2:GetSerialConsoleAccessStatus
	ec2:GetSnapshotBlockPublicAccessState
	ec2:GetSpotPlacementScores
	ec2:GetSubnetCidrReservations
	ec2:GetTransitGatewayAttachmentPropagations
	ec2：GetTransitGatewayMeteringPolicyEntries
	ec2:GetTransitGatewayMulticastDomainAssociations
	ec2:GetTransitGatewayPolicyTableAssociations
	ec2:GetTransitGatewayPolicyTableEntries
	ec2:GetTransitGatewayPrefixListReferences
	ec2:GetTransitGatewayRouteTableAssociations
	ec2:GetTransitGatewayRouteTablePropagations
	ec2:GetVerifiedAccessEndpointPolicy
	ec2:GetVerifiedAccessEndpointTargets
	ec2:GetVerifiedAccessGroupPolicy
	ec2：GetVpcResourcesBlockingEncryptionEnforcement
	ec2:GetVpnConnectionDeviceSampleConfiguration
	ec2:GetVpnConnectionDeviceTypes
	ec2:GetVpnTunnelReplacementStatus
	ec2:ImportClientVpnClientCertificateRevocationList
	ec2:ImportImage

服務前綴	動作
	ec2:ImportInstance
	ec2：ImportKeyPair
	ec2:ImportSnapshot
	ec2:ImportVolume
	ec2:ListImagesInRecycleBin
	ec2:ListSnapshotsInRecycleBin
	ec2：ListVolumesInRecycleBin
	ec2:LockSnapshot
	ec2:ModifyAddressAttribute
	ec2:ModifyAvailabilityZoneGroup
	ec2:ModifyCapacityReservation
	ec2:ModifyCapacityReservationFleet
	ec2:ModifyClientVpnEndpoint
	ec2:ModifyDefaultCreditSpecification
	ec2:ModifyEbsDefaultKmsKeyId
	ec2:ModifyFleet
	ec2:ModifyFpgaImageAttribute
	ec2:ModifyHosts
	ec2:ModifyIdFormat
	ec2:ModifyIdentityIdFormat
	ec2:ModifyImageAttribute

服務前綴	動作
	ec2:ModifyInstanceAttribute
	ec2:ModifyInstanceCapacityReservationAttributes
	ec2：ModifyInstanceConnectEndpoint
	ec2:ModifyInstanceCpuOptions
	ec2:ModifyInstanceCreditSpecification
	ec2:ModifyInstanceEventStartTime
	ec2:ModifyInstanceEventWindow
	ec2:ModifyInstanceMaintenanceOptions
	ec2:ModifyInstanceMetadataDefaults
	ec2:ModifyInstanceMetadataOptions
	ec2:ModifyInstanceNetworkPerformanceOptions
	ec2:ModifyInstancePlacement
	ec2:ModifyIpam
	ec2：ModifyIpamPolicyAllocationRules
	ec2:ModifyIpamPool
	ec2：ModifyIpamPoolAllocation
	ec2：ModifyIpamPrefixListResolver
	ec2：ModifyIpamPrefixListResolverTarget
	ec2:ModifyIpamResourceCidr
	ec2:ModifyIpamResourceDiscovery
	ec2:ModifyIpamScope

服務前綴	動作
	ec2:ModifyLaunchTemplate
	ec2:ModifyLocalGatewayRoute
	ec2:ModifyManagedPrefixList
	ec2：ModifyManagedResourceVisibility
	ec2:ModifyNetworkInterfaceAttribute
	ec2:ModifyPrivateDnsNameOptions
	ec2:ModifyPublicIpDnsNameOptions
	ec2:ModifyReservedInstances
	ec2:ModifyRouteServer
	ec2:ModifySecurityGroupRules
	ec2:ModifySnapshotAttribute
	ec2:ModifySnapshotTier
	ec2:ModifySpotFleetRequest
	ec2：ModifySubnetAttribute
	ec2:ModifyTrafficMirrorFilterNetworkServices
	ec2:ModifyTrafficMirrorFilterRule
	ec2:ModifyTrafficMirrorSession
	ec2:ModifyTransitGateway
	ec2：ModifyTransitGatewayMeteringPolicy
	ec2:ModifyTransitGatewayPrefixListReference
	ec2:ModifyTransitGatewayVpcAttachment

服務前綴	動作
	ec2:ModifyVerifiedAccessEndpoint
	ec2:ModifyVerifiedAccessEndpointPolicy
	ec2:ModifyVerifiedAccessGroup
	ec2:ModifyVerifiedAccessGroupPolicy
	ec2:ModifyVerifiedAccessInstance
	ec2:ModifyVerifiedAccessInstanceLoggingConfiguration
	ec2:ModifyVerifiedAccessTrustProvider
	ec2:ModifyVolume
	ec2:ModifyVolumeAttribute
	ec2：ModifyVpcAttribute
	ec2:ModifyVpcBlockPublicAccessExclusion
	ec2:ModifyVpcBlockPublicAccessOptions
	ec2：ModifyVpcEncryptionControl
	ec2:ModifyVpcEndpoint
	ec2:ModifyVpcEndpointConnectionNotification
	ec2:ModifyVpcEndpointServiceConfiguration
	ec2:ModifyVpcEndpointServicePayerResponsibility
	ec2:ModifyVpcEndpointServicePermissions
	ec2:ModifyVpcPeeringConnectionOptions
	ec2:ModifyVpcTenancy
	ec2:ModifyVpnConnection

服務前綴	動作
	ec2:ModifyVpnConnectionOptions
	ec2:ModifyVpnTunnelCertificate
	ec2:ModifyVpnTunnelOptions
	ec2:MonitorInstances
	ec2:MoveAddressToVpc
	ec2:MoveByoipCidrToIpam
	ec2:MoveCapacityReservationInstances
	ec2:ProvisionByoipCidr
	ec2:ProvisionIpamByoasn
	ec2:ProvisionIpamPoolCidr
	ec2:ProvisionPublicIpv4PoolCidr
	ec2:PurchaseCapacityBlockExtension
	ec2:PurchaseHostReservation
	ec2:PurchaseReservedInstancesOffering
	ec2:PurchaseScheduledInstances
	ec2:RebootInstances
	ec2:RegisterImage
	ec2:RegisterInstanceEventNotificationAttributes
	ec2:RegisterTransitGatewayMulticastGroupMembers
	ec2:RegisterTransitGatewayMulticastGroupSources
	ec2:RejectCapacityReservationBillingOwnership

服務前綴	動作
	ec2：RejectTransitGatewayClientVpnAttachment
	ec2:RejectTransitGatewayMulticastDomainAssociations
	ec2:RejectTransitGatewayPeeringAttachment
	ec2:RejectTransitGatewayVpcAttachment
	ec2:RejectVpcEndpointConnections
	ec2:RejectVpcPeeringConnection
	ec2:ReleaseAddress
	ec2:ReleaseHosts
	ec2:ReleaseIpamPoolAllocation
	ec2:ReplaceIamInstanceProfileAssociation
	ec2:ReplaceImageCriteriaInAllowedImagesSettings
	ec2:ReplaceNetworkAclAssociation
	ec2:ReplaceNetworkAclEntry
	ec2:ReplaceRoute
	ec2:ReplaceRouteTableAssociation
	ec2:ReplaceTransitGatewayRoute
	ec2:ReplaceVpnTunnel
	ec2:ReportInstanceStatus
	ec2:RequestSpotFleet
	ec2:RequestSpotInstances
	ec2:ResetAddressAttribute

服務前綴	動作
	ec2:ResetEbsDefaultKmsKeyId
	ec2:ResetFpgaImageAttribute
	ec2:ResetImageAttribute
	ec2:ResetInstanceAttribute
	ec2:ResetNetworkInterfaceAttribute
	ec2:ResetSnapshotAttribute
	ec2:RestoreAddressToClassic
	ec2:RestoreImageFromRecycleBin
	ec2:RestoreManagedPrefixListVersion
	ec2:RestoreSnapshotFromRecycleBin
	ec2:RestoreSnapshotTier
	ec2：RestoreVolumeFromRecycleBin
	ec2:RevokeClientVpnIngress
	ec2:RevokeSecurityGroupEgress
	ec2:RevokeSecurityGroupIngress
	ec2:RunInstances
	ec2:RunScheduledInstances
	ec2:SearchLocalGatewayRoutes
	ec2:SearchTransitGatewayMulticastGroups
	ec2:SearchTransitGatewayRoutes
	ec2:SendDiagnosticInterrupt

服務前綴	動作
	ec2:StartDeclarativePoliciesReport
	ec2:StartInstances
	ec2:StartNetworkInsightsAccessScopeAnalysis
	ec2:StartNetworkInsightsAnalysis
	ec2:StartVpcEndpointServicePrivateDnsVerification
	ec2:TerminateClientVpnConnections
	ec2:UnassignIpv6Addresses
	ec2:UnassignPrivateIpAddresses
	ec2:UnassignPrivateNatGatewayAddress
	ec2:UnlockSnapshot
	ec2:UnmonitorInstances
	ec2：UpdateCapacityManagerOrganizationsAccess
	ec2：UpdateInterruptibleCapacityReservationAllocation
	ec2:UpdateSecurityGroupRuleDescriptionsEgress
	ec2:UpdateSecurityGroupRuleDescriptionsIngress
	ec2:WithdrawByoipCidr

服務前綴	動作
ecr	ecr:BatchCheckLayerAvailability ecr:BatchDeleteImage ecr:BatchGetImage ecr:BatchGetRepositoryScanningConfiguration ecr:CompleteLayerUpload ecr:CreatePullThroughCacheRule ecr:CreateRepositoryCreationTemplate ecr:DeleteLifecyclePolicy ecr:DeletePullThroughCacheRule ecr:DeleteRegistryPolicy ecr:DeleteRepository ecr:DeleteRepositoryCreationTemplate ecr:DeleteRepositoryPolicy ecr：DeleteSigningConfiguration ecr:DescribeImageReplicationStatus ecr:DescribeImageScanFindings ecr:DescribeImages ecr:DescribePullThroughCacheRules ecr:DescribeRegistry ecr:DescribeRepositories ecr:DescribeRepositoryCreationTemplates

服務前綴	動作
	ecr:GetAccountSetting ecr:GetAuthorizationToken ecr:GetDownloadUrlForLayer ecr:GetLifecyclePolicy ecr:GetLifecyclePolicyPreview ecr:GetRegistryPolicy ecr:GetRegistryScanningConfiguration ecr:GetRepositoryPolicy ecr：GetSigningConfiguration ecr:InitiateLayerUpload ecr:ListImages ecr：ListPullTimeUpdateExclusions ecr:PutAccountSetting ecr:PutImage ecr:PutImageScanningConfiguration ecr:PutRegistryPolicy ecr:PutRegistryScanningConfiguration ecr:PutReplicationConfiguration ecr:StartImageScan ecr:StartLifecyclePolicyPreview ecr:UpdatePullThroughCacheRule

服務前綴	動作
	ecr:UpdateRepositoryCreationTemplate ecr:UploadLayerPart ecr:ValidatePullThroughCacheRule

服務前綴	動作
ecr-public	ecr-public:BatchCheckLayerAvailability
	ecr-public:BatchDeleteImage
	ecr-public:CompleteLayerUpload
	ecr-public:CreateRepository
	ecr-public:DeleteRepository
	ecr-public:DeleteRepositoryPolicy
	ecr-public:DescribeImages
	ecr-public:DescribeRegistries
	ecr-public:DescribeRepositories
	ecr-public:GetAuthorizationToken
	ecr-public:GetRegistryCatalogData
	ecr-public:GetRepositoryCatalogData
	ecr-public:GetRepositoryPolicy
	ecr-public:InitiateLayerUpload
	ecr-public:PutImage
	ecr-public:PutRegistryCatalogData
	ecr-public:PutRepositoryCatalogData
	ecr-public:SetRepositoryPolicy
	ecr-public:UploadLayerPart

服務前綴	動作
ecs	ecs：ContinueServiceDeployment ecs:CreateCapacityProvider ecs:CreateCluster ecs：CreateDaemon ecs:CreateService ecs:CreateTaskSet ecs:DeleteAccountSetting ecs:DeleteAttributes ecs:DeleteCapacityProvider ecs:DeleteCluster ecs：DeleteDaemon ecs：DeleteDaemonTaskDefinition ecs：DeleteExpressGatewayService ecs:DeleteService ecs:DeleteTaskDefinitions ecs:DeleteTaskSet ecs:DeregisterContainerInstance ecs:DeregisterTaskDefinition ecs:DescribeCapacityProviders ecs:DescribeClusters ecs:DescribeContainerInstances

服務前綴	動作
	ecs：DescribeDaemon ecs：DescribeDaemonDeployments ecs：DescribeDaemonRevisions ecs：DescribeDaemonTaskDefinition ecs：DescribeExpressGatewayService ecs:DescribeServiceDeployments ecs:DescribeServiceRevisions ecs:DescribeServices ecs:DescribeTaskDefinition ecs:DescribeTaskSets ecs:DescribeTasks ecs:DiscoverPollEndpoint ecs:ExecuteCommand ecs:GetTaskProtection ecs:ListAccountSettings ecs:ListAttributes ecs:ListClusters ecs:ListContainerInstances ecs：ListDaemonDeployments ecs：ListDaemonTaskDefinitions ecs：ListDaemons

服務前綴	動作
	ecs:ListServiceDeployments ecs:ListServices ecs:ListServicesByNamespace ecs:ListTaskDefinitionFamilies ecs:ListTaskDefinitions ecs:ListTasks ecs:PutAccountSetting ecs:PutAccountSettingDefault ecs:PutAttributes ecs:PutClusterCapacityProviders ecs:RegisterContainerInstance ecs：RegisterDaemonTaskDefinition ecs:RunTask ecs:StartTask ecs:StopServiceDeployment ecs:StopTask ecs:SubmitAttachmentStateChanges ecs:SubmitContainerStateChange ecs:SubmitTaskStateChange ecs:UpdateCapacityProvider ecs:UpdateCluster

服務前綴	動作
	ecs:UpdateClusterSettings ecs:UpdateContainerAgent ecs:UpdateContainerInstancesState ecs：UpdateDaemon ecs：UpdateExpressGatewayService ecs:UpdateService ecs:UpdateServicePrimaryTaskSet ecs:UpdateTaskProtection ecs:UpdateTaskSet

服務前綴	動作
eks	eks:AssociateAccessPolicy
	eks:AssociateEncryptionConfig
	eks:AssociateIdentityProviderConfig
	eks：CancelUpdate
	eks:CreateAccessEntry
	eks:CreateAddon
	eks:CreateCluster
	eks:CreateEksAnywhereSubscription
	eks:CreateFargateProfile
	eks:CreateNodegroup
	eks:DeleteAccessEntry
	eks:DeleteAddon
	eks：DeleteCapability
	eks:DeleteCluster
	eks:DeleteEksAnywhereSubscription
	eks:DeleteFargateProfile
	eks:DeleteNodegroup
	eks:DeletePodIdentityAssociation
	eks:DeregisterCluster
	eks:DescribeAccessEntry
	eks:DescribeAddon

服務前綴	動作
	eks:DescribeAddonConfiguration
	eks:DescribeAddonVersions
	eks：DescribeCapability
	eks:DescribeCluster
	eks:DescribeClusterVersions
	eks:DescribeEksAnywhereSubscription
	eks:DescribeFargateProfile
	eks:DescribeIdentityProviderConfig
	eks:DescribeInsight
	eks：DescribeInsightsRefresh
	eks:DescribeNodegroup
	eks:DescribePodIdentityAssociation
	eks:DescribeUpdate
	eks:DisassociateAccessPolicy
	eks:DisassociateIdentityProviderConfig
	eks:ListAccessEntries
	eks:ListAccessPolicies
	eks:ListAddons
	eks:ListAssociatedAccessPolicies
	eks：ListCapabilities
	eks:ListClusters

服務前綴	動作
	eks:ListEksAnywhereSubscriptions
	eks:ListFargateProfiles
	eks:ListIdentityProviderConfigs
	eks:ListInsights
	eks:ListNodegroups
	eks:ListPodIdentityAssociations
	eks:ListUpdates
	eks:RegisterCluster
	eks：StartInsightsRefresh
	eks:UpdateAccessEntry
	eks:UpdateAddon
	eks：UpdateCapability
	eks:UpdateClusterConfig
	eks:UpdateClusterVersion
	eks:UpdateEksAnywhereSubscription
	eks:UpdateNodegroupConfig
	eks:UpdateNodegroupVersion
	eks:UpdatePodIdentityAssociation

服務前綴	動作
elasticache	elasticache:AuthorizeCacheSecurityGroupIngress elasticache:BatchApplyUpdateAction elasticache:BatchStopUpdateAction elasticache:CompleteMigration elasticache:CopyServerlessCacheSnapshot elasticache:CopySnapshot elasticache:CreateCacheCluster elasticache:CreateCacheParameterGroup elasticache:CreateCacheSecurityGroup elasticache:CreateCacheSubnetGroup elasticache:CreateGlobalReplicationGroup elasticache:CreateReplicationGroup elasticache:CreateServerlessCache elasticache:CreateServerlessCacheSnapshot elasticache:CreateSnapshot elasticache:CreateUser elasticache:CreateUserGroup elasticache:DecreaseNodeGroupsInGlobalReplicationGroup elasticache:DecreaseReplicaCount elasticache:DeleteCacheCluster elasticache:DeleteCacheParameterGroup

服務前綴	動作
	elasticache:DeleteCacheSecurityGroup elasticache:DeleteCacheSubnetGroup elasticache:DeleteGlobalReplicationGroup elasticache:DeleteReplicationGroup elasticache:DeleteServerlessCache elasticache:DeleteServerlessCacheSnapshot elasticache:DeleteSnapshot elasticache:DeleteUser elasticache:DeleteUserGroup elasticache:DescribeCacheClusters elasticache:DescribeCacheEngineVersions elasticache:DescribeCacheParameterGroups elasticache:DescribeCacheParameters elasticache:DescribeCacheSecurityGroups elasticache:DescribeCacheSubnetGroups elasticache:DescribeEngineDefaultParameters elasticache:DescribeEvents elasticache:DescribeGlobalReplicationGroups elasticache:DescribeReplicationGroups elasticache:DescribeReservedCacheNodes elasticache:DescribeReservedCacheNodesOfferings

服務前綴	動作
	elasticache:DescribeServerlessCacheSnapshots elasticache:DescribeServerlessCaches elasticache:DescribeServiceUpdates elasticache:DescribeSnapshots elasticache:DescribeUpdateActions elasticache:DescribeUserGroups elasticache:DescribeUsers elasticache:DisassociateGlobalReplicationGroup elasticache:ExportServerlessCacheSnapshot elasticache:FailoverGlobalReplicationGroup elasticache:IncreaseNodeGroupsInGlobalReplicationGroup elasticache:IncreaseReplicaCount elasticache:ListAllowedNodeTypeModifications elasticache:ModifyCacheCluster elasticache:ModifyCacheParameterGroup elasticache:ModifyCacheSubnetGroup elasticache:ModifyGlobalReplicationGroup elasticache:ModifyReplicationGroup elasticache:ModifyReplicationGroupShardConfiguration elasticache:ModifyServerlessCache elasticache:ModifyUser

服務前綴	動作
	elasticache:ModifyUserGroup elasticache:PurchaseReservedCacheNodesOffering elasticache:RebalanceSlotsInGlobalReplicationGroup elasticache:RebootCacheCluster elasticache:ResetCacheParameterGroup elasticache:RevokeCacheSecurityGroupIngress elasticache:StartMigration elasticache:TestFailover elasticache:TestMigration

服務前綴	動作
elasticbeanstalk	elasticbeanstalk:AbortEnvironmentUpdate
	elasticbeanstalk:ApplyEnvironmentManagedAction
	elasticbeanstalk:AssociateEnvironmentOperationsRole
	elasticbeanstalk:CheckDNSAvailability
	elasticbeanstalk:ComposeEnvironments
	elasticbeanstalk:CreateApplication
	elasticbeanstalk:CreateApplicationVersion
	elasticbeanstalk:CreateConfigurationTemplate
	elasticbeanstalk:CreateEnvironment
	elasticbeanstalk:CreatePlatformVersion
	elasticbeanstalk:CreateStorageLocation
	elasticbeanstalk:DeleteApplication
	elasticbeanstalk:DeleteApplicationVersion
	elasticbeanstalk:DeleteConfigurationTemplate
	elasticbeanstalk:DeleteEnvironmentConfiguration
	elasticbeanstalk:DeletePlatformVersion
	elasticbeanstalk:DescribeAccountAttributes
	elasticbeanstalk:DescribeApplicationVersions
	elasticbeanstalk:DescribeApplications
	elasticbeanstalk:DescribeConfigurationOptions
	elasticbeanstalk:DescribeConfigurationSettings

服務前綴	動作
	elasticbeanstalk:DescribeEnvironmentHealth
	elasticbeanstalk:DescribeEnvironmentManagedActionHistory
	elasticbeanstalk:DescribeEnvironmentManagedActions
	elasticbeanstalk:DescribeEnvironmentResources
	elasticbeanstalk:DescribeEnvironments
	elasticbeanstalk:DescribeEvents
	elasticbeanstalk:DescribeInstancesHealth
	elasticbeanstalk:DescribePlatformVersion
	elasticbeanstalk:DisassociateEnvironmentOperationsRole
	elasticbeanstalk:ListAvailableSolutionStacks
	elasticbeanstalk:ListPlatformBranches
	elasticbeanstalk:ListPlatformVersions
	elasticbeanstalk:RebuildEnvironment
	elasticbeanstalk:RequestEnvironmentInfo
	elasticbeanstalk:RestartAppServer
	elasticbeanstalk:RetrieveEnvironmentInfo
	elasticbeanstalk:SwapEnvironmentCNAMEs
	elasticbeanstalk:TerminateEnvironment
	elasticbeanstalk:UpdateApplication
	elasticbeanstalk:UpdateApplicationResourceLifecycle
	elasticbeanstalk:UpdateApplicationVersion

服務前綴	動作
	elasticbeanstalk:UpdateConfigurationTemplate elasticbeanstalk:UpdateEnvironment elasticbeanstalk:ValidateConfigurationSettings

服務前綴	動作
elasticfilesystem	elasticfilesystem:CreateAccessPoint elasticfilesystem:CreateFileSystem elasticfilesystem:CreateMountTarget elasticfilesystem:CreateReplicationConfiguration elasticfilesystem:DeleteAccessPoint elasticfilesystem:DeleteFileSystem elasticfilesystem:DeleteFileSystemPolicy elasticfilesystem:DeleteMountTarget elasticfilesystem:DeleteReplicationConfiguration elasticfilesystem:DescribeAccessPoints elasticfilesystem:DescribeAccountPreferences elasticfilesystem:DescribeBackupPolicy elasticfilesystem:DescribeFileSystemPolicy elasticfilesystem:DescribeFileSystems elasticfilesystem:DescribeLifecycleConfiguration elasticfilesystem:DescribeMountTargetSecurityGroups elasticfilesystem:DescribeMountTargets elasticfilesystem:DescribeReplicationConfigurations elasticfilesystem:ModifyMountTargetSecurityGroups elasticfilesystem:PutAccountPreferences elasticfilesystem:PutBackupPolicy

服務前綴	動作
	elasticfilesystem:PutFileSystemPolicy elasticfilesystem:PutLifecycleConfiguration elasticfilesystem:UpdateFileSystem elasticfilesystem:UpdateFileSystemProtection

服務前綴	動作
elasticloadbalancing	elasticloadbalancing:AddListenerCertificates
	elasticloadbalancing:AddTrustStoreRevocations
	elasticloadbalancing:ApplySecurityGroupsToLoadBalancer
	elasticloadbalancing:AttachLoadBalancerToSubnets
	elasticloadbalancing:ConfigureHealthCheck
	elasticloadbalancing:CreateAppCookieStickinessPolicy
	elasticloadbalancing:CreateLBCookieStickinessPolicy
	elasticloadbalancing:CreateListener
	elasticloadbalancing:CreateLoadBalancer
	elasticloadbalancing:CreateLoadBalancerListeners
	elasticloadbalancing:CreateLoadBalancerPolicy
	elasticloadbalancing:CreateRule
	elasticloadbalancing:CreateTargetGroup
	elasticloadbalancing:CreateTrustStore
	elasticloadbalancing:DeleteListener
	elasticloadbalancing:DeleteLoadBalancer
	elasticloadbalancing:DeleteLoadBalancerListeners
	elasticloadbalancing:DeleteLoadBalancerPolicy
	elasticloadbalancing:DeleteRule
	elasticloadbalancing:DeleteSharedTrustStoreAssociation
	elasticloadbalancing:DeleteTargetGroup

服務前綴	動作
	elasticloadbalancing:DeleteTrustStore
	elasticloadbalancing:DeregisterInstancesFromLoadBalancer
	elasticloadbalancing:DeregisterTargets
	elasticloadbalancing:DescribeAccountLimits
	elasticloadbalancing:DescribeCapacityReservation
	elasticloadbalancing:DescribeInstanceHealth
	elasticloadbalancing:DescribeListenerAttributes
	elasticloadbalancing:DescribeListenerCertificates
	elasticloadbalancing:DescribeListeners
	elasticloadbalancing:DescribeLoadBalancerAttributes
	elasticloadbalancing:DescribeLoadBalancerPolicies
	elasticloadbalancing:DescribeLoadBalancerPolicyTypes
	elasticloadbalancing:DescribeLoadBalancers
	elasticloadbalancing:DescribeRules
	elasticloadbalancing:DescribeSSLPolicies
	elasticloadbalancing:DescribeTargetGroupAttributes
	elasticloadbalancing:DescribeTargetGroups
	elasticloadbalancing:DescribeTargetHealth
	elasticloadbalancing:DescribeTrustStoreAssociations
	elasticloadbalancing:DescribeTrustStoreRevocations
	elasticloadbalancing:DescribeTrustStores

服務前綴	動作
	elasticloadbalancing:DetachLoadBalancerFromSubnets
	elasticloadbalancing:DisableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:EnableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:GetResourcePolicy
	elasticloadbalancing:GetTrustStoreCaCertificatesBundle
	elasticloadbalancing:GetTrustStoreRevocationContent
	elasticloadbalancing:ModifyCapacityReservation
	elasticloadbalancing:ModifyIpPools
	elasticloadbalancing:ModifyListener
	elasticloadbalancing:ModifyLoadBalancerAttributes
	elasticloadbalancing:ModifyRule
	elasticloadbalancing:ModifyTargetGroup
	elasticloadbalancing:ModifyTargetGroupAttributes
	elasticloadbalancing:ModifyTrustStore
	elasticloadbalancing:RegisterInstancesWithLoadBalancer
	elasticloadbalancing:RegisterTargets
	elasticloadbalancing:RemoveListenerCertificates
	elasticloadbalancing:RemoveTrustStoreRevocations
	elasticloadbalancing:SetIpAddressType
	elasticloadbalancing:SetLoadBalancerListenerSSLCertificate
	elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer

服務前綴	動作
	elasticloadbalancing:SetLoadBalancerPoliciesOfListener elasticloadbalancing:SetRulePriorities elasticloadbalancing:SetSecurityGroups elasticloadbalancing:SetSubnets

服務前綴	動作
emr-containers	emr-containers:CancelJobRun emr-containers:CreateJobTemplate emr-containers:CreateManagedEndpoint emr-containers:CreateSecurityConfiguration emr-containers:CreateVirtualCluster emr-containers:DeleteJobTemplate emr-containers:DeleteManagedEndpoint emr-containers：DeleteSecurityConfiguration emr-containers:DeleteVirtualCluster emr-containers:DescribeJobRun emr-containers:DescribeJobTemplate emr-containers:DescribeManagedEndpoint emr-containers:DescribeSecurityConfiguration emr-containers:DescribeVirtualCluster emr-containers:GetManagedEndpointSessionCredentials emr-containers:ListJobRuns emr-containers:ListJobTemplates emr-containers:ListManagedEndpoints emr-containers:ListSecurityConfigurations emr-containers:ListVirtualClusters emr-containers:StartJobRun

服務前綴	動作
emr-serverless	emr-serverless:CancelJobRun
	emr-serverless:CreateApplication
	emr-serverless:DeleteApplication
	emr-serverless:GetApplication
	emr-serverless:GetDashboardForJobRun
	emr-serverless:GetJobRun
	emr-serverless：GetResourceDashboard
	emr-serverless：GetSession
	emr-serverless：GetSessionEndpoint
	emr-serverless:ListApplications
	emr-serverless:ListJobRunAttempts
	emr-serverless:ListJobRuns
	emr-serverless：ListSessions
	emr-serverless:StartApplication
	emr-serverless:StartJobRun
	emr-serverless:StopApplication
	emr-serverless：TerminateSession
	emr-serverless:UpdateApplication

服務前綴	動作
es	es:AcceptInboundConnection es:AcceptInboundCrossClusterSearchConnection es:AssociatePackage es:AuthorizeVpcEndpointAccess es:CancelElasticsearchServiceSoftwareUpdate es:CancelServiceSoftwareUpdate es:CreateDomain es:CreateElasticsearchDomain es：CreateIndex es:CreateOutboundConnection es:CreateOutboundCrossClusterSearchConnection es:CreatePackage es:CreateVpcEndpoint es:DeleteDomain es:DeleteElasticsearchDomain es:DeleteElasticsearchServiceRole es:DeleteInboundConnection es:DeleteInboundCrossClusterSearchConnection es：DeleteIndex es:DeleteOutboundConnection es:DeleteOutboundCrossClusterSearchConnection

服務前綴	動作
	es:DeletePackage es:DeleteVpcEndpoint es:DescribeDomain es:DescribeDomainAutoTunes es:DescribeDomainChangeProgress es:DescribeDomainConfig es:DescribeDomainHealth es:DescribeDomainNodes es:DescribeDomains es:DescribeDryRunProgress es:DescribeElasticsearchDomain es:DescribeElasticsearchDomainConfig es:DescribeElasticsearchDomains es:DescribeElasticsearchInstanceTypeLimits es:DescribeInboundConnections es:DescribeInboundCrossClusterSearchConnections es:DescribeInstanceTypeLimits es:DescribeOutboundConnections es:DescribeOutboundCrossClusterSearchConnections es:DescribePackages es:DescribeReservedElasticsearchInstanceOfferings

服務前綴	動作
	es:DescribeReservedElasticsearchInstances es:DescribeReservedInstanceOfferings es:DescribeReservedInstances es:DescribeVpcEndpoints es:DissociatePackage es:DissociatePackages es:GetCompatibleElasticsearchVersions es:GetCompatibleVersions es:GetDataSource es:GetDomainMaintenanceStatus es:GetPackageVersionHistory es:GetUpgradeHistory es:GetUpgradeStatus es:ListDataSources es:ListDomainNames es:ListDomainsForPackage es:ListElasticsearchInstanceTypes es:ListElasticsearchVersions es：ListInsights es:ListInstanceTypeDetails es:ListPackagesForDomain

服務前綴	動作
	es:ListScheduledActions es:ListVersions es:ListVpcEndpointAccess es:ListVpcEndpoints es:ListVpcEndpointsForDomain es:PurchaseReservedElasticsearchInstanceOffering es:PurchaseReservedInstanceOffering es:RejectInboundConnection es:RejectInboundCrossClusterSearchConnection es:RevokeVpcEndpointAccess es：RollbackServiceSoftwareUpdate es:StartDomainMaintenance es:StartElasticsearchServiceSoftwareUpdate es:StartServiceSoftwareUpdate es:UpdateDataSource es:UpdateDomainConfig es:UpdateElasticsearchDomainConfig es：UpdateIndex es:UpdatePackage es:UpdatePackageScope es:UpdateScheduledAction

服務前綴	動作
	es:UpdateVpcEndpoint es:UpgradeDomain es:UpgradeElasticsearchDomain

服務前綴	動作
events	events:ActivateEventSource
	events:CancelReplay
	events:CreateApiDestination
	events:CreateArchive
	events:CreateConnection
	events:CreateEndpoint
	events:CreateEventBus
	events:CreatePartnerEventSource
	events:DeactivateEventSource
	events:DeauthorizeConnection
	events:DeleteApiDestination
	events:DeleteArchive
	events:DeleteConnection
	events:DeleteEndpoint
	events:DeleteEventBus
	events:DeletePartnerEventSource
	events:DeleteRule
	events:DescribeApiDestination
	events:DescribeArchive
	events:DescribeConnection
	events:DescribeEndpoint

服務前綴	動作
	events:DescribeEventBus
	events:DescribeEventSource
	events:DescribePartnerEventSource
	events:DescribeReplay
	events:DescribeRule
	events:DisableRule
	events:EnableRule
	events:ListApiDestinations
	events:ListArchives
	events:ListConnections
	events:ListEndpoints
	events:ListEventBuses
	events:ListEventSources
	events:ListPartnerEventSourceAccounts
	events:ListPartnerEventSources
	events:ListReplays
	events:ListRuleNamesByTarget
	events:ListRules
	events:ListTargetsByRule
	events:PutPermission
	events:PutRule

服務前綴	動作
	events:PutTargets
	events:RemovePermission
	events:RemoveTargets
	events:StartReplay
	events:TestEventPattern
	events:UpdateApiDestination
	events:UpdateArchive
	events:UpdateConnection
	events:UpdateEndpoint
	events:UpdateEventBus

服務前綴	動作
finspace	finspace:CreateEnvironment finspace:CreateKxChangeset finspace:CreateKxCluster finspace:CreateKxDatabase finspace:CreateKxDataview finspace:CreateKxEnvironment finspace:CreateKxScalingGroup finspace:CreateKxUser finspace:CreateKxVolume finspace:CreateUser finspace:DeleteEnvironment finspace:DeleteKxCluster finspace:DeleteKxClusterNode finspace:DeleteKxDatabase finspace:DeleteKxDataview finspace:DeleteKxEnvironment finspace:DeleteKxScalingGroup finspace:DeleteKxUser finspace:DeleteKxVolume finspace:GetEnvironment finspace:GetKxChangeset

服務前綴	動作
	finspace:GetKxCluster finspace:GetKxConnectionString finspace:GetKxDatabase finspace:GetKxDataview finspace:GetKxEnvironment finspace:GetKxScalingGroup finspace:GetKxUser finspace:GetKxVolume finspace:GetLoadSampleDataSetGroupIntoEnvironmentStatus finspace:GetUser finspace:ListEnvironments finspace:ListKxChangesets finspace:ListKxClusterNodes finspace:ListKxClusters finspace:ListKxDatabases finspace:ListKxDataviews finspace:ListKxEnvironments finspace:ListKxScalingGroups finspace:ListKxUsers finspace:ListKxVolumes finspace:ListUsers

服務前綴	動作
	finspace:LoadSampleDataSetGroupIntoEnvironment finspace:ResetUserPassword finspace:UpdateEnvironment finspace:UpdateKxClusterCodeConfiguration finspace:UpdateKxClusterDatabases finspace:UpdateKxDatabase finspace:UpdateKxDataview finspace:UpdateKxEnvironment finspace:UpdateKxEnvironmentNetwork finspace:UpdateKxUser finspace:UpdateKxVolume finspace:UpdateUser
firehose	firehose:CreateDeliveryStream firehose:DeleteDeliveryStream firehose:DescribeDeliveryStream firehose:ListDeliveryStreams firehose:StartDeliveryStreamEncryption firehose:StopDeliveryStreamEncryption firehose:UpdateDestination

服務前綴	動作
fis	fis:CreateExperimentTemplate fis:CreateTargetAccountConfiguration fis:DeleteExperimentTemplate fis:DeleteTargetAccountConfiguration fis:GetAction fis:GetExperiment fis:GetExperimentTargetAccountConfiguration fis:GetExperimentTemplate fis:GetSafetyLever fis:GetTargetAccountConfiguration fis:GetTargetResourceType fis:ListActions fis:ListExperimentResolvedTargets fis:ListExperimentTargetAccountConfigurations fis:ListExperimentTemplates fis:ListExperiments fis:ListTargetAccountConfigurations fis:ListTargetResourceTypes fis:StartExperiment fis:StopExperiment fis:UpdateExperimentTemplate

服務前綴	動作
	fis:UpdateSafetyLeverState fis:UpdateTargetAccountConfiguration

服務前綴	動作
fms	fms:AssociateAdminAccount
	fms:AssociateThirdPartyFirewall
	fms:BatchAssociateResource
	fms:BatchDisassociateResource
	fms:DeleteAppsList
	fms:DeleteNotificationChannel
	fms:DeletePolicy
	fms:DeleteProtocolsList
	fms:DeleteResourceSet
	fms:DisassociateAdminAccount
	fms:DisassociateThirdPartyFirewall
	fms:GetAdminAccount
	fms:GetAdminScope
	fms:GetAppsList
	fms:GetComplianceDetail
	fms:GetNotificationChannel
	fms:GetPolicy
	fms:GetProtectionStatus
	fms:GetProtocolsList
	fms:GetResourceSet
	fms:GetThirdPartyFirewallAssociationStatus

服務前綴	動作
	fms:GetViolationDetails
	fms:ListAdminAccountsForOrganization
	fms:ListAdminsManagingAccount
	fms:ListAppsLists
	fms:ListComplianceStatus
	fms:ListDiscoveredResources
	fms:ListMemberAccounts
	fms:ListPolicies
	fms:ListProtocolsLists
	fms:ListResourceSetResources
	fms:ListResourceSets
	fms:ListThirdPartyFirewallFirewallPolicies
	fms:PutAdminAccount
	fms:PutAppsList
	fms:PutNotificationChannel
	fms:PutPolicy
	fms:PutProtocolsList
	fms:PutResourceSet

服務前綴	動作
frauddetector	frauddetector:BatchCreateVariable frauddetector:BatchGetVariable frauddetector:CancelBatchImportJob frauddetector:CancelBatchPredictionJob frauddetector:CreateBatchImportJob frauddetector:CreateBatchPredictionJob frauddetector:CreateDetectorVersion frauddetector:CreateList frauddetector:CreateModel frauddetector:CreateModelVersion frauddetector:CreateRule frauddetector:CreateVariable frauddetector:DeleteBatchImportJob frauddetector:DeleteBatchPredictionJob frauddetector:DeleteDetector frauddetector:DeleteDetectorVersion frauddetector:DeleteEntityType frauddetector:DeleteEvent frauddetector:DeleteEventType frauddetector:DeleteEventsByEventType frauddetector:DeleteExternalModel

服務前綴	動作
	frauddetector:DeleteLabel frauddetector:DeleteList frauddetector:DeleteModel frauddetector:DeleteModelVersion frauddetector:DeleteOutcome frauddetector:DeleteRule frauddetector:DeleteVariable frauddetector:DescribeDetector frauddetector:DescribeModelVersions frauddetector:GetBatchImportJobs frauddetector:GetBatchPredictionJobs frauddetector:GetDeleteEventsByEventTypeStatus frauddetector:GetDetectorVersion frauddetector:GetDetectors frauddetector:GetEntityTypes frauddetector:GetEvent frauddetector:GetEventPrediction frauddetector:GetEventPredictionMetadata frauddetector:GetEventTypes frauddetector:GetExternalModels frauddetector:GetKMSEncryptionKey

服務前綴	動作
	frauddetector:GetLabels frauddetector:GetListElements frauddetector:GetListsMetadata frauddetector:GetModelVersion frauddetector:GetModels frauddetector:GetOutcomes frauddetector:GetRules frauddetector:GetVariables frauddetector:ListEventPredictions frauddetector:PutDetector frauddetector:PutEntityType frauddetector:PutEventType frauddetector:PutExternalModel frauddetector:PutKMSEncryptionKey frauddetector:PutLabel frauddetector:PutOutcome frauddetector:SendEvent frauddetector:UpdateDetectorVersion frauddetector:UpdateDetectorVersionMetadata frauddetector:UpdateDetectorVersionStatus frauddetector:UpdateEventLabel

服務前綴	動作
	frauddetector:UpdateList frauddetector:UpdateModel frauddetector:UpdateModelVersion frauddetector:UpdateModelVersionStatus frauddetector:UpdateRuleMetadata frauddetector:UpdateRuleVersion frauddetector:UpdateVariable

服務前綴	動作
fsx	fsx:AssociateFileSystemAliases
	fsx:CancelDataRepositoryTask
	fsx:CopyBackup
	fsx:CreateDataRepositoryTask
	fsx:CreateFileCache
	fsx:CreateFileSystem
	fsx:CreateFileSystemFromBackup
	fsx:CreateSnapshot
	fsx:CreateStorageVirtualMachine
	fsx:CreateVolume
	fsx:CreateVolumeFromBackup
	fsx:DeleteBackup
	fsx:DeleteFileCache
	fsx:DeleteFileSystem
	fsx:DeleteSnapshot
	fsx:DeleteStorageVirtualMachine
	fsx:DeleteVolume
	fsx:DescribeBackups
	fsx:DescribeDataRepositoryAssociations
	fsx:DescribeDataRepositoryTasks
	fsx:DescribeFileCaches

服務前綴	動作
	fsx:DescribeFileSystemAliases
	fsx:DescribeFileSystems
	fsx:DescribeS3AccessPointAttachments
	fsx:DescribeSharedVpcConfiguration
	fsx:DescribeSnapshots
	fsx:DescribeStorageVirtualMachines
	fsx:DescribeVolumes
	fsx:DetachAndDeleteS3AccessPoint
	fsx:DisassociateFileSystemAliases
	fsx:ReleaseFileSystemNfsV3Locks
	fsx:RestoreVolumeFromSnapshot
	fsx:StartMisconfiguredStateRecovery
	fsx:UpdateDataRepositoryAssociation
	fsx:UpdateFileCache
	fsx:UpdateFileSystem
	fsx:UpdateSharedVpcConfiguration
	fsx:UpdateSnapshot
	fsx:UpdateStorageVirtualMachine
	fsx:UpdateVolume

服務前綴	動作
gamelift	gamelift:AcceptMatch gamelift:ClaimGameServer gamelift:CreateAlias gamelift:CreateBuild gamelift:CreateContainerGroupDefinition gamelift:CreateFleet gamelift:CreateFleetLocations gamelift:CreateGameServerGroup gamelift:CreateGameSession gamelift:CreateGameSessionQueue gamelift:CreateLocation gamelift:CreateMatchmakingConfiguration gamelift:CreateMatchmakingRuleSet gamelift:CreatePlayerSession gamelift:CreatePlayerSessions gamelift:CreateScript gamelift:CreateVpcPeeringAuthorization gamelift:CreateVpcPeeringConnection gamelift:DeleteAlias gamelift:DeleteBuild gamelift:DeleteContainerGroupDefinition

服務前綴	動作
	gamelift:DeleteFleet gamelift:DeleteFleetLocations gamelift:DeleteGameServerGroup gamelift:DeleteGameSessionQueue gamelift:DeleteLocation gamelift:DeleteMatchmakingConfiguration gamelift:DeleteMatchmakingRuleSet gamelift:DeleteScalingPolicy gamelift:DeleteScript gamelift:DeleteVpcPeeringAuthorization gamelift:DeleteVpcPeeringConnection gamelift:DeregisterCompute gamelift:DeregisterGameServer gamelift:DescribeAlias gamelift:DescribeBuild gamelift:DescribeCompute gamelift:DescribeContainerFleet gamelift:DescribeContainerGroupDefinition gamelift：DescribeContainerGroupPortMappings gamelift:DescribeEC2InstanceLimits gamelift:DescribeFleetAttributes

服務前綴	動作
	gamelift:DescribeFleetCapacity gamelift:DescribeFleetEvents gamelift:DescribeFleetLocationAttributes gamelift:DescribeFleetLocationCapacity gamelift:DescribeFleetLocationUtilization gamelift:DescribeFleetPortSettings gamelift:DescribeFleetUtilization gamelift:DescribeGameServer gamelift:DescribeGameServerGroup gamelift:DescribeGameServerInstances gamelift:DescribeGameSessionDetails gamelift:DescribeGameSessionPlacement gamelift:DescribeGameSessionQueues gamelift:DescribeGameSessions gamelift:DescribeInstances gamelift:DescribeMatchmaking gamelift:DescribeMatchmakingConfigurations gamelift:DescribeMatchmakingRuleSets gamelift:DescribePlayerSessions gamelift:DescribeRuntimeConfiguration gamelift:DescribeScalingPolicies

服務前綴	動作
	gamelift:DescribeScript gamelift:DescribeVpcPeeringAuthorizations gamelift:DescribeVpcPeeringConnections gamelift:GetComputeAccess gamelift:GetComputeAuthToken gamelift:GetGameSessionLogUrl gamelift:GetInstanceAccess gamelift：GetPlayerConnectionDetails gamelift:ListAliases gamelift:ListBuilds gamelift:ListCompute gamelift:ListContainerFleets gamelift:ListContainerGroupDefinitionVersions gamelift:ListContainerGroupDefinitions gamelift:ListFleetDeployments gamelift:ListFleets gamelift:ListGameServerGroups gamelift:ListGameServers gamelift:ListLocations gamelift:ListScripts gamelift:PutScalingPolicy

服務前綴	動作
	gamelift:RegisterCompute gamelift:RegisterGameServer gamelift:RequestUploadCredentials gamelift:ResolveAlias gamelift:ResumeGameServerGroup gamelift:SearchGameSessions gamelift:StartFleetActions gamelift:StartGameSessionPlacement gamelift:StartMatchBackfill gamelift:StartMatchmaking gamelift:StopFleetActions gamelift:StopGameSessionPlacement gamelift:StopMatchmaking gamelift:SuspendGameServerGroup gamelift:TerminateGameSession gamelift:UpdateAlias gamelift:UpdateBuild gamelift:UpdateContainerGroupDefinition gamelift:UpdateFleetAttributes gamelift:UpdateFleetCapacity gamelift:UpdateFleetPortSettings

服務前綴	動作
	gamelift:UpdateGameServer gamelift:UpdateGameServerGroup gamelift:UpdateGameSession gamelift:UpdateGameSessionQueue gamelift:UpdateMatchmakingConfiguration gamelift:UpdateRuntimeConfiguration gamelift:UpdateScript gamelift:ValidateMatchmakingRuleSet

服務前綴	動作
geo	geo:AssociateTrackerConsumer
	geo:BatchDeleteDevicePositionHistory
	geo:BatchDeleteGeofence
	geo:BatchEvaluateGeofences
	geo:BatchGetDevicePosition
	geo:BatchPutGeofence
	geo:BatchUpdateDevicePosition
	geo:CalculateRoute
	geo:CalculateRouteMatrix
	geo：CancelJob
	geo:CreateGeofenceCollection
	geo:CreateMap
	geo:CreatePlaceIndex
	geo:CreateRouteCalculator
	geo:CreateTracker
	geo:DeleteGeofenceCollection
	geo:DeleteKey
	geo:DeleteMap
	geo:DeletePlaceIndex
	geo:DeleteRouteCalculator
	geo:DeleteTracker

服務前綴	動作
	geo:DescribeGeofenceCollection
	geo:DescribeKey
	geo:DescribeMap
	geo:DescribePlaceIndex
	geo:DescribeRouteCalculator
	geo:DescribeTracker
	geo:DisassociateTrackerConsumer
	geo:ForecastGeofenceEvents
	geo:GetDevicePosition
	geo:GetDevicePositionHistory
	geo:GetGeofence
	geo：GetJob
	geo:GetMapGlyphs
	geo:GetMapSprites
	geo:GetMapStyleDescriptor
	geo:GetMapTile
	geo:GetPlace
	geo:ListDevicePositions
	geo:ListGeofenceCollections
	geo:ListGeofences
	geo：ListJobs

服務前綴	動作
	geo:ListKeys
	geo:ListMaps
	geo:ListPlaceIndexes
	geo:ListRouteCalculators
	geo:ListTrackerConsumers
	geo:ListTrackers
	geo:PutGeofence
	geo:SearchPlaceIndexForPosition
	geo:SearchPlaceIndexForSuggestions
	geo:SearchPlaceIndexForText
	geo:UpdateGeofenceCollection
	geo:UpdateKey
	geo:UpdateMap
	geo:UpdatePlaceIndex
	geo:UpdateRouteCalculator
	geo:UpdateTracker
	geo:VerifyDevicePosition

服務前綴	動作
glacier	glacier:AbortMultipartUpload glacier:AbortVaultLock glacier:CompleteMultipartUpload glacier:CompleteVaultLock glacier:CreateVault glacier:DeleteArchive glacier:DeleteVault glacier:DeleteVaultAccessPolicy glacier:DeleteVaultNotifications glacier:DescribeJob glacier:DescribeVault glacier:GetDataRetrievalPolicy glacier:GetJobOutput glacier:GetVaultAccessPolicy glacier:GetVaultLock glacier:GetVaultNotifications glacier:InitiateJob glacier:InitiateMultipartUpload glacier:InitiateVaultLock glacier:ListJobs glacier:ListMultipartUploads

服務前綴	動作
	glacier:ListParts glacier:ListProvisionedCapacity glacier:ListVaults glacier:PurchaseProvisionedCapacity glacier:SetDataRetrievalPolicy glacier:SetVaultAccessPolicy glacier:SetVaultNotifications glacier:UploadArchive glacier:UploadMultipartPart

服務前綴	動作
grafana	grafana:AssociateLicense
	grafana:CreateWorkspace
	grafana:CreateWorkspaceApiKey
	grafana:CreateWorkspaceServiceAccount
	grafana:CreateWorkspaceServiceAccountToken
	grafana:DeleteWorkspace
	grafana:DeleteWorkspaceApiKey
	grafana:DeleteWorkspaceServiceAccount
	grafana:DeleteWorkspaceServiceAccountToken
	grafana:DescribeWorkspace
	grafana:DescribeWorkspaceAuthentication
	grafana:DescribeWorkspaceConfiguration
	grafana:DisassociateLicense
	grafana:ListPermissions
	grafana:ListVersions
	grafana:ListWorkspaceServiceAccountTokens
	grafana:ListWorkspaceServiceAccounts
	grafana:ListWorkspaces
	grafana:UpdatePermissions
	grafana:UpdateWorkspace
	grafana:UpdateWorkspaceAuthentication

服務前綴	動作
	grafana:UpdateWorkspaceConfiguration

服務前綴	動作
greengrass	greengrass:AssociateRoleToGroup greengrass:AssociateServiceRoleToAccount greengrass:BatchAssociateClientDeviceWithCoreDevice greengrass:BatchDisassociateClientDeviceFromCoreDevice greengrass:CancelDeployment greengrass:CreateComponentVersion greengrass:CreateConnectorDefinition greengrass:CreateConnectorDefinitionVersion greengrass:CreateCoreDefinition greengrass:CreateCoreDefinitionVersion greengrass:CreateDeployment greengrass:CreateDeviceDefinition greengrass:CreateDeviceDefinitionVersion greengrass:CreateFunctionDefinition greengrass:CreateFunctionDefinitionVersion greengrass:CreateGroup greengrass:CreateGroupCertificateAuthority greengrass:CreateGroupVersion greengrass:CreateLoggerDefinition greengrass:CreateLoggerDefinitionVersion greengrass:CreateResourceDefinition

服務前綴	動作
	greengrass:CreateResourceDefinitionVersion greengrass:CreateSoftwareUpdateJob greengrass:CreateSubscriptionDefinition greengrass:CreateSubscriptionDefinitionVersion greengrass:DeleteComponent greengrass:DeleteConnectorDefinition greengrass:DeleteCoreDefinition greengrass:DeleteCoreDevice greengrass:DeleteDeployment greengrass:DeleteDeviceDefinition greengrass:DeleteFunctionDefinition greengrass:DeleteGroup greengrass:DeleteLoggerDefinition greengrass:DeleteResourceDefinition greengrass:DeleteSubscriptionDefinition greengrass:DescribeComponent greengrass:DisassociateRoleFromGroup greengrass:DisassociateServiceRoleFromAccount greengrass:GetAssociatedRole greengrass:GetBulkDeploymentStatus greengrass:GetComponent

服務前綴	動作
	greengrass:GetComponentVersionArtifact greengrass:GetConnectivityInfo greengrass:GetConnectorDefinition greengrass:GetConnectorDefinitionVersion greengrass:GetCoreDefinition greengrass:GetCoreDefinitionVersion greengrass:GetCoreDevice greengrass:GetDeployment greengrass:GetDeploymentStatus greengrass:GetDeviceDefinition greengrass:GetDeviceDefinitionVersion greengrass:GetFunctionDefinition greengrass:GetFunctionDefinitionVersion greengrass:GetGroup greengrass:GetGroupCertificateAuthority greengrass:GetGroupCertificateConfiguration greengrass:GetGroupVersion greengrass:GetLoggerDefinition greengrass:GetLoggerDefinitionVersion greengrass:GetResourceDefinition greengrass:GetResourceDefinitionVersion

服務前綴	動作
	greengrass:GetServiceRoleForAccount greengrass:GetSubscriptionDefinition greengrass:GetSubscriptionDefinitionVersion greengrass:GetThingRuntimeConfiguration greengrass:ListBulkDeploymentDetailedReports greengrass:ListBulkDeployments greengrass:ListClientDevicesAssociatedWithCoreDevice greengrass:ListComponentVersions greengrass:ListComponents greengrass:ListConnectorDefinitionVersions greengrass:ListConnectorDefinitions greengrass:ListCoreDefinitionVersions greengrass:ListCoreDefinitions greengrass:ListCoreDevices greengrass:ListDeployments greengrass:ListDeviceDefinitionVersions greengrass:ListDeviceDefinitions greengrass:ListEffectiveDeployments greengrass:ListFunctionDefinitionVersions greengrass:ListFunctionDefinitions greengrass:ListGroupCertificateAuthorities

服務前綴	動作
	greengrass:ListGroupVersions greengrass:ListGroups greengrass:ListInstalledComponents greengrass:ListLoggerDefinitionVersions greengrass:ListLoggerDefinitions greengrass:ListResourceDefinitionVersions greengrass:ListResourceDefinitions greengrass:ListSubscriptionDefinitionVersions greengrass:ListSubscriptionDefinitions greengrass:ResetDeployments greengrass:StartBulkDeployment greengrass:StopBulkDeployment greengrass:UpdateConnectivityInfo greengrass:UpdateConnectorDefinition greengrass:UpdateCoreDefinition greengrass:UpdateDeviceDefinition greengrass:UpdateFunctionDefinition greengrass:UpdateGroup greengrass:UpdateGroupCertificateConfiguration greengrass:UpdateLoggerDefinition greengrass:UpdateResourceDefinition

服務前綴	動作
	greengrass:UpdateSubscriptionDefinition greengrass:UpdateThingRuntimeConfiguration

服務前綴	動作
groundstation	groundstation:CancelContact
	groundstation:CreateConfig
	groundstation:CreateDataflowEndpointGroup
	groundstation：CreateDataflowEndpointGroupV2
	groundstation:CreateEphemeris
	groundstation:CreateMissionProfile
	groundstation:DeleteConfig
	groundstation:DeleteDataflowEndpointGroup
	groundstation:DeleteEphemeris
	groundstation:DeleteMissionProfile
	groundstation:DescribeContact
	groundstation：DescribeContactVersion
	groundstation:DescribeEphemeris
	groundstation:GetConfig
	groundstation:GetDataflowEndpointGroup
	groundstation:GetMinuteUsage
	groundstation:GetMissionProfile
	groundstation:GetSatellite
	groundstation：ListAntennas
	groundstation:ListConfigs
	groundstation:ListContacts

服務前綴	動作
	groundstation:ListDataflowEndpointGroups
	groundstation:ListEphemerides
	groundstation：ListGroundStationReservations
	groundstation:ListGroundStations
	groundstation:ListMissionProfiles
	groundstation:ListSatellites
	groundstation:RegisterAgent
	groundstation:ReserveContact
	groundstation:UpdateAgentStatus
	groundstation:UpdateConfig
	groundstation:UpdateEphemeris
	groundstation:UpdateMissionProfile

服務前綴	動作
guardduty	guardduty:AcceptAdministratorInvitation guardduty:AcceptInvitation guardduty:ArchiveFindings guardduty:CreateDetector guardduty:CreateFilter guardduty:CreateIPSet guardduty：CreateInvestigation guardduty:CreateMalwareProtectionPlan guardduty:CreateMembers guardduty:CreatePublishingDestination guardduty:CreateSampleFindings guardduty:CreateThreatEntitySet guardduty:CreateThreatIntelSet guardduty:CreateTrustedEntitySet guardduty:DeclineInvitations guardduty:DeleteDetector guardduty:DeleteFilter guardduty:DeleteIPSet guardduty:DeleteInvitations guardduty:DeleteMalwareProtectionPlan guardduty:DeleteMembers

服務前綴	動作
	guardduty:DeletePublishingDestination guardduty:DeleteThreatEntitySet guardduty:DeleteThreatIntelSet guardduty:DeleteTrustedEntitySet guardduty:DescribeMalwareScans guardduty:DescribeOrganizationConfiguration guardduty:DescribePublishingDestination guardduty:DisableOrganizationAdminAccount guardduty:DisassociateFromAdministratorAccount guardduty:DisassociateFromMasterAccount guardduty:DisassociateMembers guardduty:EnableOrganizationAdminAccount guardduty:GetAdministratorAccount guardduty:GetCoverageStatistics guardduty:GetDetector guardduty:GetFilter guardduty:GetFindings guardduty:GetFindingsStatistics guardduty:GetIPSet guardduty：GetInvestigation guardduty:GetInvitationsCount

服務前綴	動作
	guardduty:GetMalwareProtectionPlan guardduty：GetMalwareScan guardduty:GetMalwareScanSettings guardduty:GetMasterAccount guardduty:GetMemberDetectors guardduty:GetMembers guardduty:GetOrganizationStatistics guardduty:GetRemainingFreeTrialDays guardduty:GetThreatEntitySet guardduty:GetThreatIntelSet guardduty:GetTrustedEntitySet guardduty:GetUsageStatistics guardduty:InviteMembers guardduty:ListCoverage guardduty:ListDetectors guardduty:ListFilters guardduty:ListFindings guardduty:ListIPSets guardduty：ListInvestigations guardduty:ListInvitations guardduty:ListMalwareProtectionPlans

服務前綴	動作
	guardduty：ListMalwareScans guardduty:ListMembers guardduty:ListOrganizationAdminAccounts guardduty:ListPublishingDestinations guardduty:ListThreatEntitySets guardduty:ListThreatIntelSets guardduty:ListTrustedEntitySets guardduty:StartMalwareScan guardduty:StartMonitoringMembers guardduty:StopMonitoringMembers guardduty:UnarchiveFindings guardduty:UpdateDetector guardduty:UpdateFilter guardduty:UpdateFindingsFeedback guardduty:UpdateIPSet guardduty:UpdateMalwareProtectionPlan guardduty:UpdateMalwareScanSettings guardduty:UpdateMemberDetectors guardduty:UpdateOrganizationConfiguration guardduty:UpdatePublishingDestination guardduty:UpdateThreatEntitySet

服務前綴	動作
	guardduty:UpdateThreatIntelSet guardduty:UpdateTrustedEntitySet

服務前綴	動作
healthlake	healthlake:CancelFHIRExportJobWithDelete healthlake：CreateDataTransformationProfile healthlake:CreateFHIRDatastore healthlake:CreateResource healthlake：DeleteDataTransformationProfile healthlake:DeleteFHIRDatastore healthlake:DeleteResource healthlake：DescribeDataTransformationJob healthlake:DescribeFHIRDatastore healthlake:DescribeFHIRExportJob healthlake:DescribeFHIRExportJobWithGet healthlake:DescribeFHIRImportJob healthlake:GetCapabilities healthlake：GetDataTransformationProfile healthlake：ListDataTransformationJobs healthlake：ListDataTransformationProfileVersions healthlake：ListDataTransformationProfiles healthlake:ListFHIRDatastores healthlake:ListFHIRExportJobs healthlake:ListFHIRImportJobs healthlake：PublishDataTransformationProfile

服務前綴	動作
	healthlake:ReadResource
	healthlake:SearchEverything
	healthlake:SearchWithGet
	healthlake:SearchWithPost
	healthlake:StartFHIRExportJob
	healthlake:StartFHIRExportJobWithPost
	healthlake:StartFHIRImportJob
	healthlake：UpdateDataTransformationProfile
	healthlake：UpdateFHIRDatastore
	healthlake：UpdateProfileWithAgent
	healthlake:UpdateResource

服務前綴	動作
honeycode	honeycode:BatchCreateTableRows honeycode:BatchDeleteTableRows honeycode:BatchUpdateTableRows honeycode:BatchUpsertTableRows honeycode:DescribeTableDataImportJob honeycode:GetScreenData honeycode:InvokeScreenAutomation honeycode:ListTableColumns honeycode:ListTableRows honeycode:ListTables honeycode:QueryTableRows honeycode:StartTableDataImportJob

服務前綴	動作
iam	iam:AddClientIDToOpenIDConnectProvider
	iam:AddRoleToInstanceProfile
	iam:AddUserToGroup
	iam:AttachGroupPolicy
	iam:AttachRolePolicy
	iam:AttachUserPolicy
	iam:ChangePassword
	iam:CreateAccessKey
	iam:CreateAccountAlias
	iam:CreateGroup
	iam:CreateInstanceProfile
	iam:CreateLoginProfile
	iam:CreateOpenIDConnectProvider
	iam:CreatePolicy
	iam:CreatePolicyVersion
	iam:CreateRole
	iam:CreateSAMLProvider
	iam：CreateServiceLinkedRole
	iam:CreateServiceSpecificCredential
	iam:CreateUser
	iam:CreateVirtualMFADevice

服務前綴	動作
	iam:DeactivateMFADevice
	iam:DeleteAccessKey
	iam:DeleteAccountAlias
	iam:DeleteAccountPasswordPolicy
	iam:DeleteCloudFrontPublicKey
	iam:DeleteGroup
	iam:DeleteGroupPolicy
	iam:DeleteInstanceProfile
	iam:DeleteLoginProfile
	iam:DeleteOpenIDConnectProvider
	iam:DeletePolicy
	iam:DeletePolicyVersion
	iam:DeleteRole
	iam:DeleteRolePermissionsBoundary
	iam:DeleteRolePolicy
	iam:DeleteSAMLProvider
	iam:DeleteSSHPublicKey
	iam:DeleteServerCertificate
	iam:DeleteServiceLinkedRole
	iam:DeleteServiceSpecificCredential
	iam:DeleteSigningCertificate

服務前綴	動作
	iam:DeleteUser
	iam:DeleteUserPermissionsBoundary
	iam:DeleteUserPolicy
	iam:DeleteVirtualMFADevice
	iam:DetachGroupPolicy
	iam:DetachRolePolicy
	iam:DetachUserPolicy
	iam:DisableOrganizationsRootCredentialsManagement
	iam:DisableOrganizationsRootSessions
	iam：DisableOutboundWebIdentityFederation
	iam:EnableMFADevice
	iam:EnableOrganizationsRootCredentialsManagement
	iam:EnableOrganizationsRootSessions
	iam：EnableOutboundWebIdentityFederation
	iam:GenerateCredentialReport
	iam:GenerateOrganizationsAccessReport
	iam:GenerateServiceLastAccessedDetails
	iam:GetAccessKeyLastUsed
	iam:GetAccountAuthorizationDetails
	iam:GetAccountEmailAddress
	iam:GetAccountName

服務前綴	動作
	iam:GetAccountPasswordPolicy
	iam:GetAccountSummary
	iam:GetCloudFrontPublicKey
	iam:GetContextKeysForCustomPolicy
	iam:GetContextKeysForPrincipalPolicy
	iam:GetCredentialReport
	iam:GetGroup
	iam:GetGroupPolicy
	iam:GetInstanceProfile
	iam:GetLoginProfile
	iam:GetMFADevice
	iam:GetOpenIDConnectProvider
	iam:GetOrganizationsAccessReport
	iam：GetOutboundWebIdentityFederationInfo
	iam:GetPolicy
	iam:GetPolicyVersion
	iam:GetRole
	iam:GetRolePolicy
	iam:GetSAMLProvider
	iam:GetSSHPublicKey
	iam:GetServerCertificate

服務前綴	動作
	iam:GetServiceLastAccessedDetails
	iam:GetServiceLastAccessedDetailsWithEntities
	iam:GetServiceLinkedRoleDeletionStatus
	iam:GetUser
	iam:GetUserPolicy
	iam:ListAccessKeys
	iam:ListAccountAliases
	iam:ListAttachedGroupPolicies
	iam:ListAttachedRolePolicies
	iam:ListAttachedUserPolicies
	iam:ListCloudFrontPublicKeys
	iam：ListDelegationRequests
	iam:ListEntitiesForPolicy
	iam:ListGroupPolicies
	iam:ListGroups
	iam:ListGroupsForUser
	iam:ListInstanceProfiles
	iam:ListInstanceProfilesForRole
	iam:ListMFADevices
	iam:ListOpenIDConnectProviders
	iam:ListOrganizationsFeatures

服務前綴	動作
	iam:ListPolicies
	iam:ListPoliciesGrantingServiceAccess
	iam:ListPolicyVersions
	iam:ListRolePolicies
	iam:ListRoles
	iam:ListSAMLProviders
	iam:ListSSHPublicKeys
	iam:ListSTSRegionalEndpointsStatus
	iam:ListServerCertificates
	iam:ListServiceSpecificCredentials
	iam:ListSigningCertificates
	iam:ListUserPolicies
	iam:ListUsers
	iam:ListVirtualMFADevices
	iam:PutGroupPolicy
	iam:PutRolePermissionsBoundary
	iam：PutRolePolicy
	iam:PutUserPermissionsBoundary
	iam:PutUserPolicy
	iam:RemoveClientIDFromOpenIDConnectProvider
	iam:RemoveRoleFromInstanceProfile

服務前綴	動作
	iam:RemoveUserFromGroup
	iam:ResetServiceSpecificCredential
	iam:ResyncMFADevice
	iam:SetDefaultPolicyVersion
	iam:SetSTSRegionalEndpointStatus
	iam:SetSecurityTokenServicePreferences
	iam:SimulateCustomPolicy
	iam:SimulatePrincipalPolicy
	iam:UpdateAccessKey
	iam:UpdateAccountEmailAddress
	iam:UpdateAccountName
	iam:UpdateAccountPasswordPolicy
	iam:UpdateAssumeRolePolicy
	iam:UpdateCloudFrontPublicKey
	iam:UpdateGroup
	iam:UpdateLoginProfile
	iam:UpdateOpenIDConnectProviderThumbprint
	iam:UpdateRole
	iam:UpdateRoleDescription
	iam:UpdateSAMLProvider
	iam:UpdateSSHPublicKey

服務前綴	動作
	iam:UpdateServerCertificate
	iam:UpdateServiceSpecificCredential
	iam:UpdateSigningCertificate
	iam:UpdateUser
	iam:UploadCloudFrontPublicKey
	iam:UploadSSHPublicKey
	iam:UploadServerCertificate
	iam:UploadSigningCertificate

服務前綴	動作
identitystore	identitystore:CreateGroup identitystore:CreateGroupMembership identitystore:CreateUser identitystore:DeleteGroup identitystore:DeleteGroupMembership identitystore:DeleteUser identitystore:DescribeGroup identitystore:DescribeGroupMembership identitystore:DescribeUser identitystore:GetGroupId identitystore:GetGroupMembershipId identitystore:GetUserId identitystore:IsMemberInGroups identitystore:ListGroupMemberships identitystore:ListGroupMembershipsForMember identitystore:ListGroups identitystore:ListUsers identitystore:UpdateGroup identitystore:UpdateUser

服務前綴	動作
imagebuilder	imagebuilder:CancelImageCreation imagebuilder:CancelLifecycleExecution imagebuilder:CreateComponent imagebuilder:CreateContainerRecipe imagebuilder:CreateDistributionConfiguration imagebuilder:CreateImage imagebuilder:CreateImagePipeline imagebuilder:CreateImageRecipe imagebuilder:CreateInfrastructureConfiguration imagebuilder:CreateLifecyclePolicy imagebuilder:CreateWorkflow imagebuilder:DeleteComponent imagebuilder:DeleteContainerRecipe imagebuilder:DeleteDistributionConfiguration imagebuilder:DeleteImage imagebuilder:DeleteImagePipeline imagebuilder:DeleteImageRecipe imagebuilder:DeleteInfrastructureConfiguration imagebuilder:DeleteLifecyclePolicy imagebuilder:DeleteWorkflow imagebuilder：DistributeImage

服務前綴	動作
	imagebuilder:GetComponentPolicy
	imagebuilder:GetContainerRecipePolicy
	imagebuilder:GetImagePolicy
	imagebuilder:GetImageRecipePolicy
	imagebuilder:GetLifecycleExecution
	imagebuilder:GetLifecyclePolicy
	imagebuilder:GetMarketplaceResource
	imagebuilder:GetWorkflowExecution
	imagebuilder:GetWorkflowStepExecution
	imagebuilder:ImportComponent
	imagebuilder:ImportDiskImage
	imagebuilder:ImportVmImage
	imagebuilder:ListComponentBuildVersions
	imagebuilder:ListComponents
	imagebuilder:ListContainerRecipes
	imagebuilder:ListDistributionConfigurations
	imagebuilder:ListImageBuildVersions
	imagebuilder:ListImagePackages
	imagebuilder:ListImagePipelineImages
	imagebuilder:ListImagePipelines
	imagebuilder:ListImageRecipes

服務前綴	動作
	imagebuilder:ListImageScanFindingAggregations
	imagebuilder:ListImageScanFindings
	imagebuilder:ListImages
	imagebuilder:ListInfrastructureConfigurations
	imagebuilder:ListLifecycleExecutionResources
	imagebuilder:ListLifecycleExecutions
	imagebuilder:ListLifecyclePolicies
	imagebuilder:ListWaitingWorkflowSteps
	imagebuilder:ListWorkflowExecutions
	imagebuilder:ListWorkflowStepExecutions
	imagebuilder:ListWorkflows
	imagebuilder:PutComponentPolicy
	imagebuilder:PutContainerRecipePolicy
	imagebuilder:PutImagePolicy
	imagebuilder:PutImageRecipePolicy
	imagebuilder：RetryImage
	imagebuilder:SendWorkflowStepAction
	imagebuilder:StartImagePipelineExecution
	imagebuilder:StartResourceStateUpdate
	imagebuilder:UpdateDistributionConfiguration
	imagebuilder:UpdateImagePipeline

服務前綴	動作
	imagebuilder:UpdateInfrastructureConfiguration

服務前綴	動作
inspector	inspector:AddAttributesToFindings inspector:CreateAssessmentTarget inspector:CreateAssessmentTemplate inspector:CreateExclusionsPreview inspector:CreateResourceGroup inspector:DeleteAssessmentRun inspector:DeleteAssessmentTarget inspector:DeleteAssessmentTemplate inspector:DescribeAssessmentRuns inspector:DescribeAssessmentTargets inspector:DescribeAssessmentTemplates inspector:DescribeCrossAccountAccessRole inspector:DescribeExclusions inspector:DescribeFindings inspector:DescribeResourceGroups inspector:DescribeRulesPackages inspector:GetAssessmentReport inspector:GetExclusionsPreview inspector:GetTelemetryMetadata inspector:ListAssessmentRunAgents inspector:ListAssessmentRuns

服務前綴	動作
	inspector:ListAssessmentTargets inspector:ListAssessmentTemplates inspector:ListEventSubscriptions inspector:ListExclusions inspector:ListFindings inspector:ListRulesPackages inspector:PreviewAgents inspector:RegisterCrossAccountAccessRole inspector:RemoveAttributesFromFindings inspector:StartAssessmentRun inspector:StopAssessmentRun inspector:SubscribeToEvent inspector:UnsubscribeFromEvent inspector:UpdateAssessmentTarget

服務前綴	動作
inspector2	inspector2:AssociateMember
	inspector2:BatchGetAccountStatus
	inspector2:BatchGetCodeSnippet
	inspector2:BatchGetFindingDetails
	inspector2:BatchGetFreeTrialInfo
	inspector2:BatchGetMemberEc2DeepInspectionStatus
	inspector2:BatchUpdateMemberEc2DeepInspectionStatus
	inspector2:CancelFindingsReport
	inspector2:CancelSbomExport
	inspector2:CreateCisScanConfiguration
	inspector2:CreateCodeSecurityIntegration
	inspector2：CreateConnector
	inspector2:CreateFilter
	inspector2:CreateFindingsReport
	inspector2:CreateSbomExport
	inspector2:DeleteCisScanConfiguration
	inspector2:DeleteCodeSecurityIntegration
	inspector2：DeleteConnector
	inspector2:DeleteFilter
	inspector2:DescribeOrganizationConfiguration
	inspector2:Disable

服務前綴	動作
	inspector2:DisableDelegatedAdminAccount
	inspector2:DisassociateMember
	inspector2:Enable
	inspector2:EnableDelegatedAdminAccount
	inspector2:GetCisScanReport
	inspector2:GetCisScanResultDetails
	inspector2:GetClustersForImage
	inspector2:GetCodeSecurityIntegration
	inspector2:GetCodeSecurityScan
	inspector2:GetConfiguration
	inspector2:GetDelegatedAdminAccount
	inspector2:GetEc2DeepInspectionConfiguration
	inspector2:GetEncryptionKey
	inspector2:GetFindingsReportStatus
	inspector2:GetMember
	inspector2:GetSbomExport
	inspector2:ListAccountPermissions
	inspector2:ListCisScanConfigurations
	inspector2:ListCisScanResultsAggregatedByChecks
	inspector2:ListCisScanResultsAggregatedByTargetResource
	inspector2:ListCisScans

服務前綴	動作
	inspector2:ListCodeSecurityIntegrations
	inspector2:ListCodeSecurityScanConfigurations
	inspector2：ListConnectorScanConfigurations
	inspector2：ListConnectors
	inspector2:ListCoverage
	inspector2:ListCoverageStatistics
	inspector2:ListDelegatedAdminAccounts
	inspector2:ListFilters
	inspector2:ListFindingAggregations
	inspector2:ListFindings
	inspector2:ListMembers
	inspector2:ListUsageTotals
	inspector2:ResetEncryptionKey
	inspector2:SearchVulnerabilities
	inspector2:SendCisSessionHealth
	inspector2:SendCisSessionTelemetry
	inspector2:StartCisSession
	inspector2:StartCodeSecurityScan
	inspector2:StopCisSession
	inspector2:UpdateCisScanConfiguration
	inspector2:UpdateCodeSecurityIntegration

服務前綴	動作
	inspector2:UpdateConfiguration inspector2：UpdateConnector inspector2:UpdateEc2DeepInspectionConfiguration inspector2:UpdateEncryptionKey inspector2:UpdateFilter inspector2:UpdateOrgEc2DeepInspectionConfiguration inspector2:UpdateOrganizationConfiguration

服務前綴	動作
iot	iot:AcceptCertificateTransfer iot:AddThingToBillingGroup iot:AddThingToThingGroup iot:AssociateSbomWithPackageVersion iot:AssociateTargetsWithJob iot:AttachPolicy iot:AttachPrincipalPolicy iot:AttachSecurityProfile iot:AttachThingPrincipal iot:CancelAuditMitigationActionsTask iot:CancelAuditTask iot:CancelCertificateTransfer iot:CancelDetectMitigationActionsTask iot:CancelJob iot:CancelJobExecution iot:ClearDefaultAuthorizer iot:ConfirmTopicRuleDestination iot:CreateAuditSuppression iot:CreateAuthorizer iot:CreateBillingGroup iot:CreateCertificateFromCsr

服務前綴	動作
	iot:CreateCertificateProvider iot:CreateCommand iot:CreateCustomMetric iot:CreateDimension iot:CreateDomainConfiguration iot:CreateDynamicThingGroup iot:CreateFleetMetric iot:CreateJob iot:CreateJobTemplate iot:CreateKeysAndCertificate iot:CreateMitigationAction iot:CreateOTAUpdate iot:CreatePackage iot:CreatePackageVersion iot:CreatePolicy iot:CreatePolicyVersion iot:CreateProvisioningClaim iot:CreateProvisioningTemplate iot:CreateProvisioningTemplateVersion iot:CreateRoleAlias iot:CreateScheduledAudit

服務前綴	動作
	iot:CreateSecurityProfile iot:CreateStream iot:CreateThing iot:CreateThingGroup iot:CreateThingType iot:CreateTopicRule iot:CreateTopicRuleDestination iot:DeleteAccountAuditConfiguration iot:DeleteAuditSuppression iot:DeleteAuthorizer iot:DeleteBillingGroup iot:DeleteCACertificate iot:DeleteCertificate iot:DeleteCertificateProvider iot:DeleteCommand iot:DeleteCustomMetric iot:DeleteDimension iot:DeleteDomainConfiguration iot:DeleteDynamicThingGroup iot:DeleteFleetMetric iot:DeleteJob

服務前綴	動作
	iot:DeleteJobExecution iot:DeleteJobTemplate iot:DeleteMitigationAction iot:DeleteOTAUpdate iot:DeletePackage iot:DeletePackageVersion iot:DeletePolicy iot:DeletePolicyVersion iot:DeleteProvisioningTemplate iot:DeleteProvisioningTemplateVersion iot:DeleteRegistrationCode iot:DeleteRoleAlias iot:DeleteScheduledAudit iot:DeleteSecurityProfile iot:DeleteStream iot:DeleteThing iot:DeleteThingGroup iot:DeleteThingType iot:DeleteTopicRule iot:DeleteTopicRuleDestination iot:DeleteV2LoggingLevel

服務前綴	動作
	iot:DeprecateThingType iot:DescribeAccountAuditConfiguration iot:DescribeAuditFinding iot:DescribeAuditMitigationActionsTask iot:DescribeAuditSuppression iot:DescribeAuditTask iot:DescribeAuthorizer iot:DescribeBillingGroup iot:DescribeCACertificate iot:DescribeCertificate iot:DescribeCertificateProvider iot:DescribeCustomMetric iot:DescribeDefaultAuthorizer iot:DescribeDetectMitigationActionsTask iot:DescribeDimension iot:DescribeDomainConfiguration iot:DescribeEncryptionConfiguration iot:DescribeEndpoint iot:DescribeEventConfigurations iot:DescribeFleetMetric iot:DescribeIndex

服務前綴	動作
	iot:DescribeJob iot:DescribeJobExecution iot:DescribeJobTemplate iot:DescribeManagedJobTemplate iot:DescribeMitigationAction iot:DescribeProvisioningTemplate iot:DescribeProvisioningTemplateVersion iot:DescribeRoleAlias iot:DescribeScheduledAudit iot:DescribeSecurityProfile iot:DescribeStream iot:DescribeThing iot:DescribeThingGroup iot:DescribeThingRegistrationTask iot:DescribeThingType iot:DetachPolicy iot:DetachPrincipalPolicy iot:DetachSecurityProfile iot:DetachThingPrincipal iot:DisableTopicRule iot:DisassociateSbomFromPackageVersion

服務前綴	動作
	iot:EnableTopicRule iot:GetBehaviorModelTrainingSummaries iot:GetBucketsAggregation iot:GetCardinality iot:GetCommand iot:GetEffectivePolicies iot:GetJobDocument iot:GetLoggingOptions iot:GetOTAUpdate iot:GetPackage iot:GetPackageConfiguration iot:GetPackageVersion iot:GetPercentiles iot:GetPolicy iot:GetPolicyVersion iot:GetRegistrationCode iot:GetStatistics iot:GetThingConnectivityData iot:GetTopicRule iot:GetTopicRuleDestination iot:GetV2LoggingOptions

服務前綴	動作
	iot:ListActiveViolations iot:ListAttachedPolicies iot:ListAuditFindings iot:ListAuditMitigationActionsExecutions iot:ListAuditMitigationActionsTasks iot:ListAuditSuppressions iot:ListAuditTasks iot:ListAuthorizers iot:ListBillingGroups iot:ListCACertificates iot:ListCertificateProviders iot:ListCertificates iot:ListCertificatesByCA iot:ListCommands iot:ListCustomMetrics iot:ListDetectMitigationActionsExecutions iot:ListDetectMitigationActionsTasks iot:ListDimensions iot:ListDomainConfigurations iot:ListFleetMetrics iot:ListIndices

服務前綴	動作
	iot:ListJobExecutionsForJob iot:ListJobExecutionsForThing iot:ListJobTemplates iot:ListJobs iot:ListManagedJobTemplates iot:ListMetricValues iot:ListMitigationActions iot:ListOTAUpdates iot:ListOutgoingCertificates iot:ListPackageVersions iot:ListPackages iot:ListPolicies iot:ListPolicyPrincipals iot:ListPolicyVersions iot:ListPrincipalPolicies iot:ListPrincipalThings iot:ListPrincipalThingsV2 iot:ListProvisioningTemplateVersions iot:ListProvisioningTemplates iot:ListRelatedResourcesForAuditFinding iot:ListRoleAliases

服務前綴	動作
	iot:ListSbomValidationResults iot:ListScheduledAudits iot:ListSecurityProfiles iot:ListSecurityProfilesForTarget iot:ListStreams iot:ListTargetsForPolicy iot:ListTargetsForSecurityProfile iot:ListThingGroups iot:ListThingGroupsForThing iot:ListThingPrincipals iot:ListThingPrincipalsV2 iot:ListThingRegistrationTaskReports iot:ListThingRegistrationTasks iot:ListThingTypes iot:ListThings iot:ListThingsInBillingGroup iot:ListThingsInThingGroup iot:ListTopicRuleDestinations iot:ListTopicRules iot:ListV2LoggingLevels iot:ListViolationEvents

服務前綴	動作
	iot:PutVerificationStateOnViolation iot:RegisterCACertificate iot:RegisterCertificate iot:RegisterCertificateWithoutCA iot:RegisterThing iot:RejectCertificateTransfer iot:RemoveThingFromBillingGroup iot:RemoveThingFromThingGroup iot:ReplaceTopicRule iot:SearchIndex iot:SetDefaultAuthorizer iot:SetDefaultPolicyVersion iot:SetLoggingOptions iot:SetV2LoggingLevel iot:SetV2LoggingOptions iot:StartAuditMitigationActionsTask iot:StartDetectMitigationActionsTask iot:StartOnDemandAuditTask iot:StartThingRegistrationTask iot:StopThingRegistrationTask iot:TestAuthorization

服務前綴	動作
	iot:TestInvokeAuthorizer iot:TransferCertificate iot:UpdateAccountAuditConfiguration iot:UpdateAuditSuppression iot:UpdateAuthorizer iot:UpdateBillingGroup iot:UpdateCACertificate iot:UpdateCertificate iot:UpdateCertificateProvider iot:UpdateCommand iot:UpdateCustomMetric iot:UpdateDimension iot:UpdateDomainConfiguration iot:UpdateDynamicThingGroup iot:UpdateEncryptionConfiguration iot:UpdateEventConfigurations iot:UpdateFleetMetric iot:UpdateIndexingConfiguration iot:UpdateJob iot:UpdateMitigationAction iot:UpdatePackage

服務前綴	動作
	iot:UpdatePackageConfiguration iot:UpdatePackageVersion iot:UpdateProvisioningTemplate iot:UpdateRoleAlias iot:UpdateScheduledAudit iot:UpdateSecurityProfile iot:UpdateStream iot:UpdateThing iot:UpdateThingGroup iot:UpdateThingGroupsForThing iot:UpdateThingType iot:UpdateTopicRuleDestination iot:ValidateSecurityProfileBehaviors

服務前綴	動作
iotanalytics	iotanalytics:CancelPipelineReprocessing
	iotanalytics:CreateChannel
	iotanalytics:CreateDataset
	iotanalytics:CreateDatasetContent
	iotanalytics:CreateDatastore
	iotanalytics:CreatePipeline
	iotanalytics:DeleteChannel
	iotanalytics:DeleteDataset
	iotanalytics:DeleteDatasetContent
	iotanalytics:DeleteDatastore
	iotanalytics:DeletePipeline
	iotanalytics:DescribeChannel
	iotanalytics:DescribeDataset
	iotanalytics:DescribeDatastore
	iotanalytics:DescribeLoggingOptions
	iotanalytics:DescribePipeline
	iotanalytics:GetDatasetContent
	iotanalytics:ListChannels
	iotanalytics:ListDatasetContents
	iotanalytics:ListDatasets
	iotanalytics:ListDatastores

服務前綴	動作
	iotanalytics:ListPipelines iotanalytics:PutLoggingOptions iotanalytics:RunPipelineActivity iotanalytics:SampleChannelData iotanalytics:StartPipelineReprocessing iotanalytics:UpdateChannel iotanalytics:UpdateDataset iotanalytics:UpdateDatastore iotanalytics:UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor:CreateSuiteDefinition iotdeviceadvisor:DeleteSuiteDefinition iotdeviceadvisor:GetEndpoint iotdeviceadvisor:GetSuiteDefinition iotdeviceadvisor:GetSuiteRun iotdeviceadvisor:GetSuiteRunReport iotdeviceadvisor:ListSuiteDefinitions iotdeviceadvisor:ListSuiteRuns iotdeviceadvisor:StartSuiteRun iotdeviceadvisor:StopSuiteRun iotdeviceadvisor:UpdateSuiteDefinition

服務前綴	動作
iotevents	iotevents:BatchAcknowledgeAlarm
	iotevents:BatchDeleteDetector
	iotevents:BatchDisableAlarm
	iotevents:BatchEnableAlarm
	iotevents:BatchResetAlarm
	iotevents:BatchSnoozeAlarm
	iotevents:BatchUpdateDetector
	iotevents:CreateAlarmModel
	iotevents:CreateDetectorModel
	iotevents:CreateInput
	iotevents:DeleteAlarmModel
	iotevents:DeleteDetectorModel
	iotevents:DeleteInput
	iotevents:DescribeAlarm
	iotevents:DescribeAlarmModel
	iotevents:DescribeDetector
	iotevents:DescribeDetectorModel
	iotevents:DescribeDetectorModelAnalysis
	iotevents:DescribeInput
	iotevents:DescribeLoggingOptions
	iotevents:GetDetectorModelAnalysisResults

服務前綴	動作
	iotevents:ListAlarmModelVersions iotevents:ListAlarmModels iotevents:ListAlarms iotevents:ListDetectorModelVersions iotevents:ListDetectorModels iotevents:ListDetectors iotevents:ListInputRoutings iotevents:ListInputs iotevents:PutLoggingOptions iotevents:StartDetectorModelAnalysis iotevents:UpdateAlarmModel iotevents:UpdateDetectorModel iotevents:UpdateInput
iotfleethub	iotfleethub:CreateApplication iotfleethub:DeleteApplication iotfleethub:DescribeApplication iotfleethub:ListApplications iotfleethub:UpdateApplication

服務前綴	動作
iotsitewise	iotsitewise:AssociateAssets
	iotsitewise:AssociateTimeSeriesToAssetProperty
	iotsitewise:BatchAssociateProjectAssets
	iotsitewise:BatchDisassociateProjectAssets
	iotsitewise:CreateAccessPolicy
	iotsitewise:CreateAsset
	iotsitewise:CreateAssetModel
	iotsitewise:CreateAssetModelCompositeModel
	iotsitewise:CreateBulkImportJob
	iotsitewise:CreateComputationModel
	iotsitewise:CreateDashboard
	iotsitewise:CreateDataset
	iotsitewise:CreateGateway
	iotsitewise:CreatePortal
	iotsitewise:CreateProject
	iotsitewise:DeleteAccessPolicy
	iotsitewise:DeleteAsset
	iotsitewise:DeleteAssetModel
	iotsitewise:DeleteAssetModelCompositeModel
	iotsitewise:DeleteComputationModel
	iotsitewise:DeleteDashboard

服務前綴	動作
	iotsitewise:DeleteDataset
	iotsitewise:DeleteGateway
	iotsitewise:DeletePortal
	iotsitewise:DeleteProject
	iotsitewise:DeleteTimeSeries
	iotsitewise:DescribeAccessPolicy
	iotsitewise:DescribeAsset
	iotsitewise:DescribeAssetCompositeModel
	iotsitewise:DescribeAssetModel
	iotsitewise:DescribeAssetModelCompositeModel
	iotsitewise:DescribeAssetModelInterfaceRelationship
	iotsitewise:DescribeAssetProperty
	iotsitewise:DescribeBulkImportJob
	iotsitewise:DescribeComputationModel
	iotsitewise:DescribeComputationModelExecutionSummary
	iotsitewise:DescribeDashboard
	iotsitewise:DescribeDataset
	iotsitewise:DescribeDefaultEncryptionConfiguration
	iotsitewise:DescribeExecution
	iotsitewise:DescribeGateway
	iotsitewise:DescribeGatewayCapabilityConfiguration

服務前綴	動作
	iotsitewise:DescribeLoggingOptions
	iotsitewise:DescribePortal
	iotsitewise:DescribeProject
	iotsitewise:DescribeStorageConfiguration
	iotsitewise:DescribeTimeSeries
	iotsitewise:DisassociateAssets
	iotsitewise:DisassociateTimeSeriesFromAssetProperty
	iotsitewise:ExecuteAction
	iotsitewise:ExecuteQuery
	iotsitewise:ListAccessPolicies
	iotsitewise:ListActions
	iotsitewise:ListAssetModelCompositeModels
	iotsitewise:ListAssetModelProperties
	iotsitewise:ListAssetModels
	iotsitewise:ListAssetProperties
	iotsitewise:ListAssetRelationships
	iotsitewise:ListAssets
	iotsitewise:ListAssociatedAssets
	iotsitewise:ListBulkImportJobs
	iotsitewise:ListCompositionRelationships
	iotsitewise:ListComputationModelDataBindingUsages

服務前綴	動作
	iotsitewise:ListComputationModelResolveToResources
	iotsitewise:ListComputationModels
	iotsitewise:ListDashboards
	iotsitewise:ListDatasets
	iotsitewise:ListExecutions
	iotsitewise:ListGateways
	iotsitewise:ListInterfaceRelationships
	iotsitewise:ListPortals
	iotsitewise:ListProjectAssets
	iotsitewise:ListProjects
	iotsitewise:ListTimeSeries
	iotsitewise:PutDefaultEncryptionConfiguration
	iotsitewise:PutLoggingOptions
	iotsitewise:PutStorageConfiguration
	iotsitewise:UpdateAccessPolicy
	iotsitewise:UpdateAsset
	iotsitewise:UpdateAssetModel
	iotsitewise:UpdateAssetModelCompositeModel
	iotsitewise:UpdateAssetProperty
	iotsitewise:UpdateComputationModel
	iotsitewise:UpdateDashboard

服務前綴	動作
	iotsitewise:UpdateDataset iotsitewise:UpdateGateway iotsitewise:UpdateGatewayCapabilityConfiguration iotsitewise:UpdatePortal iotsitewise:UpdateProject

服務前綴	動作
iottwinmaker	iottwinmaker:CancelMetadataTransferJob iottwinmaker:CreateComponentType iottwinmaker:CreateEntity iottwinmaker:CreateMetadataTransferJob iottwinmaker:CreateScene iottwinmaker:CreateSyncJob iottwinmaker:CreateWorkspace iottwinmaker:DeleteComponentType iottwinmaker:DeleteEntity iottwinmaker:DeleteScene iottwinmaker:DeleteSyncJob iottwinmaker:DeleteWorkspace iottwinmaker:ExecuteQuery iottwinmaker:GetMetadataTransferJob iottwinmaker:GetPricingPlan iottwinmaker:GetScene iottwinmaker:GetSyncJob iottwinmaker:ListComponentTypes iottwinmaker:ListComponents iottwinmaker:ListEntities iottwinmaker:ListMetadataTransferJobs

服務前綴	動作
	iottwinmaker:ListProperties iottwinmaker:ListScenes iottwinmaker:ListSyncJobs iottwinmaker:ListSyncResources iottwinmaker:ListWorkspaces iottwinmaker:UpdateComponentType iottwinmaker:UpdateEntity iottwinmaker:UpdatePricingPlan iottwinmaker:UpdateScene iottwinmaker:UpdateWorkspace

服務前綴	動作
iotwireless	iotwireless:AssociateAwsAccountWithPartnerAccount
	iotwireless:AssociateMulticastGroupWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithMulticastGroup
	iotwireless:AssociateWirelessDeviceWithThing
	iotwireless:AssociateWirelessGatewayWithCertificate
	iotwireless:AssociateWirelessGatewayWithThing
	iotwireless:CancelMulticastGroupSession
	iotwireless:CreateDestination
	iotwireless:CreateDeviceProfile
	iotwireless:CreateFuotaTask
	iotwireless:CreateMulticastGroup
	iotwireless:CreateNetworkAnalyzerConfiguration
	iotwireless:CreateServiceProfile
	iotwireless:CreateWirelessDevice
	iotwireless:CreateWirelessGateway
	iotwireless:CreateWirelessGatewayTask
	iotwireless:CreateWirelessGatewayTaskDefinition
	iotwireless:DeleteDestination
	iotwireless:DeleteDeviceProfile
	iotwireless:DeleteFuotaTask

服務前綴	動作
	iotwireless:DeleteMulticastGroup
	iotwireless:DeleteNetworkAnalyzerConfiguration
	iotwireless:DeleteQueuedMessages
	iotwireless:DeleteServiceProfile
	iotwireless:DeleteWirelessDevice
	iotwireless:DeleteWirelessDeviceImportTask
	iotwireless:DeleteWirelessGateway
	iotwireless:DeleteWirelessGatewayTask
	iotwireless:DeleteWirelessGatewayTaskDefinition
	iotwireless:DeregisterWirelessDevice
	iotwireless:DisassociateAwsAccountFromPartnerAccount
	iotwireless:DisassociateMulticastGroupFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromMulticastGroup
	iotwireless:DisassociateWirelessDeviceFromThing
	iotwireless:DisassociateWirelessGatewayFromCertificate
	iotwireless:DisassociateWirelessGatewayFromThing
	iotwireless:GetDestination
	iotwireless:GetDeviceProfile
	iotwireless:GetEventConfigurationByResourceTypes
	iotwireless:GetFuotaTask

服務前綴	動作
	iotwireless:GetLogLevelsByResourceTypes
	iotwireless:GetMetricConfiguration
	iotwireless:GetMetrics
	iotwireless:GetMulticastGroup
	iotwireless:GetMulticastGroupSession
	iotwireless:GetNetworkAnalyzerConfiguration
	iotwireless:GetPartnerAccount
	iotwireless:GetPosition
	iotwireless:GetPositionConfiguration
	iotwireless:GetPositionEstimate
	iotwireless:GetResourceEventConfiguration
	iotwireless:GetResourceLogLevel
	iotwireless:GetResourcePosition
	iotwireless:GetServiceEndpoint
	iotwireless:GetServiceProfile
	iotwireless:GetWirelessDevice
	iotwireless:GetWirelessDeviceImportTask
	iotwireless:GetWirelessDeviceStatistics
	iotwireless:GetWirelessGateway
	iotwireless:GetWirelessGatewayCertificate
	iotwireless:GetWirelessGatewayFirmwareInformation

服務前綴	動作
	iotwireless:GetWirelessGatewayStatistics
	iotwireless:GetWirelessGatewayTask
	iotwireless:GetWirelessGatewayTaskDefinition
	iotwireless:ListDestinations
	iotwireless:ListDeviceProfiles
	iotwireless:ListDevicesForWirelessDeviceImportTask
	iotwireless:ListEventConfigurations
	iotwireless:ListFuotaTasks
	iotwireless:ListMulticastGroups
	iotwireless:ListMulticastGroupsByFuotaTask
	iotwireless:ListNetworkAnalyzerConfigurations
	iotwireless:ListPartnerAccounts
	iotwireless:ListPositionConfigurations
	iotwireless:ListQueuedMessages
	iotwireless:ListServiceProfiles
	iotwireless:ListWirelessDeviceImportTasks
	iotwireless:ListWirelessDevices
	iotwireless:ListWirelessGatewayTaskDefinitions
	iotwireless:ListWirelessGateways
	iotwireless:PutPositionConfiguration
	iotwireless:PutResourceLogLevel

服務前綴	動作
	iotwireless:ResetAllResourceLogLevels
	iotwireless:ResetResourceLogLevel
	iotwireless:SendDataToMulticastGroup
	iotwireless:SendDataToWirelessDevice
	iotwireless:StartBulkAssociateWirelessDeviceWithMulticastGroup
	iotwireless:StartBulkDisassociateWirelessDeviceFromMulticastGroup
	iotwireless:StartFuotaTask
	iotwireless:StartMulticastGroupSession
	iotwireless:StartNetworkAnalyzerStream
	iotwireless:StartSingleWirelessDeviceImportTask
	iotwireless:StartWirelessDeviceImportTask
	iotwireless:TestWirelessDevice
	iotwireless:UpdateDestination
	iotwireless:UpdateEventConfigurationByResourceTypes
	iotwireless:UpdateFuotaTask
	iotwireless:UpdateLogLevelsByResourceTypes
	iotwireless:UpdateMetricConfiguration
	iotwireless:UpdateMulticastGroup
	iotwireless:UpdateNetworkAnalyzerConfiguration
	iotwireless:UpdatePartnerAccount

服務前綴	動作
	iotwireless:UpdatePosition iotwireless:UpdateResourceEventConfiguration iotwireless:UpdateResourcePosition iotwireless:UpdateWirelessDevice iotwireless:UpdateWirelessDeviceImportTask iotwireless:UpdateWirelessGateway

服務前綴	動作
ivs	ivs:BatchGetChannel ivs:BatchGetStreamKey ivs:BatchStartViewerSessionRevocation ivs：CreateAdConfiguration ivs:CreateChannel ivs:CreateEncoderConfiguration ivs:CreateIngestConfiguration ivs:CreateParticipantToken ivs:CreatePlaybackRestrictionPolicy ivs:CreateRecordingConfiguration ivs:CreateStorageConfiguration ivs:CreateStreamKey ivs：DeleteAdConfiguration ivs:DeleteChannel ivs:DeleteEncoderConfiguration ivs:DeleteIngestConfiguration ivs:DeletePlaybackKeyPair ivs:DeletePlaybackRestrictionPolicy ivs:DeletePublicKey ivs:DeleteRecordingConfiguration ivs:DeleteStorageConfiguration

服務前綴	動作
	ivs:DeleteStreamKey ivs:DisconnectParticipant ivs：GetAdConfiguration ivs:GetChannel ivs:GetComposition ivs:GetEncoderConfiguration ivs:GetIngestConfiguration ivs:GetParticipant ivs:GetPlaybackKeyPair ivs:GetPlaybackRestrictionPolicy ivs:GetPublicKey ivs:GetRecordingConfiguration ivs:GetStorageConfiguration ivs:GetStream ivs:GetStreamKey ivs:GetStreamSession ivs:ImportPlaybackKeyPair ivs:ImportPublicKey ivs：InsertAdBreak ivs：ListAdConfigurations ivs:ListChannels

服務前綴	動作
	ivs:ListCompositions ivs:ListEncoderConfigurations ivs:ListIngestConfigurations ivs:ListParticipantEvents ivs:ListParticipantReplicas ivs:ListParticipants ivs:ListPlaybackKeyPairs ivs:ListPlaybackRestrictionPolicies ivs:ListPublicKeys ivs:ListRecordingConfigurations ivs:ListStorageConfigurations ivs:ListStreamKeys ivs:ListStreamSessions ivs:ListStreams ivs:PutMetadata ivs:StartComposition ivs:StartViewerSessionRevocation ivs:StopComposition ivs:StopStream ivs：UpdateAdConfiguration ivs:UpdateChannel

服務前綴	動作
	ivs:UpdateIngestConfiguration ivs:UpdatePlaybackRestrictionPolicy
ivschat	ivschat:CreateChatToken ivschat:CreateLoggingConfiguration ivschat:CreateRoom ivschat:DeleteLoggingConfiguration ivschat:DeleteMessage ivschat:DeleteRoom ivschat:DisconnectUser ivschat:GetLoggingConfiguration ivschat:GetRoom ivschat:ListLoggingConfigurations ivschat:ListRooms ivschat:SendEvent ivschat:UpdateLoggingConfiguration ivschat:UpdateRoom

服務前綴	動作
kafka	kafka:BatchAssociateScramSecret kafka:BatchDisassociateScramSecret kafka:CreateCluster kafka:CreateClusterV2 kafka:CreateConfiguration kafka:CreateReplicator kafka:CreateVpcConnection kafka:DeleteCluster kafka:DeleteClusterPolicy kafka:DeleteConfiguration kafka:DeleteReplicator kafka:DeleteVpcConnection kafka:DescribeCluster kafka:DescribeClusterOperation kafka:DescribeClusterOperationV2 kafka:DescribeClusterV2 kafka:DescribeConfiguration kafka:DescribeConfigurationRevision kafka:DescribeVpcConnection kafka:GetBootstrapBrokers kafka:GetClusterPolicy

服務前綴	動作
	kafka:GetCompatibleKafkaVersions kafka:ListClientVpcConnections kafka:ListClusterOperations kafka:ListClusterOperationsV2 kafka:ListClusters kafka:ListClustersV2 kafka:ListConfigurationRevisions kafka:ListConfigurations kafka:ListKafkaVersions kafka:ListNodes kafka:ListReplicators kafka:ListScramSecrets kafka:ListVpcConnections kafka:PutClusterPolicy kafka:RebootBroker kafka:RejectClientVpcConnection kafka:UpdateBrokerCount kafka:UpdateBrokerStorage kafka:UpdateBrokerType kafka:UpdateClusterConfiguration kafka:UpdateClusterKafkaVersion

服務前綴	動作
	kafka:UpdateConfiguration kafka:UpdateConnectivity kafka:UpdateMonitoring kafka：UpdateRebalancing kafka:UpdateReplicationInfo kafka:UpdateSecurity kafka:UpdateStorage
kafkaconnect	kafkaconnect:CreateConnector kafkaconnect:CreateCustomPlugin kafkaconnect:CreateWorkerConfiguration kafkaconnect:DeleteConnector kafkaconnect:DeleteCustomPlugin kafkaconnect:DeleteWorkerConfiguration kafkaconnect:DescribeConnector kafkaconnect:DescribeCustomPlugin kafkaconnect:DescribeWorkerConfiguration kafkaconnect:ListConnectorOperations kafkaconnect:ListConnectors kafkaconnect:ListCustomPlugins kafkaconnect:ListWorkerConfigurations kafkaconnect:UpdateConnector

服務前綴	動作
kendra	kendra:AssociateEntitiesToExperience kendra:AssociatePersonasToEntities kendra:BatchDeleteDocument kendra:BatchDeleteFeaturedResultsSet kendra:BatchGetDocumentStatus kendra:BatchPutDocument kendra:ClearQuerySuggestions kendra:CreateAccessControlConfiguration kendra:CreateDataSource kendra:CreateExperience kendra:CreateFaq kendra:CreateFeaturedResultsSet kendra:CreateIndex kendra:CreateQuerySuggestionsBlockList kendra:CreateThesaurus kendra:DeleteDataSource kendra:DeleteExperience kendra:DeleteFaq kendra:DeleteIndex kendra:DeletePrincipalMapping kendra:DeleteQuerySuggestionsBlockList

服務前綴	動作
	kendra:DeleteThesaurus kendra:DescribeAccessControlConfiguration kendra:DescribeDataSource kendra:DescribeExperience kendra:DescribeFaq kendra:DescribeFeaturedResultsSet kendra:DescribeIndex kendra:DescribePrincipalMapping kendra:DescribeQuerySuggestionsBlockList kendra:DescribeQuerySuggestionsConfig kendra:DescribeThesaurus kendra:DisassociateEntitiesFromExperience kendra:DisassociatePersonasFromEntities kendra:GetQuerySuggestions kendra:GetSnapshots kendra:ListAccessControlConfigurations kendra:ListDataSourceSyncJobs kendra:ListDataSources kendra:ListEntityPersonas kendra:ListExperienceEntities kendra:ListExperiences

服務前綴	動作
	kendra:ListFaqs kendra:ListFeaturedResultsSets kendra:ListGroupsOlderThanOrderingId kendra:ListIndices kendra:ListQuerySuggestionsBlockLists kendra:ListThesauri kendra:PutPrincipalMapping kendra:Query kendra:Retrieve kendra:StartDataSourceSyncJob kendra:StopDataSourceSyncJob kendra:SubmitFeedback kendra:UpdateDataSource kendra:UpdateExperience kendra:UpdateFeaturedResultsSet kendra:UpdateIndex kendra:UpdateQuerySuggestionsBlockList kendra:UpdateQuerySuggestionsConfig kendra:UpdateThesaurus

服務前綴	動作
kinesis	kinesis:CreateStream
	kinesis:DecreaseStreamRetentionPeriod
	kinesis:DeleteStream
	kinesis:DeregisterStreamConsumer
	kinesis：DescribeAccountSettings
	kinesis:DescribeLimits
	kinesis:DescribeStream
	kinesis:DescribeStreamConsumer
	kinesis:DescribeStreamSummary
	kinesis:DisableEnhancedMonitoring
	kinesis:EnableEnhancedMonitoring
	kinesis:IncreaseStreamRetentionPeriod
	kinesis:ListShards
	kinesis:ListStreamConsumers
	kinesis:ListStreams
	kinesis:MergeShards
	kinesis:RegisterStreamConsumer
	kinesis:SplitShard
	kinesis:StartStreamEncryption
	kinesis:StopStreamEncryption
	kinesis：UpdateAccountSettings

服務前綴	動作
	kinesis:UpdateShardCount kinesis:UpdateStreamMode

服務前綴	動作
kinesisanalytics	kinesisanalytics:AddApplicationCloudWatchLoggingOption kinesisanalytics:AddApplicationInput kinesisanalytics:AddApplicationInputProcessingConfiguration kinesisanalytics:AddApplicationOutput kinesisanalytics:AddApplicationReferenceDataSource kinesisanalytics:AddApplicationVpcConfiguration kinesisanalytics:CreateApplication kinesisanalytics:CreateApplicationPresignedUrl kinesisanalytics:CreateApplicationSnapshot kinesisanalytics:DeleteApplication kinesisanalytics:DeleteApplicationCloudWatchLoggingOption kinesisanalytics:DeleteApplicationInputProcessingConfiguration kinesisanalytics:DeleteApplicationOutput kinesisanalytics:DeleteApplicationReferenceDataSource kinesisanalytics:DeleteApplicationSnapshot kinesisanalytics:DeleteApplicationVpcConfiguration kinesisanalytics:DescribeApplication kinesisanalytics:DescribeApplicationOperation kinesisanalytics:DescribeApplicationSnapshot kinesisanalytics:DescribeApplicationVersion kinesisanalytics:DiscoverInputSchema

服務前綴	動作
	kinesisanalytics:ListApplicationOperations kinesisanalytics:ListApplicationSnapshots kinesisanalytics:ListApplicationVersions kinesisanalytics:ListApplications kinesisanalytics:RollbackApplication kinesisanalytics:StartApplication kinesisanalytics:StopApplication kinesisanalytics:UpdateApplication kinesisanalytics:UpdateApplicationMaintenanceConfiguration

服務前綴	動作
kms	kms:CancelKeyDeletion kms:ConnectCustomKeyStore kms:CreateAlias kms:CreateCustomKeyStore kms:創建權限 kms:CreateKey kms:解密 kms:DeleteAlias kms:DeleteCustomKeyStore kms:DeleteImportedKeyMaterial kms:DeriveSharedSecret kms:DescribeCustomKeyStores kms:描述金鑰 kms:DisableKey kms:DisableKeyRotation kms:DisconnectCustomKeyStore kms:EnableKey kms:EnableKeyRotation kms:Encrypt kms:產生資料金鑰 kms:GenerateDataKeyPair

服務前綴	動作
	kms:GenerateDataKeyPairWithoutPlaintext
	kms:GenerateDataKeyWithoutPlaintext
	kms:GenerateMac
	kms:GenerateRandom
	kms:GetKeyLastUsage
	kms:GetKeyPolicy
	kms:GetKeyRotationStatus
	kms:GetParametersForImport
	kms:GetPublicKey
	kms:ImportKeyMaterial
	kms:ListAliases
	kms:ListGrants
	kms:ListKeyPolicies
	kms:ListKeyRotations
	kms:ListKeys
	kms:ListRetirableGrants
	kms:ReplicateKey
	kms:RetireGrant
	kms:RevokeGrant
	kms:RotateKeyOnDemand
	kms:ScheduleKeyDeletion

服務前綴	動作
	kms:Sign kms:UpdateAlias kms:UpdateCustomKeyStore kms:UpdateKeyDescription kms:UpdatePrimaryRegion kms:Verify kms:VerifyMac

服務前綴	動作
lambda	lambda:AddLayerVersionPermission lambda:AddPermission lambda:CreateAlias lambda:CreateCodeSigningConfig lambda:CreateEventSourceMapping lambda:CreateFunction lambda:CreateFunctionUrlConfig lambda:DeleteAlias lambda：DeleteCapacityProvider lambda:DeleteCodeSigningConfig lambda:DeleteEventSourceMapping lambda:DeleteFunction lambda:DeleteFunctionCodeSigningConfig lambda:DeleteFunctionConcurrency lambda:DeleteFunctionEventInvokeConfig lambda:DeleteFunctionUrlConfig lambda:DeleteLayerVersion lambda:DeleteProvisionedConcurrencyConfig lambda:GetAccountSettings lambda:GetAlias lambda：GetCapacityProvider

服務前綴	動作
	lambda:GetCodeSigningConfig lambda:GetEventSourceMapping lambda:GetFunction lambda:GetFunctionCodeSigningConfig lambda:GetFunctionConcurrency lambda:GetFunctionConfiguration lambda:GetFunctionEventInvokeConfig lambda:GetFunctionRecursionConfig lambda：GetFunctionScalingConfig lambda:GetFunctionUrlConfig lambda:GetLayerVersion lambda:GetLayerVersionPolicy lambda:GetPolicy lambda:GetProvisionedConcurrencyConfig lambda:GetRuntimeManagementConfig lambda:ListAliases lambda：ListCapacityProviders lambda:ListCodeSigningConfigs lambda：ListDurableExecutionsByFunction lambda:ListEventSourceMappings lambda:ListFunctionEventInvokeConfigs

服務前綴	動作
	lambda:ListFunctionUrlConfigs lambda:ListFunctions lambda:ListFunctionsByCodeSigningConfig lambda:ListLayerVersions lambda:ListLayers lambda:ListProvisionedConcurrencyConfigs lambda:ListVersionsByFunction lambda:PublishLayerVersion lambda:PublishVersion lambda:PutFunctionCodeSigningConfig lambda:PutFunctionConcurrency lambda:PutFunctionEventInvokeConfig lambda:PutFunctionRecursionConfig lambda：PutFunctionScalingConfig lambda:PutProvisionedConcurrencyConfig lambda:PutRuntimeManagementConfig lambda:RemoveLayerVersionPermission lambda:RemovePermission lambda:UpdateAlias lambda：UpdateCapacityProvider lambda:UpdateCodeSigningConfig

服務前綴	動作
	lambda:UpdateEventSourceMapping lambda:UpdateFunctionCode lambda:UpdateFunctionConfiguration lambda:UpdateFunctionEventInvokeConfig lambda:UpdateFunctionUrlConfig

服務前綴	動作
lex	lex:BatchCreateCustomVocabularyItem
	lex:BatchDeleteCustomVocabularyItem
	lex:BatchUpdateCustomVocabularyItem
	lex:BuildBotLocale
	lex:CreateBotAlias
	lex:CreateBotReplica
	lex:CreateBotVersion
	lex:CreateExport
	lex:CreateIntentVersion
	lex:CreateResourcePolicy
	lex:CreateSlotTypeVersion
	lex:CreateTestSetDiscrepancyReport
	lex:CreateUploadUrl
	lex:DeleteBot
	lex：DeleteBotAnalyzerRecommendation
	lex:DeleteBotChannelAssociation
	lex:DeleteBotReplica
	lex:DeleteExport
	lex:DeleteImport
	lex:DeleteIntentVersion
	lex:DeleteResourcePolicy

服務前綴	動作
	lex:DeleteSlotTypeVersion
	lex:DeleteTestSet
	lex:DeleteUtterances
	lex:DescribeBotAlias
	lex：DescribeBotAnalyzerRecommendation
	lex:DescribeBotRecommendation
	lex:DescribeBotReplica
	lex:DescribeBotResourceGeneration
	lex:DescribeBotVersion
	lex:DescribeCustomVocabularyMetadata
	lex:DescribeExport
	lex:DescribeImport
	lex:DescribeResourcePolicy
	lex:DescribeTestExecution
	lex:DescribeTestSet
	lex:DescribeTestSetDiscrepancyReport
	lex:DescribeTestSetGeneration
	lex:GenerateBotElement
	lex:GetBot
	lex:GetBotAlias
	lex:GetBotAliases

服務前綴	動作
	lex:GetBotChannelAssociation
	lex:GetBotChannelAssociations
	lex:GetBotVersions
	lex:GetBots
	lex:GetBuiltinIntent
	lex:GetBuiltinIntents
	lex:GetBuiltinSlotTypes
	lex:GetExport
	lex:GetImport
	lex:GetIntent
	lex:GetIntentVersions
	lex:GetIntents
	lex:GetMigration
	lex:GetMigrations
	lex:GetSlotType
	lex:GetSlotTypeVersions
	lex:GetSlotTypes
	lex:GetTestExecutionArtifactsUrl
	lex:GetUtterancesView
	lex:ListBotAliasReplicas
	lex:ListBotAliases

服務前綴	動作
	lex:ListBotRecommendations
	lex:ListBotReplicas
	lex:ListBotResourceGenerations
	lex:ListBotVersionReplicas
	lex:ListBotVersions
	lex:ListBots
	lex:ListBuiltInIntents
	lex:ListBuiltInSlotTypes
	lex:ListCustomVocabularyItems
	lex:ListExports
	lex:ListImports
	lex:ListIntentMetrics
	lex:ListIntentPaths
	lex:ListRecommendedIntents
	lex:ListSessionAnalyticsData
	lex:ListSessionMetrics
	lex:ListTestExecutionResultItems
	lex:ListTestExecutions
	lex:ListTestSets
	lex:PutBot
	lex:PutBotAlias

服務前綴	動作
	lex:PutIntent
	lex:PutSlotType
	lex:SearchAssociatedTranscripts
	lex：StartBotAnalyzer
	lex:StartBotRecommendation
	lex:StartImport
	lex:StartMigration
	lex:StartTestExecution
	lex:StartTestSetGeneration
	lex：StopBotAnalyzer
	lex:StopBotRecommendation
	lex:UpdateBotAlias
	lex:UpdateBotRecommendation
	lex:UpdateExport
	lex:UpdateResourcePolicy

服務前綴	動作
license-manager-linux-subscriptions	license-manager-linux-subscriptions:DeregisterSubscriptionProvider license-manager-linux-subscriptions:GetRegisteredSubscriptionProvider license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions:ListRegisteredSubscriptionProviders license-manager-linux-subscriptions:RegisterSubscriptionProvider license-manager-linux-subscriptions:UpdateServiceSettings

服務前綴	動作
lightsail	lightsail:AllocateStaticIp
	lightsail:AttachCertificateToDistribution
	lightsail:AttachDisk
	lightsail:AttachInstancesToLoadBalancer
	lightsail:AttachLoadBalancerTlsCertificate
	lightsail:AttachStaticIp
	lightsail:CloseInstancePublicPorts
	lightsail:CopySnapshot
	lightsail:CreateBucket
	lightsail:CreateBucketAccessKey
	lightsail:CreateCertificate
	lightsail:CreateCloudFormationStack
	lightsail:CreateContactMethod
	lightsail:CreateContainerService
	lightsail:CreateContainerServiceDeployment
	lightsail:CreateContainerServiceRegistryLogin
	lightsail:CreateDisk
	lightsail:CreateDiskFromSnapshot
	lightsail:CreateDiskSnapshot
	lightsail:CreateDistribution
	lightsail:CreateDomain

服務前綴	動作
	lightsail:CreateGUISessionAccessDetails
	lightsail:CreateInstanceSnapshot
	lightsail:CreateInstances
	lightsail:CreateInstancesFromSnapshot
	lightsail:CreateKeyPair
	lightsail:CreateLoadBalancer
	lightsail:CreateLoadBalancerTlsCertificate
	lightsail:CreateRelationalDatabase
	lightsail:CreateRelationalDatabaseFromSnapshot
	lightsail:CreateRelationalDatabaseSnapshot
	lightsail:DeleteAlarm
	lightsail:DeleteAutoSnapshot
	lightsail:DeleteBucket
	lightsail:DeleteBucketAccessKey
	lightsail:DeleteCertificate
	lightsail:DeleteContactMethod
	lightsail:DeleteContainerImage
	lightsail:DeleteContainerService
	lightsail:DeleteDisk
	lightsail:DeleteDiskSnapshot
	lightsail:DeleteDistribution

服務前綴	動作
	lightsail:DeleteDomain
	lightsail:DeleteDomainEntry
	lightsail:DeleteInstance
	lightsail:DeleteInstanceSnapshot
	lightsail:DeleteKeyPair
	lightsail:DeleteKnownHostKeys
	lightsail:DeleteLoadBalancer
	lightsail:DeleteLoadBalancerTlsCertificate
	lightsail:DeleteRelationalDatabase
	lightsail:DeleteRelationalDatabaseSnapshot
	lightsail:DetachCertificateFromDistribution
	lightsail:DetachDisk
	lightsail:DetachInstancesFromLoadBalancer
	lightsail:DetachStaticIp
	lightsail:DisableAddOn
	lightsail:DownloadDefaultKeyPair
	lightsail:EnableAddOn
	lightsail:ExportSnapshot
	lightsail:GetActiveNames
	lightsail:GetAlarms
	lightsail:GetAutoSnapshots

服務前綴	動作
	lightsail:GetBlueprints
	lightsail:GetBucketAccessKeys
	lightsail:GetBucketBundles
	lightsail:GetBucketMetricData
	lightsail:GetBuckets
	lightsail:GetBundles
	lightsail:GetCertificates
	lightsail:GetCloudFormationStackRecords
	lightsail:GetContactMethods
	lightsail:GetContainerAPIMetadata
	lightsail:GetContainerImages
	lightsail:GetContainerLog
	lightsail:GetContainerServiceDeployments
	lightsail:GetContainerServiceMetricData
	lightsail:GetContainerServicePowers
	lightsail:GetContainerServices
	lightsail:GetCostEstimate
	lightsail:GetDisk
	lightsail:GetDiskSnapshot
	lightsail:GetDiskSnapshots
	lightsail:GetDisks

服務前綴	動作
	lightsail:GetDistributionBundles
	lightsail:GetDistributionLatestCacheReset
	lightsail:GetDistributionMetricData
	lightsail:GetDistributions
	lightsail:GetDomain
	lightsail:GetExportSnapshotRecords
	lightsail:GetInstance
	lightsail:GetInstanceMetricData
	lightsail:GetInstancePortStates
	lightsail:GetInstanceSnapshot
	lightsail:GetInstanceSnapshots
	lightsail:GetInstanceState
	lightsail:GetInstances
	lightsail:GetKeyPair
	lightsail:GetKeyPairs
	lightsail:GetLoadBalancer
	lightsail:GetLoadBalancerMetricData
	lightsail:GetLoadBalancerTlsCertificates
	lightsail:GetLoadBalancerTlsPolicies
	lightsail:GetLoadBalancers
	lightsail:GetOperation

服務前綴	動作
	lightsail:GetOperations
	lightsail:GetOperationsForResource
	lightsail:GetRegions
	lightsail:GetRelationalDatabase
	lightsail:GetRelationalDatabaseBlueprints
	lightsail:GetRelationalDatabaseBundles
	lightsail:GetRelationalDatabaseEvents
	lightsail:GetRelationalDatabaseLogEvents
	lightsail:GetRelationalDatabaseLogStreams
	lightsail:GetRelationalDatabaseMasterUserPassword
	lightsail:GetRelationalDatabaseMetricData
	lightsail:GetRelationalDatabaseParameters
	lightsail:GetRelationalDatabaseSnapshot
	lightsail:GetRelationalDatabaseSnapshots
	lightsail:GetRelationalDatabases
	lightsail:GetSetupHistory
	lightsail:GetStaticIp
	lightsail:GetStaticIps
	lightsail:ImportKeyPair
	lightsail:IsVpcPeered
	lightsail:OpenInstancePublicPorts

服務前綴	動作
	lightsail:PeerVpc
	lightsail:PutAlarm
	lightsail:PutInstancePublicPorts
	lightsail:RebootInstance
	lightsail:RebootRelationalDatabase
	lightsail:RegisterContainerImage
	lightsail:ReleaseStaticIp
	lightsail:ResetDistributionCache
	lightsail:SendContactMethodVerification
	lightsail:SetIpAddressType
	lightsail:SetResourceAccessForBucket
	lightsail:SetupInstanceHttps
	lightsail:StartGUISession
	lightsail:StartInstance
	lightsail:StartRelationalDatabase
	lightsail:StopGUISession
	lightsail:StopInstance
	lightsail:StopRelationalDatabase
	lightsail:TestAlarm
	lightsail:UnpeerVpc
	lightsail:UpdateBucket

服務前綴	動作
	lightsail:UpdateBucketBundle lightsail:UpdateContainerService lightsail:UpdateDistribution lightsail:UpdateDistributionBundle lightsail:UpdateDomainEntry lightsail:UpdateInstanceMetadataOptions lightsail:UpdateLoadBalancerAttribute lightsail:UpdateRelationalDatabase lightsail:UpdateRelationalDatabaseParameters

服務前綴	動作
日誌	logs:AssociateKmsKey log：AssociateSourceToS3TableIntegration logs:CancelExportTask log：CancelImportTask logs:CreateDelivery logs:CreateExportTask logs:CreateLogAnomalyDetector logs:CreateLogGroup logs:CreateLogStream log：CreateLookupTable logs:DeleteDataProtectionPolicy logs:DeleteDelivery logs:DeleteDeliveryDestination logs:DeleteDeliveryDestinationPolicy logs:DeleteDeliverySource logs:DeleteDestination logs:DeleteIndexPolicy logs:DeleteIntegration logs:DeleteLogAnomalyDetector logs:DeleteLogGroup logs:DeleteLogStream

服務前綴	動作
	log：DeleteLookupTable logs:DeleteMetricFilter logs:DeleteQueryDefinition logs:DeleteResourcePolicy logs:DeleteRetentionPolicy log：DeleteScheduledQuery logs:DeleteSubscriptionFilter log：DeleteSyslogConfiguration logs:DeleteTransformer logs:DescribeAccountPolicies logs:DescribeConfigurationTemplates logs:DescribeDeliveries logs:DescribeDeliveryDestinations logs:DescribeDeliverySources logs:DescribeDestinations logs:DescribeExportTasks logs:DescribeFieldIndexes log：DescribeImportTaskBatches log：DescribeImportTasks logs:DescribeIndexPolicies logs:DescribeLogGroups

服務前綴	動作
	logs:DescribeLogStreams log：DescribeLookupTables logs:DescribeMetricFilters logs:DescribeQueries logs:DescribeQueryDefinitions logs:DescribeResourcePolicies logs:DescribeSubscriptionFilters logs:DisassociateKmsKey log：DisassociateSourceFromS3TableIntegration logs:GetDataProtectionPolicy logs:GetDelivery logs:GetDeliveryDestination logs:GetDeliveryDestinationPolicy logs:GetDeliverySource logs:GetIntegration logs:GetLogAnomalyDetector log：GetLogFields logs:GetLogGroupFields logs:GetLogRecord log：GetLookupTable logs:GetQueryResults

服務前綴	動作
	log：GetScheduledQuery log：GetScheduledQueryHistory log：GetStorageTierPolicy logs:GetTransformer logs:ListAnomalies logs:ListIntegrations logs:ListLogAnomalyDetectors logs:ListLogGroupsForQuery log：ListScheduledQueries log：ListSourcesForS3TableIntegration log：ListSyslogConfigurations log：PutBearerTokenAuthentication logs:PutDataProtectionPolicy logs:PutDeliveryDestination logs:PutDeliveryDestinationPolicy logs:PutDeliverySource logs:PutDestination logs:PutDestinationPolicy logs:PutIndexPolicy logs:PutIntegration log：PutLogGroupDeletionProtection

服務前綴	動作
	logs:PutMetricFilter logs:PutQueryDefinition logs:PutResourcePolicy logs:PutRetentionPolicy log：PutStorageTierPolicy logs:PutSubscriptionFilter log：PutSyslogConfiguration logs:PutTransformer logs:StartLiveTail logs:StartQuery logs:StopQuery logs:TestMetricFilter logs:TestTransformer logs:UpdateAnomaly logs:UpdateDeliveryConfiguration logs:UpdateLogAnomalyDetector log：UpdateLookupTable

服務前綴	動作
lookoutequipment	lookoutequipment:CreateDataset lookoutequipment:CreateInferenceScheduler lookoutequipment:CreateLabel lookoutequipment:CreateLabelGroup lookoutequipment:CreateModel lookoutequipment:DeleteDataset lookoutequipment:DeleteInferenceScheduler lookoutequipment:DeleteLabel lookoutequipment:DeleteLabelGroup lookoutequipment:DeleteModel lookoutequipment:DeleteResourcePolicy lookoutequipment:DeleteRetrainingScheduler lookoutequipment:DescribeDataIngestionJob lookoutequipment:DescribeDataset lookoutequipment:DescribeInferenceScheduler lookoutequipment:DescribeLabelGroup lookoutequipment:DescribeModel lookoutequipment:DescribeModelVersion lookoutequipment:DescribeResourcePolicy lookoutequipment:DescribeRetrainingScheduler lookoutequipment:Describelabel

服務前綴	動作
	lookoutequipment:ImportDataset
	lookoutequipment:ImportModelVersion
	lookoutequipment:ListDataIngestionJobs
	lookoutequipment:ListDatasets
	lookoutequipment:ListInferenceEvents
	lookoutequipment:ListInferenceExecutions
	lookoutequipment:ListInferenceSchedulers
	lookoutequipment:ListLabelGroups
	lookoutequipment:ListLabels
	lookoutequipment:ListModelVersions
	lookoutequipment:ListModels
	lookoutequipment:ListRetrainingSchedulers
	lookoutequipment:ListSensorStatistics
	lookoutequipment:PutResourcePolicy
	lookoutequipment:StartDataIngestionJob
	lookoutequipment:StartInferenceScheduler
	lookoutequipment:StartRetrainingScheduler
	lookoutequipment:StopInferenceScheduler
	lookoutequipment:StopRetrainingScheduler
	lookoutequipment:UpdateActiveModelVersion
	lookoutequipment:UpdateInferenceScheduler

服務前綴	動作
	lookoutequipment:UpdateLabelGroup lookoutequipment:UpdateModel lookoutequipment:UpdateRetrainingScheduler

服務前綴	動作
lookoutmetrics	lookoutmetrics:ActivateAnomalyDetector lookoutmetrics:BackTestAnomalyDetector lookoutmetrics:CreateAlert lookoutmetrics:CreateAnomalyDetector lookoutmetrics:CreateMetricSet lookoutmetrics:DeactivateAnomalyDetector lookoutmetrics:DeleteAlert lookoutmetrics:DeleteAnomalyDetector lookoutmetrics:DescribeAlert lookoutmetrics:DescribeAnomalyDetectionExecutions lookoutmetrics:DescribeAnomalyDetector lookoutmetrics:DescribeMetricSet lookoutmetrics:DetectMetricSetConfig lookoutmetrics:GetAnomalyGroup lookoutmetrics:GetDataQualityMetrics lookoutmetrics:GetFeedback lookoutmetrics:GetSampleData lookoutmetrics:ListAlerts lookoutmetrics:ListAnomalyDetectors lookoutmetrics:ListAnomalyGroupRelatedMetrics lookoutmetrics:ListAnomalyGroupSummaries

服務前綴	動作
	lookoutmetrics:ListAnomalyGroupTimeSeries lookoutmetrics:ListMetricSets lookoutmetrics:PutFeedback lookoutmetrics:UpdateAlert lookoutmetrics:UpdateAnomalyDetector lookoutmetrics:UpdateMetricSet

服務前綴	動作
m2	m2:CancelBatchJobExecution
	m2:CreateApplication
	m2:CreateDataSetExportTask
	m2:CreateDataSetImportTask
	m2:CreateDeployment
	m2:CreateEnvironment
	m2:DeleteApplication
	m2:DeleteApplicationFromEnvironment
	m2:DeleteEnvironment
	m2:GetApplication
	m2:GetApplicationVersion
	m2:GetBatchJobExecution
	m2:GetDataSetDetails
	m2:GetDataSetExportTask
	m2:GetDataSetImportTask
	m2:GetDeployment
	m2:GetEnvironment
	m2:GetSignedBluinsightsUrl
	m2:ListApplicationVersions
	m2:ListApplications
	m2:ListBatchJobDefinitions

服務前綴	動作
	m2:ListBatchJobExecutions
	m2:ListBatchJobRestartPoints
	m2:ListDataSetExportHistory
	m2:ListDataSetImportHistory
	m2:ListDataSets
	m2:ListDeployments
	m2:ListEngineVersions
	m2:ListEnvironments
	m2:StartApplication
	m2:StartBatchJob
	m2:StopApplication
	m2:UpdateApplication
	m2:UpdateEnvironment

服務前綴	動作
managedblockchain	managedblockchain:CreateAccessor managedblockchain:CreateMember managedblockchain:CreateNetwork managedblockchain:CreateNode managedblockchain:CreateProposal managedblockchain:DeleteAccessor managedblockchain:DeleteMember managedblockchain:DeleteNode managedblockchain:GetAccessor managedblockchain:GetMember managedblockchain:GetNetwork managedblockchain:GetNode managedblockchain:GetProposal managedblockchain:InvokeRpcPolygonMainnet managedblockchain:InvokeRpcPolygonMumbaiTestnet managedblockchain:ListAccessors managedblockchain:ListInvitations managedblockchain:ListMembers managedblockchain:ListNetworks managedblockchain:ListNodes managedblockchain:ListProposalVotes

服務前綴	動作
	managedblockchain:ListProposals managedblockchain:RejectInvitation managedblockchain:UpdateMember managedblockchain:UpdateNode managedblockchain:VoteOnProposal

服務前綴	動作
mediaconnect	mediaconnect:AddBridgeOutputs
	mediaconnect:AddBridgeSources
	mediaconnect:AddFlowMediaStreams
	mediaconnect:AddFlowOutputs
	mediaconnect:AddFlowSources
	mediaconnect:AddFlowVpcInterfaces
	mediaconnect:CreateBridge
	mediaconnect:CreateFlow
	mediaconnect:CreateGateway
	mediaconnect:DeleteBridge
	mediaconnect:DeleteFlow
	mediaconnect:DeleteGateway
	mediaconnect：DeleteRouterInput
	mediaconnect：DeleteRouterNetworkInterface
	mediaconnect：DeleteRouterOutput
	mediaconnect:DeregisterGatewayInstance
	mediaconnect:DescribeBridge
	mediaconnect:DescribeFlow
	mediaconnect:DescribeFlowSourceMetadata
	mediaconnect:DescribeFlowSourceThumbnail
	mediaconnect:DescribeGateway

服務前綴	動作
	mediaconnect:DescribeGatewayInstance
	mediaconnect:DescribeOffering
	mediaconnect:DescribeReservation
	mediaconnect：GetRouterInput
	mediaconnect：GetRouterInputSourceMetadata
	mediaconnect：GetRouterInputThumbnail
	mediaconnect：GetRouterNetworkInterface
	mediaconnect：GetRouterOutput
	mediaconnect:GrantFlowEntitlements
	mediaconnect:ListBridges
	mediaconnect:ListEntitlements
	mediaconnect:ListFlows
	mediaconnect:ListGatewayInstances
	mediaconnect:ListGateways
	mediaconnect:ListOfferings
	mediaconnect:ListReservations
	mediaconnect：ListRouterInputs
	mediaconnect：ListRouterNetworkInterfaces
	mediaconnect：ListRouterOutputs
	mediaconnect:PurchaseOffering
	mediaconnect:RemoveBridgeOutput

服務前綴	動作
	mediaconnect:RemoveBridgeSource
	mediaconnect:RemoveFlowMediaStream
	mediaconnect:RemoveFlowOutput
	mediaconnect:RemoveFlowSource
	mediaconnect:RemoveFlowVpcInterface
	mediaconnect：RestartRouterInput
	mediaconnect：RestartRouterOutput
	mediaconnect:RevokeFlowEntitlement
	mediaconnect:StartFlow
	mediaconnect：StartRouterInput
	mediaconnect：StartRouterOutput
	mediaconnect:StopFlow
	mediaconnect：StopRouterInput
	mediaconnect：StopRouterOutput
	mediaconnect：TakeRouterInput
	mediaconnect:UpdateBridge
	mediaconnect:UpdateBridgeOutput
	mediaconnect:UpdateBridgeSource
	mediaconnect:UpdateBridgeState
	mediaconnect:UpdateFlow
	mediaconnect:UpdateFlowEntitlement

服務前綴	動作
	mediaconnect:UpdateFlowMediaStream mediaconnect:UpdateGatewayInstance

服務前綴	動作
mediaconvert	mediaconvert:AssociateCertificate mediaconvert:CancelJob mediaconvert:CreateJob mediaconvert:CreateJobTemplate mediaconvert:CreatePreset mediaconvert:CreateQueue mediaconvert：CreateResourceShare mediaconvert:DeleteJobTemplate mediaconvert:DeletePolicy mediaconvert:DeletePreset mediaconvert:DeleteQueue mediaconvert:DescribeEndpoints mediaconvert:DisassociateCertificate mediaconvert:GetJob mediaconvert:GetJobTemplate mediaconvert:GetPolicy mediaconvert:GetPreset mediaconvert:GetQueue mediaconvert:ListJobTemplates mediaconvert:ListJobs mediaconvert:ListPresets

服務前綴	動作
	mediaconvert:ListQueues mediaconvert:ListVersions mediaconvert:Probe mediaconvert:PutPolicy mediaconvert:SearchJobs mediaconvert:UpdateJobTemplate mediaconvert:UpdatePreset mediaconvert:UpdateQueue

服務前綴	動作
medialive	medialive:AcceptInputDeviceTransfer
	medialive:BatchDelete
	medialive:BatchStart
	medialive:BatchStop
	medialive:BatchUpdateSchedule
	medialive:CancelInputDeviceTransfer
	medialive:ClaimDevice
	medialive:CreateChannel
	medialive:CreateChannelPlacementGroup
	medialive:CreateCloudWatchAlarmTemplate
	medialive:CreateCloudWatchAlarmTemplateGroup
	medialive:CreateCluster
	medialive:CreateEventBridgeRuleTemplate
	medialive:CreateEventBridgeRuleTemplateGroup
	medialive:CreateInput
	medialive:CreateInputSecurityGroup
	medialive:CreateMultiplex
	medialive:CreateMultiplexProgram
	medialive:CreateNetwork
	medialive:CreateNode
	medialive:CreateNodeRegistrationScript

服務前綴	動作
	medialive:CreatePartnerInput
	medialive:CreateSdiSource
	medialive:CreateSignalMap
	medialive:DeleteChannel
	medialive:DeleteChannelPlacementGroup
	medialive:DeleteCloudWatchAlarmTemplate
	medialive:DeleteCloudWatchAlarmTemplateGroup
	medialive:DeleteCluster
	medialive:DeleteEventBridgeRuleTemplate
	medialive:DeleteEventBridgeRuleTemplateGroup
	medialive:DeleteInput
	medialive:DeleteInputSecurityGroup
	medialive:DeleteMultiplex
	medialive:DeleteMultiplexProgram
	medialive:DeleteNetwork
	medialive:DeleteNode
	medialive:DeleteReservation
	medialive:DeleteSchedule
	medialive:DeleteSdiSource
	medialive:DeleteSignalMap
	medialive:DescribeAccountConfiguration

服務前綴	動作
	medialive:DescribeChannel
	medialive:DescribeChannelPlacementGroup
	medialive:DescribeCluster
	medialive:DescribeInput
	medialive:DescribeInputDevice
	medialive:DescribeInputDeviceThumbnail
	medialive:DescribeInputSecurityGroup
	medialive:DescribeMultiplex
	medialive:DescribeMultiplexProgram
	medialive:DescribeNetwork
	medialive:DescribeNode
	medialive:DescribeOffering
	medialive:DescribeReservation
	medialive:DescribeSchedule
	medialive:DescribeSdiSource
	medialive:DescribeThumbnails
	medialive:GetCloudWatchAlarmTemplate
	medialive:GetCloudWatchAlarmTemplateGroup
	medialive:GetEventBridgeRuleTemplate
	medialive:GetEventBridgeRuleTemplateGroup
	medialive:GetSignalMap

服務前綴	動作
	medialive：ListAlerts
	medialive:ListChannelPlacementGroups
	medialive:ListChannels
	medialive:ListCloudWatchAlarmTemplateGroups
	medialive:ListCloudWatchAlarmTemplates
	medialive：ListClusterAlerts
	medialive:ListClusters
	medialive:ListEventBridgeRuleTemplateGroups
	medialive:ListEventBridgeRuleTemplates
	medialive:ListInputDeviceTransfers
	medialive:ListInputDevices
	medialive:ListInputSecurityGroups
	medialive:ListInputs
	medialive：ListMultiplexAlerts
	medialive:ListMultiplexPrograms
	medialive:ListMultiplexes
	medialive:ListNetworks
	medialive:ListNodes
	medialive:ListOfferings
	medialive:ListReservations
	medialive:ListSdiSources

服務前綴	動作
	medialive:ListSignalMaps
	medialive:ListVersions
	medialive:PurchaseOffering
	medialive:RebootInputDevice
	medialive:RejectInputDeviceTransfer
	medialive:RestartChannelPipelines
	medialive:StartChannel
	medialive:StartDeleteMonitorDeployment
	medialive:StartInputDevice
	medialive:StartInputDeviceMaintenanceWindow
	medialive:StartMonitorDeployment
	medialive:StartMultiplex
	medialive:StartUpdateSignalMap
	medialive:StopChannel
	medialive:StopInputDevice
	medialive:StopMultiplex
	medialive:TransferInputDevice
	medialive:UpdateAccountConfiguration
	medialive:UpdateChannel
	medialive:UpdateChannelClass
	medialive:UpdateChannelPlacementGroup

服務前綴	動作
	medialive:UpdateCloudWatchAlarmTemplate
	medialive:UpdateCloudWatchAlarmTemplateGroup
	medialive:UpdateCluster
	medialive:UpdateEventBridgeRuleTemplate
	medialive:UpdateEventBridgeRuleTemplateGroup
	medialive:UpdateInput
	medialive:UpdateInputDevice
	medialive:UpdateInputSecurityGroup
	medialive:UpdateMultiplex
	medialive:UpdateMultiplexProgram
	medialive:UpdateNetwork
	medialive:UpdateNode
	medialive:UpdateNodeState
	medialive:UpdateReservation
	medialive:UpdateSdiSource

服務前綴	動作
mediatailor	mediatailor:ConfigureLogsForPlaybackConfiguration mediatailor:CreateChannel mediatailor:CreateLiveSource mediatailor:CreatePrefetchSchedule mediatailor:CreateProgram mediatailor:CreateSourceLocation mediatailor:CreateVodSource mediatailor:DeleteChannel mediatailor:DeleteChannelPolicy mediatailor:DeleteLiveSource mediatailor:DeletePlaybackConfiguration mediatailor:DeletePrefetchSchedule mediatailor:DeleteProgram mediatailor:DeleteSourceLocation mediatailor:DeleteVodSource mediatailor:DescribeChannel mediatailor:DescribeLiveSource mediatailor:DescribeProgram mediatailor:DescribeSourceLocation mediatailor:DescribeVodSource mediatailor:GetChannelPolicy

服務前綴	動作
	mediatailor:GetChannelSchedule mediatailor:GetPlaybackConfiguration mediatailor:GetPrefetchSchedule mediatailor:ListAlerts mediatailor:ListChannels mediatailor:ListLiveSources mediatailor:ListPlaybackConfigurations mediatailor:ListPrefetchSchedules mediatailor:ListSourceLocations mediatailor:ListVodSources mediatailor:PutChannelPolicy mediatailor:PutPlaybackConfiguration mediatailor:StartChannel mediatailor:StopChannel mediatailor:UpdateChannel mediatailor:UpdateLiveSource mediatailor:UpdateProgram mediatailor:UpdateSourceLocation mediatailor:UpdateVodSource

服務前綴	動作
memorydb	memorydb:BatchUpdateCluster
	memorydb:CopySnapshot
	memorydb:CreateAcl
	memorydb:CreateCluster
	memorydb:CreateMultiRegionCluster
	memorydb:CreateParameterGroup
	memorydb:CreateSnapshot
	memorydb:CreateSubnetGroup
	memorydb:CreateUser
	memorydb:DeleteAcl
	memorydb:DeleteCluster
	memorydb:DeleteMultiRegionCluster
	memorydb:DeleteParameterGroup
	memorydb:DeleteSnapshot
	memorydb:DeleteSubnetGroup
	memorydb:DeleteUser
	memorydb:DescribeAcls
	memorydb:DescribeClusters
	memorydb:DescribeEngineVersions
	memorydb:DescribeEvents
	memorydb:DescribeMultiRegionClusters

服務前綴	動作
	memorydb：DescribeMultiRegionParameterGroups
	memorydb：DescribeMultiRegionParameters
	memorydb:DescribeParameterGroups
	memorydb:DescribeParameters
	memorydb:DescribeReservedNodes
	memorydb:DescribeReservedNodesOfferings
	memorydb:DescribeServiceUpdates
	memorydb:DescribeSnapshots
	memorydb:DescribeSubnetGroups
	memorydb:DescribeUsers
	memorydb:FailoverShard
	memorydb:ListAllowedMultiRegionClusterUpdates
	memorydb:ListAllowedNodeTypeUpdates
	memorydb:PurchaseReservedNodesOffering
	memorydb:ResetParameterGroup
	memorydb:UpdateAcl
	memorydb:UpdateCluster
	memorydb:UpdateMultiRegionCluster
	memorydb:UpdateParameterGroup
	memorydb:UpdateSubnetGroup
	memorydb:UpdateUser

服務前綴	動作
mgh	mgh:AssociateCreatedArtifact mgh:AssociateDiscoveredResource mgh:AssociateSourceResource mgh:CreateHomeRegionControl mgh:CreateProgressUpdateStream mgh:DeleteHomeRegionControl mgh:DeleteProgressUpdateStream mgh:DescribeApplicationState mgh:DescribeHomeRegionControls mgh:DescribeMigrationTask mgh:DisassociateCreatedArtifact mgh:DisassociateDiscoveredResource mgh:DisassociateSourceResource mgh:GetHomeRegion mgh:ImportMigrationTask mgh:ListApplicationStates mgh:ListCreatedArtifacts mgh:ListDiscoveredResources mgh:ListMigrationTaskUpdates mgh:ListMigrationTasks mgh:ListProgressUpdateStreams

服務前綴	動作
	mgh:ListSourceResources mgh:NotifyApplicationState mgh:NotifyMigrationTaskState mgh:PutResourceAttributes

服務前綴	動作
mgn	mgn:ArchiveApplication mgn:ArchiveWave mgn:AssociateApplications mgn:AssociateSourceServers mgn:ChangeServerLifeCycleState mgn:CreateApplication mgn:CreateConnector mgn:CreateLaunchConfigurationTemplate mgn：CreateNetworkMigrationDefinition mgn:CreateReplicationConfigurationTemplate mgn:CreateWave mgn:DeleteApplication mgn:DeleteConnector mgn:DeleteJob mgn:DeleteLaunchConfigurationTemplate mgn：DeleteNetworkMigrationDefinition mgn:DeleteReplicationConfigurationTemplate mgn:DeleteSourceServer mgn:DeleteVcenterClient mgn:DeleteWave mgn:DescribeJobLogItems

服務前綴	動作
	mgn:DescribeJobs
	mgn:DescribeLaunchConfigurationTemplates
	mgn:DescribeReplicationConfigurationTemplates
	mgn:DescribeVcenterClients
	mgn:DisassociateApplications
	mgn:DisassociateSourceServers
	mgn:DisconnectFromService
	mgn:FinalizeCutover
	mgn：GetNetworkMigrationDefinition
	mgn：GetNetworkMigrationMapperSegmentConstruct
	mgn:GetReplicationConfiguration
	mgn:InitializeService
	mgn:ListConnectors
	mgn:ListExportErrors
	mgn:ListExports
	mgn:ListImportErrors
	mgn：ListImportFileEnrichments
	mgn:ListImports
	mgn:ListManagedAccounts
	mgn：ListNetworkMigrationAnalyses
	mgn：ListNetworkMigrationAnalysisResults

服務前綴	動作
	mgn：ListNetworkMigrationCodeGenerationSegments
	mgn：ListNetworkMigrationCodeGenerations
	mgn：ListNetworkMigrationDefinitions
	mgn：ListNetworkMigrationDeployedStacks
	mgn：ListNetworkMigrationDeployments
	mgn：ListNetworkMigrationExecutions
	mgn：ListNetworkMigrationMapperSegmentConstructs
	mgn：ListNetworkMigrationMapperSegments
	mgn：ListNetworkMigrationMappingUpdates
	mgn：ListNetworkMigrationMappings
	mgn:ListSourceServerActions
	mgn:ListTemplateActions
	mgn:MarkAsArchived
	mgn:PauseReplication
	mgn:PutSourceServerAction
	mgn:PutTemplateAction
	mgn:RemoveSourceServerAction
	mgn:RemoveTemplateAction
	mgn:ResumeReplication
	mgn:RetryDataReplication
	mgn:StartCutover

服務前綴	動作
	mgn:StartExport
	mgn:StartImport
	mgn：StartImportFileEnrichment
	mgn：StartNetworkMigrationCodeGeneration
	mgn：StartNetworkMigrationMapping
	mgn：StartNetworkMigrationMappingUpdate
	mgn:StartReplication
	mgn:StartTest
	mgn:StopReplication
	mgn:TerminateTargetInstances
	mgn:UnarchiveApplication
	mgn:UnarchiveWave
	mgn:UpdateApplication
	mgn:UpdateConnector
	mgn:UpdateLaunchConfigurationTemplate
	mgn：UpdateNetworkMigrationDefinition
	mgn：UpdateNetworkMigrationMapperSegment
	mgn:UpdateReplicationConfiguration
	mgn:UpdateReplicationConfigurationTemplate
	mgn:UpdateSourceServer
	mgn:UpdateSourceServerReplicationType

服務前綴	動作
	mgn:UpdateWave

服務前綴	動作
migrationhub-strategy	migrationhub-strategy:GetAntiPattern migrationhub-strategy:GetApplicationComponentDetails migrationhub-strategy:GetApplicationComponentStrategies migrationhub-strategy:GetAssessment migrationhub-strategy:GetImportFileTask migrationhub-strategy:GetLatestAssessmentId migrationhub-strategy:GetMessage migrationhub-strategy:GetPortfolioPreferences migrationhub-strategy:GetPortfolioSummary migrationhub-strategy:GetRecommendationReportDetails migrationhub-strategy:GetServerDetails migrationhub-strategy:GetServerStrategies migrationhub-strategy:ListAnalyzableServers migrationhub-strategy:ListAntiPatterns migrationhub-strategy:ListApplicationComponents migrationhub-strategy:ListCollectors migrationhub-strategy:ListImportFileTask migrationhub-strategy:ListJarArtifacts migrationhub-strategy:ListServers migrationhub-strategy:PutLogData migrationhub-strategy:PutMetricData

服務前綴	動作
	migrationhub-strategy:PutPortfolioPreferences migrationhub-strategy:RegisterCollector migrationhub-strategy:SendMessage migrationhub-strategy:StartAssessment migrationhub-strategy:StartImportFileTask migrationhub-strategy:StartRecommendationReportGeneration migrationhub-strategy:StopAssessment migrationhub-strategy:UpdateApplicationComponentConfig migrationhub-strategy:UpdateCollectorConfiguration migrationhub-strategy:UpdateServerConfig

服務前綴	動作
mobiletargeting	mobiletargeting:CreateApp
	mobiletargeting:CreateCampaign
	mobiletargeting:CreateEmailTemplate
	mobiletargeting:CreateExportJob
	mobiletargeting:CreateImportJob
	mobiletargeting:CreateInAppTemplate
	mobiletargeting:CreateJourney
	mobiletargeting:CreatePushTemplate
	mobiletargeting:CreateRecommenderConfiguration
	mobiletargeting:CreateSegment
	mobiletargeting:CreateSmsTemplate
	mobiletargeting:CreateVoiceTemplate
	mobiletargeting:DeleteAdmChannel
	mobiletargeting:DeleteApnsChannel
	mobiletargeting:DeleteApnsSandboxChannel
	mobiletargeting:DeleteApnsVoipChannel
	mobiletargeting:DeleteApnsVoipSandboxChannel
	mobiletargeting:DeleteApp
	mobiletargeting:DeleteBaiduChannel
	mobiletargeting:DeleteCampaign
	mobiletargeting:DeleteEmailChannel

服務前綴	動作
	mobiletargeting:DeleteEmailTemplate
	mobiletargeting:DeleteEndpoint
	mobiletargeting:DeleteEventStream
	mobiletargeting:DeleteGcmChannel
	mobiletargeting:DeleteInAppTemplate
	mobiletargeting:DeleteJourney
	mobiletargeting:DeletePushTemplate
	mobiletargeting:DeleteRecommenderConfiguration
	mobiletargeting:DeleteSegment
	mobiletargeting:DeleteSmsChannel
	mobiletargeting:DeleteSmsTemplate
	mobiletargeting:DeleteUserEndpoints
	mobiletargeting:DeleteVoiceChannel
	mobiletargeting:DeleteVoiceTemplate
	mobiletargeting:GetAdmChannel
	mobiletargeting:GetApnsChannel
	mobiletargeting:GetApnsSandboxChannel
	mobiletargeting:GetApnsVoipChannel
	mobiletargeting:GetApnsVoipSandboxChannel
	mobiletargeting:GetApp
	mobiletargeting:GetApplicationDateRangeKpi

服務前綴	動作
	mobiletargeting:GetApplicationSettings
	mobiletargeting:GetApps
	mobiletargeting:GetBaiduChannel
	mobiletargeting:GetCampaign
	mobiletargeting:GetCampaignActivities
	mobiletargeting:GetCampaignDateRangeKpi
	mobiletargeting:GetCampaignVersion
	mobiletargeting:GetCampaignVersions
	mobiletargeting:GetCampaigns
	mobiletargeting:GetChannels
	mobiletargeting:GetEmailChannel
	mobiletargeting:GetEmailTemplate
	mobiletargeting:GetEndpoint
	mobiletargeting:GetEventStream
	mobiletargeting:GetExportJob
	mobiletargeting:GetExportJobs
	mobiletargeting:GetGcmChannel
	mobiletargeting:GetImportJob
	mobiletargeting:GetImportJobs
	mobiletargeting:GetInAppMessages
	mobiletargeting:GetInAppTemplate

服務前綴	動作
	mobiletargeting:GetJourney
	mobiletargeting:GetJourneyDateRangeKpi
	mobiletargeting:GetJourneyExecutionActivityMetrics
	mobiletargeting:GetJourneyExecutionMetrics
	mobiletargeting:GetJourneyRunExecutionActivityMetrics
	mobiletargeting:GetJourneyRunExecutionMetrics
	mobiletargeting:GetJourneyRuns
	mobiletargeting:GetPushTemplate
	mobiletargeting:GetRecommenderConfiguration
	mobiletargeting:GetRecommenderConfigurations
	mobiletargeting:GetSegment
	mobiletargeting:GetSegmentExportJobs
	mobiletargeting:GetSegmentImportJobs
	mobiletargeting:GetSegmentVersion
	mobiletargeting:GetSegmentVersions
	mobiletargeting:GetSegments
	mobiletargeting:GetSmsChannel
	mobiletargeting:GetSmsTemplate
	mobiletargeting:GetUserEndpoints
	mobiletargeting:GetVoiceChannel
	mobiletargeting:GetVoiceTemplate

服務前綴	動作
	mobiletargeting:ListJourneys
	mobiletargeting:ListTemplateVersions
	mobiletargeting:ListTemplates
	mobiletargeting:PhoneNumberValidate
	mobiletargeting:PutEventStream
	mobiletargeting:RemoveAttributes
	mobiletargeting:UpdateAdmChannel
	mobiletargeting:UpdateApnsChannel
	mobiletargeting:UpdateApnsSandboxChannel
	mobiletargeting:UpdateApnsVoipChannel
	mobiletargeting:UpdateApnsVoipSandboxChannel
	mobiletargeting:UpdateApplicationSettings
	mobiletargeting:UpdateBaiduChannel
	mobiletargeting:UpdateCampaign
	mobiletargeting:UpdateEmailChannel
	mobiletargeting:UpdateEmailTemplate
	mobiletargeting:UpdateEndpoint
	mobiletargeting:UpdateEndpointsBatch
	mobiletargeting:UpdateGcmChannel
	mobiletargeting:UpdateInAppTemplate
	mobiletargeting:UpdateJourney

服務前綴	動作
	mobiletargeting:UpdateJourneyState
	mobiletargeting:UpdatePushTemplate
	mobiletargeting:UpdateRecommenderConfiguration
	mobiletargeting:UpdateSegment
	mobiletargeting:UpdateSmsChannel
	mobiletargeting:UpdateSmsTemplate
	mobiletargeting:UpdateTemplateActiveVersion
	mobiletargeting:UpdateVoiceChannel
	mobiletargeting:UpdateVoiceTemplate
	mobiletargeting:VerifyOTPMessage

服務前綴	動作
mq	mq:CreateBroker mq:CreateConfiguration mq:CreateUser mq:DeleteBroker mq:DeleteConfiguration mq:DeleteUser mq:DescribeBroker mq:DescribeBrokerEngineTypes mq:DescribeBrokerInstanceOptions mq:DescribeConfiguration mq:DescribeConfigurationRevision mq：DescribeSharedResources mq:DescribeUser mq:ListBrokers mq:ListConfigurationRevisions mq:ListConfigurations mq:ListUsers mq:Promote mq:RebootBroker mq:UpdateBroker mq:UpdateConfiguration

服務前綴	動作
	mq:UpdateUser

服務前綴	動作
networkmanager	networkmanager:AcceptAttachment networkmanager:AssociateConnectPeer networkmanager:AssociateCustomerGateway networkmanager:AssociateLink networkmanager:AssociateTransitGatewayConnectPeer networkmanager:CreateConnectAttachment networkmanager:CreateConnectPeer networkmanager:CreateConnection networkmanager:CreateCoreNetwork networkmanager:CreateDevice networkmanager:CreateDirectConnectGatewayAttachment networkmanager:CreateGlobalNetwork networkmanager:CreateLink networkmanager:CreateSite networkmanager:CreateSiteToSiteVpnAttachment networkmanager:CreateTransitGatewayPeering networkmanager:CreateTransitGatewayRouteTableAttachment networkmanager:CreateVpcAttachment networkmanager:DeleteAttachment networkmanager:DeleteConnectPeer networkmanager:DeleteConnection

服務前綴	動作
	networkmanager:DeleteCoreNetwork
	networkmanager:DeleteCoreNetworkPolicyVersion
	networkmanager:DeleteDevice
	networkmanager:DeleteGlobalNetwork
	networkmanager:DeleteLink
	networkmanager:DeletePeering
	networkmanager:DeleteResourcePolicy
	networkmanager:DeleteSite
	networkmanager:DeregisterTransitGateway
	networkmanager:DescribeGlobalNetworks
	networkmanager:DisassociateConnectPeer
	networkmanager:DisassociateCustomerGateway
	networkmanager:DisassociateLink
	networkmanager:DisassociateTransitGatewayConnectPeer
	networkmanager:ExecuteCoreNetworkChangeSet
	networkmanager:GetConnectAttachment
	networkmanager:GetConnectPeer
	networkmanager:GetConnectPeerAssociations
	networkmanager:GetConnections
	networkmanager:GetCoreNetwork
	networkmanager:GetCoreNetworkChangeEvents

服務前綴	動作
	networkmanager:GetCoreNetworkChangeSet
	networkmanager:GetCoreNetworkPolicy
	networkmanager:GetCustomerGatewayAssociations
	networkmanager:GetDevices
	networkmanager:GetLinkAssociations
	networkmanager:GetLinks
	networkmanager:GetNetworkResourceCounts
	networkmanager:GetNetworkResourceRelationships
	networkmanager:GetNetworkResources
	networkmanager:GetNetworkRoutes
	networkmanager:GetNetworkTelemetry
	networkmanager:GetResourcePolicy
	networkmanager:GetRouteAnalysis
	networkmanager:GetSiteToSiteVpnAttachment
	networkmanager:GetSites
	networkmanager:GetTransitGatewayConnectPeerAssociations
	networkmanager:GetTransitGatewayPeering
	networkmanager:GetTransitGatewayRegistrations
	networkmanager:GetTransitGatewayRouteTableAttachment
	networkmanager:GetVpcAttachment
	networkmanager：ListAttachmentRoutingPolicyAssociations

服務前綴	動作
	networkmanager:ListAttachments
	networkmanager:ListConnectPeers
	networkmanager:ListCoreNetworkPolicyVersions
	networkmanager：ListCoreNetworkPrefixListAssociations
	networkmanager：ListCoreNetworkRoutingInformation
	networkmanager:ListCoreNetworks
	networkmanager:ListOrganizationServiceAccessStatus
	networkmanager:ListPeerings
	networkmanager：PutAttachmentRoutingPolicyLabel
	networkmanager:PutCoreNetworkPolicy
	networkmanager:PutResourcePolicy
	networkmanager:RegisterTransitGateway
	networkmanager:RejectAttachment
	networkmanager：RemoveAttachmentRoutingPolicyLabel
	networkmanager:RestoreCoreNetworkPolicyVersion
	networkmanager:StartOrganizationServiceAccessUpdate
	networkmanager:StartRouteAnalysis
	networkmanager:UpdateConnection
	networkmanager:UpdateCoreNetwork
	networkmanager:UpdateDevice
	networkmanager:UpdateDirectConnectGatewayAttachment

服務前綴	動作
	networkmanager:UpdateGlobalNetwork networkmanager:UpdateLink networkmanager:UpdateNetworkResourceMetadata networkmanager:UpdateSite networkmanager:UpdateVpcAttachment

服務前綴	動作
nimble	nimble:AcceptEulas nimble:CreateLaunchProfile nimble:CreateStreamingImage nimble:CreateStreamingSession nimble:CreateStreamingSessionStream nimble:CreateStudio nimble:CreateStudioComponent nimble:DeleteLaunchProfile nimble:DeleteLaunchProfileMember nimble:DeleteStreamingImage nimble:DeleteStreamingSession nimble:DeleteStudio nimble:DeleteStudioComponent nimble:DeleteStudioMember nimble:GetEula nimble:GetLaunchProfileDetails nimble:GetStreamingImage nimble:GetStreamingSession nimble:GetStreamingSessionBackup nimble:GetStreamingSessionStream nimble:GetStudio

服務前綴	動作
	nimble:GetStudioComponent nimble:GetStudioMember nimble:ListEulas nimble:ListLaunchProfileMembers nimble:ListLaunchProfiles nimble:ListStreamingImages nimble:ListStreamingSessionBackups nimble:ListStreamingSessions nimble:ListStudioComponents nimble:ListStudioMembers nimble:ListStudios nimble:PutLaunchProfileMembers nimble:PutStudioMembers nimble:StartStreamingSession nimble:StartStudioSSOConfigurationRepair nimble:StopStreamingSession nimble:UpdateLaunchProfile nimble:UpdateLaunchProfileMember nimble:UpdateStreamingImage nimble:UpdateStudio nimble:UpdateStudioComponent

服務前綴	動作
omics	omics:AbortMultipartReadSetUpload
	omics:AcceptShare
	omics:BatchDeleteReadSet
	omics:CancelAnnotationImportJob
	omics:CancelRun
	omics：CancelRunBatch
	omics:CancelVariantImportJob
	omics:CompleteMultipartReadSetUpload
	omics:CreateAnnotationStore
	omics:CreateAnnotationStoreVersion
	omics：CreateConfiguration
	omics:CreateMultipartReadSetUpload
	omics:CreateReferenceStore
	omics:CreateRunGroup
	omics:CreateSequenceStore
	omics:CreateShare
	omics:CreateVariantStore
	omics:CreateWorkflow
	omics:CreateWorkflowVersion
	omics:DeleteAnnotationStore
	omics:DeleteAnnotationStoreVersions

服務前綴	動作
	omics：DeleteBatch
	omics：DeleteConfiguration
	omics:DeleteReference
	omics:DeleteReferenceStore
	omics:DeleteRun
	omics：DeleteRunBatch
	omics:DeleteRunGroup
	omics:DeleteSequenceStore
	omics:DeleteShare
	omics:DeleteVariantStore
	omics:DeleteWorkflow
	omics:DeleteWorkflowVersion
	omics:GetAnnotationImportJob
	omics:GetAnnotationStore
	omics:GetAnnotationStoreVersion
	omics：GetBatch
	omics：GetConfiguration
	omics:GetReadSet
	omics:GetReadSetActivationJob
	omics:GetReadSetExportJob
	omics:GetReadSetImportJob

服務前綴	動作
	omics:GetReadSetMetadata
	omics:GetReference
	omics:GetReferenceImportJob
	omics:GetReferenceMetadata
	omics:GetReferenceStore
	omics:GetRun
	omics:GetRunGroup
	omics:GetRunTask
	omics:GetSequenceStore
	omics:GetShare
	omics:GetVariantImportJob
	omics:GetVariantStore
	omics:GetWorkflow
	omics:GetWorkflowVersion
	omics:ListAnnotationImportJobs
	omics:ListAnnotationStoreVersions
	omics:ListAnnotationStores
	omics：ListBatch
	omics：ListConfigurations
	omics:ListMultipartReadSetUploads
	omics:ListReadSetActivationJobs

服務前綴	動作
	omics:ListReadSetExportJobs
	omics:ListReadSetImportJobs
	omics:ListReadSetUploadParts
	omics:ListReadSets
	omics:ListReferenceImportJobs
	omics:ListReferenceStores
	omics:ListReferences
	omics:ListRunGroups
	omics:ListRunTasks
	omics:ListRuns
	omics：ListRunsInBatch
	omics:ListSequenceStores
	omics:ListShares
	omics:ListVariantImportJobs
	omics:ListVariantStores
	omics:ListWorkflowVersions
	omics:ListWorkflows
	omics:StartAnnotationImportJob
	omics:StartReadSetActivationJob
	omics:StartReadSetExportJob
	omics:StartReadSetImportJob

服務前綴	動作
	omics:StartReferenceImportJob omics:StartRun omics：StartRunBatch omics:StartVariantImportJob omics:UpdateAnnotationStore omics:UpdateAnnotationStoreVersion omics:UpdateRunGroup omics:UpdateVariantStore omics:UpdateWorkflow omics:UpdateWorkflowVersion omics:UploadReadSetPart

服務前綴	動作
opsworks	opsworks:AssignInstance opsworks:AssignVolume opsworks:AssociateElasticIp opsworks:AttachElasticLoadBalancer opsworks:CloneStack opsworks:CreateApp opsworks:CreateDeployment opsworks:CreateInstance opsworks:CreateLayer opsworks:CreateStack opsworks:CreateUserProfile opsworks:DeleteApp opsworks:DeleteInstance opsworks:DeleteLayer opsworks:DeleteStack opsworks:DeleteUserProfile opsworks:DeregisterEcsCluster opsworks:DeregisterElasticIp opsworks:DeregisterInstance opsworks:DeregisterRdsDbInstance opsworks:DeregisterVolume

服務前綴	動作
	opsworks:DescribeAgentVersions opsworks:DescribeApps opsworks:DescribeCommands opsworks:DescribeDeployments opsworks:DescribeEcsClusters opsworks:DescribeElasticIps opsworks:DescribeElasticLoadBalancers opsworks:DescribeInstances opsworks:DescribeLayers opsworks:DescribeLoadBasedAutoScaling opsworks:DescribeMyUserProfile opsworks:DescribeOperatingSystems opsworks:DescribePermissions opsworks:DescribeRaidArrays opsworks:DescribeRdsDbInstances opsworks:DescribeServiceErrors opsworks:DescribeStackProvisioningParameters opsworks:DescribeStackSummary opsworks:DescribeStacks opsworks:DescribeTimeBasedAutoScaling opsworks:DescribeUserProfiles

服務前綴	動作
	opsworks:DescribeVolumes opsworks:DetachElasticLoadBalancer opsworks:DisassociateElasticIp opsworks:GetHostnameSuggestion opsworks:GrantAccess opsworks:RebootInstance opsworks:RegisterEcsCluster opsworks:RegisterElasticIp opsworks:RegisterInstance opsworks:RegisterRdsDbInstance opsworks:RegisterVolume opsworks:SetLoadBasedAutoScaling opsworks:SetPermission opsworks:SetTimeBasedAutoScaling opsworks:StartInstance opsworks:StartStack opsworks:StopInstance opsworks:StopStack opsworks:UnassignInstance opsworks:UnassignVolume opsworks:UpdateApp

服務前綴	動作
	opsworks:UpdateElasticIp opsworks:UpdateInstance opsworks:UpdateLayer opsworks:UpdateMyUserProfile opsworks:UpdateRdsDbInstance opsworks:UpdateStack opsworks:UpdateUserProfile opsworks:UpdateVolume

服務前綴	動作
opsworks-cm	opsworks-cm:AssociateNode
	opsworks-cm:CreateBackup
	opsworks-cm:CreateServer
	opsworks-cm:DeleteBackup
	opsworks-cm:DeleteServer
	opsworks-cm:DescribeAccountAttributes
	opsworks-cm:DescribeBackups
	opsworks-cm:DescribeEvents
	opsworks-cm:DescribeNodeAssociationStatus
	opsworks-cm:DescribeServers
	opsworks-cm:DisassociateNode
	opsworks-cm:ExportServerEngineAttribute
	opsworks-cm:RestoreServer
	opsworks-cm:StartMaintenance
	opsworks-cm:UpdateServer
	opsworks-cm:UpdateServerEngineAttributes

服務前綴	動作
組織	organizations:AcceptHandshake organizations:AttachPolicy organizations:CancelHandshake organizations:CloseAccount organizations:CreateAccount organizations:CreateGovCloudAccount organizations:CreateOrganization organizations:CreateOrganizationalUnit organizations:CreatePolicy organizations:DeclineHandshake organizations:DeleteOrganization organizations:DeleteOrganizationalUnit organizations:DeletePolicy organizations:DeleteResourcePolicy organizations:DeregisterDelegatedAdministrator organizations:DescribeAccount organizations:DescribeCreateAccountStatus organizations:DescribeEffectivePolicy organizations:DescribeHandshake organizations：DescribeOrganization organizations:DescribeOrganizationalUnit

服務前綴	動作
	organizations:DescribePolicy organizations:DescribeResourcePolicy organizations：DescribeResponsibilityTransfer organizations:DetachPolicy organizations:DisableAWSServiceAccess organizations:DisablePolicyType organizations:EnableAWSServiceAccess organizations:EnableAllFeatures organizations:EnablePolicyType organizations:InviteAccountToOrganization organizations:LeaveOrganization organizations：ListAWSServiceAccessForOrganization organizations:ListAccounts organizations:ListAccountsForParent organizations：ListAccountsWithInvalidEffectivePolicy organizations:ListChildren organizations:ListCreateAccountStatus organizations:ListDelegatedAdministrators organizations:ListDelegatedServicesForAccount organizations:ListHandshakesForAccount organizations:ListHandshakesForOrganization

服務前綴	動作
	organizations：ListInboundResponsibilityTransfers organizations:ListOrganizationalUnitsForParent organizations：ListOutboundResponsibilityTransfers organizations:ListParents organizations:ListPolicies organizations:ListPoliciesForTarget organizations:ListRoots organizations:ListTargetsForPolicy organizations:MoveAccount organizations:PutResourcePolicy organizations:RegisterDelegatedAdministrator organizations:RemoveAccountFromOrganization organizations：TerminateResponsibilityTransfer organizations:UpdateOrganizationalUnit organizations:UpdatePolicy organizations：UpdateResponsibilityTransfer

服務前綴	動作
outposts	outposts:CancelCapacityTask
	outposts:CancelOrder
	outposts:CreateOrder
	outposts:CreateOutpost
	outposts:CreatePrivateConnectivityConfig
	outposts：CreateQuote
	outposts：CreateRenewal
	outposts:CreateSite
	outposts:DeleteOutpost
	outposts：DeleteQuote
	outposts:DeleteSite
	outposts:GetCapacityTask
	outposts:GetCatalogItem
	outposts:GetConnection
	outposts:GetOrder
	outposts:GetOutpost
	outposts:GetOutpostBillingInformation
	outposts:GetOutpostInstanceTypes
	outposts:GetOutpostSupportedInstanceTypes
	outposts:GetPrivateConnectivityConfig
	outposts：GetQuote

服務前綴	動作
	outposts：GetRenewalPricing
	outposts:GetSite
	outposts:GetSiteAddress
	outposts:ListAssetInstances
	outposts:ListAssets
	outposts:ListBlockingInstancesForCapacityTask
	outposts:ListCapacityTasks
	outposts:ListCatalogItems
	outposts：ListOrderableInstanceTypes
	outposts:ListOrders
	outposts:ListOutposts
	outposts：ListQuotes
	outposts:ListSites
	outposts:StartCapacityTask
	outposts:StartConnection
	outposts:UpdateOutpost
	outposts：UpdateQuote
	outposts:UpdateSite
	outposts:UpdateSiteAddress
	outposts:UpdateSiteRackPhysicalProperties

服務前綴	動作
pi	pi:CreatePerformanceAnalysisReport pi:DeletePerformanceAnalysisReport pi:DescribeDimensionKeys pi:GetDimensionKeyDetails pi:GetPerformanceAnalysisReport pi:GetResourceMetadata pi:GetResourceMetrics pi:ListAvailableResourceDimensions pi:ListAvailableResourceMetrics pi:ListPerformanceAnalysisReports
pipes	pipes:CreatePipe pipes:DeletePipe pipes:DescribePipe pipes:ListPipes pipes:StartPipe pipes:StopPipe pipes:UpdatePipe

服務前綴	動作
polly	polly:DeleteLexicon polly:DescribeVoices polly:GetLexicon polly:GetSpeechSynthesisTask polly:ListLexicons polly:ListSpeechSynthesisTasks polly:PutLexicon polly:StartSpeechSynthesisTask polly:SynthesizeSpeech

服務前綴	動作
profile	profile:AddProfileKey profile:BatchGetCalculatedAttributeForProfile profile:BatchGetProfile profile:CreateCalculatedAttributeDefinition profile:CreateDomain profile:CreateEventStream profile:CreateProfile profile：CreateRecommender profile：CreateRecommenderFilter profile：CreateRecommenderSchema profile:CreateSegmentDefinition profile:CreateSegmentEstimate profile:CreateSegmentSnapshot profile:CreateUploadJob profile:DeleteCalculatedAttributeDefinition profile:DeleteDomain profile：DeleteDomainObjectType profile:DeleteEventStream profile:DeleteIntegration profile:DeleteProfile profile:DeleteProfileKey

服務前綴	動作
	profile:DeleteProfileObject
	profile:DeleteProfileObjectType
	profile：DeleteRecommender
	profile：DeleteRecommenderFilter
	profile：DeleteRecommenderSchema
	profile:DeleteSegmentDefinition
	profile:DeleteWorkflow
	profile:DetectProfileObjectType
	profile:GetAutoMergingPreview
	profile:GetCalculatedAttributeDefinition
	profile:GetCalculatedAttributeForProfile
	profile:GetDomain
	profile：GetDomainObjectType
	profile:GetEventStream
	profile:GetIdentityResolutionJob
	profile:GetIntegration
	profile:GetMatches
	profile：GetObjectTypeAttributeStatistics
	profile:GetProfileObjectType
	profile:GetProfileObjectTypeTemplate
	profile：GetProfileRecommendations

服務前綴	動作
	profile：GetRecommender
	profile：GetRecommenderFilter
	profile：GetRecommenderSchema
	profile:GetSegmentDefinition
	profile:GetSegmentEstimate
	profile:GetSegmentMembership
	profile:GetSegmentSnapshot
	profile:GetSimilarProfiles
	profile:GetUploadJob
	profile:GetUploadJobPath
	profile:GetWorkflow
	profile:GetWorkflowSteps
	profile:ListAccountIntegrations
	profile:ListCalculatedAttributeDefinitions
	profile:ListCalculatedAttributesForProfile
	profile:ListDomainLayouts
	profile：ListDomainObjectTypes
	profile:ListDomains
	profile:ListEventStreams
	profile:ListIdentityResolutionJobs
	profile:ListIntegrations

服務前綴	動作
	profile：ListObjectTypeAttributeValues
	profile:ListObjectTypeAttributes
	profile:ListProfileAttributeValues
	profile:ListProfileObjectTypeTemplates
	profile:ListProfileObjectTypes
	profile:ListProfileObjects
	profile：ListRecommenderFilters
	profile：ListRecommenderRecipes
	profile：ListRecommenderSchemas
	profile：ListRecommenders
	profile:ListRuleBasedMatches
	profile:ListSegmentDefinitions
	profile:ListUploadJobs
	profile:ListWorkflows
	profile:MergeProfiles
	profile：PutDomainObjectType
	profile:PutIntegration
	profile:PutProfileObject
	profile:PutProfileObjectType
	profile:SearchProfiles
	profile：StartRecommender

服務前綴	動作
	profile:StartUploadJob profile：StopRecommender profile:StopUploadJob profile:UpdateCalculatedAttributeDefinition profile:UpdateDomain profile:UpdateProfile profile：UpdateRecommender

服務前綴	動作
qldb	qldb:CancelJournalKinesisStream qldb:CreateLedger qldb:DeleteLedger qldb:DescribeJournalKinesisStream qldb:DescribeJournalS3Export qldb:DescribeLedger qldb:ExportJournalToS3 qldb:GetBlock qldb:GetDigest qldb:GetRevision qldb:ListJournalKinesisStreamsForLedger qldb:ListJournalS3Exports qldb:ListJournalS3ExportsForLedger qldb:ListLedgers qldb:StreamJournalToKinesis qldb:UpdateLedger qldb:UpdateLedgerPermissionsMode

服務前綴	動作
ram	ram:AcceptResourceShareInvitation
	ram:AssociateResourceShare
	ram:AssociateResourceSharePermission
	ram:CreatePermission
	ram:CreatePermissionVersion
	ram:CreateResourceShare
	ram:DeletePermission
	ram:DeletePermissionVersion
	ram:DeleteResourceShare
	ram:DisassociateResourceShare
	ram:DisassociateResourceSharePermission
	ram:EnableSharingWithAwsOrganization
	ram:GetPermission
	ram:GetResourcePolicies
	ram:GetResourceShareAssociations
	ram:GetResourceShareInvitations
	ram:GetResourceShares
	ram:ListPendingInvitationResources
	ram:ListPermissionAssociations
	ram:ListPermissionVersions
	ram:ListPermissions

服務前綴	動作
	ram:ListPrincipals ram:ListReplacePermissionAssociationsWork ram:ListResourceSharePermissions ram:ListResourceTypes ram:ListResources ram：ListSourceAssociations ram:PromotePermissionCreatedFromPolicy ram:PromoteResourceShareCreatedFromPolicy ram:RejectResourceShareInvitation ram:ReplacePermissionAssociations ram:SetDefaultPermissionVersion ram:UpdateResourceShare
rbin	rbin:CreateRule rbin:DeleteRule rbin:GetRule rbin:ListRules rbin:LockRule rbin:UnlockRule rbin:UpdateRule

服務前綴	動作
rds	rds:AddRoleToDBCluster
	rds:AddRoleToDBInstance
	rds:AddSourceIdentifierToSubscription
	rds:ApplyPendingMaintenanceAction
	rds:AuthorizeDBSecurityGroupIngress
	rds:BacktrackDBCluster
	rds:CancelExportTask
	rds:CopyDBClusterParameterGroup
	rds:CopyDBClusterSnapshot
	rds:CopyDBParameterGroup
	rds:CopyDBSnapshot
	rds:CopyOptionGroup
	rds:CreateCustomDBEngineVersion
	rds:CreateDBClusterParameterGroup
	rds:CreateDBParameterGroup
	rds:CreateDBProxy
	rds:CreateDBProxyEndpoint
	rds:CreateDBSecurityGroup
	rds:CreateDBSubnetGroup
	rds:CreateEventSubscription
	rds:CreateGlobalCluster

服務前綴	動作
	rds:CreateOptionGroup
	rds:DeleteBlueGreenDeployment
	rds:DeleteDBClusterAutomatedBackup
	rds:DeleteDBClusterParameterGroup
	rds:DeleteDBClusterSnapshot
	rds:DeleteDBInstanceAutomatedBackup
	rds:DeleteDBParameterGroup
	rds:DeleteDBProxy
	rds:DeleteDBProxyEndpoint
	rds:DeleteDBSecurityGroup
	rds:DeleteDBSnapshot
	rds:DeleteDBSubnetGroup
	rds:DeleteEventSubscription
	rds:DeleteGlobalCluster
	rds:DeleteOptionGroup
	rds:DeregisterDBProxyTargets
	rds:DescribeAccountAttributes
	rds:DescribeBlueGreenDeployments
	rds:DescribeCertificates
	rds:DescribeDBClusterAutomatedBackups
	rds:DescribeDBClusterBacktracks

服務前綴	動作
	rds:DescribeDBClusterEndpoints
	rds:DescribeDBClusterParameterGroups
	rds:DescribeDBClusterParameters
	rds:DescribeDBClusterSnapshotAttributes
	rds:DescribeDBClusterSnapshots
	rds:DescribeDBClusters
	rds:DescribeDBEngineVersions
	rds:DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rds:DescribeDBLogFiles
	rds:DescribeDBMajorEngineVersions
	rds:DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rds:DescribeDBProxyEndpoints
	rds:DescribeDBProxyTargetGroups
	rds:DescribeDBProxyTargets
	rds:DescribeDBRecommendations
	rds:DescribeDBSecurityGroups
	rds:DescribeDBSnapshotAttributes
	rds:DescribeDBSnapshotTenantDatabases

服務前綴	動作
	rds:DescribeDBSnapshots
	rds:DescribeDBSubnetGroups
	rds:DescribeEngineDefaultClusterParameters
	rds:DescribeEngineDefaultParameters
	rds:DescribeEventCategories
	rds:DescribeEventSubscriptions
	rds:DescribeEvents
	rds:DescribeExportTasks
	rds:DescribeGlobalClusters
	rds:DescribeIntegrations
	rds:DescribeOptionGroupOptions
	rds:DescribeOptionGroups
	rds:DescribeOrderableDBInstanceOptions
	rds:DescribePendingMaintenanceActions
	rds:DescribeReservedDBInstances
	rds:DescribeReservedDBInstancesOfferings
	rds:DescribeSourceRegions
	rds:DescribeTenantDatabases
	rds:DescribeValidDBInstanceModifications
	rds:DownloadCompleteDBLogFile
	rds:DownloadDBLogFilePortion

服務前綴	動作
	rds:FailoverDBCluster
	rds:FailoverGlobalCluster
	rds:ModifyActivityStream
	rds:ModifyCertificates
	rds:ModifyCurrentDBClusterCapacity
	rds:ModifyDBClusterEndpoint
	rds:ModifyDBClusterParameterGroup
	rds:ModifyDBClusterSnapshotAttribute
	rds:ModifyDBParameterGroup
	rds:ModifyDBProxy
	rds:ModifyDBProxyEndpoint
	rds:ModifyDBProxyTargetGroup
	rds:ModifyDBRecommendation
	rds:ModifyDBSnapshot
	rds:ModifyDBSnapshotAttribute
	rds:ModifyDBSubnetGroup
	rds:ModifyEventSubscription
	rds:ModifyGlobalCluster
	rds:ModifyOptionGroup
	rds:ModifyTenantDatabase
	rds:PurchaseReservedDBInstancesOffering

服務前綴	動作
	rds:RebootDBCluster
	rds:RegisterDBProxyTargets
	rds:RemoveFromGlobalCluster
	rds:RemoveRoleFromDBCluster
	rds:RemoveRoleFromDBInstance
	rds:RemoveSourceIdentifierFromSubscription
	rds:ResetDBClusterParameterGroup
	rds:ResetDBParameterGroup
	rds:RestoreDBClusterFromS3
	rds:RestoreDBClusterFromSnapshot
	rds:RestoreDBClusterToPointInTime
	rds:RestoreDBInstanceFromDBSnapshot
	rds:RestoreDBInstanceFromS3
	rds:RestoreDBInstanceToPointInTime
	rds:RevokeDBSecurityGroupIngress
	rds:StartActivityStream
	rds:StartDBCluster
	rds:StartDBInstance
	rds:StartDBInstanceAutomatedBackupsReplication
	rds:StartExportTask
	rds:StopActivityStream

服務前綴	動作
	rds:StopDBCluster rds:StopDBInstance rds:StopDBInstanceAutomatedBackupsReplication rds:SwitchoverBlueGreenDeployment rds:SwitchoverGlobalCluster rds:SwitchoverReadReplica

服務前綴	動作
redshift	redshift:AcceptReservedNodeExchange redshift:AddPartner redshift:AssociateDataShareConsumer redshift:AuthorizeClusterSecurityGroupIngress redshift:AuthorizeDataShare redshift:AuthorizeEndpointAccess redshift:AuthorizeSnapshotAccess redshift:BatchDeleteClusterSnapshots redshift:BatchModifyClusterSnapshots redshift:CancelResize redshift:CopyClusterSnapshot redshift:CreateAuthenticationProfile redshift:CreateCluster redshift:CreateClusterParameterGroup redshift:CreateClusterSecurityGroup redshift:CreateClusterSnapshot redshift:CreateClusterSubnetGroup redshift:CreateCustomDomainAssociation redshift:CreateEndpointAccess redshift:CreateEventSubscription redshift:CreateHsmClientCertificate

服務前綴	動作
	redshift:CreateHsmConfiguration redshift:CreateIntegration redshift:CreateRedshiftIdcApplication redshift:CreateScheduledAction redshift:CreateSnapshotCopyGrant redshift:CreateSnapshotSchedule redshift:CreateUsageLimit redshift:DeauthorizeDataShare redshift:DeleteAuthenticationProfile redshift:DeleteCluster redshift:DeleteClusterParameterGroup redshift:DeleteClusterSecurityGroup redshift:DeleteClusterSnapshot redshift:DeleteClusterSubnetGroup redshift:DeleteCustomDomainAssociation redshift:DeleteEndpointAccess redshift:DeleteEventSubscription redshift:DeleteHsmClientCertificate redshift:DeleteHsmConfiguration redshift:DeletePartner redshift:DeleteRedshiftIdcApplication

服務前綴	動作
	redshift:DeleteResourcePolicy redshift:DeleteScheduledAction redshift:DeleteSnapshotCopyGrant redshift:DeleteSnapshotSchedule redshift:DeleteUsageLimit redshift:DeregisterNamespace redshift:DescribeAccountAttributes redshift:DescribeAuthenticationProfiles redshift:DescribeClusterDbRevisions redshift:DescribeClusterParameterGroups redshift:DescribeClusterParameters redshift:DescribeClusterSecurityGroups redshift:DescribeClusterSnapshots redshift:DescribeClusterSubnetGroups redshift:DescribeClusterTracks redshift:DescribeClusterVersions redshift:DescribeClusters redshift:DescribeCustomDomainAssociations redshift:DescribeDataShares redshift:DescribeDataSharesForConsumer redshift:DescribeDataSharesForProducer

服務前綴	動作
	redshift:DescribeDefaultClusterParameters redshift:DescribeEndpointAccess redshift:DescribeEndpointAuthorization redshift:DescribeEventCategories redshift:DescribeEventSubscriptions redshift:DescribeEvents redshift:DescribeHsmClientCertificates redshift:DescribeHsmConfigurations redshift:DescribeInboundIntegrations redshift:DescribeIntegrations redshift:DescribeLoggingStatus redshift:DescribeNodeConfigurationOptions redshift:DescribeOrderableClusterOptions redshift:DescribePartners redshift:DescribeRedshiftIdcApplications redshift:DescribeReservedNodeExchangeStatus redshift:DescribeReservedNodeOfferings redshift:DescribeReservedNodes redshift:DescribeResize redshift:DescribeScheduledActions redshift:DescribeSnapshotCopyGrants

服務前綴	動作
	redshift:DescribeSnapshotSchedules redshift:DescribeStorage redshift:DescribeTableRestoreStatus redshift:DescribeUsageLimits redshift:DisableLogging redshift:DisableSnapshotCopy redshift:DisassociateDataShareConsumer redshift:EnableLogging redshift:EnableSnapshotCopy redshift:FailoverPrimaryCompute redshift:GetClusterCredentials redshift:GetClusterCredentialsWithIAM redshift：GetIdentityCenterAuthToken redshift:GetReservedNodeExchangeConfigurationOptions redshift:GetReservedNodeExchangeOfferings redshift:GetResourcePolicy redshift:ListRecommendations redshift:ModifyAquaConfiguration redshift:ModifyAuthenticationProfile redshift:ModifyCluster redshift:ModifyClusterDbRevision

服務前綴	動作
	redshift:ModifyClusterIamRoles redshift:ModifyClusterMaintenance redshift:ModifyClusterParameterGroup redshift:ModifyClusterSnapshot redshift:ModifyClusterSnapshotSchedule redshift:ModifyClusterSubnetGroup redshift:ModifyCustomDomainAssociation redshift:ModifyEndpointAccess redshift:ModifyEventSubscription redshift:ModifyRedshiftIdcApplication redshift:ModifyScheduledAction redshift:ModifySnapshotCopyRetentionPeriod redshift:ModifySnapshotSchedule redshift:ModifyUsageLimit redshift:PauseCluster redshift:PurchaseReservedNodeOffering redshift:PutResourcePolicy redshift:RebootCluster redshift:RegisterNamespace redshift:RejectDataShare redshift:ResetClusterParameterGroup

服務前綴	動作
	redshift:ResizeCluster redshift:RestoreFromClusterSnapshot redshift:RestoreTableFromClusterSnapshot redshift:ResumeCluster redshift:RevokeClusterSecurityGroupIngress redshift:RevokeEndpointAccess redshift:RevokeSnapshotAccess redshift:RotateEncryptionKey redshift:UpdatePartnerStatus
redshift-data	redshift-data:BatchExecuteStatement redshift-data:CancelStatement redshift-data:DescribeStatement redshift-data:DescribeTable redshift-data:ExecuteStatement redshift-data:GetStatementResult redshift-data:ListDatabases redshift-data:ListSchemas redshift-data：ListSessions redshift-data:ListStatements redshift-data:ListTables

服務前綴	動作
refactor-spaces	refactor-spaces:CreateApplication refactor-spaces:CreateEnvironment refactor-spaces:CreateRoute refactor-spaces:CreateService refactor-spaces:DeleteApplication refactor-spaces:DeleteEnvironment refactor-spaces:DeleteResourcePolicy refactor-spaces:DeleteRoute refactor-spaces:DeleteService refactor-spaces:GetApplication refactor-spaces:GetEnvironment refactor-spaces:GetResourcePolicy refactor-spaces:GetRoute refactor-spaces:GetService refactor-spaces:ListApplications refactor-spaces:ListEnvironmentVpcs refactor-spaces:ListEnvironments refactor-spaces:ListRoutes refactor-spaces:ListServices refactor-spaces:PutResourcePolicy refactor-spaces:UpdateRoute

服務前綴	動作
rekognition	rekognition:AssociateFaces
	rekognition:CompareFaces
	rekognition:CopyProjectVersion
	rekognition:CreateCollection
	rekognition:CreateDataset
	rekognition:CreateFaceLivenessSession
	rekognition:CreateProject
	rekognition:CreateProjectVersion
	rekognition:CreateStreamProcessor
	rekognition:CreateUser
	rekognition:DeleteCollection
	rekognition:DeleteDataset
	rekognition:DeleteFaces
	rekognition:DeleteProject
	rekognition:DeleteProjectPolicy
	rekognition:DeleteProjectVersion
	rekognition:DeleteStreamProcessor
	rekognition:DeleteUser
	rekognition:DescribeCollection
	rekognition:DescribeDataset
	rekognition:DescribeProjectVersions

服務前綴	動作
	rekognition:DescribeProjects
	rekognition:DescribeStreamProcessor
	rekognition:DetectCustomLabels
	rekognition:DetectFaces
	rekognition:DetectLabels
	rekognition:DetectModerationLabels
	rekognition:DetectProtectiveEquipment
	rekognition:DetectText
	rekognition:DisassociateFaces
	rekognition:DistributeDatasetEntries
	rekognition:GetCelebrityInfo
	rekognition:GetCelebrityRecognition
	rekognition:GetContentModeration
	rekognition:GetFaceDetection
	rekognition:GetFaceLivenessSessionResults
	rekognition:GetFaceSearch
	rekognition:GetLabelDetection
	rekognition:GetMediaAnalysisJob
	rekognition:GetPersonTracking
	rekognition:GetSegmentDetection
	rekognition:GetTextDetection

服務前綴	動作
	rekognition:IndexFaces
	rekognition:ListCollections
	rekognition:ListDatasetEntries
	rekognition:ListDatasetLabels
	rekognition:ListFaces
	rekognition:ListMediaAnalysisJobs
	rekognition:ListProjectPolicies
	rekognition:ListStreamProcessors
	rekognition:ListUsers
	rekognition:PutProjectPolicy
	rekognition:RecognizeCelebrities
	rekognition:SearchFaces
	rekognition:SearchFacesByImage
	rekognition:SearchUsers
	rekognition:SearchUsersByImage
	rekognition:StartCelebrityRecognition
	rekognition:StartContentModeration
	rekognition:StartFaceDetection
	rekognition:StartFaceLivenessSession
	rekognition:StartFaceSearch
	rekognition:StartLabelDetection

服務前綴	動作
	rekognition:StartMediaAnalysisJob
	rekognition:StartPersonTracking
	rekognition:StartProjectVersion
	rekognition:StartSegmentDetection
	rekognition:StartStreamProcessor
	rekognition:StartTextDetection
	rekognition:StopProjectVersion
	rekognition:StopStreamProcessor
	rekognition:UpdateDatasetEntries
	rekognition:UpdateStreamProcessor

服務前綴	動作
resiliencehub	resiliencehub:AcceptResourceGroupingRecommendations resiliencehub:AddDraftAppVersionResourceMappings resiliencehub:BatchUpdateRecommendationStatus resiliencehub:CreateApp resiliencehub:CreateAppVersionAppComponent resiliencehub:CreateAppVersionResource resiliencehub:CreateRecommendationTemplate resiliencehub:CreateResiliencyPolicy resiliencehub:DeleteApp resiliencehub:DeleteAppAssessment resiliencehub:DeleteAppInputSource resiliencehub:DeleteAppVersionAppComponent resiliencehub:DeleteAppVersionResource resiliencehub:DeleteRecommendationTemplate resiliencehub:DeleteResiliencyPolicy resiliencehub:DescribeApp resiliencehub:DescribeAppAssessment resiliencehub:DescribeAppVersion resiliencehub:DescribeAppVersionAppComponent resiliencehub:DescribeAppVersionResource resiliencehub:DescribeAppVersionResourcesResolutionStatus

服務前綴	動作
	resiliencehub:DescribeAppVersionTemplate resiliencehub:DescribeDraftAppVersionResourcesImportStatus resiliencehub:DescribeMetricsExport resiliencehub:DescribeResiliencyPolicy resiliencehub:DescribeResourceGroupingRecommendationTask resiliencehub:ImportResourcesToDraftAppVersion resiliencehub:ListAlarmRecommendations resiliencehub:ListAppAssessmentComplianceDrifts resiliencehub:ListAppAssessmentResourceDrifts resiliencehub:ListAppAssessments resiliencehub:ListAppComponentCompliances resiliencehub:ListAppComponentRecommendations resiliencehub:ListAppInputSources resiliencehub:ListAppVersionAppComponents resiliencehub:ListAppVersionResourceMappings resiliencehub:ListAppVersionResources resiliencehub:ListAppVersions resiliencehub:ListApps resiliencehub:ListMetrics resiliencehub:ListRecommendationTemplates resiliencehub:ListResiliencyPolicies

服務前綴	動作
	resiliencehub:ListResourceGroupingRecommendations resiliencehub:ListSopRecommendations resiliencehub:ListSuggestedResiliencyPolicies resiliencehub:ListTestRecommendations resiliencehub:ListUnsupportedAppVersionResources resiliencehub:PublishAppVersion resiliencehub:PutDraftAppVersionTemplate resiliencehub:RejectResourceGroupingRecommendations resiliencehub:RemoveDraftAppVersionResourceMappings resiliencehub:ResolveAppVersionResources resiliencehub:StartAppAssessment resiliencehub:StartResourceGroupingRecommendationTask resiliencehub:UpdateApp resiliencehub:UpdateAppVersion resiliencehub:UpdateAppVersionAppComponent resiliencehub:UpdateAppVersionResource resiliencehub:UpdateResiliencyPolicy

服務前綴	動作
resource-explorer-2	resource-explorer-2:AssociateDefaultView
	resource-explorer-2:BatchGetView
	resource-explorer-2:CreateIndex
	resource-explorer-2：CreateResourceExplorerSetup
	resource-explorer-2:CreateView
	resource-explorer-2:DeleteIndex
	resource-explorer-2：DeleteResourceExplorerSetup
	resource-explorer-2:DeleteView
	resource-explorer-2:DisassociateDefaultView
	resource-explorer-2:GetAccountLevelServiceConfiguration
	resource-explorer-2:GetDefaultView
	resource-explorer-2:GetIndex
	resource-explorer-2:GetManagedView
	resource-explorer-2：GetResourceExplorerSetup
	resource-explorer-2：GetServiceIndex
	resource-explorer-2：GetServiceView
	resource-explorer-2:ListIndexes
	resource-explorer-2:ListIndexesForMembers
	resource-explorer-2:ListManagedViews
	resource-explorer-2：ListServiceIndexes
	resource-explorer-2：ListServiceViews

服務前綴	動作
	resource-explorer-2：ListStreamingAccessForServices resource-explorer-2:ListSupportedResourceTypes resource-explorer-2:ListViews resource-explorer-2:Search resource-explorer-2:UpdateIndexType resource-explorer-2:UpdateView

服務前綴	動作
resource-groups	resource-groups:CancelTagSyncTask resource-groups:GetAccountSettings resource-groups:GetGroup resource-groups:GetGroupConfiguration resource-groups:GetGroupQuery resource-groups:GetTagSyncTask resource-groups:GroupResources resource-groups:ListGroupResources resource-groups:ListGroupingStatuses resource-groups:ListGroups resource-groups:ListTagSyncTasks resource-groups:PutGroupConfiguration resource-groups:SearchResources resource-groups:StartTagSyncTask resource-groups:UngroupResources resource-groups:UpdateAccountSettings resource-groups:UpdateGroup resource-groups:UpdateGroupQuery

服務前綴	動作
robomaker	robomaker:BatchDeleteWorlds
	robomaker:BatchDescribeSimulationJob
	robomaker:CancelDeploymentJob
	robomaker:CancelSimulationJob
	robomaker:CancelSimulationJobBatch
	robomaker:CancelWorldExportJob
	robomaker:CancelWorldGenerationJob
	robomaker:CreateDeploymentJob
	robomaker:CreateFleet
	robomaker:CreateRobot
	robomaker:CreateRobotApplication
	robomaker:CreateRobotApplicationVersion
	robomaker:CreateSimulationApplication
	robomaker:CreateSimulationApplicationVersion
	robomaker:CreateSimulationJob
	robomaker:CreateWorldExportJob
	robomaker:CreateWorldGenerationJob
	robomaker:CreateWorldTemplate
	robomaker:DeleteFleet
	robomaker:DeleteRobot
	robomaker:DeleteRobotApplication

服務前綴	動作
	robomaker:DeleteSimulationApplication
	robomaker:DeleteWorldTemplate
	robomaker:DeregisterRobot
	robomaker:DescribeDeploymentJob
	robomaker:DescribeFleet
	robomaker:DescribeRobot
	robomaker:DescribeRobotApplication
	robomaker:DescribeSimulationApplication
	robomaker:DescribeSimulationJob
	robomaker:DescribeSimulationJobBatch
	robomaker:DescribeWorld
	robomaker:DescribeWorldExportJob
	robomaker:DescribeWorldGenerationJob
	robomaker:DescribeWorldTemplate
	robomaker:GetWorldTemplateBody
	robomaker:ListDeploymentJobs
	robomaker:ListFleets
	robomaker:ListRobotApplications
	robomaker:ListRobots
	robomaker:ListSimulationApplications
	robomaker:ListSimulationJobBatches

服務前綴	動作
	robomaker:ListSimulationJobs
	robomaker:ListWorldExportJobs
	robomaker:ListWorldGenerationJobs
	robomaker:ListWorldTemplates
	robomaker:ListWorlds
	robomaker:RegisterRobot
	robomaker:RestartSimulationJob
	robomaker:StartSimulationJobBatch
	robomaker:SyncDeploymentJob
	robomaker:UpdateRobotApplication
	robomaker:UpdateSimulationApplication
	robomaker:UpdateWorldTemplate

服務前綴	動作
rolesanywhere	rolesanywhere:CreateProfile rolesanywhere:CreateTrustAnchor rolesanywhere:DeleteAttributeMapping rolesanywhere:DeleteCrl rolesanywhere:DeleteProfile rolesanywhere:DeleteTrustAnchor rolesanywhere:DisableCrl rolesanywhere:DisableProfile rolesanywhere:DisableTrustAnchor rolesanywhere:EnableCrl rolesanywhere:EnableProfile rolesanywhere:EnableTrustAnchor rolesanywhere:GetCrl rolesanywhere:GetProfile rolesanywhere:GetSubject rolesanywhere:GetTrustAnchor rolesanywhere:ImportCrl rolesanywhere:ListCrls rolesanywhere:ListProfiles rolesanywhere:ListSubjects rolesanywhere:ListTrustAnchors

服務前綴	動作
	rolesanywhere:PutAttributeMapping rolesanywhere:PutNotificationSettings rolesanywhere:ResetNotificationSettings rolesanywhere:UpdateCrl rolesanywhere:UpdateProfile rolesanywhere:UpdateTrustAnchor

服務前綴	動作
route53	route53:ActivateKeySigningKey
	route53:AssociateVPCWithHostedZone
	route53:ChangeCidrCollection
	route53:ChangeResourceRecordSets
	route53:CreateCidrCollection
	route53:CreateHealthCheck
	route53:CreateHostedZone
	route53:CreateKeySigningKey
	route53:CreateQueryLoggingConfig
	route53:CreateReusableDelegationSet
	route53:CreateTrafficPolicy
	route53:CreateTrafficPolicyInstance
	route53:CreateTrafficPolicyVersion
	route53:CreateVPCAssociationAuthorization
	route53:DeactivateKeySigningKey
	route53:DeleteCidrCollection
	route53:DeleteHealthCheck
	route53:DeleteHostedZone
	route53:DeleteKeySigningKey
	route53:DeleteQueryLoggingConfig
	route53:DeleteReusableDelegationSet

服務前綴	動作
	route53:DeleteTrafficPolicy
	route53:DeleteTrafficPolicyInstance
	route53:DeleteVPCAssociationAuthorization
	route53:DisableHostedZoneDNSSEC
	route53:DisassociateVPCFromHostedZone
	route53:EnableHostedZoneDNSSEC
	route53:GetAccountLimit
	route53:GetChange
	route53:GetCheckerIpRanges
	route53:GetDNSSEC
	route53:GetGeoLocation
	route53:GetHealthCheck
	route53:GetHealthCheckCount
	route53:GetHealthCheckLastFailureReason
	route53:GetHealthCheckStatus
	route53:GetHostedZone
	route53:GetHostedZoneCount
	route53:GetHostedZoneLimit
	route53:GetQueryLoggingConfig
	route53:GetReusableDelegationSet
	route53:GetReusableDelegationSetLimit

服務前綴	動作
	route53:GetTrafficPolicy
	route53:GetTrafficPolicyInstance
	route53:GetTrafficPolicyInstanceCount
	route53:ListCidrBlocks
	route53:ListCidrCollections
	route53:ListCidrLocations
	route53:ListGeoLocations
	route53:ListHealthChecks
	route53:ListHostedZones
	route53:ListHostedZonesByName
	route53:ListHostedZonesByVPC
	route53:ListQueryLoggingConfigs
	route53:ListResourceRecordSets
	route53:ListReusableDelegationSets
	route53:ListTrafficPolicies
	route53:ListTrafficPolicyInstances
	route53:ListTrafficPolicyInstancesByHostedZone
	route53:ListTrafficPolicyInstancesByPolicy
	route53:ListTrafficPolicyVersions
	route53:ListVPCAssociationAuthorizations
	route53:TestDNSAnswer

服務前綴	動作
	route53:UpdateHealthCheck route53:UpdateHostedZoneComment route53:UpdateTrafficPolicyComment route53:UpdateTrafficPolicyInstance

服務前綴	動作
route53-recovery-control-config	route53-recovery-control-config:CreateCluster route53-recovery-control-config:CreateControlPanel route53-recovery-control-config:CreateRoutingControl route53-recovery-control-config:CreateSafetyRule route53-recovery-control-config:DeleteCluster route53-recovery-control-config:DeleteControlPanel route53-recovery-control-config:DeleteRoutingControl route53-recovery-control-config:DeleteSafetyRule route53-recovery-control-config:DescribeCluster route53-recovery-control-config:DescribeControlPanel route53-recovery-control-config:DescribeRoutingControl route53-recovery-control-config:DescribeSafetyRule route53-recovery-control-config:GetResourcePolicy route53-recovery-control-config:ListAssociatedRoute53HealthChecks route53-recovery-control-config:ListClusters route53-recovery-control-config:ListControlPanels route53-recovery-control-config:ListRoutingControls route53-recovery-control-config:ListSafetyRules route53-recovery-control-config:UpdateCluster route53-recovery-control-config:UpdateControlPanel

服務前綴	動作
	route53-recovery-control-config:UpdateRoutingControl route53-recovery-control-config:UpdateSafetyRule

服務前綴	動作
route53-recovery-readiness	route53-recovery-readiness:CreateCell route53-recovery-readiness:CreateCrossAccountAuthorization route53-recovery-readiness:CreateReadinessCheck route53-recovery-readiness:CreateRecoveryGroup route53-recovery-readiness:CreateResourceSet route53-recovery-readiness:DeleteCell route53-recovery-readiness:DeleteCrossAccountAuthorization route53-recovery-readiness:DeleteReadinessCheck route53-recovery-readiness:DeleteRecoveryGroup route53-recovery-readiness:DeleteResourceSet route53-recovery-readiness:GetArchitectureRecommendations route53-recovery-readiness:GetCell route53-recovery-readiness:GetCellReadinessSummary route53-recovery-readiness:GetReadinessCheck route53-recovery-readiness:GetReadinessCheckResourceStatus route53-recovery-readiness:GetReadinessCheckStatus route53-recovery-readiness:GetRecoveryGroup route53-recovery-readiness:GetRecoveryGroupReadinessSummary route53-recovery-readiness:GetResourceSet route53-recovery-readiness:ListCells route53-recovery-readiness:ListCrossAccountAuthorizations

服務前綴	動作
	route53-recovery-readiness:ListReadinessChecks
	route53-recovery-readiness:ListRecoveryGroups
	route53-recovery-readiness:ListResourceSets
	route53-recovery-readiness:ListRules
	route53-recovery-readiness:UpdateCell
	route53-recovery-readiness:UpdateReadinessCheck
	route53-recovery-readiness:UpdateRecoveryGroup
	route53-recovery-readiness:UpdateResourceSet

服務前綴	動作
route53resolver	route53resolver:AssociateFirewallRuleGroup route53resolver:AssociateResolverEndpointIpAddress route53resolver:AssociateResolverQueryLogConfig route53resolver:AssociateResolverRule route53resolver:CreateFirewallDomainList route53resolver:CreateFirewallRule route53resolver:CreateFirewallRuleGroup route53resolver:CreateResolverEndpoint route53resolver:CreateResolverQueryLogConfig route53resolver:CreateResolverRule route53resolver:DeleteFirewallDomainList route53resolver:DeleteFirewallRule route53resolver:DeleteFirewallRuleGroup route53resolver:DeleteOutpostResolver route53resolver:DeleteResolverEndpoint route53resolver:DeleteResolverQueryLogConfig route53resolver:DeleteResolverRule route53resolver:DisassociateFirewallRuleGroup route53resolver:DisassociateResolverEndpointIpAddress route53resolver:DisassociateResolverQueryLogConfig route53resolver:DisassociateResolverRule

服務前綴	動作
	route53resolver:GetFirewallConfig route53resolver:GetFirewallDomainList route53resolver:GetFirewallRuleGroup route53resolver:GetFirewallRuleGroupAssociation route53resolver:GetFirewallRuleGroupPolicy route53resolver:GetOutpostResolver route53resolver:GetResolverConfig route53resolver:GetResolverDnssecConfig route53resolver:GetResolverEndpoint route53resolver:GetResolverQueryLogConfig route53resolver:GetResolverQueryLogConfigAssociation route53resolver:GetResolverQueryLogConfigPolicy route53resolver:GetResolverRule route53resolver:GetResolverRuleAssociation route53resolver:GetResolverRulePolicy route53resolver:ImportFirewallDomains route53resolver:ListFirewallConfigs route53resolver:ListFirewallDomainLists route53resolver:ListFirewallDomains route53resolver:ListFirewallRuleGroupAssociations route53resolver:ListFirewallRuleGroups

服務前綴	動作
	route53resolver：ListFirewallRuleTypes route53resolver:ListFirewallRules route53resolver:ListOutpostResolvers route53resolver:ListResolverConfigs route53resolver:ListResolverDnssecConfigs route53resolver:ListResolverEndpointIpAddresses route53resolver:ListResolverEndpoints route53resolver:ListResolverQueryLogConfigAssociations route53resolver:ListResolverQueryLogConfigs route53resolver:ListResolverRuleAssociations route53resolver:ListResolverRules route53resolver:PutFirewallRuleGroupPolicy route53resolver:PutResolverQueryLogConfigPolicy route53resolver:UpdateFirewallConfig route53resolver:UpdateFirewallDomains route53resolver:UpdateFirewallRule route53resolver:UpdateFirewallRuleGroupAssociation route53resolver:UpdateOutpostResolver route53resolver:UpdateResolverConfig route53resolver:UpdateResolverDnssecConfig route53resolver:UpdateResolverEndpoint

服務前綴	動作
	route53resolver:UpdateResolverRule
rum	rum:BatchCreateRumMetricDefinitions rum:BatchDeleteRumMetricDefinitions rum:BatchGetRumMetricDefinitions rum:CreateAppMonitor rum:DeleteAppMonitor rum:DeleteResourcePolicy rum:DeleteRumMetricsDestination rum:GetAppMonitor rum:GetAppMonitorData rum:GetResourcePolicy rum:ListAppMonitors rum:ListRumMetricsDestinations rum:PutResourcePolicy rum:PutRumMetricsDestination rum:UpdateAppMonitor rum:UpdateRumMetricDefinition

服務前綴	動作
s3	s3:AssociateAccessGrantsIdentityCenter s3:CreateAccessGrant s3:CreateAccessGrantsInstance s3:CreateAccessGrantsLocation s3:CreateAccessPoint s3:CreateAccessPointForObjectLambda s3:CreateBucket s3:CreateBucketMetadataTableConfiguration s3:CreateJob s3:CreateMultiRegionAccessPoint s3:DeleteAccessGrant s3:DeleteAccessGrantsInstance s3:DeleteAccessGrantsInstanceResourcePolicy s3:DeleteAccessGrantsLocation s3:DeleteAccessPoint s3:DeleteAccessPointForObjectLambda s3:DeleteAccessPointPolicy s3:DeleteAccessPointPolicyForObjectLambda s3:DeleteBucket s3:DeleteBucketMetadataTableConfiguration s3:DeleteBucketPolicy

服務前綴	動作
	s3:DeleteBucketWebsite s3:DeleteMultiRegionAccessPoint s3:DeleteStorageLensConfiguration s3:DescribeJob s3:DescribeMultiRegionAccessPointOperation s3:DissociateAccessGrantsIdentityCenter s3:GetAccelerateConfiguration s3:GetAccessGrant s3:GetAccessGrantsInstance s3:GetAccessGrantsInstanceForPrefix s3:GetAccessGrantsInstanceResourcePolicy s3:GetAccessGrantsLocation s3:GetAccessPoint s3:GetAccessPointConfigurationForObjectLambda s3:GetAccessPointForObjectLambda s3:GetAccessPointPolicy s3:GetAccessPointPolicyForObjectLambda s3:GetAccessPointPolicyStatus s3:GetAccessPointPolicyStatusForObjectLambda s3:GetAccountPublicAccessBlock s3:GetAnalyticsConfiguration

服務前綴	動作
	s3：GetBucketAbac s3:GetBucketAcl s3:GetBucketCORS s3:GetBucketLocation s3:GetBucketLogging s3:GetBucketNotification s3:GetBucketObjectLockConfiguration s3:GetBucketOwnershipControls s3:GetBucketPolicy s3:GetBucketPolicyStatus s3:GetBucketPublicAccessBlock s3:GetBucketRequestPayment s3:GetBucketVersioning s3:GetBucketWebsite s3:GetDataAccess s3:GetEncryptionConfiguration s3:GetIntelligentTieringConfiguration s3:GetInventoryConfiguration s3:GetLifecycleConfiguration s3:GetMetricsConfiguration s3:GetMultiRegionAccessPoint

服務前綴	動作
	s3:GetMultiRegionAccessPointPolicy s3:GetMultiRegionAccessPointPolicyStatus s3:GetMultiRegionAccessPointRoutes s3:GetReplicationConfiguration s3:GetStorageLensConfiguration s3:GetStorageLensDashboard s3:ListAccessGrants s3:ListAccessGrantsInstances s3:ListAccessGrantsLocations s3:ListAccessPoints s3:ListAccessPointsForObjectLambda s3:ListAllMyBuckets s3:ListBucketMultipartUploads s3:ListCallerAccessGrants s3:ListJobs s3:ListMultiRegionAccessPoints s3:ListStorageLensConfigurations s3:PutAccelerateConfiguration s3:PutAccessGrantsInstanceResourcePolicy s3:PutAccessPointConfigurationForObjectLambda s3:PutAccessPointPolicy

服務前綴	動作
	s3:PutAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:PutAnalyticsConfiguration s3：PutBucketAbac s3:PutBucketAcl s3:PutBucketCORS s3:PutBucketLogging s3:PutBucketNotification s3:PutBucketObjectLockConfiguration s3:PutBucketOwnershipControls s3:PutBucketPolicy s3:PutBucketPublicAccessBlock s3:PutBucketRequestPayment s3:PutBucketVersioning s3:PutBucketWebsite s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutMetricsConfiguration s3:PutMultiRegionAccessPointPolicy

服務前綴	動作
	s3:PutReplicationConfiguration s3:PutStorageLensConfiguration s3:SubmitMultiRegionAccessPointRoutes s3:UpdateAccessGrantsLocation s3：UpdateBucketMetadataAnnotationTableConfiguration s3:UpdateBucketMetadataJournalTableConfiguration s3:UpdateJobPriority s3:UpdateJobStatus
s3-outposts	s3-outposts:CreateEndpoint s3-outposts:DeleteEndpoint s3-outposts:ListEndpoints s3-outposts:ListOutpostsWithS3 s3-outposts:ListSharedEndpoints

服務前綴	動作
sagemaker-geospatial	sagemaker-geospatial:DeleteEarthObservationJob sagemaker-geospatial:DeleteVectorEnrichmentJob sagemaker-geospatial:ExportEarthObservationJob sagemaker-geospatial:ExportVectorEnrichmentJob sagemaker-geospatial:GetEarthObservationJob sagemaker-geospatial:GetRasterDataCollection sagemaker-geospatial:GetTile sagemaker-geospatial:GetVectorEnrichmentJob sagemaker-geospatial:ListEarthObservationJobs sagemaker-geospatial:ListRasterDataCollections sagemaker-geospatial:ListVectorEnrichmentJobs sagemaker-geospatial:SearchRasterDataCollection sagemaker-geospatial:StartEarthObservationJob sagemaker-geospatial:StartVectorEnrichmentJob sagemaker-geospatial:StopEarthObservationJob sagemaker-geospatial:StopVectorEnrichmentJob

服務前綴	動作
savingsplans	savingsplans:CreateSavingsPlan savingsplans:DeleteQueuedSavingsPlan savingsplans:DescribeSavingsPlanRates savingsplans:DescribeSavingsPlans savingsplans:DescribeSavingsPlansOfferingRates savingsplans:DescribeSavingsPlansOfferings savingsplans:ReturnSavingsPlan

服務前綴	動作
schemas	schemas:CreateDiscoverer schemas:CreateRegistry schemas:CreateSchema schemas:DeleteDiscoverer schemas:DeleteRegistry schemas:DeleteResourcePolicy schemas:DeleteSchema schemas:DeleteSchemaVersion schemas:DescribeCodeBinding schemas:DescribeDiscoverer schemas:DescribeRegistry schemas:DescribeSchema schemas:ExportSchema schemas:GetCodeBindingSource schemas:GetDiscoveredSchema schemas:GetResourcePolicy schemas:ListDiscoverers schemas:ListRegistries schemas:ListSchemaVersions schemas:ListSchemas schemas:PutCodeBinding

服務前綴	動作
	schemas:PutResourcePolicy schemas:SearchSchemas schemas:StartDiscoverer schemas:StopDiscoverer schemas:UpdateDiscoverer schemas:UpdateRegistry schemas:UpdateSchema
sdb	sdb:CreateDomain sdb:DeleteDomain sdb:DomainMetadata sdb:ListDomains

服務前綴	動作
secretsmanager	secretsmanager:CancelRotateSecret secretsmanager:CreateSecret secretsmanager:DeleteResourcePolicy secretsmanager:DeleteSecret secretsmanager:DescribeSecret secretsmanager:GetRandomPassword secretsmanager:GetResourcePolicy secretsmanager:GetSecretValue secretsmanager:ListSecretVersionIds secretsmanager:ListSecrets secretsmanager:PutResourcePolicy secretsmanager:PutSecretValue secretsmanager:RemoveRegionsFromReplication secretsmanager:ReplicateSecretToRegions secretsmanager:RestoreSecret secretsmanager:RotateSecret secretsmanager:StopReplicationToReplica secretsmanager:UpdateSecret secretsmanager:ValidateResourcePolicy

服務前綴	動作
securityhub	securityhub:AcceptAdministratorInvitation
	securityhub:AcceptInvitation
	securityhub:BatchDeleteAutomationRules
	securityhub:BatchDisableStandards
	securityhub:BatchEnableStandards
	securityhub:BatchGetAutomationRules
	securityhub:BatchGetConfigurationPolicyAssociations
	securityhub：BatchGetEnabledRegionsV2
	securityhub:BatchGetSecurityControls
	securityhub:BatchGetStandardsControlAssociations
	securityhub:BatchImportFindings
	securityhub:BatchUpdateAutomationRules
	securityhub:BatchUpdateFindings
	securityhub:BatchUpdateStandardsControlAssociations
	securityhub:ConnectorRegistrationsV2
	securityhub:CreateActionTarget
	securityhub:CreateAggregatorV2
	securityhub:CreateAutomationRule
	securityhub:CreateAutomationRuleV2
	securityhub:CreateConfigurationPolicy
	securityhub：CreateConnector

服務前綴	動作
	securityhub:CreateConnectorV2
	securityhub:CreateFindingAggregator
	securityhub:CreateInsight
	securityhub:CreateMembers
	securityhub:CreateTicketV2
	securityhub:DeclineInvitations
	securityhub:DeleteActionTarget
	securityhub:DeleteAggregatorV2
	securityhub:DeleteAutomationRuleV2
	securityhub:DeleteConfigurationPolicy
	securityhub：DeleteConnector
	securityhub:DeleteConnectorV2
	securityhub:DeleteFindingAggregator
	securityhub:DeleteInsight
	securityhub:DeleteInvitations
	securityhub:DeleteMembers
	securityhub:DescribeActionTargets
	securityhub:DescribeHub
	securityhub:DescribeOrganizationConfiguration
	securityhub:DescribeProducts
	securityhub:DescribeSecurityHubV2

服務前綴	動作
	securityhub:DescribeStandards
	securityhub:DisableImportFindingsForProduct
	securityhub:DisableOrganizationAdminAccount
	securityhub:DisableSecurityHub
	securityhub：DisableSecurityHubFeatureV2
	securityhub:DisableSecurityHubV2
	securityhub:DisassociateFromAdministratorAccount
	securityhub:DisassociateFromMasterAccount
	securityhub:DisassociateMembers
	securityhub:EnableImportFindingsForProduct
	securityhub:EnableOrganizationAdminAccount
	securityhub:EnableSecurityHub
	securityhub：EnableSecurityHubFeatureV2
	securityhub:GetAdministratorAccount
	securityhub:GetAggregatorV2
	securityhub:GetAutomationRuleV2
	securityhub:GetConfigurationPolicy
	securityhub:GetConfigurationPolicyAssociation
	securityhub：GetConnector
	securityhub:GetConnectorV2
	securityhub：GetCoverageStatisticsV2

服務前綴	動作
	securityhub:GetEnabledStandards
	securityhub:GetFindingAggregator
	securityhub:GetFindingHistory
	securityhub:GetFindings
	securityhub:GetInsightResults
	securityhub:GetInsights
	securityhub:GetInvitationsCount
	securityhub:GetMasterAccount
	securityhub:GetMembers
	securityhub:GetSecurityControlDefinition
	securityhub:InviteMembers
	securityhub:ListAggregatorsV2
	securityhub:ListAutomationRules
	securityhub:ListAutomationRulesV2
	securityhub:ListConfigurationPolicies
	securityhub:ListConfigurationPolicyAssociations
	securityhub：ListConnectors
	securityhub:ListConnectorsV2
	securityhub:ListEnabledProductsForImport
	securityhub:ListFindingAggregators
	securityhub:ListInvitations

服務前綴	動作
	securityhub:ListMembers
	securityhub:ListOrganizationAdminAccounts
	securityhub:ListSecurityControlDefinitions
	securityhub:ListStandardsControlAssociations
	securityhub:StartConfigurationPolicyAssociation
	securityhub:StartConfigurationPolicyDisassociation
	securityhub:UpdateActionTarget
	securityhub:UpdateAggregatorV2
	securityhub:UpdateAutomationRuleV2
	securityhub:UpdateConfigurationPolicy
	securityhub：UpdateConnector
	securityhub:UpdateConnectorV2
	securityhub:UpdateFindingAggregator
	securityhub:UpdateFindings
	securityhub:UpdateInsight
	securityhub:UpdateOrganizationConfiguration
	securityhub:UpdateSecurityControl
	securityhub:UpdateSecurityHubConfiguration

服務前綴	動作
securitylake	securitylake:CreateAwsLogSource securitylake:CreateCustomLogSource securitylake:CreateDataLakeExceptionSubscription securitylake:CreateDataLakeOrganizationConfiguration securitylake:CreateSubscriber securitylake:CreateSubscriberNotification securitylake:DeleteAwsLogSource securitylake:DeleteCustomLogSource securitylake:DeleteDataLakeExceptionSubscription securitylake:DeleteDataLakeOrganizationConfiguration securitylake:DeleteSubscriber securitylake:DeleteSubscriberNotification securitylake:DeregisterDataLakeDelegatedAdministrator securitylake:GetDataLakeExceptionSubscription securitylake:GetDataLakeOrganizationConfiguration securitylake:GetDataLakeSources securitylake:GetSubscriber securitylake:ListDataLakes securitylake:ListLogSources securitylake:ListSubscribers securitylake:RegisterDataLakeDelegatedAdministrator

服務前綴	動作
	securitylake:UpdateDataLakeExceptionSubscription securitylake:UpdateSubscriber securitylake:UpdateSubscriberNotification
serverlessrepo	serverlessrepo:CreateApplication serverlessrepo:CreateApplicationVersion serverlessrepo:CreateCloudFormationChangeSet serverlessrepo:CreateCloudFormationTemplate serverlessrepo:DeleteApplication serverlessrepo:GetApplication serverlessrepo:GetApplicationPolicy serverlessrepo:GetCloudFormationTemplate serverlessrepo:ListApplicationDependencies serverlessrepo:ListApplicationVersions serverlessrepo:ListApplications serverlessrepo:PutApplicationPolicy serverlessrepo:UnshareApplication serverlessrepo:UpdateApplication

服務前綴	動作
servicecatalog	servicecatalog:AcceptPortfolioShare servicecatalog:AssociateBudgetWithResource servicecatalog:AssociatePrincipalWithPortfolio servicecatalog:AssociateProductWithPortfolio servicecatalog:AssociateServiceActionWithProvisioningArtifact servicecatalog:BatchAssociateServiceActionWithProvisioningArtifact servicecatalog:BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog:CopyProduct servicecatalog:CreateAttributeGroup servicecatalog:CreateConstraint servicecatalog:CreatePortfolio servicecatalog:CreatePortfolioShare servicecatalog:CreateProduct servicecatalog:CreateProvisionedProductPlan servicecatalog:CreateProvisioningArtifact servicecatalog:CreateServiceAction servicecatalog:DeleteAttributeGroup servicecatalog:DeleteConstraint servicecatalog:DeletePortfolio servicecatalog:DeletePortfolioShare

服務前綴	動作
	servicecatalog:DeleteProduct servicecatalog:DeleteProvisionedProductPlan servicecatalog:DeleteProvisioningArtifact servicecatalog:DeleteServiceAction servicecatalog:DescribeConstraint servicecatalog:DescribeCopyProductStatus servicecatalog:DescribePortfolio servicecatalog:DescribePortfolioShareStatus servicecatalog:DescribePortfolioShares servicecatalog:DescribeProduct servicecatalog:DescribeProductAsAdmin servicecatalog:DescribeProductView servicecatalog:DescribeProvisionedProduct servicecatalog:DescribeProvisionedProductPlan servicecatalog:DescribeProvisioningArtifact servicecatalog:DescribeProvisioningParameters servicecatalog:DescribeRecord servicecatalog:DescribeServiceAction servicecatalog:DescribeServiceActionExecutionParameters servicecatalog:DisableAWSOrganizationsAccess servicecatalog:DisassociateBudgetFromResource

服務前綴	動作
	servicecatalog:DisassociatePrincipalFromPortfolio servicecatalog:DisassociateProductFromPortfolio servicecatalog:DisassociateServiceActionFromProvisioningArtifact servicecatalog:EnableAWSOrganizationsAccess servicecatalog:ExecuteProvisionedProductPlan servicecatalog:ExecuteProvisionedProductServiceAction servicecatalog:GetAWSOrganizationsAccessStatus servicecatalog:GetProvisionedProductOutputs servicecatalog:ImportAsProvisionedProduct servicecatalog:ListAcceptedPortfolioShares servicecatalog:ListAttributeGroups servicecatalog:ListBudgetsForResource servicecatalog:ListConstraintsForPortfolio servicecatalog:ListLaunchPaths servicecatalog:ListOrganizationPortfolioAccess servicecatalog:ListPortfolioAccess servicecatalog:ListPortfolios servicecatalog:ListPortfoliosForProduct servicecatalog:ListPrincipalsForPortfolio servicecatalog:ListProvisionedProductPlans servicecatalog:ListProvisioningArtifacts

服務前綴	動作
	servicecatalog:ListProvisioningArtifactsForServiceAction servicecatalog:ListRecordHistory servicecatalog:ListServiceActions servicecatalog:ListServiceActionsForProvisioningArtifact servicecatalog:ListStackInstancesForProvisionedProduct servicecatalog:NotifyProvisionProductEngineWorkflowResult servicecatalog:NotifyTerminateProvisionedProductEngine WorkflowResult servicecatalog:NotifyUpdateProvisionedProductEngineWor kflowResult servicecatalog:ProvisionProduct servicecatalog:RejectPortfolioShare servicecatalog:ScanProvisionedProducts servicecatalog:SearchProducts servicecatalog:SearchProductsAsAdmin servicecatalog:SearchProvisionedProducts servicecatalog:TerminateProvisionedProduct servicecatalog:UpdateConstraint servicecatalog:UpdatePortfolio servicecatalog:UpdatePortfolioShare servicecatalog:UpdateProduct servicecatalog:UpdateProvisionedProduct

服務前綴	動作
	servicecatalog:UpdateProvisionedProductProperties servicecatalog:UpdateProvisioningArtifact servicecatalog:UpdateServiceAction

服務前綴	動作
servicediscovery	servicediscovery:CreateHttpNamespace
	servicediscovery:CreatePrivateDnsNamespace
	servicediscovery:CreatePublicDnsNamespace
	servicediscovery:CreateService
	servicediscovery:DeleteNamespace
	servicediscovery:DeleteService
	servicediscovery:DeleteServiceAttributes
	servicediscovery:DeregisterInstance
	servicediscovery:GetInstance
	servicediscovery:GetInstancesHealthStatus
	servicediscovery:GetNamespace
	servicediscovery:GetOperation
	servicediscovery:GetService
	servicediscovery:ListInstances
	servicediscovery:ListNamespaces
	servicediscovery:ListOperations
	servicediscovery:ListServices
	servicediscovery:RegisterInstance
	servicediscovery:UpdateHttpNamespace
	servicediscovery:UpdateInstanceCustomHealthStatus
	servicediscovery:UpdatePrivateDnsNamespace

服務前綴	動作
	servicediscovery:UpdatePublicDnsNamespace servicediscovery:UpdateService servicediscovery:UpdateServiceAttributes

服務前綴	動作
servicequotas	servicequotas:AssociateServiceQuotaTemplate servicequotas:CreateSupportCase servicequotas:DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas:DisassociateServiceQuotaTemplate servicequotas:GetAWSDefaultServiceQuota servicequotas:GetAssociationForServiceQuotaTemplate servicequotas：GetAutoManagementConfiguration servicequotas：GetQuotaUtilizationReport servicequotas:GetRequestedServiceQuotaChange servicequotas:GetServiceQuota servicequotas:GetServiceQuotaIncreaseRequestFromTemplate servicequotas:ListAWSDefaultServiceQuotas servicequotas:ListRequestedServiceQuotaChangeHistory servicequotas:ListRequestedServiceQuotaChangeHistoryByQuota servicequotas:ListServiceQuotaIncreaseRequestsInTemplate servicequotas:ListServiceQuotas servicequotas:ListServices servicequotas:PutServiceQuotaIncreaseRequestIntoTemplate servicequotas:RequestServiceQuotaIncrease servicequotas：StartAutoManagement servicequotas：StartQuotaUtilizationReport

服務前綴	動作
	servicequotas：StopAutoManagement servicequotas：UpdateAutoManagement

服務前綴	動作
ses	ses:BatchGetMetricData
	ses:CloneReceiptRuleSet
	ses:CreateAddonInstance
	ses:CreateAddonSubscription
	ses:CreateAddressList
	ses:CreateAddressListImportJob
	ses:CreateArchive
	ses:CreateConfigurationSet
	ses:CreateConfigurationSetEventDestination
	ses:CreateConfigurationSetTrackingOptions
	ses:CreateContact
	ses:CreateContactList
	ses:CreateCustomVerificationEmailTemplate
	ses:CreateDedicatedIpPool
	ses:CreateDeliverabilityTestReport
	ses:CreateEmailIdentity
	ses:CreateEmailIdentityPolicy
	ses:CreateEmailTemplate
	ses:CreateImportJob
	ses:CreateIngressPoint
	ses:CreateMultiRegionEndpoint

服務前綴	動作
	ses:CreateReceiptFilter
	ses:CreateReceiptRule
	ses:CreateReceiptRuleSet
	ses:CreateRelay
	ses:CreateRuleSet
	ses:CreateTemplate
	ses:CreateTenant
	ses:CreateTenantResourceAssociation
	ses:CreateTrafficPolicy
	ses:DeleteAddonInstance
	ses:DeleteAddonSubscription
	ses:DeleteAddressList
	ses:DeleteArchive
	ses:DeleteConfigurationSet
	ses:DeleteConfigurationSetEventDestination
	ses:DeleteConfigurationSetTrackingOptions
	ses:DeleteContact
	ses:DeleteContactList
	ses:DeleteCustomVerificationEmailTemplate
	ses:DeleteDedicatedIpPool
	ses:DeleteEmailIdentity

服務前綴	動作
	ses:DeleteEmailIdentityPolicy
	ses:DeleteEmailTemplate
	ses:DeleteIdentity
	ses:DeleteIdentityPolicy
	ses:DeleteIngressPoint
	ses:DeleteMultiRegionEndpoint
	ses:DeleteReceiptFilter
	ses:DeleteReceiptRule
	ses:DeleteReceiptRuleSet
	ses:DeleteRelay
	ses:DeleteRuleSet
	ses:DeleteSuppressedDestination
	ses:DeleteTemplate
	ses:DeleteTenant
	ses:DeleteTenantResourceAssociation
	ses:DeleteTrafficPolicy
	ses:DeleteVerifiedEmailAddress
	ses:DeregisterMemberFromAddressList
	ses:DescribeActiveReceiptRuleSet
	ses:DescribeConfigurationSet
	ses:DescribeReceiptRule

服務前綴	動作
	ses:DescribeReceiptRuleSet
	ses:GetAccount
	ses:GetAccountSendingEnabled
	ses:GetAddonInstance
	ses:GetAddonSubscription
	ses:GetAddressList
	ses:GetArchive
	ses:GetArchiveExport
	ses:GetArchiveMessage
	ses:GetArchiveMessageContent
	ses:GetArchiveSearch
	ses:GetArchiveSearchResults
	ses:GetBlacklistReports
	ses:GetConfigurationSet
	ses:GetConfigurationSetEventDestinations
	ses:GetContact
	ses:GetContactList
	ses:GetCustomVerificationEmailTemplate
	ses:GetDedicatedIp
	ses:GetDedicatedIpPool
	ses:GetDedicatedIps

服務前綴	動作
	ses:GetDeliverabilityDashboardOptions
	ses:GetDeliverabilityTestReport
	ses:GetDomainDeliverabilityCampaign
	ses:GetDomainStatisticsReport
	ses：GetEmailAddressInsights
	ses:GetEmailIdentity
	ses:GetEmailIdentityPolicies
	ses:GetEmailTemplate
	ses:GetIdentityDkimAttributes
	ses:GetIdentityMailFromDomainAttributes
	ses:GetIdentityNotificationAttributes
	ses:GetIdentityPolicies
	ses:GetIdentityVerificationAttributes
	ses:GetImportJob
	ses:GetIngressPoint
	ses:GetMemberOfAddressList
	ses:GetMessageInsights
	ses:GetMultiRegionEndpoint
	ses:GetRelay
	ses:GetRuleSet
	ses:GetSendQuota

服務前綴	動作
	ses:GetSendStatistics
	ses:GetSuppressedDestination
	ses:GetTemplate
	ses:GetTenant
	ses:GetTrafficPolicy
	ses:ListAddonInstances
	ses:ListAddonSubscriptions
	ses:ListAddressListImportJobs
	ses:ListAddressLists
	ses:ListArchiveExports
	ses:ListArchiveSearches
	ses:ListArchives
	ses:ListConfigurationSets
	ses:ListContactLists
	ses:ListContacts
	ses:ListCustomVerificationEmailTemplates
	ses:ListDedicatedIpPools
	ses:ListDeliverabilityTestReports
	ses:ListDomainDeliverabilityCampaigns
	ses:ListEmailIdentities
	ses:ListEmailTemplates

服務前綴	動作
	ses:ListExportJobs
	ses:ListIdentities
	ses:ListIdentityPolicies
	ses:ListImportJobs
	ses:ListIngressPoints
	ses:ListMembersOfAddressList
	ses:ListMultiRegionEndpoints
	ses:ListReceiptFilters
	ses:ListReceiptRuleSets
	ses:ListRecommendations
	ses:ListRelays
	ses:ListReputationEntities
	ses:ListResourceTenants
	ses:ListRuleSets
	ses:ListSuppressedDestinations
	ses:ListTemplates
	ses:ListTenantResources
	ses:ListTenants
	ses:ListTrafficPolicies
	ses:ListVerifiedEmailAddresses
	ses:PutAccountDedicatedIpWarmupAttributes

服務前綴	動作
	ses:PutAccountDetails
	ses:PutAccountSendingAttributes
	ses:PutAccountSuppressionAttributes
	ses:PutAccountVdmAttributes
	ses:PutConfigurationSetArchivingOptions
	ses:PutConfigurationSetDeliveryOptions
	ses:PutConfigurationSetReputationOptions
	ses:PutConfigurationSetSendingOptions
	ses:PutConfigurationSetSuppressionOptions
	ses:PutConfigurationSetTrackingOptions
	ses:PutConfigurationSetVdmOptions
	ses:PutDedicatedIpInPool
	ses:PutDedicatedIpPoolScalingAttributes
	ses:PutDedicatedIpWarmupAttributes
	ses:PutDeliverabilityDashboardOption
	ses:PutEmailIdentityConfigurationSetAttributes
	ses:PutEmailIdentityDkimAttributes
	ses:PutEmailIdentityDkimSigningAttributes
	ses:PutEmailIdentityFeedbackAttributes
	ses:PutEmailIdentityMailFromAttributes
	ses:PutIdentityPolicy

服務前綴	動作
	ses:PutSuppressedDestination
	ses：PutTenantSuppressionAttributes
	ses:RegisterMemberToAddressList
	ses:ReorderReceiptRuleSet
	ses:SendBounce
	ses:SendCustomVerificationEmail
	ses:SetActiveReceiptRuleSet
	ses:SetIdentityDkimEnabled
	ses:SetIdentityFeedbackForwardingEnabled
	ses:SetIdentityHeadersInNotificationsEnabled
	ses:SetIdentityMailFromDomain
	ses:SetIdentityNotificationTopic
	ses:SetReceiptRulePosition
	ses:StartArchiveExport
	ses:StartArchiveSearch
	ses:StopArchiveExport
	ses:StopArchiveSearch
	ses:TestRenderEmailTemplate
	ses:TestRenderTemplate
	ses:UpdateAccountSendingEnabled
	ses:UpdateArchive

服務前綴	動作
	ses:UpdateConfigurationSetEventDestination
	ses:UpdateConfigurationSetReputationMetricsEnabled
	ses:UpdateConfigurationSetSendingEnabled
	ses:UpdateConfigurationSetTrackingOptions
	ses:UpdateContact
	ses:UpdateContactList
	ses:UpdateCustomVerificationEmailTemplate
	ses:UpdateEmailIdentityPolicy
	ses:UpdateEmailTemplate
	ses:UpdateIngressPoint
	ses:UpdateReceiptRule
	ses:UpdateRelay
	ses:UpdateRuleSet
	ses:UpdateTemplate
	ses:UpdateTrafficPolicy
	ses:VerifyDomainDkim
	ses:VerifyDomainIdentity
	ses:VerifyEmailAddress
	ses:VerifyEmailIdentity

服務前綴	動作
shield	shield:AssociateDRTLogBucket shield:AssociateHealthCheck shield:AssociateProactiveEngagementDetails shield:CreateProtection shield:CreateProtectionGroup shield:CreateSubscription shield:DeleteProtection shield:DeleteProtectionGroup shield:DeleteSubscription shield:DescribeAttack shield:DescribeAttackStatistics shield:DescribeDRTAccess shield:DescribeEmergencyContactSettings shield:DescribeProtection shield:DescribeProtectionGroup shield:DescribeSubscription shield:DisableApplicationLayerAutomaticResponse shield:DisableProactiveEngagement shield:DisassociateDRTLogBucket shield:DisassociateDRTRole shield:DisassociateHealthCheck

服務前綴	動作
	shield:EnableApplicationLayerAutomaticResponse shield:EnableProactiveEngagement shield:GetSubscriptionState shield:ListAttacks shield:ListProtectionGroups shield:ListProtections shield:ListResourcesInProtectionGroup shield:UpdateApplicationLayerAutomaticResponse shield:UpdateEmergencyContactSettings shield:UpdateProtectionGroup shield:UpdateSubscription

服務前綴	動作
signer	signer:AddProfilePermission signer:CancelSigningProfile signer:DescribeSigningJob signer:GetSigningPlatform signer:GetSigningProfile signer:ListProfilePermissions signer:ListSigningJobs signer:ListSigningPlatforms signer:ListSigningProfiles signer:PutSigningProfile signer:RemoveProfilePermission signer:RevokeSignature signer:RevokeSigningProfile signer:SignPayload signer:StartSigningJob

服務前綴	動作
simspaceweaver	simspaceweaver:CreateSnapshot simspaceweaver:DeleteApp simspaceweaver:DeleteSimulation simspaceweaver:DescribeApp simspaceweaver:DescribeSimulation simspaceweaver:ListApps simspaceweaver:ListSimulations simspaceweaver:StartApp simspaceweaver:StartClock simspaceweaver:StartSimulation simspaceweaver:StopApp simspaceweaver:StopClock simspaceweaver:StopSimulation

服務前綴	動作
sms	sms:CreateApp
	sms:CreateReplicationJob
	sms:DeleteApp
	sms:DeleteAppLaunchConfiguration
	sms:DeleteAppReplicationConfiguration
	sms:DeleteAppValidationConfiguration
	sms:DeleteReplicationJob
	sms:DeleteServerCatalog
	sms:DisassociateConnector
	sms:GenerateChangeSet
	sms:GenerateTemplate
	sms:GetApp
	sms:GetAppLaunchConfiguration
	sms:GetAppReplicationConfiguration
	sms:GetAppValidationConfiguration
	sms:GetAppValidationOutput
	sms:GetConnectors
	sms:GetReplicationJobs
	sms:GetReplicationRuns
	sms:GetServers
	sms:ImportAppCatalog

服務前綴	動作
	sms:ImportServerCatalog
	sms:LaunchApp
	sms:ListApps
	sms:NotifyAppValidationOutput
	sms:PutAppLaunchConfiguration
	sms:PutAppReplicationConfiguration
	sms:PutAppValidationConfiguration
	sms:StartAppReplication
	sms:StartOnDemandAppReplication
	sms:StartOnDemandReplicationRun
	sms:StopAppReplication
	sms:TerminateApp
	sms:UpdateApp
	sms:UpdateReplicationJob

服務前綴	動作
sms-voice	sms-voice:AssociateProtectConfiguration sms-voice:CreateConfigurationSet sms-voice:CreateConfigurationSetEventDestination sms-voice:CreateEventDestination sms-voice:CreateOptOutList sms-voice:CreatePool sms-voice:CreateProtectConfiguration sms-voice:CreateRegistration sms-voice:CreateRegistrationAssociation sms-voice:CreateRegistrationAttachment sms-voice:CreateRegistrationVersion sms-voice:CreateVerifiedDestinationNumber sms-voice:DeleteAccountDefaultProtectConfiguration sms-voice:DeleteConfigurationSet sms-voice:DeleteConfigurationSetEventDestination sms-voice:DeleteDefaultMessageType sms-voice:DeleteDefaultSenderId sms-voice:DeleteEventDestination sms-voice:DeleteKeyword sms-voice:DeleteMediaMessageSpendLimitOverride sms-voice:DeleteOptOutList

服務前綴	動作
	sms-voice:DeleteOptedOutNumber sms-voice:DeletePool sms-voice:DeleteProtectConfiguration sms-voice:DeleteProtectConfigurationRuleSetNumberOverride sms-voice:DeleteRegistration sms-voice:DeleteRegistrationAttachment sms-voice:DeleteResourcePolicy sms-voice:DeleteTextMessageSpendLimitOverride sms-voice:DeleteVerifiedDestinationNumber sms-voice:DeleteVoiceMessageSpendLimitOverride sms-voice:DescribeAccountAttributes sms-voice:DescribeAccountLimits sms-voice:DescribeConfigurationSets sms-voice:DescribeKeywords sms-voice:DescribeOptOutLists sms-voice:DescribeOptedOutNumbers sms-voice:DescribePhoneNumbers sms-voice:DescribePools sms-voice:DescribeProtectConfigurations sms-voice:DescribeRegistrationAttachments sms-voice:DescribeRegistrationFieldDefinitions

服務前綴	動作
	sms-voice:DescribeRegistrationFieldValues sms-voice:DescribeRegistrationSectionDefinitions sms-voice:DescribeRegistrationTypeDefinitions sms-voice:DescribeRegistrationVersions sms-voice:DescribeRegistrations sms-voice:DescribeSenderIds sms-voice:DescribeSpendLimits sms-voice:DescribeVerifiedDestinationNumbers sms-voice:DisassociateOriginationIdentity sms-voice:DisassociateProtectConfiguration sms-voice:DiscardRegistrationVersion sms-voice:GetConfigurationSetEventDestinations sms-voice:GetProtectConfigurationCountryRuleSet sms-voice:GetResourcePolicy sms-voice:ListConfigurationSets sms-voice:ListPoolOriginationIdentities sms-voice:ListProtectConfigurationRuleSetNumberOverrides sms-voice:ListRegistrationAssociations sms-voice:PutKeyword sms-voice:PutOptedOutNumber sms-voice:PutProtectConfigurationRuleSetNumberOverride

服務前綴	動作
	sms-voice:PutResourcePolicy sms-voice:ReleasePhoneNumber sms-voice:ReleaseSenderId sms-voice:RequestPhoneNumber sms-voice:RequestSenderId sms-voice:SendDestinationNumberVerificationCode sms-voice:SetAccountDefaultProtectConfiguration sms-voice:SetDefaultMessageFeedbackEnabled sms-voice:SetDefaultMessageType sms-voice:SetDefaultSenderId sms-voice:SetMediaMessageSpendLimitOverride sms-voice:SetTextMessageSpendLimitOverride sms-voice:SetVoiceMessageSpendLimitOverride sms-voice:SubmitRegistrationVersion sms-voice:UpdateConfigurationSetEventDestination sms-voice:UpdateEventDestination sms-voice:UpdatePhoneNumber sms-voice:UpdatePool sms-voice:UpdateProtectConfiguration sms-voice:UpdateProtectConfigurationCountryRuleSet sms-voice:UpdateSenderId

服務前綴	動作
snowball	snowball:CancelCluster
	snowball:CancelJob
	snowball:CreateAddress
	snowball:CreateCluster
	snowball:CreateJob
	snowball:CreateLongTermPricing
	snowball:CreateReturnShippingLabel
	snowball:DescribeAddress
	snowball:DescribeAddresses
	snowball:DescribeCluster
	snowball:DescribeJob
	snowball:DescribeReturnShippingLabel
	snowball:GetJobManifest
	snowball:GetJobUnlockCode
	snowball:GetSnowballUsage
	snowball:GetSoftwareUpdates
	snowball:ListClusterJobs
	snowball:ListClusters
	snowball:ListCompatibleImages
	snowball:ListJobs
	snowball:ListLongTermPricing

服務前綴	動作
	snowball:ListPickupLocations snowball:ListServiceVersions snowball:UpdateCluster snowball:UpdateJob snowball:UpdateJobShipmentState snowball:UpdateLongTermPricing
sqs	sqs:AddPermission sqs:CancelMessageMoveTask sqs:CreateQueue sqs:DeleteQueue sqs:PurgeQueue sqs:RemovePermission sqs:SetQueueAttributes

服務前綴	動作
ssm	ssm:AssociateOpsItemRelatedItem
	ssm:CancelCommand
	ssm:CancelMaintenanceWindowExecution
	ssm:CreateActivation
	ssm:CreateAssociation
	ssm:CreateAssociationBatch
	ssm:CreateDocument
	ssm:CreateMaintenanceWindow
	ssm:CreateOpsItem
	ssm:CreateOpsMetadata
	ssm:CreatePatchBaseline
	ssm:CreateResourceDataSync
	ssm:DeleteActivation
	ssm:DeleteAssociation
	ssm：DeleteCloudConnector
	ssm:DeleteDocument
	ssm:DeleteInventory
	ssm:DeleteMaintenanceWindow
	ssm:DeleteOpsItem
	ssm:DeleteOpsMetadata
	ssm:DeleteParameter

服務前綴	動作
	ssm:DeleteParameters
	ssm:DeletePatchBaseline
	ssm:DeleteResourceDataSync
	ssm:DeleteResourcePolicy
	ssm:DeregisterManagedInstance
	ssm:DeregisterPatchBaselineForPatchGroup
	ssm:DeregisterTargetFromMaintenanceWindow
	ssm:DeregisterTaskFromMaintenanceWindow
	ssm:DescribeActivations
	ssm:DescribeAssociation
	ssm:DescribeAssociationExecutionTargets
	ssm:DescribeAssociationExecutions
	ssm:DescribeAutomationExecutions
	ssm:DescribeAutomationStepExecutions
	ssm:DescribeAvailablePatches
	ssm:DescribeDocument
	ssm:DescribeDocumentParameters
	ssm:DescribeDocumentPermission
	ssm:DescribeEffectiveInstanceAssociations
	ssm:DescribeEffectivePatchesForPatchBaseline
	ssm:DescribeInstanceAssociationsStatus

服務前綴	動作
	ssm:DescribeInstanceInformation
	ssm:DescribeInstancePatchStates
	ssm:DescribeInstancePatchStatesForPatchGroup
	ssm:DescribeInstancePatches
	ssm:DescribeInstanceProperties
	ssm:DescribeInventoryDeletions
	ssm:DescribeMaintenanceWindowExecutionTaskInvocations
	ssm:DescribeMaintenanceWindowExecutionTasks
	ssm:DescribeMaintenanceWindowExecutions
	ssm:DescribeMaintenanceWindowSchedule
	ssm:DescribeMaintenanceWindowTargets
	ssm:DescribeMaintenanceWindowTasks
	ssm:DescribeMaintenanceWindows
	ssm:DescribeMaintenanceWindowsForTarget
	ssm:DescribeOpsItems
	ssm:DescribeParameters
	ssm:DescribePatchBaselines
	ssm:DescribePatchGroupState
	ssm:DescribePatchGroups
	ssm:DescribePatchProperties
	ssm:DescribeSessions

服務前綴	動作
	ssm:DisassociateOpsItemRelatedItem
	ssm:GetAccessToken
	ssm:GetAutomationExecution
	ssm:GetCalendarState
	ssm：GetCloudConnector
	ssm:GetCommandInvocation
	ssm:GetConnectionStatus
	ssm:GetDefaultPatchBaseline
	ssm:GetDeployablePatchSnapshotForInstance
	ssm:GetDocument
	ssm:GetExecutionPreview
	ssm:GetInventory
	ssm:GetInventorySchema
	ssm:GetMaintenanceWindow
	ssm:GetMaintenanceWindowExecution
	ssm:GetMaintenanceWindowExecutionTask
	ssm:GetMaintenanceWindowExecutionTaskInvocation
	ssm:GetMaintenanceWindowTask
	ssm:GetOpsItem
	ssm:GetOpsMetadata
	ssm:GetOpsSummary

服務前綴	動作
	ssm:GetParameter
	ssm:GetParameterHistory
	ssm:GetParameters
	ssm:GetParametersByPath
	ssm:GetPatchBaseline
	ssm:GetPatchBaselineForPatchGroup
	ssm:GetResourcePolicies
	ssm:GetServiceSetting
	ssm:LabelParameterVersion
	ssm:ListAssociationVersions
	ssm:ListAssociations
	ssm：ListCloudConnectors
	ssm:ListCommandInvocations
	ssm:ListCommands
	ssm:ListComplianceItems
	ssm:ListComplianceSummaries
	ssm:ListDocumentMetadataHistory
	ssm:ListDocumentVersions
	ssm:ListDocuments
	ssm:ListInstanceAssociations
	ssm:ListInventoryEntries

服務前綴	動作
	ssm:ListNodes
	ssm:ListNodesSummary
	ssm:ListOpsItemEvents
	ssm:ListOpsItemRelatedItems
	ssm:ListOpsMetadata
	ssm:ListResourceComplianceSummaries
	ssm:ListResourceDataSync
	ssm:ModifyDocumentPermission
	ssm:PutComplianceItems
	ssm:PutInventory
	ssm:PutParameter
	ssm:PutResourcePolicy
	ssm:RegisterDefaultPatchBaseline
	ssm:RegisterManagedInstance
	ssm:RegisterPatchBaselineForPatchGroup
	ssm:RegisterTargetWithMaintenanceWindow
	ssm:RegisterTaskWithMaintenanceWindow
	ssm:ResetServiceSetting
	ssm:ResumeSession
	ssm:SendAutomationSignal
	ssm:SendCommand

服務前綴	動作
	ssm:StartAssociationsOnce
	ssm:StartAutomationExecution
	ssm:StartChangeRequestExecution
	ssm:StartSession
	ssm:StopAutomationExecution
	ssm:TerminateSession
	ssm:UnlabelParameterVersion
	ssm:UpdateAssociation
	ssm:UpdateAssociationStatus
	ssm：UpdateCloudConnector
	ssm:UpdateDocument
	ssm:UpdateDocumentDefaultVersion
	ssm:UpdateDocumentMetadata
	ssm:UpdateInstanceInformation
	ssm:UpdateMaintenanceWindow
	ssm:UpdateMaintenanceWindowTarget
	ssm:UpdateMaintenanceWindowTask
	ssm:UpdateManagedInstanceRole
	ssm:UpdateOpsItem
	ssm:UpdateOpsMetadata
	ssm:UpdatePatchBaseline

服務前綴	動作
	ssm:UpdateResourceDataSync ssm:UpdateServiceSetting ssm：ValidateCloudConnector

服務前綴	動作
ssm-incidents	ssm-incidents:BatchGetIncidentFindings ssm-incidents:CreateReplicationSet ssm-incidents:CreateResponsePlan ssm-incidents:CreateTimelineEvent ssm-incidents:DeleteIncidentRecord ssm-incidents:DeleteReplicationSet ssm-incidents:DeleteResourcePolicy ssm-incidents:DeleteResponsePlan ssm-incidents:DeleteTimelineEvent ssm-incidents:GetIncidentRecord ssm-incidents:GetReplicationSet ssm-incidents:GetResourcePolicies ssm-incidents:GetResponsePlan ssm-incidents:GetTimelineEvent ssm-incidents:ListIncidentFindings ssm-incidents:ListIncidentRecords ssm-incidents:ListRelatedItems ssm-incidents:ListReplicationSets ssm-incidents:ListResponsePlans ssm-incidents:ListTimelineEvents ssm-incidents:PutResourcePolicy

服務前綴	動作
	ssm-incidents:StartIncident ssm-incidents:UpdateDeletionProtection ssm-incidents:UpdateIncidentRecord ssm-incidents:UpdateRelatedItems ssm-incidents:UpdateReplicationSet ssm-incidents:UpdateResponsePlan ssm-incidents:UpdateTimelineEvent

服務前綴	動作
ssm-sap	ssm-sap:BackupDatabase
	ssm-sap:DeleteResourcePermission
	ssm-sap:DeregisterApplication
	ssm-sap:GetApplication
	ssm-sap:GetComponent
	ssm-sap：GetConfigurationCheckOperation
	ssm-sap:GetDatabase
	ssm-sap:GetOperation
	ssm-sap:GetResourcePermission
	ssm-sap:ListApplications
	ssm-sap:ListComponents
	ssm-sap：ListConfigurationCheckDefinitions
	ssm-sap：ListConfigurationCheckOperations
	ssm-sap:ListDatabases
	ssm-sap:ListOperationEvents
	ssm-sap:ListOperations
	ssm-sap：ListSubCheckResults
	ssm-sap：ListSubCheckRuleResults
	ssm-sap:PutResourcePermission
	ssm-sap:RegisterApplication
	ssm-sap:RestoreDatabase

服務前綴	動作
	ssm-sap:StartApplication ssm-sap:StartApplicationRefresh ssm-sap：StartConfigurationChecks ssm-sap:StopApplication ssm-sap:UpdateApplicationSettings ssm-sap:UpdateHANABackupSettings

服務前綴	動作
states	states:CreateActivity states:CreateStateMachine states:CreateStateMachineAlias states:DeleteActivity states:DeleteStateMachine states:DeleteStateMachineAlias states:DeleteStateMachineVersion states:DescribeActivity states:DescribeExecution states:DescribeMapRun states:DescribeStateMachine states:DescribeStateMachineAlias states:DescribeStateMachineForExecution states:GetExecutionHistory states:ListActivities states:ListExecutions states:ListMapRuns states:ListStateMachineAliases states:ListStateMachineVersions states:ListStateMachines states:SendTaskFailure

服務前綴	動作
	states:SendTaskHeartbeat states:SendTaskSuccess states:StartExecution states:StopExecution states:UpdateMapRun states:UpdateStateMachine states:UpdateStateMachineAlias states:ValidateStateMachineDefinition
sts	sts:AssumeRole sts:AssumeRoleWithSAML sts:AssumeRoleWithWebIdentity sts:DecodeAuthorizationMessage sts:GetAccessKeyInfo sts:GetCallerIdentity sts:GetFederationToken sts:GetSessionToken sts：GetWebIdentityToken

服務前綴	動作
swf	swf:DeleteActivityType swf:DeleteWorkflowType swf:DeprecateActivityType swf:DeprecateDomain swf:DeprecateWorkflowType swf:DescribeActivityType swf:DescribeDomain swf:DescribeWorkflowType swf:ListActivityTypes swf:ListDomains swf:ListWorkflowTypes swf:RegisterActivityType swf:RegisterDomain swf:RegisterWorkflowType swf:UndeprecateActivityType swf:UndeprecateDomain swf:UndeprecateWorkflowType

服務前綴	動作
synthetics	synthetics:AssociateResource
	synthetics:CreateCanary
	synthetics:CreateGroup
	synthetics:DeleteCanary
	synthetics:DeleteGroup
	synthetics:DescribeCanaries
	synthetics:DescribeCanariesLastRun
	synthetics:DescribeRuntimeVersions
	synthetics:DisassociateResource
	synthetics:GetCanary
	synthetics:GetCanaryRuns
	synthetics:GetGroup
	synthetics:ListAssociatedGroups
	synthetics:ListGroupResources
	synthetics:ListGroups
	synthetics:StartCanary
	synthetics:StartCanaryDryRun
	synthetics:StopCanary
	synthetics:UpdateCanary

服務前綴	動作
標籤	tag:DescribeReportCreation tag:GetComplianceSummary tag:GetResources tag:StartReportCreation

服務前綴	動作
textract	textract:AnalyzeDocument
	textract:AnalyzeExpense
	textract:AnalyzeID
	textract:CreateAdapter
	textract:CreateAdapterVersion
	textract:DeleteAdapter
	textract:DeleteAdapterVersion
	textract:DetectDocumentText
	textract:GetAdapter
	textract:GetAdapterVersion
	textract:GetDocumentAnalysis
	textract:GetDocumentTextDetection
	textract:GetExpenseAnalysis
	textract:GetLendingAnalysis
	textract:GetLendingAnalysisSummary
	textract:ListAdapterVersions
	textract:ListAdapters
	textract:StartDocumentAnalysis
	textract:StartDocumentTextDetection
	textract:StartExpenseAnalysis
	textract:StartLendingAnalysis

服務前綴	動作
	textract:UpdateAdapter

服務前綴	動作
timestream	timestream:CancelQuery timestream:CreateDatabase timestream:CreateScheduledQuery timestream:CreateTable timestream:DeleteDatabase timestream:DeleteScheduledQuery timestream:DeleteTable timestream:DescribeAccountSettings timestream:DescribeDatabase timestream:DescribeScheduledQuery timestream:DescribeTable timestream:ExecuteScheduledQuery timestream:ListBatchLoadTasks timestream:ListDatabases timestream:ListScheduledQueries timestream:ListTables timestream:PrepareQuery timestream:UpdateAccountSettings timestream:UpdateDatabase timestream:UpdateScheduledQuery timestream:UpdateTable

服務前綴	動作
tnb	tnb:CancelSolNetworkOperation
	tnb:CreateSolFunctionPackage
	tnb:CreateSolNetworkInstance
	tnb:CreateSolNetworkPackage
	tnb:DeleteSolFunctionPackage
	tnb:DeleteSolNetworkInstance
	tnb:DeleteSolNetworkPackage
	tnb:GetSolFunctionInstance
	tnb:GetSolFunctionPackage
	tnb:GetSolFunctionPackageContent
	tnb:GetSolFunctionPackageDescriptor
	tnb:GetSolNetworkInstance
	tnb:GetSolNetworkOperation
	tnb:GetSolNetworkPackage
	tnb:GetSolNetworkPackageContent
	tnb:GetSolNetworkPackageDescriptor
	tnb:InstantiateSolNetworkInstance
	tnb:ListSolFunctionInstances
	tnb:ListSolFunctionPackages
	tnb:ListSolNetworkInstances
	tnb:ListSolNetworkOperations

服務前綴	動作
	tnb:ListSolNetworkPackages tnb:PutSolFunctionPackageContent tnb:PutSolNetworkPackageContent tnb:TerminateSolNetworkInstance tnb:UpdateSolFunctionPackage tnb:UpdateSolNetworkInstance tnb:UpdateSolNetworkPackage tnb:ValidateSolFunctionPackageContent tnb:ValidateSolNetworkPackageContent

服務前綴	動作
transcribe	transcribe:CreateCallAnalyticsCategory transcribe:CreateLanguageModel transcribe:CreateMedicalVocabulary transcribe:CreateVocabulary transcribe:CreateVocabularyFilter transcribe:DeleteCallAnalyticsCategory transcribe:DeleteCallAnalyticsJob transcribe:DeleteLanguageModel transcribe:DeleteMedicalScribeJob transcribe:DeleteMedicalTranscriptionJob transcribe:DeleteMedicalVocabulary transcribe:DeleteTranscriptionJob transcribe:DeleteVocabulary transcribe:DeleteVocabularyFilter transcribe:DescribeLanguageModel transcribe:GetCallAnalyticsCategory transcribe:GetCallAnalyticsJob transcribe:GetMedicalScribeJob transcribe:GetMedicalTranscriptionJob transcribe:GetMedicalVocabulary transcribe:GetTranscriptionJob

服務前綴	動作
	transcribe:GetVocabulary transcribe:GetVocabularyFilter transcribe:ListCallAnalyticsCategories transcribe:ListCallAnalyticsJobs transcribe:ListLanguageModels transcribe:ListMedicalScribeJobs transcribe:ListMedicalTranscriptionJobs transcribe:ListMedicalVocabularies transcribe:ListTranscriptionJobs transcribe:ListVocabularies transcribe:ListVocabularyFilters transcribe:StartCallAnalyticsJob transcribe:StartCallAnalyticsStreamTranscription transcribe:StartCallAnalyticsStreamTranscriptionWebSocket transcribe:StartMedicalScribeJob transcribe:StartMedicalStreamTranscription transcribe:StartMedicalStreamTranscriptionWebSocket transcribe:StartMedicalTranscriptionJob transcribe:StartStreamTranscription transcribe:StartStreamTranscriptionWebSocket transcribe:StartTranscriptionJob

服務前綴	動作
	transcribe:UpdateCallAnalyticsCategory transcribe:UpdateMedicalVocabulary transcribe:UpdateVocabulary transcribe:UpdateVocabularyFilter

服務前綴	動作
傳輸	transfer:CreateAccess
	transfer:CreateAgreement
	transfer:CreateConnector
	transfer:CreateProfile
	transfer:CreateServer
	transfer:CreateUser
	transfer:CreateWebApp
	transfer:CreateWorkflow
	transfer:DeleteAccess
	transfer:DeleteAgreement
	transfer:DeleteCertificate
	transfer:DeleteConnector
	transfer:DeleteHostKey
	transfer:DeleteProfile
	transfer:DeleteServer
	transfer:DeleteSshPublicKey
	transfer:DeleteUser
	transfer:DeleteWebApp
	transfer:DeleteWebAppCustomization
	transfer:DeleteWorkflow
	transfer:DescribeAccess

服務前綴	動作
	transfer:DescribeAgreement
	transfer:DescribeCertificate
	transfer:DescribeConnector
	transfer:DescribeExecution
	transfer:DescribeHostKey
	transfer:DescribeProfile
	transfer:DescribeSecurityPolicy
	transfer:DescribeServer
	transfer:DescribeUser
	transfer:DescribeWebApp
	transfer:DescribeWebAppCustomization
	transfer:DescribeWorkflow
	transfer:ImportCertificate
	transfer:ImportHostKey
	transfer:ImportSshPublicKey
	transfer:ListAccesses
	transfer:ListCertificates
	transfer:ListConnectors
	transfer:ListExecutions
	transfer:ListFileTransferResults
	transfer:ListHostKeys

服務前綴	動作
	transfer:ListProfiles
	transfer:ListSecurityPolicies
	transfer:ListServers
	transfer:ListUsers
	transfer:ListWebApps
	transfer:ListWorkflows
	transfer:SendWorkflowStepState
	transfer:StartDirectoryListing
	transfer:StartFileTransfer
	transfer:StartRemoteDelete
	transfer:StartRemoteMove
	transfer:StartServer
	transfer:StopServer
	transfer:TestConnection
	transfer:TestIdentityProvider
	transfer:UpdateAccess
	transfer:UpdateAgreement
	transfer:UpdateCertificate
	transfer:UpdateConnector
	transfer:UpdateHostKey
	transfer:UpdateProfile

服務前綴	動作
	transfer:UpdateServer transfer:UpdateUser transfer:UpdateWebApp transfer:UpdateWebAppCustomization
translate	translate:CreateParallelData translate:DeleteParallelData translate:DeleteTerminology translate:DescribeTextTranslationJob translate:GetParallelData translate:GetTerminology translate:ImportTerminology translate:ListLanguages translate:ListParallelData translate:ListTerminologies translate:ListTextTranslationJobs translate:StartTextTranslationJob translate:StopTextTranslationJob translate:TranslateDocument translate:TranslateText translate:UpdateParallelData

服務前綴	動作
voiceid	voiceid:AssociateFraudster
	voiceid:CreateDomain
	voiceid:CreateWatchlist
	voiceid:DeleteDomain
	voiceid:DeleteFraudster
	voiceid:DeleteSpeaker
	voiceid:DeleteWatchlist
	voiceid:DescribeDomain
	voiceid:DescribeFraudster
	voiceid:DescribeFraudsterRegistrationJob
	voiceid:DescribeSpeaker
	voiceid:DescribeSpeakerEnrollmentJob
	voiceid:DescribeWatchlist
	voiceid:DisassociateFraudster
	voiceid:EvaluateSession
	voiceid:ListDomains
	voiceid:ListFraudsterRegistrationJobs
	voiceid:ListFraudsters
	voiceid:ListSpeakerEnrollmentJobs
	voiceid:ListSpeakers
	voiceid:ListWatchlists

服務前綴	動作
	voiceid:OptOutSpeaker voiceid:StartFraudsterRegistrationJob voiceid:StartSpeakerEnrollmentJob voiceid:UpdateDomain voiceid:UpdateWatchlist

服務前綴	動作
vpc-lattice	vpc-lattice:CreateAccessLogSubscription vpc-lattice:CreateListener vpc-lattice:CreateResourceConfiguration vpc-lattice:CreateResourceGateway vpc-lattice:CreateRule vpc-lattice:CreateService vpc-lattice:CreateServiceNetwork vpc-lattice:CreateServiceNetworkResourceAssociation vpc-lattice:CreateServiceNetworkServiceAssociation vpc-lattice:CreateServiceNetworkVpcAssociation vpc-lattice:CreateTargetGroup vpc-lattice:DeleteAccessLogSubscription vpc-lattice:DeleteAuthPolicy vpc-lattice：DeleteDomainVerification vpc-lattice:DeleteListener vpc-lattice:DeleteResourceConfiguration vpc-lattice:DeleteResourceEndpointAssociation vpc-lattice:DeleteResourceGateway vpc-lattice:DeleteResourcePolicy vpc-lattice:DeleteRule vpc-lattice:DeleteService

服務前綴	動作
	vpc-lattice:DeleteServiceNetwork vpc-lattice:DeleteServiceNetworkResourceAssociation vpc-lattice:DeleteServiceNetworkServiceAssociation vpc-lattice:DeleteServiceNetworkVpcAssociation vpc-lattice:DeleteTargetGroup vpc-lattice:DeregisterTargets vpc-lattice:GetAccessLogSubscription vpc-lattice:GetAuthPolicy vpc-lattice：GetDomainVerification vpc-lattice:GetListener vpc-lattice:GetResourceConfiguration vpc-lattice:GetResourceGateway vpc-lattice:GetResourcePolicy vpc-lattice:GetRule vpc-lattice:GetService vpc-lattice:GetServiceNetwork vpc-lattice:GetServiceNetworkResourceAssociation vpc-lattice:GetServiceNetworkServiceAssociation vpc-lattice:GetServiceNetworkVpcAssociation vpc-lattice:GetTargetGroup vpc-lattice:ListAccessLogSubscriptions

服務前綴	動作
	vpc-lattice：ListDomainVerifications vpc-lattice:ListListeners vpc-lattice:ListResourceConfigurations vpc-lattice:ListResourceEndpointAssociations vpc-lattice:ListResourceGateways vpc-lattice:ListRules vpc-lattice:ListServiceNetworkResourceAssociations vpc-lattice:ListServiceNetworkServiceAssociations vpc-lattice:ListServiceNetworkVpcAssociations vpc-lattice:ListServiceNetworkVpcEndpointAssociations vpc-lattice:ListServiceNetworks vpc-lattice:ListServices vpc-lattice:ListTargetGroups vpc-lattice:ListTargets vpc-lattice:PutAuthPolicy vpc-lattice:PutResourcePolicy vpc-lattice:RegisterTargets vpc-lattice：StartDomainVerification vpc-lattice:UpdateAccessLogSubscription vpc-lattice:UpdateListener vpc-lattice:UpdateResourceConfiguration

服務前綴	動作
	vpc-lattice:UpdateResourceGateway vpc-lattice:UpdateRule vpc-lattice:UpdateService vpc-lattice:UpdateServiceNetwork vpc-lattice:UpdateServiceNetworkVpcAssociation vpc-lattice:UpdateTargetGroup

服務前綴	動作
wafv2	wafv2:AssociateWebACL
	wafv2:CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2:CreateRegexPatternSet
	wafv2:CreateRuleGroup
	wafv2:CreateWebACL
	wafv2:DeleteAPIKey
	wafv2:DeleteFirewallManagerRuleGroups
	wafv2:DeleteIPSet
	wafv2:DeleteLoggingConfiguration
	wafv2:DeletePermissionPolicy
	wafv2:DeleteRegexPatternSet
	wafv2:DeleteRuleGroup
	wafv2:DeleteWebACL
	wafv2:DescribeAllManagedProducts
	wafv2:DescribeManagedProductsByVendor
	wafv2:DescribeManagedRuleGroup
	wafv2:DisassociateWebACL
	wafv2:GenerateMobileSdkReleaseUrl
	wafv2:GetDecryptedAPIKey

服務前綴	動作
	wafv2:GetIPSet
	wafv2:GetLoggingConfiguration
	wafv2:GetManagedRuleSet
	wafv2:GetMobileSdkRelease
	wafv2:GetRateBasedStatementManagedKeys
	wafv2:GetRegexPatternSet
	wafv2：GetRevenueStatistics
	wafv2：GetRevenueStatisticsSummary
	wafv2：GetRevenueStatisticsTimeSeries
	wafv2:GetRuleGroup
	wafv2:GetSampledRequests
	wafv2:GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2:ListAvailableManagedRuleGroupVersions
	wafv2:ListAvailableManagedRuleGroups
	wafv2:ListIPSets
	wafv2:ListLoggingConfigurations
	wafv2:ListManagedRuleSets
	wafv2:ListMobileSdkReleases
	wafv2:ListRegexPatternSets
	wafv2:ListResourcesForWebACL

服務前綴	動作
	wafv2:ListRuleGroups
	wafv2：ListSettlementRecords
	wafv2:ListWebACLs
	wafv2:PutLoggingConfiguration
	wafv2:PutManagedRuleSetVersions
	wafv2:UpdateIPSet
	wafv2:UpdateManagedRuleSetVersionExpiryDate
	wafv2:UpdateRegexPatternSet
	wafv2:UpdateRuleGroup
	wafv2:UpdateWebACL

服務前綴	動作
wellarchitected	wellarchitected:AssociateLenses wellarchitected:AssociateProfiles wellarchitected:CreateLensShare wellarchitected:CreateLensVersion wellarchitected:CreateMilestone wellarchitected:CreateProfile wellarchitected:CreateProfileShare wellarchitected:CreateReviewTemplate wellarchitected:CreateWorkload wellarchitected:CreateWorkloadShare wellarchitected:DeleteLens wellarchitected:DeleteLensShare wellarchitected:DeleteProfile wellarchitected:DeleteProfileShare wellarchitected:DeleteReviewTemplate wellarchitected:DeleteTemplateShare wellarchitected:DeleteWorkload wellarchitected:DeleteWorkloadShare wellarchitected:DisassociateLenses wellarchitected:DisassociateProfiles wellarchitected:ExportLens

服務前綴	動作
	wellarchitected:GetAnswer wellarchitected:GetConsolidatedReport wellarchitected:GetGlobalSettings wellarchitected:GetLens wellarchitected:GetLensReview wellarchitected:GetLensReviewReport wellarchitected:GetLensVersionDifference wellarchitected:GetMilestone wellarchitected:GetProfile wellarchitected:GetProfileTemplate wellarchitected:GetReviewTemplate wellarchitected:GetReviewTemplateAnswer wellarchitected:GetReviewTemplateLensReview wellarchitected:GetWorkload wellarchitected:ImportLens wellarchitected:ListAnswers wellarchitected:ListCheckDetails wellarchitected:ListCheckSummaries wellarchitected:ListLensReviewImprovements wellarchitected:ListLensReviews wellarchitected:ListLensShares

服務前綴	動作
	wellarchitected:ListLenses wellarchitected:ListMilestones wellarchitected:ListNotifications wellarchitected:ListProfileNotifications wellarchitected:ListProfileShares wellarchitected:ListProfiles wellarchitected:ListReviewTemplateAnswers wellarchitected:ListReviewTemplates wellarchitected:ListShareInvitations wellarchitected:ListTemplateShares wellarchitected:ListWorkloadShares wellarchitected:ListWorkloads wellarchitected:UpdateAnswer wellarchitected:UpdateGlobalSettings wellarchitected:UpdateIntegration wellarchitected:UpdateLensReview wellarchitected:UpdateProfile wellarchitected:UpdateReviewTemplate wellarchitected:UpdateReviewTemplateLensReview wellarchitected:UpdateShareInvitation wellarchitected:UpdateWorkload

服務前綴	動作
	wellarchitected:UpdateWorkloadShare wellarchitected:UpgradeLensReview wellarchitected:UpgradeProfileVersion wellarchitected:UpgradeReviewTemplateLensReview

服務前綴	動作
wisdom	wisdom:CreateAssistant wisdom:CreateAssistantAssociation wisdom:CreateContent wisdom:CreateKnowledgeBase wisdom:CreateQuickResponse wisdom:CreateSession wisdom:DeleteAssistant wisdom:DeleteAssistantAssociation wisdom:DeleteContent wisdom:DeleteImportJob wisdom:DeleteKnowledgeBase wisdom:DeleteQuickResponse wisdom:GetAssistant wisdom:GetAssistantAssociation wisdom:GetContent wisdom:GetContentAssociation wisdom:GetContentSummary wisdom:GetImportJob wisdom:GetKnowledgeBase wisdom:GetRecommendations wisdom:GetSession

服務前綴	動作
	wisdom:ListAssistantAssociations
	wisdom:ListAssistants
	wisdom:ListContentAssociations
	wisdom:ListContents
	wisdom:ListImportJobs
	wisdom:ListKnowledgeBases
	wisdom:ListQuickResponses
	wisdom:NotifyRecommendationsReceived
	wisdom:QueryAssistant
	wisdom:RemoveKnowledgeBaseTemplateUri
	wisdom:SearchContent
	wisdom:SearchQuickResponses
	wisdom:SearchSessions
	wisdom:StartContentUpload
	wisdom:StartImportJob
	wisdom:UpdateContent
	wisdom:UpdateKnowledgeBaseTemplateUri
	wisdom:UpdateQuickResponse
	wisdom:UpdateSession

服務前綴	動作
worklink	worklink:AssociateDomain worklink:AssociateWebsiteAuthorizationProvider worklink:AssociateWebsiteCertificateAuthority worklink:CreateFleet worklink:DeleteFleet worklink:DescribeAuditStreamConfiguration worklink:DescribeCompanyNetworkConfiguration worklink:DescribeDevice worklink:DescribeDevicePolicyConfiguration worklink:DescribeDomain worklink:DescribeFleetMetadata worklink:DescribeIdentityProviderConfiguration worklink:DescribeWebsiteCertificateAuthority worklink:DisassociateDomain worklink:DisassociateWebsiteAuthorizationProvider worklink:DisassociateWebsiteCertificateAuthority worklink:ListDevices worklink:ListDomains worklink:ListFleets worklink:ListWebsiteAuthorizationProviders worklink:ListWebsiteCertificateAuthorities

服務前綴	動作
	worklink:RestoreDomainAccess worklink:RevokeDomainAccess worklink:SignOutUser worklink:UpdateAuditStreamConfiguration worklink:UpdateCompanyNetworkConfiguration worklink:UpdateDevicePolicyConfiguration worklink:UpdateDomainMetadata worklink:UpdateFleetMetadata worklink:UpdateIdentityProviderConfiguration

服務前綴	動作
工作區	workspaces:AcceptAccountLinkInvitation workspaces:AssociateConnectionAlias workspaces:AssociateIpGroups workspaces:AssociateWorkspaceApplication workspaces:CopyWorkspaceImage workspaces:CreateAccountLinkInvitation workspaces:CreateConnectClientAddIn workspaces:CreateConnectionAlias workspaces:CreateIpGroup workspaces:CreateStandbyWorkspaces workspaces:CreateUpdatedWorkspaceImage workspaces:CreateWorkspaceBundle workspaces:CreateWorkspaceImage workspaces:CreateWorkspaces workspaces:CreateWorkspacesPool workspaces:DeleteAccountLinkInvitation workspaces:DeleteClientBranding workspaces:DeleteConnectClientAddIn workspaces:DeleteConnectionAlias workspaces:DeleteIpGroup workspaces:DeleteWorkspaceBundle

服務前綴	動作
	workspaces:DeleteWorkspaceImage
	workspaces:DeployWorkspaceApplications
	workspaces:DeregisterWorkspaceDirectory
	workspaces:DescribeAccount
	workspaces:DescribeAccountModifications
	workspaces:DescribeApplicationAssociations
	workspaces:DescribeApplications
	workspaces:DescribeBundleAssociations
	workspaces:DescribeClientBranding
	workspaces:DescribeClientProperties
	workspaces:DescribeConnectClientAddIns
	workspaces:DescribeConnectionAliasPermissions
	workspaces:DescribeConnectionAliases
	workspaces:DescribeCustomWorkspaceImageImport
	workspaces:DescribeImageAssociations
	workspaces:DescribeIpGroups
	workspaces:DescribeWorkspaceAssociations
	workspaces:DescribeWorkspaceBundles
	workspaces:DescribeWorkspaceDirectories
	workspaces:DescribeWorkspaceImagePermissions
	workspaces:DescribeWorkspaceSnapshots

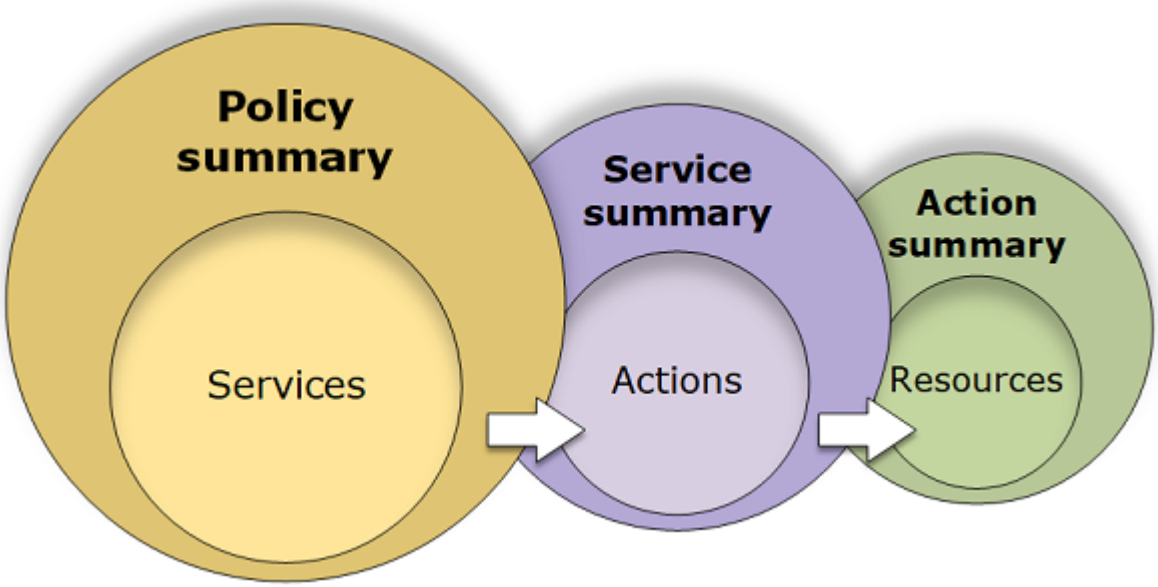
服務前綴	動作
	workspaces:DescribeWorkspaces
	workspaces:DescribeWorkspacesConnectionStatus
	workspaces:DescribeWorkspacesPoolSessions
	workspaces:DescribeWorkspacesPools
	workspaces:DisassociateConnectionAlias
	workspaces:DisassociateIpGroups
	workspaces:DisassociateWorkspaceApplication
	workspaces:GetAccountLink
	workspaces:ImportClientBranding
	workspaces:ImportWorkspaceImage
	workspaces:ListAccountLinks
	workspaces:ListAvailableManagementCidrRanges
	workspaces:MigrateWorkspace
	workspaces:ModifyAccount
	workspaces:ModifyCertificateBasedAuthProperties
	workspaces:ModifyClientProperties
	workspaces:ModifyEndpointEncryptionMode
	workspaces:ModifySamlProperties
	workspaces:ModifySelfservicePermissions
	workspaces:ModifyStreamingProperties
	workspaces:ModifyWorkspaceAccessProperties

服務前綴	動作
	workspaces:ModifyWorkspaceCreationProperties
	workspaces:ModifyWorkspaceProperties
	workspaces:ModifyWorkspaceState
	workspaces:RebootWorkspaces
	workspaces:RebuildWorkspaces
	workspaces:RegisterWorkspaceDirectory
	workspaces:RejectAccountLinkInvitation
	workspaces:RestoreWorkspace
	workspaces:StartWorkspaces
	workspaces:StartWorkspacesPool
	workspaces:StopWorkspaces
	workspaces:StopWorkspacesPool
	workspaces:TerminateWorkspaces
	workspaces:TerminateWorkspacesPool
	workspaces:TerminateWorkspacesPoolSession
	workspaces:UpdateConnectClientAddIn
	workspaces:UpdateConnectionAliasPermission
	workspaces:UpdateWorkspaceBundle
	workspaces:UpdateWorkspaceImagePermission
	workspaces:UpdateWorkspacesPool

服務前綴	動作
xray	xray:CreateGroup xray:CreateSamplingRule xray:DeleteGroup xray:DeleteResourcePolicy xray:DeleteSamplingRule xray:GetEncryptionConfig xray:GetGroup xray:GetGroups xray:GetInsight xray:GetInsightEvents xray:GetInsightImpactGraph xray:GetInsightSummaries xray:GetSamplingRules xray:ListResourcePolicies xray:PutEncryptionConfig xray:PutResourcePolicy xray:UpdateGroup xray:UpdateSamplingRule

政策摘要

IAM 主控台中提供了政策摘要表，這些表總結政策中對每個服務允許或拒絕的存取級別、資源和條件。政策摘要列於三個表中：政策摘要、服務摘要以及動作摘要。政策摘要表中包含服務清單。選擇其中一個服務來查看服務摘要。此摘要表包含所選服務的動作與相關許可的清單。您可以選擇該表中的動作以查看動作摘要。此表格包含所選動作的資源和條件清單。



您可以在 Users (使用者) 頁面或 Roles (角色) 頁面上查看連接到該使用者的所有政策 (受管和內嵌) 的政策摘要。在 Policies (政策) 頁面上檢視所有受管政策的摘要。受管政策包括 AWS 受管政策、受 AWS 管任務職能政策和客戶受管政策。您可以在 Policies (政策) 頁面上查看這些政策的摘要，無論它們是連接到使用者或其他 IAM 身分。

您可以使用政策摘要中的資訊來了解政策允許或拒絕的許可。政策摘要可協助您排除故障並修復未提供您預期的許可的政策。

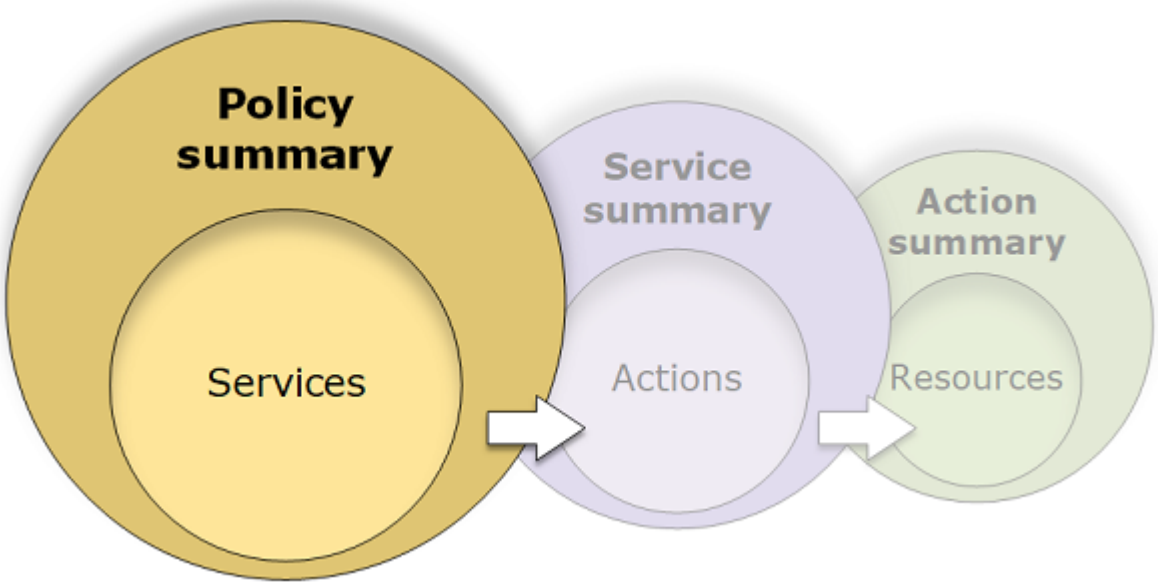
如需檢視摘要的step-by-step說明，請參閱 檢視政策摘要、 檢視服務摘要和 檢視動作摘要。

主題

- 政策摘要 (服務清單)
- 政策摘要中的存取層級
- 服務摘要 (動作清單)
- 動作摘要 (資源清單)
- 政策摘要的範例

政策摘要 (服務清單)

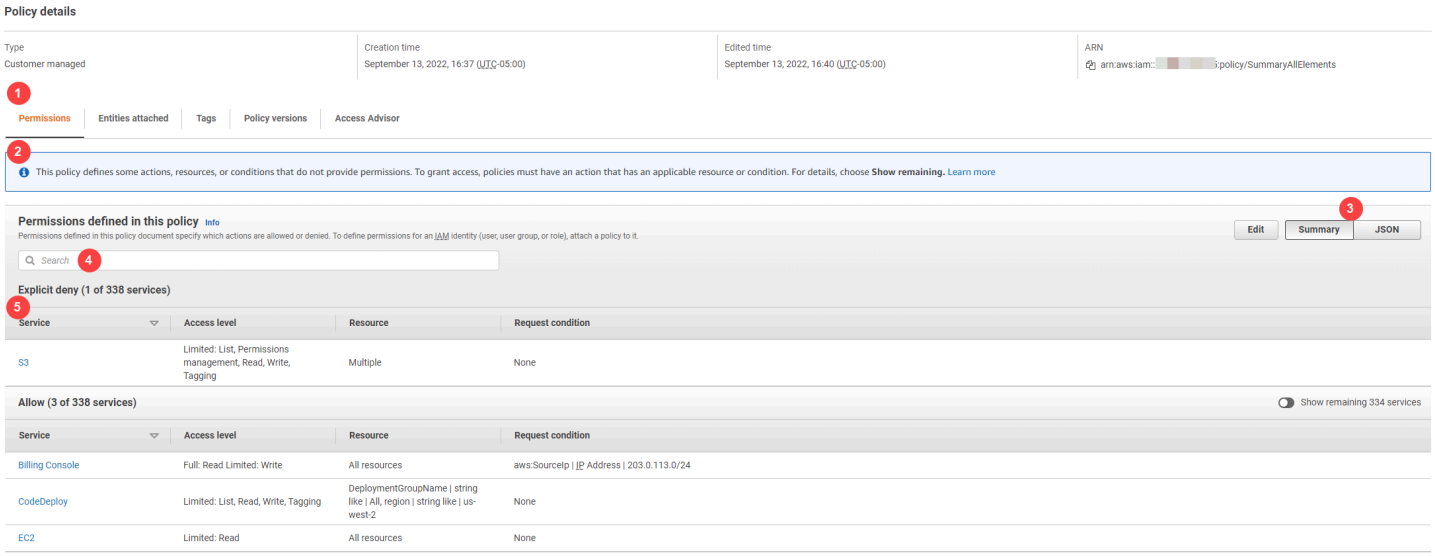
政策摘要列於三個表中：政策摘要、服務摘要以及動作摘要。政策摘要表包括由所選政策定義的服務清單和許可摘要。



政策摘要表格是分成一或多個 Uncategorized services (未分類服務)、Explicit deny (明確拒絕)、以及 Allow (允許) 等部分。如果政策包含 IAM 無法辨識的服務，則該服務將包含在表格的 Uncategorized services (未分類服務) 部分中。如果 IAM 能夠辨識該服務，則它將包含在表格的 Explicit deny (明確拒絕) 或 Allow (允許) 部分中，取決於政策的效果 (`Deny` 或 `Allow`)。

了解政策摘要的元素

在下列政策詳細資訊頁面範例中，SummaryAllElements 政策為直接連接到使用者的受管政策 (客戶管理政策)。此政策已展開並顯示了政策摘要。



在上圖中，政策摘要在政策頁面顯示：

1. 許可索引標籤包括政策中定義的許可。
2. 如果政策未授予許可給政策中定義的所有操作、資源和條件，則將在頁面頂部顯示警告或錯誤橫幅。政策摘要中包含關於問題的詳細資訊。若要了解政策摘可如何協助您了解政策授予的許權並進行相關問題故障排除，請參閱the section called “我的政策未授與預期的許可”.。
3. 使用摘要和 JSON 按鈕，可在政策摘要和 JSON 政策文件之間切換。
4. 使用搜尋方塊，可減少服務清單並查詢特定服務。
5. 展開的視圖顯示了 SummaryAllElements 政策的更多詳細資訊。

下面的政策摘要表圖像顯示了在政策詳細資訊頁面上已展開的 SummaryAllElements 政策。

Explicit deny (1 of 338 services) A

Service B	Access level C	Resource D	Request condition E
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	None

Allow (3 of 338 services) F Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeDeploy	Limited: List, Read, Write, Tagging	DeploymentGroupName | string like | All, region | string like | us-west-2	None
EC2	Limited: Read	All resources	None

在上圖中，政策摘要在政策頁面顯示：

A. 對於 IAM 辨識的那些服務，它根據政策是允許還是明確拒絕使用該服務來安排服務。在此範例中，政策包含用於 Amazon S3 服務的 `Deny` 陳述式，以及用於帳單、CodeDeploy 和 Amazon EC2 服務的 `Allow` 陳述式。

B. Service (服務) – 此欄列出在政策內定義的服務並提供每項服務的詳細資訊。政策摘要表中的每個服務名稱都是指向服務摘要表的一個連結，服務摘要 (動作清單) 中對其進行了說明。在此範例中，為 Amazon S3、帳單、CodeDeploy 和 Amazon EC2 服務定義了許可。

C. 存取層級 – 此欄指出了在每個存取級別中的動作 (`List`、`Read`、`Write`、`Permission Management` 以及 `Tagging`) 是擁有 `Full` 許可還是政策中定義的 `Limited` 許可。有關存取許可級別摘要的更多詳細資訊和範例，請參閱 政策摘要中的存取層級。

- Full access (完整存取) – 此項目顯示服務對於該服務可用的全部四個存取層級中的所有動作都擁有存取許可。
- 如果該項不包含 Full access (完整存取)，則服務可以存取部分但不是全部用於該服務的動作。然後，根據每個存取級別分類 (`List`、`Read`、`Write`、`Permission Management` 以及 `Tagging`) 的說明，來定義存取許可：

  Full (完整)：政策提供對列出的每個存取級別分類中的所有動作的存取許可。在此範例中，政策提供對所有帳單 `Read` 操作的存取許可。

  Limited (限制)：政策提供對所列每個存取級別分類內的一或多個但非全部動作的存取許可。在此範例中，政策提供對部分帳單 `Write` 操作的存取許可。

D. Resource (資源) – 此欄顯示政策為每項服務指定的資源。

- Multiple (多個) – 政策包含服務內的多項資源，但並非全部。在此範例中，明確拒絕對多個 Amazon S3 資源的存取權限。
- 所有資源 – 為服務內的所有資源而定義政策。在此範例中，政策允許對所有帳單資源執行列出的操作。
- 資源文字 – 該政策包含服務內的一個資源。在此範例中，只允許對 `DeploymentGroupName` CodeDeploy 資源執行列出的動作。根據服務提供給 IAM 的資訊，您可能會看到一個 ARN，或者可能會看到定義的資源類型。

> **Note**
>
> 此欄位可能包含不同服務的資源。如果包含資源的政策陳述式，不包含來自相同服務的動作和資源，則您的政策包含不相符的資源。在建立政策或在政策摘要中查看政策時，IAM

不會警告您關於無法配對的資源。如果此欄位包含不相符的資源，則您應該檢閱您的政策錯誤。若要更全面了解您的政策，請一律以政策模擬器來測試。

E. Request condition (請求條件) – 此欄顯示與資源關聯的服務或動作是否受條件約束。

- None (無) – 政策對服務不包含任何條件。在此範例中，沒有條件適用於 Amazon S3 服務中拒絕的操作。
- 條件文字 – 政策包含服務的一項條件。在此範例中，僅當來源的 IP 地址與 `203.0.113.0/24` 匹配時，列出的帳單動作才會獲允許。
- Multiple (多項) – 政策包含服務的多項條件。若要檢視政策多項條件中的每一項條件，請選擇 JSON 來檢視政策文件。

F. 顯示剩餘的服務 – 切換此按鈕可展開資料表，以包括政策未定義的服務。這些服務在該政策中被隱含拒絕 (或根據預設拒絕)。但是，另一政策中的陳述中可能仍然允許或明確拒絕使用該服務。政策摘要匯總了單一政策的許可。若要了解 AWS 服務如何決定是否應允許或拒絕特定請求，請參閱 政策評估邏輯。

當政策或政策中的元素未授予許可時，IAM 在政策摘要中提供額外的警告和資訊。下面的政策摘要表顯示了在 SummaryAllElements 政策詳細資訊頁面上展開的顯示剩餘的服務，並附上可能的警告。

Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

在上圖中，您可以看到包含沒有許可的已定義操作、資源或條件的所有服務：

a. Resource warnings (資源警告) – 對於沒有為所有包含的動作或資源提供許可的服務，您將在資料表的 Resource (資源) 欄中看到以下警告之一：

- 

  未定義資源。– 這表示該服務具有已定義的動作，但政策中不包含支援的資源。

- 

  一或多個動作沒有適用資源。– 表示該服務具有已定義的動作，但其中一些動作沒有支援的資源。

- 

  一或多個資源沒有適用動作。– 表示該服務具有已定義的資源，但其中一些資源沒有支援的動作。

  如果服務同時包含沒有適用資源的動作和沒有適用資源的資源，系統只會顯示一個或多個資源沒有適用的動作警告。這是因為當您查看服務的服務摘要時，不會顯示不適用於任何動作的資源。對於 `ListAllMyBuckets` 動作，此政策包含最後一條警告，因為該動作不支援資源級許可，也不支援 `s3:x-amz-acl` 條件索引鍵。如果您修復資源問題或條件問題，剩餘的問題會出現在詳細的警告中。

b. Request condition warnings (請求條件警告) – 對於沒有為所有包含的條件提供許可的服務，您將在資料表的 Request condition (請求條件) 欄中看到以下警告之一：

- 

  一或多個動作沒有適用條件。– 表示該服務具有已定義的動作，但其中一些動作沒有支援的條件。

- 

  一或多個條件沒有適用動作。– 表示該服務具有已定義的條件，但其中一些條件沒有支援的動作。

c. Multiple (多個) |



一或多個動作沒有適用資源。– Amazon S3 的 Deny 陳述式包含一個以上的資源。它還包含多個動作，但其中一些動作支援資源，一些不支援。若要查看此政策，請參閱 the section called "SummaryAllElements JSON 政策文件"。在這種情況下，政策包含所有 Amazon S3 動作，只會拒絕可在儲存貯體或儲存貯體物件上執行的動作。

d. 

No resources are defined (未定義資源) – 服務具有已定義的動作，但政策中不包含支援的資源，因此服務不提供任何許可。在這種情況下，政策包含 CodeCommit 動作，但沒有 CodeCommit 資源。

e. DeploymentGroupName | 類似的字串 | 全部、區域 | 類似的字串 | us-west-2

|

一個或多個動作沒有適用的資源。– 服務具有定義的動作，且至少有額外一個沒有支援資源的動作。

f. 無 |



一個或多個條件沒有適用的動作。– 服務具有至少一個沒有支援動作的條件索引鍵。

SummaryAllElements JSON 政策文件

SummaryAllElements 政策不適用於在帳戶中定義許可。包含它的目的在於說明您在查看政策摘要時可能會遇到的錯誤和警告。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-
project"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
```

```
                    "other"
                ]
            }
        }
    }
]
}
```

檢視政策摘要

您可以檢視連接到 IAM 使用者或角色的任何政策的政策摘要。對於受管政策，您可以在政策頁面上檢視政策摘要。如果您的政策不包含政策摘要，請參閱 缺少政策摘要 以了解原因。

從政策頁面檢視政策摘要

您可以在 Policies (政策) 頁面上檢視受管政策的政策摘要。

從 Policies (政策) 頁面查看政策摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇您要檢視的政策名稱。
4. 在政策的政策詳細資訊頁面上，檢視許可索引標籤，可查看政策摘要。

檢視連接到使用者的政策的政策摘要

您可以檢視連接到 IAM 使用者的任何政策的政策摘要。

查看連接到使用者的政策摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 從導覽窗格選擇 Users (使用者)。
3. 在使用者清單中，選擇要檢視其政策的使用者的名稱。
4. 在使用者的 Summary (摘要) 頁面上，檢視 Permissions (許可) 標籤來查看直接連接到使用者或從群組連接的政策清單。
5. 在使用者的政策表中，展開您要檢視的政策列。

檢視連接到角色的政策的政策摘要

您可以檢視連接到角色的任何政策的政策摘要。

查看連接到角色的政策摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色)。
3. 在角色清單中，選擇要檢視其政策的角色名稱。
4. 在角色的 Summary (摘要) 頁面上，檢視 Permissions (許可) 標籤來查看直接連接到角色的政策清單。
5. 在角色的政策表中，展開您要檢視的政策列。

編輯政策以修正警告

在查看政策摘要時，您可能會發現拼寫錯誤，或者注意到政策未提供所需的許可。您無法直接編輯政策摘要。不過，您可以使用視覺化政策編輯器編輯客戶託管政策，該編輯器會捕捉政策摘要報告中許多相同的錯誤和警告。然後，您可以在政策摘要中查看更改以確認已修正所有問題。若要了解如何編輯內嵌政策，請參閱 the section called “編輯 IAM 政策”。您無法編輯 AWS 受管政策。

您可以使用視覺化選項來編輯政策摘要的政策。

使用視覺化選項可編輯政策摘要的政策

1. 按照上述步驟中的說明開啟政策摘要。
2. 選擇編輯。

   如果您已打開 Users (使用者) 頁面，並選擇編輯連接到該使用者的客戶受管政策，系統會將您重新引導 Policies (政策) 頁面。您只能在 Policies (政策) 頁面上編輯客戶受管政策。
3. 選擇視覺化選項可檢視您的政策之可編輯視覺化形式。IAM 可能會調整您的政策結構以針對視覺化編輯器進行最佳化，並使您更輕鬆地查詢和解決任何問題。頁面上的警告和錯誤訊息可以引導您解決政策中的任何問題。如需有關 IAM 重新建構這些政策的詳細資訊，請參閱 政策結構調整。
4. 編輯您的政策，然後選擇下一步，可查看已在政策摘要中反映的變更。如果仍出現問題，請選擇 Previous (上一步) 以返回到編輯螢幕。
5. 選擇儲存變更，以儲存您所做的變更。

您可以使用 JSON 選項來編輯政策摘要的政策。

若要使用 JSON 選項編輯政策摘要的政策

1. 按照上述步驟中的說明開啟政策摘要。
2. 您可以使用摘要和 JSON 按鈕，將政策摘要與 JSON 政策文件進行比較。您可以使用該資訊來決定要更改政策文件中的哪些行。
3. 選擇編輯，然後選擇 JSON 選項，以編輯 JSON 政策文件。

> **Note**
>
> 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器選項中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

如果您已打開 Users (使用者) 頁面，並選擇編輯連接到該使用者的客戶受管政策，系統會將您重新引導 Policies (政策) 頁面。您只能在 Policies (政策) 頁面上編輯客戶受管政策。

4. 編輯政策。解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。如果仍出現問題，請選擇 Previous (上一步) 以返回到編輯螢幕。
5. 選擇儲存變更，以儲存您所做的變更。

政策摘要中的存取層級

AWS 存取層級摘要

政策摘要包括說明針對政策中所提及的每項服務定義的動作許可的存取層級摘要。若要進一步了解政策摘要，請參閱政策摘要。從存取層級摘要可知政策中是否已為每個存取層級中的動作 (`List`、`Read`、`Tagging`、`Write` 和 `Permissions management`) 定義 `Full` 或 `Limited` 許可。若要檢視指派給服務中每個動作的存取層級分類，請參閱 AWS 服務的動作、資源和條件金鑰。

以下範例說明針對特定服務政策所提供的存取權。如需完整的 JSON 政策文件及其相關摘要的範例，請參閱政策摘要的範例。

服務	存取層級	此政策提供下列
IAM	完整存取	存取 IAM 服務內的所有動作

服務	存取層級	此政策提供下列
CloudWatch	完整：清單	存取 List 存取層級中的所有 CloudWatch 動作，但是無法存取具 Read、Write 或 Permissions management 存取層級分類的動作。
Data Pipeline	有限：清單、讀取	存取 List和 存取Read層級中至少一個但並非所有 AWS Data Pipeline 動作，但無法存取 Write或 Permissions management 動作。
EC2	完整：清單、讀取有限：寫入	存取所有 Amazon EC2 List 和 Read 動作，以及存取至少一個但並非所有 Amazon EC2 Write 動作，但是無法存取具 Permissions management 存取層級分類的動作。
S3	有限：讀取、寫入、許可管理	存取至少一個但並非所有 Amazon S3 Read、Write 和 Permissions management 動作。
codedploy	(空白)	未知存取，因為 IAM 無法辨識此服務。
API Gateway	無	政策中未定義任何存取權。
CodeBuild	⚠ 未定義任何動作。	無法存取，因為未針對服務定義任何動作。若要了解如何了解和故障排除此問題，請參閱the section called “我的政策未授與預期的許可”。

在政策摘要中，完整存取表示政策提供服務內所有動作的存取權。政策提供存取服務內部分但非所有動作，並會進一步根據存取層級分類分組。這是根據下列存取層級的群組指出：

- Full (完整)：政策提供對指定的存取層級分類中的所有動作的存取權。
- Limited (有限)：政策提供指定存取層級分類內的一或多個但非全部動作的存取權。
- None (無)：政策不提供存取權。

- (空)：IAM 無法辨識此服務。如果服務名稱包含錯別字，則政策不提供服務的存取權。如果服務名稱正確，則該服務可能不支援政策摘要，或者可能處於預覽狀態。在這種情況下，政策可能會提供存取權，但無法顯示在政策摘要內。若請求全面供應 (GA) 服務的政策摘要支援，請參閱 服務不支援 IAM 政策摘要。

包含有限 （ 部分 ） 動作存取的存取層級摘要會使用 AWS 存取層級分類 `List`、`Read`、`Tagging`、`Write`或 進行分組`Permissions management`。

AWS 存取層級

AWS 會為服務中的動作定義下列存取層級分類：

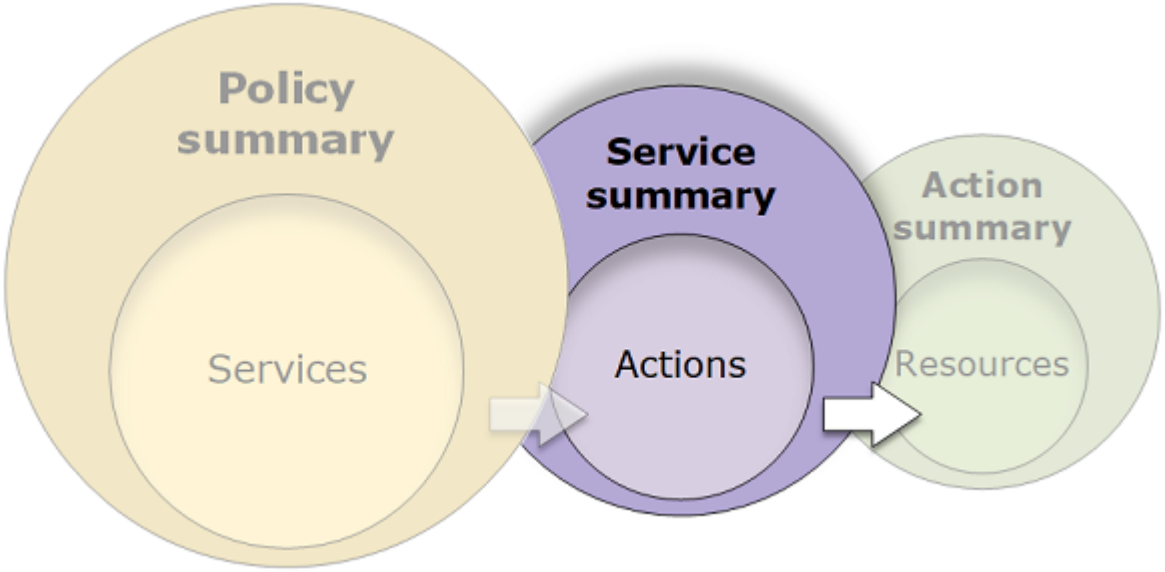
- List (清單)：列出服務內資源的許可，以判斷物件是否存在。具有此層級存取權的動作，可以列出物件，但無法查看資源的內容。例如，Amazon S3 動作 `ListBucket` 具有 List (清單) 存取層級。
- Read (讀取)：可讀取但無法編輯服務內資源的內容和屬性的許可。例如，Amazon S3 動作 `GetObject` 和 `GetBucketLocation` 具有 Read (讀取) 存取層級。
- Tagging (標記)：執行僅變更資源標籤狀態之動作的許可。例如，IAM 動作 `TagRole` 及 `UntagRole` 具有 Tagging (標記) 存取層級，因為它們只允許標記或取消標記角色。不過，當您建立該角色時，`CreateRole` 動作允許標記角色資源。由於動作不會僅新增標籤，所以它具有 `Write` 存取層級。
- Write (寫入)：可建立、刪除或修改服務內資源的許可。例如，Amazon S3 動作 `CreateBucket`、`DeleteBucket` 及 `PutObject` 具有 Write (寫入) 存取層級。`Write` 動作可能也允許修改資源標籤。不過，動作僅允許變更具有 `Tagging` 存取層級的標籤。
- 許可管理：許可管理是指控制內部存取的動作 AWS 服務，包括 IAM 和非 IAM 身分許可，但不包括安全群組等網路層級存取控制。例如，大多數 IAM 和 AWS Organizations 動作，以及 Amazon S3 動作`PutBucketPolicy`，且`DeleteBucketPolicy`具有許可管理存取層級。

> **秘訣**
>
> 為了改善 的安全性 AWS 帳戶，請限制或定期監控包含許可管理存取層級分類的政策。

若要檢視服務中所有動作的存取層級分類，請參閱 AWS 服務的動作、資源和條件金鑰。

服務摘要 (動作清單)

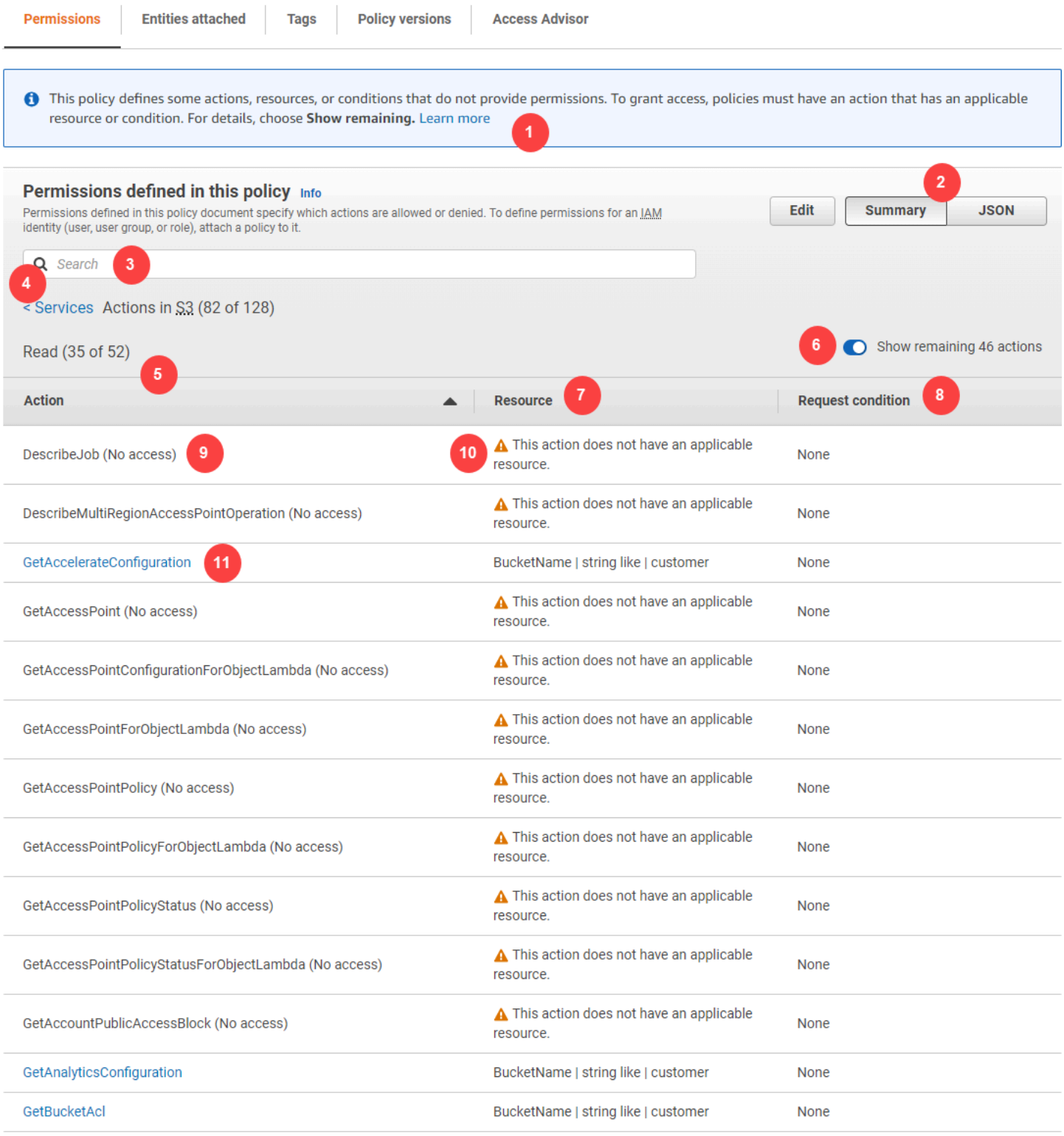
政策摘要列於三個表中：政策摘要、服務摘要以及動作摘要。服務摘要表包括動作清單和由政策針對所選服務定義的許可摘要。



您可以檢視授予許可的政策摘要中，所列的每個服務的服務摘要。該表已分組為 Uncategorized actions (未分類的動作)、Uncategorized resource types (未分類的資源類型) 和存取層級區段。如果政策包含 IAM 無法辨識的動作，則該動作將包含在資料表的 Uncategorized actions (未分類的動作) 區段中。如果 IAM 識別出動作，則該動作會包含在資料表的其中一個存取層級 (列出、讀取、寫入和許可管理) 區段下。若要檢視指派給服務中每個動作的存取層級分類，請參閱 AWS 服務的動作、資源和條件金鑰。

了解服務摘要的元素

以下範例是從政策摘要中允許的 Amazon S3 動作之服務摘要。此服務的動作會按照存取層級分組。例如，在可供服務使用的總計 52 個讀取動作中，有 35 個讀取動作已定義。



受管政策的服務摘要頁面包含以下資訊：

1. 如果政策未授予許可給政策中為服務定義的所有操作、資源和條件，則將在頁面頂部顯示警告橫幅。服務摘要中包含關於問題的詳細資訊。若要了解政策摘可如何協助您了解政策授予的許權並進行相關問題故障排除，請參閱the section called “我的政策未授與預期的許可”.。
2. 若要查看政策的其他詳細資訊，請選擇 JSON。您可以執行此操作，來查看套用到動作的所有條件。(如果您正在檢視直接連接到使用者的內嵌政策的服務摘要，您必須關閉服務摘要對話方塊並返回政策摘要，以存取 JSON 政策文件)。
3. 若要檢視特定動作的摘要，請在搜尋方塊中輸入關鍵字，以縮短可用動作清單。
4. 在服務返回箭頭旁邊，將顯示服務的名稱 (在此案例中，為 S3)。這項服務的服務摘要包含政策中定義的允許或拒絕之動作清單。如果服務出現在許可索引標籤上的 (明確拒絕) 下，系統會明確拒絕服務摘要表中列出的動作。如果服務出現在許可索引標籤上的允許下，系統會允許服務摘要表中列出的動作。
5. 動作 – 此欄位會列出在政策中定義的動作，並提供每個動作的資源和條件。如果政策授予或拒絕授予許可給動作，動作名稱會連結到動作摘要資料表。表格會根據政策允許或拒絕的存取層級，將這些動作分組到至少一個或多達五個區段。區段為列出、讀取、寫入、許可管理以及標記。計數表示在每個存取層級內提供許可的已認可動作。總計是服務的已知動作數。在這個範例中，在總計 52 個已知 Amazon S3 讀取動作中，有 35 個動作提供許可。若要檢視指派給服務中每個動作的存取層級分類，請參閱 AWS 服務的動作、資源和條件金鑰。
6. 顯示剩餘的動作 – 切換此按鈕可展開或隱藏資料表，以包含已知但不提供此服務的許可之動作。切換該按鈕也會針對不提供許可的任何元素顯示警告。
7. Resource (資源) – 此欄位顯示政策為服務定義的資源。IAM 不會檢查資源是否適用於每個動作。在此範例中，Amazon S3 服務中的動作只允許對 `developer_bucket` Amazon S3 儲存貯體資源執行。根據服務提供給 IAM 的資訊，您可能會看到一個 ARN (例如 `arn:aws:s3:::developer_bucket/*`)，或者您可能會看到定義的資源類型 (例如 `BucketName = developer_bucket`)。

   > **Note**
   >
   > 此欄位可能包含不同服務的資源。如果包含資源的政策陳述式，不包含來自相同服務的動作和資源，則您的政策包含不相符的資源。在建立政策或在服務摘要中查看政策時，IAM 不會警告您關於不相符的資源。IAM 也不會指出動作是否適用於資源或者服務是否相符。如果此欄位包含不相符的資源，則您應該檢閱您的政策錯誤。若要更全面了解您的政策，請一律以政策模擬器來測試。

8. Request condition (請求條件) – 此欄指出與資源關聯的動作是否受條件約束。若要進一步了解相關條件，請選擇 JSON 以檢視 JSON 政策文件。
9. (No access) (沒有存取) – 此政策包含不提供許可的動作。

10Resource warning (資源警告) – 對於不提供完整許可的資源動作，請參閱以下其中一個警告：

- This action does not support resource-level permissions. This requires a wildcard (*) for the resource. (此動作不支援資源層級的許可，這需要對資源使用萬用字元 (*))。– 這表示政策包含資源層級許可，但必須包含 `"Resource": ["*"]` 才能提供此動作的許可。
- This action does not have an applicable resource. (此動作沒有適用的資源)。– 這表示動作包含在政策內，而無支援的資源。
- This action does not have an applicable resource and condition. (此動作沒有適用的資源和條件)。– 這表示動作包含在政策內，而無支援的資源，也無支援的條件。在這種情況下，此服務的政策內也包含條件，但沒有適用此動作的條件。

11提供許可的動作包含連到動作摘要的連結。

檢視服務摘要

您可以檢視授予許可的政策摘要中，所列的每個服務的服務摘要。

從政策頁面檢視服務摘要

您可以在政策頁面上檢視受管政策的動作摘要。

若要檢視受管政策的服務摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇您要檢視的政策名稱。
4. 在政策的政策詳細資訊頁面上，檢視許可索引標籤，可查看政策摘要。
5. 在服務的政策摘要清單中，選擇您要檢視的服務名稱。

檢視連接到使用者的政策的服務摘要

您可以檢視連接到 IAM 使用者的任何政策的服務摘要。

若要檢視連接到使用者的政策服務摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Users (使用者)。
3. 在使用者清單中，選擇要檢視其政策的使用者的名稱。

4. 在使用者的 Summary (摘要) 頁面上，檢視 Permissions (許可) 標籤來查看直接連接到使用者或從群組連接的政策清單。
5. 在使用者的政策表中，選擇您要檢視的政策名稱。

   如果您已打開使用者頁面，並選擇檢視連接到該使用者的政策服務摘要，系統會將您重新引導至政策頁面。您只能在政策頁面上檢視服務摘要。
6. 選擇摘要。在服務的政策摘要清單中，選擇您要檢視的服務名稱。

> **Note**
>
> 如果您選擇的政策是直接連接到使用者的內嵌政策，就會顯示服務摘要表。如果政策是從群組連接過來的內嵌政策，您會被帶引到 JSON 政策文件中的該群組。如果政策是受管政策，那麼您會被帶引到 Policies (政策) 頁面上該政策的服務摘要。

檢視連接到角色的政策的服務摘要

您可以檢視連接到角色的任何政策的政策摘要。

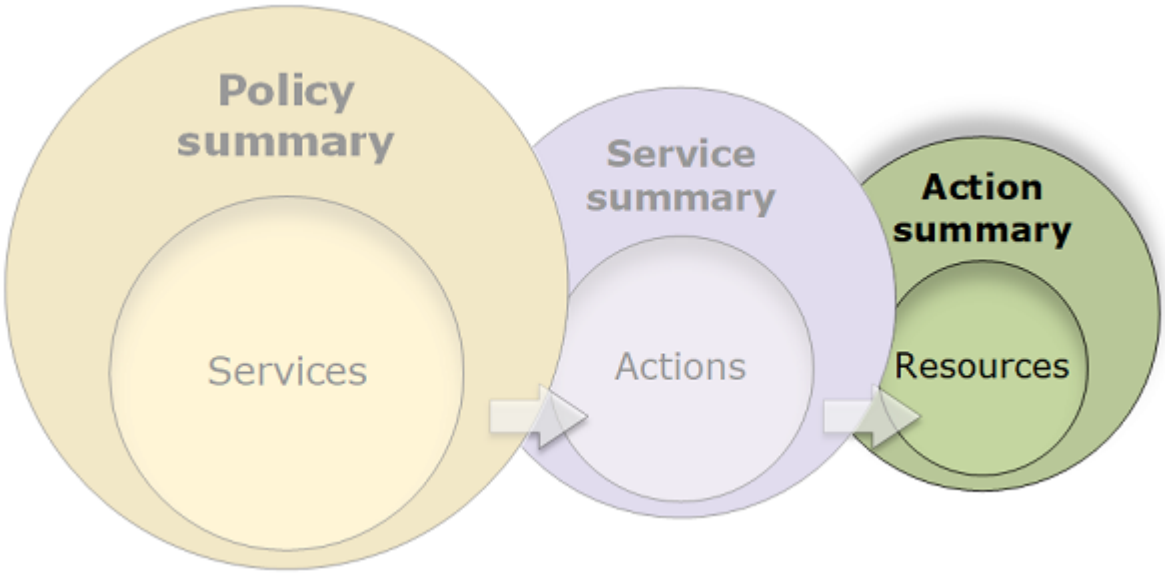
若要檢視連接到角色的政策服務摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 從導覽窗格選擇 Roles (角色)。
3. 在角色清單中，選擇要檢視其政策的角色名稱。
4. 在角色的 Summary (摘要) 頁面上，檢視 Permissions (許可) 標籤來查看直接連接到角色的政策清單。
5. 在角色的政策表中，選擇您要檢視的政策名稱。

   如果您已打開角色頁面，並選擇檢視連接到該使用者的政策服務摘要，系統會將您重新引導至政策頁面。您只能在政策頁面上檢視服務摘要。
6. 在服務的政策摘要清單中，選擇您要檢視的服務名稱。

動作摘要 (資源清單)

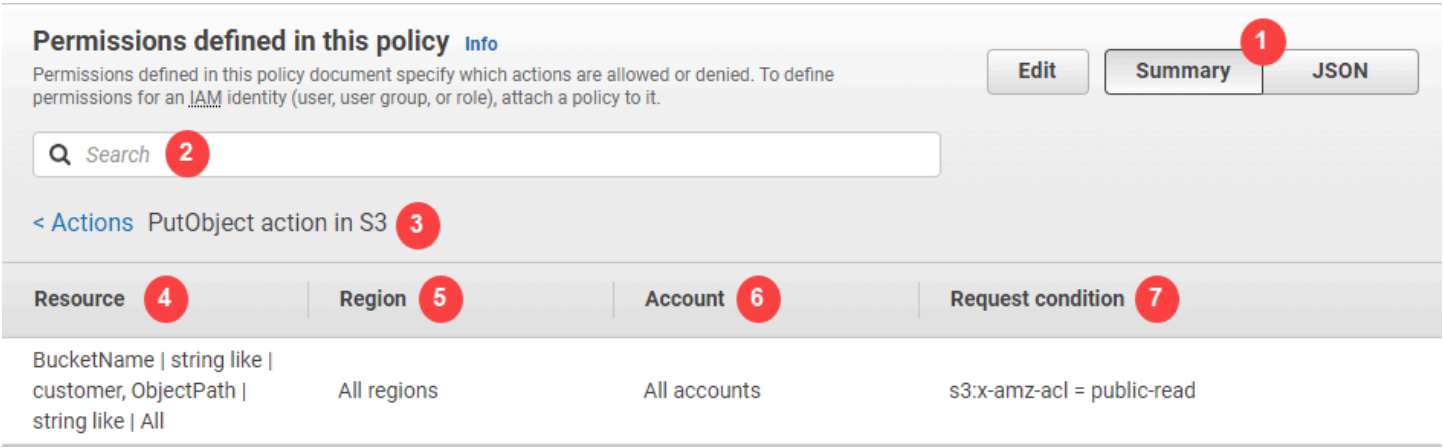
政策摘要列於三個表中：政策摘要、服務摘要以及動作摘要。動作摘要表中包含一份資源清單，以及適用於所選動作的關聯條件。



若要檢視每個授予許可的動作摘要，請選擇服務摘要中的連結。動作摘要表包含關於資源的詳細資訊，包括其 Region (區域) 和 Account (帳戶)。您也可以檢視套用到各個資源的條件。這會顯示適用某些資源，但不適用其他資源的條件。

了解動作摘要的元素

以下範例是來自 Amazon S3 服務摘要的 `PutObject` (寫入) 動作的動作摘要 (請參閱 服務摘要 (動作清單))。對於此動作，政策在單一資源定義多個條件。



動作摘要頁面包含以下資訊：

1. 選擇 JSON，可查看其他有關政策的詳細資訊，例如檢視套用到動作的多個條件。(如果您正在檢視直接連接到使用者的內嵌政策的動作摘要，則步驟有所不同。在這種情況下，您必須關閉動作摘要對話方塊並返回政策摘要，才能存取 JSON 政策文件)。

2. 若要檢視特定資源的摘要，請在搜尋方塊中輸入關鍵字，以減少可用資源清單。
3. 在動作返回箭頭旁邊，將以格式 `action name action in service` 顯示服務和動作的名稱 (在此案例中為 S3 中的 PutObject 動作)。這項服務的動作摘要包含政策中定義的資源清單。
4. Resource (資源) – 此欄列示政策為所選服務定義的資源。在此範例中，所有物件路徑都允許 PutObject 動作，但只有 `developer_bucket` Amazon S3 儲存貯體資源除外。根據服務提供給 IAM 的資訊，您可能會看到一個 ARN (例如 `arn:aws:s3:::developer_bucket/*`)，或者您可能會看到定義的資源類型 (例如 `BucketName = developer_bucket, ObjectPath = All`)。
5. Region (區域) – 此欄顯示定義資源的區域。可針對所有區域或單一區域定義資源。它們無法存在於一個以上的特定區域。
   - 所有區域 – 與資源關聯的動作適用於所有區域。在這個範例中，動作屬於全球服務 Amazon S3。屬於全球服務的動作適用於所有區域。
   - Region text (區域文字) – 與資源關聯的動作適用於一個區域。例如，政策可以指定資源的 `us-east-2` 區域。
6. Account (帳戶) – 此欄顯示與資源關聯的服務或動作是否套用於特定帳戶。資源可以存在於所有帳戶或單一帳戶。它們無法存在於一個以上的特定帳戶。
   - All accounts (所有帳戶) – 與資源關聯的動作適用於所有帳戶。在這個範例中，動作屬於全球服務 Amazon S3。屬於全球服務的動作適用於所有帳戶。
   - 這個帳戶 – 與資源關聯的動作只適用於目前的帳戶。
   - Account number (帳戶編號) – 與資源關聯的動作，套用於一個帳戶 (您目前未登入的帳戶)。例如，如果政策指定 `123456789012` 帳戶供資源使用，則帳號會出現在政策摘要中。
7. Request condition (請求條件) – 此欄顯示與資源關聯的動作是否受條件約束。此範例包含 `s3:x-amz-acl = public-read` 條件。若要進一步了解相關條件，請選擇 JSON 以檢視 JSON 政策文件。

檢視動作摘要

您可以檢視授予許可的政策摘要中所列的每個動作的動作摘要。

從政策頁面檢視動作摘要

您可以檢視受管政策的動作摘要。

檢視受管政策的動作摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。

2. 在導覽窗格中，選擇 Policies (政策)。
3. 在政策清單中，選擇您要檢視的政策名稱。
4. 在政策的政策詳細資訊頁面上，檢視許可索引標籤，可查看政策摘要。
5. 在服務的政策摘要清單中，選擇您要檢視的服務名稱。
6. 在動作的服務摘要清單中，選擇您要檢視的動作名稱。

檢視連接到使用者的政策的動作摘要

您可以檢視連接到使用者的任何政策的動作摘要。

檢視連接到使用者的政策動作摘要

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 從導覽窗格選擇 Users (使用者)。
3. 在使用者清單中，選擇要檢視其政策的使用者的名稱。
4. 在使用者的 Summary (摘要) 頁面上，檢視 Permissions (許可) 標籤來查看直接連接到使用者或從群組連接的政策清單。
5. 在使用者的政策表中，選擇您要檢視的政策名稱。

   如果您已打開使用者頁面，並選擇檢視連接到該使用者的政策服務摘要，系統會將您重新引導至政策頁面。您只能在政策頁面上檢視服務摘要。
6. 在服務的政策摘要清單中，選擇您要檢視的服務名稱。

   > **Note**
   >
   > 如果您選擇的政策是直接連接到使用者的內嵌政策，就會顯示服務摘要表。如果政策是從群組連接過來的內嵌政策，您會被帶引到 JSON 政策文件中的該群組。如果政策是受管政策，那麼您會被帶引到 Policies (政策) 頁面上該政策的服務摘要。
7. 在動作的服務摘要清單中，選擇您要檢視的動作名稱。

檢視連接到角色的政策的動作摘要

您可以檢視連接到角色的任何政策的動作摘要。

檢視連接到角色的政策動作摘要

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色)。
3. 在角色清單中，選擇要檢視其政策的角色名稱。
4. 在角色的 Summary (摘要) 頁面上，檢視 Permissions (許可) 標籤來查看直接連接到角色的政策清單。
5. 在角色的政策表中，選擇您要檢視的政策名稱。

   如果您已打開角色頁面，並選擇檢視連接到該使用者的政策服務摘要，系統會將您重新引導至政策頁面。您只能在政策頁面上檢視服務摘要。
6. 在服務的政策摘要清單中，選擇您要檢視的服務名稱。
7. 在動作的服務摘要清單中，選擇您要檢視的動作名稱。

政策摘要的範例

以下範例包括將 JSON 政策加入關聯的政策摘要、服務摘要、以及動作摘要，以協助您了解透過政策提供的許可。

政策 1：DenyCustomerBucket

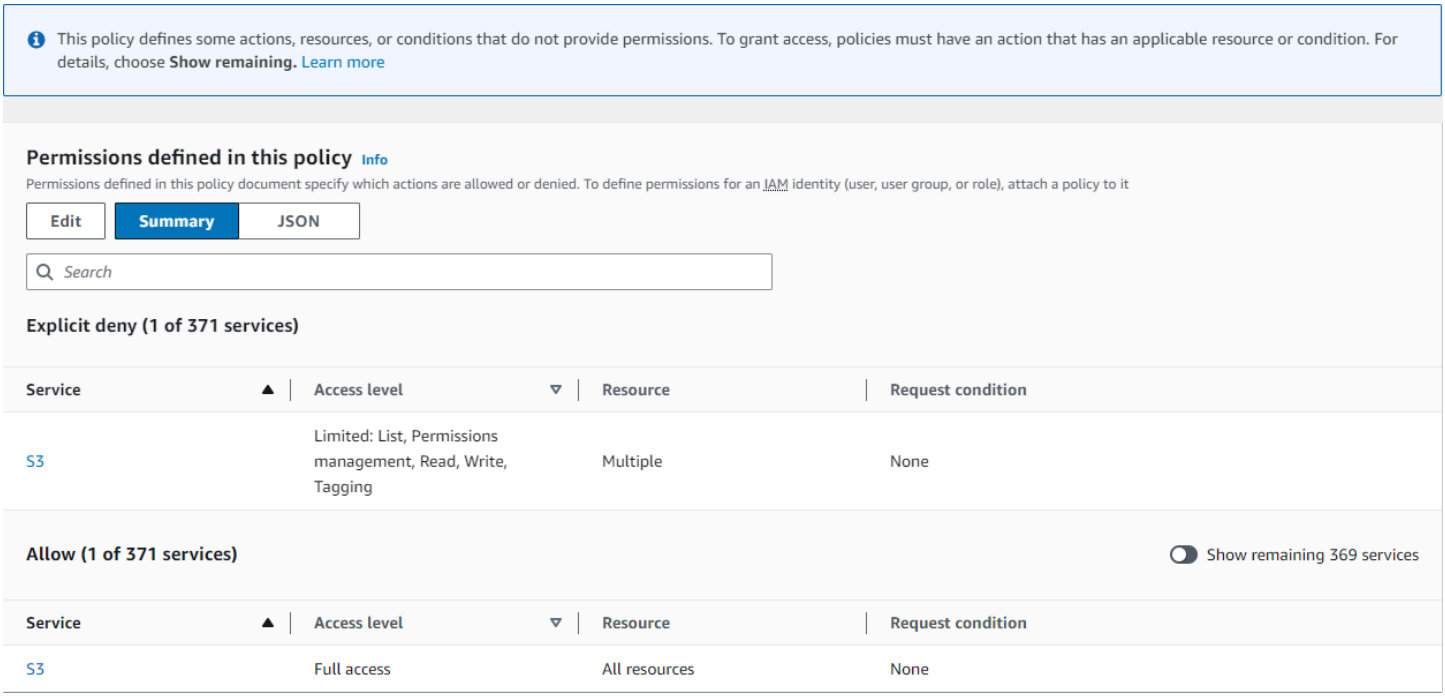
此政策示範對相同服務的允許和拒絕。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
            "Resource": ["*"]
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
```

```
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

DenyCustomerBucket 政策摘要：



DenyCustomerBucket S3 (明確拒絕) 服務摘要：

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

GetObject (讀取) 動作摘要：

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

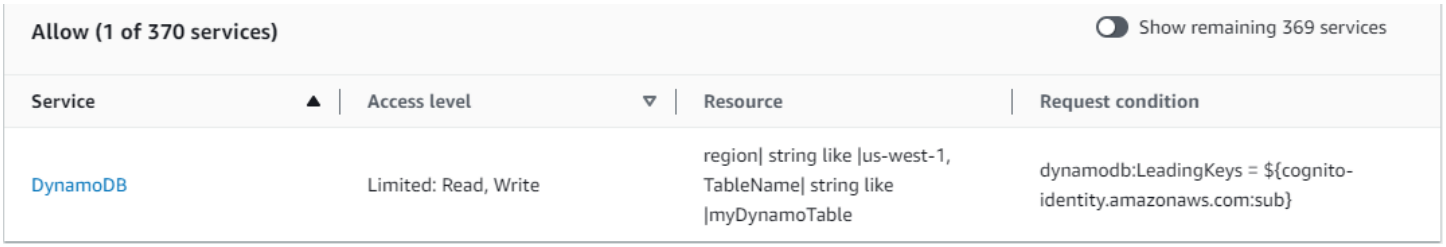
政策 2：DynamoDbRowCognitoID

此政策根據使用者的 Amazon Cognito ID 提供對 Amazon DynamoDB 的低層級存取權限。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

DynamoDbRowCognitoID 政策摘要：



DynamoDbRowCognitoID DynamoDB (允許) 服務摘要：

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

GetItem (清單) 動作摘要：

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

政策 3：MultipleResourceCondition

此政策包含多個資源和條件。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

MultipleResourceCondition 政策摘要：

Allow (1 of 370 services) Show remaining 369 services

Service	Access level	Resource	Request condition
S3	Limited: Permissions management, Write	Multiple	Multiple

MultipleResourceCondition S3 (允許) 服務摘要：

< Services Actions in S3 (2 of 130) Show remaining 128 actions

Write (1 of 47)

Action	Resource	Request condition
PutObject	Multiple	Multiple

Permission Management (1 of 15)

Action	Resource	Request condition
PutObjectAcl	Multiple	Multiple

PutObject (寫入) 動作摘要：

< Actions PutObject action in S3

Resource	Region	Account	Request condition
Multiple	-	All accounts	Multiple

政策 4：EC2_troubleshoot

以下政策可讓使用者取得執行中的 Amazon EC2 執行個體螢幕截圖，可協助執行 EC2 故障排除。此政策也允許檢視 Amazon S3 開發人員儲存貯體中項目的相關資訊。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

EC2_Troubleshoot 政策摘要：

Allow (2 of 370 services) — Show remaining 368 services

Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName| string like |developer	None

EC2_Troubleshoot S3 (允許) 服務摘要：

< Services Actions in S3 (1 of 130) Show remaining 129 actions

List (1 of 7)

Action	Resource	Request condition
ListBucket	BucketName| string like |developer	None

ListBucket (清單) 動作摘要：

< Actions ListBucket action in S3

Resource	Region	Account	Request condition
BucketName| string like |developer	-	All accounts	None

政策 5：CodeBuild_CodeCommit_CodeDeploy

此政策提供對特定 CodeBuild、CodeCommit 以及 CodeDeploy 的存取。由於這些資源對於每個服務來說都是專有的，因此只會在對應服務中顯示。如果您在 `Action` 元素中加入不符合任何服務的資源，那麼資源會顯示在所有動作摘要中。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-
project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-
east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
            ]
        }
```

```
    ]
}
```

CodeBuild_CodeCommit_CodeDeploy 政策摘要：

Allow (3 of 370 services) Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

CodeBuild_CodeCommit_CodeDeploy CodeBuild (允許) 服務摘要：

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

CodeBuild_CodeCommit_CodeDeploy StartBuild (寫入) 動作摘要：

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

存取 IAM 資源所需的許可

資源是服務中的物件。IAM 資源包括群組、使用者、角色和政策。如果您使用 AWS 帳戶根使用者 登入資料登入，則沒有管理 IAM 登入資料或 IAM 資源的限制。但是，必須明確授予 IAM 使用者管理憑證或 IAM 資源的許可。您可以透過將以身分為基礎的政策連接到使用者來執行此操作。

> **Note**
>
> 在整個 AWS 文件中，當我們參考 IAM 政策而不提及任何特定類別時，我們指的是以身分為基礎的客戶受管政策。如需政策類別的詳細資訊，請參閱the section called “政策和許可”。

管理 IAM 身分的許可

管理 IAM 群組、使用者、角色和憑證所需的許可通常對應於任務的 API 動作。例如，若要建立 IAM 使用者，您必須擁有具有對應的 API 命令的 `iam:CreateUser` 許可：`CreateUser`。若要允許 IAM 使用者建立其他 IAM 使用者，您可以將 IAM 政策 (如下所示) 連接到該使用者：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

在政策中，`Resource` 元素的值取決於動作以及動作可以影響的資源。在上述範例中，政策可讓使用者建立任何使用者 (`*` 是符合所有字串的萬用字元)。反之，允許使用者僅更改自己的存取金鑰 (API 動作 `CreateAccessKey` 和 `UpdateAccessKey`) 的政策通常具有 `Resource` 元素。在這種情況下，ARN 包含一個變數 (`${aws:username}`)，其解析至目前使用者名稱，如以下範例所示：

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

在上述範例中，${aws:username} 是一個解析為目前使用者的使用者名稱的變數。如需有關政策變數的詳細資訊，請參閱 IAM 政策元素：變數與標籤。

在動作名稱中使用萬用字元 (*) 通常可讓您更輕鬆地授予許可給與特定任務相關的所有動作。例如，若要允許使用者執行任何 IAM 動作，您可以使用 iam:* 執行動作。為了允許使用者能夠執行只與存取金鑰相關的任何動作，您可以在政策陳述式的 iam:*AccessKey* 元素中使用 Action。如此可提供使用者許可來執行 CreateAccessKey、DeleteAccessKey、GetAccessKeyLastUsed、ListAccessKeys 和 UpdateAccessKey 動作。(如果動作新增到未來具有「AccessKey」的 IAM 中，則對 iam:*AccessKey* 元素使用 Action 也將授予使用者對該新動作的許可)。下列範例顯示的政策可讓使用者執行與其自己的存取金鑰相關的所有動作 (*account-id*以您的 AWS 帳戶 ID 取代）：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
```

```
        "Resource": "arn:aws:iam::111122223333:user/${aws:username}"
    }
}
```

有些任務 (例如刪除群組) 涉及多個動作：您必須先從群組中移除使用者，然後分開或刪除群組的政策，然後實際刪除該群組。如果您希望使用者能夠刪除群組，則必須確保授予使用者執行所有相關動作的許可。

在 中工作的許可 AWS 管理主控台

上述範例顯示允許使用者使用 AWS CLI 或 AWS 開發套件執行動作的政策。

當使用者使用主控台，主控台會向 IAM 發出請求，以列出群組、使用者、角色和政策，並獲得與群組、使用者或角色關聯的政策。主控台也會發出請求，以取得委託人的相關資訊 AWS 帳戶 和資訊。主體是使用者在主控台中提出請求。

一般而言，要執行動作，您必須擁有只包含在政策中的符合動作。要建立使用者，您需要有呼叫 `CreateUser` 動作的許可。通常，當您使用主控台來執行動作時，您必須具有在主控台中顯示、列出、取得或以其他方式查看資源的許可。這是必要的，以便您可以在主控台中導覽以執行指定的動作。例如，如果使用者 Jorge 想要使用主控台來變更自己的存取金鑰，他會進入 IAM 主控台並選擇 Users (使用者)。此動作會導致主控台發出 `ListUsers` 請求。如果 Jorge 沒有 `iam:ListUsers` 動作的許可，則在嘗試列出使用者時，主控台將被拒絕存取。因此，Jorge 無法取得自己的名稱和自己的存取金鑰，即使他擁有 `CreateAccessKey` 和 `UpdateAccessKey` 動作的許可。

如果您想要授予使用者使用 管理群組、使用者、角色、政策和登入資料的許可 AWS 管理主控台，您需要包含主控台執行之動作的許可。如需可以用來授予使用者許可的一些政策範例，請參閱管理 IAM 資源的政策範例。

跨 AWS 帳戶授予許可

您可以直接向自己帳戶中的 IAM 使用者授予對您的資源的存取權限。如果其他帳戶中的使用者需要存取您的資源，您可以建立 IAM 角色，該角色是一個擁有許可的實體，但不與特定使用者相關聯。然後，其他帳戶中的使用者可以使用該角色，並根據您為該角色分配的許可存取資源。如需詳細資訊，請參閱 在您擁有的另一個 IAM 使用者中 AWS 帳戶 存取。

> **Note**
>
> 部分服務支援以資源為基礎的政策，如 以身分為基礎和以資源為基礎的政策 (例如 Amazon S3、Amazon SNS 和 Amazon SQS) 中所述。對於這些服務，使用角色的替代方法是將政策連

接到您要共用的資源 (儲存貯體、主題或佇列)。以資源為基礎的政策可以指定具有資源存取許可 AWS 的帳戶。

由一個服務來存取另一個服務的許可

許多 AWS 服務會存取其他 AWS 服務。例如，幾個 AWS 服務 (包括 Amazon EMR、Elastic Load Balancing 和 Amazon EC2 Auto Scaling) 管理 Amazon EC2 執行個體。 AWS 其他服務會使用 Amazon S3 儲存貯體、Amazon SNS 主題、Amazon SQS 佇列等。

在這些情況下管理許可的方案因服務而異。以下是一些如何處理不同服務許可的範例：

- 在 Amazon EC2 Auto Scaling 中，使用者必須具有使用 Auto Scaling 的許可，但不需要明確授予管理 Amazon EC2 執行個體的許可。
- 在 中 AWS Data Pipeline，IAM 角色會決定管道可以執行的動作；使用者需要擔任角色的許可。(有關詳細資訊，請參閱在《AWS Data Pipeline 開發人員指南》中的使用 IAM 授予管道許可)。

如需如何正確設定許可的詳細資訊，以便 AWS 服務能夠完成您想要的任務，請參閱您正在呼叫之服務的文件。若要了解如何為服務建立角色的詳細資訊，請參閱建立角色以將許可委派給 AWS 服務。

設定具有 IAM 角色的服務以代表您的工作

當您想要將 AWS 服務設定為代表您工作時，通常會提供 IAM 角色的 ARN，以定義允許服務執行的動作。 AWS 會檢查 ，以確保您具有將角色傳遞給服務的許可。如需詳細資訊，請參閱授予使用者將角色傳遞至 AWS 服務的許可。

必要的動作

動作是您可以對資源執行的動作，例如查看、建立、編輯和刪除該資源。動作由每個 AWS 服務定義。

若要允許某人執行動作，您必須在適用於呼叫身分或受影響資源的政策中包含必要的動作。一般而言，要提供執行動作所需的許可，您必須在政策中包含該動作。例如，要建立使用者，您需要將 CreateUser 動作新增到政策中。

在某些情況下，動作可能要求您在政策中包含其他相關動作。例如，若要提供某人許可，使其可以使用 AWS Directory Service 操作在 `ds:CreateDirectory` 中建立目錄，您必須在其政策中包含下列動作：

- `ds:CreateDirectory`

- ec2:DescribeSubnets
- ec2:DescribeVpcs
- ec2:CreateSecurityGroup
- ec2:CreateNetworkInterface
- ec2:DescribeNetworkInterfaces
- ec2:AuthorizeSecurityGroupIngress
- ec2:AuthorizeSecurityGroupEgress

使用視覺化編輯器建立或編輯政策時，會收到警告和提示，以幫助您選擇政策的所有必需操作。

如需在 中建立目錄所需許可的詳細資訊 AWS Directory Service，請參閱範例 2：允許使用者建立目錄。

管理 IAM 資源的政策範例

以下 IAM 政策範例允許使用者執行與管理 IAM 使用者、群組和憑證相關的任務。這包括允許使用者管理自己的密碼、存取金鑰和多重要素驗證 (MFA) 裝置的政策。

如需允許使用者使用其他 AWS 服務執行任務的政策範例，例如 Amazon S3、Amazon EC2 和 DynamoDB，請參閱 以身分為基礎的 IAM 政策範例。

主題

- 允許使用者列出帳戶的群組、使用者、政策等，以供報告之用
- 允許使用者管理群組的成員資格
- 允許使用者管理 IAM 使用者
- 允許使用者設定帳戶密碼政策
- 允許使用者產生和擷取 IAM 憑證報告
- 允許所有 IAM 動作 (管理員存取)

允許使用者列出帳戶的群組、使用者、政策等，以供報告之用

以下政策允許使用者呼叫以字串 `Get` 或 `List` 開頭的任何 IAM 動作來產生報告。若要檢視範例政策，請參閱 IAM：允許對 IAM 主控台的唯讀存取權。

允許使用者管理群組的成員資格

以下政策允許使用者更新名為 MarketingGroup 群組的成員資格。若要檢視範例政策，請參閱 IAM：允許以程式設計方式及在主控台中管理群組的成員資格。

允許使用者管理 IAM 使用者

以下政策允許使用者執行所有與管理 IAM 使用者相關的任務，但是不允許對其他實體執行操作，如建立群組或政策。允許的動作包括：

- 建立使用者 (CreateUser 動作)。
- 刪除使用者。此任務需要許可以執行下列動作：DeleteSigningCertificate、DeleteLoginProfile、RemoveUserFromGroup 和 DeleteUser。
- 列出帳戶和群組中的使用者 (GetUser、ListUsers 和 ListGroupsForUser 動作)。
- 列出並移除使用者的政策 (ListUserPolicies、ListAttachedUserPolicies、DetachUserPolicy、DeleteUserPolicy 動作)
- 更改或變更使用者路徑 (UpdateUser 動作)。Resource 元素必須包含涵蓋來源路徑和目標路徑的 ARN。如需有關路徑的詳細資訊，請參閱 易用名稱和路徑。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
```

```
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

前述政策中包含的多個許可，允許使用者在 AWS 管理主控台中執行任務。從 AWS CLI、AWS 開發套件或 IAM HTTP 查詢 API 執行使用者相關任務的使用者可能不需要特定許可。例如，如果使用者已知道從使用者取消連接的 ARN，則不需要 `iam:ListAttachedUserPolicies` 許可。使用者所需許可的確切清單取決於使用者管理其他使用者時必須執行的任務。

以下政策中的許可允許透過 AWS 管理主控台存取使用者任務：

- `iam:GetAccount*`
- `iam:ListAccount*`

允許使用者設定帳戶密碼政策

您可以授予某些使用者取得和更新您 AWS 帳戶密碼政策的許可。若要檢視範例政策，請參閱 IAM：允許以程式設計方式在主控台中設定帳戶密碼要求。

允許使用者產生和擷取 IAM 憑證報告

您可以授予使用者許可，以產生和下載列出您 中所有使用者的報告 AWS 帳戶。此報告也會列出各種使用者憑證的狀態，包括密碼、存取金鑰、MFA 裝置和簽章憑證。如需有關憑證報告的詳細資訊，請參閱 為您的 產生登入資料報告 AWS 帳戶。若要檢視範例政策，請參閱 IAM：產生和擷取 IAM 憑證報告。

允許所有 IAM 動作 (管理員存取)

您可以授予某些使用者在 IAM 中執行所有操作的管理員許可，包括管理密碼、存取金鑰、MFA 裝置和使用者憑證。以下範例政策授予這些許可。

> **Warning**
>
> 當您授予使用者 IAM 的完全存取權時，對於使用者可以向自己或他人授予的許可則沒有限制。使用者可以建立新的 IAM 實體 (使用者或角色) 並授予這些實體對您 AWS 帳戶中所有資源的完全存取權限。您授予使用者對 IAM 的完全存取權限時，實際上是授予使用者對您的 AWS 帳戶中所有資源的完全存取權限。這包括可刪除所有資源的許可。您應只將這些許可授予信任的管理員，也應該對這些管理員強制採用多重要素驗證 (MFA)。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

IAM 暫時委派

概觀

暫時委派可加速加入，並簡化 Amazon 和與您 AWS 帳戶整合之 AWS 合作夥伴的產品管理。您可以委派暫時的有限許可，允許產品提供者透過自動化部署工作流程在幾分鐘內代表您完成設定任務，而不是

手動設定多個 AWS 服務。您可以維護具有核准要求和許可界限的管理控制，而產品提供者許可會在核准的持續時間後自動過期，而不需要手動清除。如果產品需要持續存取以進行持續操作，提供者可以使用暫時委派來建立具有定義角色最大許可之許可界限的 IAM 角色。所有產品提供者活動都會透過 AWS CloudTrail 進行追蹤，以進行合規和安全監控。

若要使用暫時委派請求，您可以啟動請求並檢閱請求。如需監控的詳細資訊，請參閱通知和 CloudTrail 記錄。

> **Note**
>
> 暫時委派請求只能由 Amazon 產品和已完成功能加入程序的合格 AWS 合作夥伴建立。客戶會檢閱並核准這些請求，但無法直接建立這些請求。如果您是希望將 IAM 暫時委派整合到您的產品的 AWS 合作夥伴，請參閱 合作夥伴整合指南中的入門和整合說明。

暫時委派的運作方式

暫時委派可讓 Amazon 和 AWS 合作夥伴請求暫時、有限存取您的帳戶。在您核准後，他們可以使用委派的許可代表您執行動作。委派請求會定義產品提供者在 AWS 帳戶中部署或設定資源所需的 AWS 服務和動作的特定許可。這些許可僅在有限時間內可用，並在請求中指定的持續時間後自動過期。

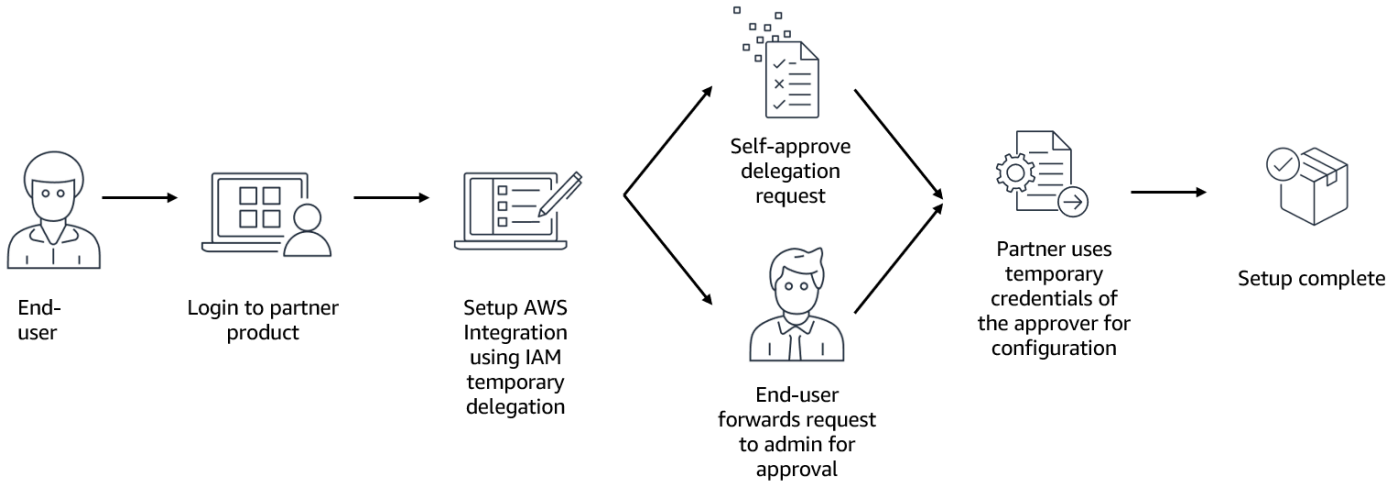
> **Note**
>
> 委派存取的持續時間上限為 12 小時。不過，根使用者只能核准持續時間為 4 小時或更短的委派請求。如果請求指定超過 4 小時，您必須使用非根身分來核准請求。如需詳細資訊，請參閱許可模擬測試版功能。

對於持續進行的任務，例如從 Amazon S3 儲存貯體讀取，委派請求可以包括建立 IAM 角色，允許在暫時存取過期後繼續存取資源和動作。產品提供者必須將許可界限連接至透過暫時委派建立的任何 IAM 角色。許可界限會限制角色的最大許可，但不會自行授予許可。您可以在核准許可界限之前，在請求中檢閱許可界限。如需詳細資訊，請參閱許可界限。

程序的運作方式如下：

1. 您登入 Amazon 或 AWS 合作夥伴產品，將其與您的 AWS 環境整合。
2. 產品提供者會代表您啟動委派請求，並將您重新導向至 AWS 管理主控台。
3. 您可以檢閱請求的許可，並決定是否核准、拒絕或轉送請求給管理員。

4. 在您或您的管理員核准請求後，產品提供者可以取得核准者的臨時憑證來執行必要的任務。
5. 產品提供者存取會在指定的時段後自動過期。不過，透過暫時委派請求建立的任何 IAM 角色會保留超過此期間，可讓產品提供者繼續存取持續管理任務的資源和動作。



> **Note**
>
> 只有在您擁有暫時委派請求中包含的服務和動作的許可時，才能將許可委派給產品提供者。如果您無法存取請求的服務和動作，當您核准請求時，產品提供者不會收到這些許可。

如果許可檢查顯示可能成功，您可以核准暫時委派請求並繼續工作流程。

如果許可檢查顯示您可能沒有足夠的許可，請將請求轉送給管理員以進行核准。我們建議您使用電子郵件或票證等您偏好的方法來通知管理員此請求。

管理員核准請求後，接下來會發生什麼情況取決於產品提供者的組態：

- 如果產品提供者請求立即存取，他們會自動收到臨時許可，且存取持續時間開始。
- 如果產品提供者請求擁有者（初始收件人）發行，您必須在存取期間開始之前返回請求以明確共用臨時帳戶存取。產品提供者通常會在需要額外的輸入時，使用此選項來完成所需的任務，例如資源選擇或組態詳細資訊。

啟動暫時委派請求

您只能從支援的 Amazon 或 AWS 合作夥伴產品啟動暫時委派請求。在支援暫時委派的工作流程期間，系統會提示您授予產品提供者暫時、有限的許可，以設定您帳戶中的必要 AWS 資源。這種自動化方法讓您無需手動設定這些資源，即可提供更簡化的體驗。

您可以在授予存取權之前檢閱委派請求詳細資訊，例如產品提供者所需的特定 IAM 角色、政策和 AWS 服務。如果您有足夠的許可，您可以自行核准請求，或將請求轉送給帳戶管理員進行核准。所有產品提供者存取都有時間限制，並可視需要進行監控和撤銷。

啟動暫時委派請求

1. 從 Amazon 或需要與 AWS 您的帳戶整合的 AWS 合作夥伴導覽至支援產品的主控台。
2. 選取使用 IAM 暫時委派部署。請注意，選項名稱可能因支援的產品而異。如需詳細資訊，請參閱產品供應商的文件。

   > **Note**
   >
   > 如果您尚未登入 AWS 管理主控台， AWS 登入頁面會開啟新視窗。我們建議您先登入 AWS 您的帳戶，再從產品主控台啟動暫時委派請求。如需如何根據您的使用者類型和您要存取 AWS 的資源登入的詳細資訊，請參閱 AWS 登入使用者指南。

3. 檢閱請求詳細資訊以確認產品提供者的產品名稱和 AWS 帳戶。您也可以檢閱產品提供者將用來代表您執行動作的 AWS 身分。
4. 檢閱透過核准此請求暫時委派之許可的存取詳細資訊。

   - 許可摘要區段提供 AI 產生的高階概觀，可協助您了解 AWS 可存取的服務類別，以及可在每個服務中執行的動作類型。
   - 選擇檢視 JSON 來檢閱產品提供者在 AWS 帳戶中部署所需的特定許可，包括存取範圍和資源限制。
   - 如果產品提供者在暫時委派請求中建立 IAM 角色，則必須將許可界限連接到角色。這些 IAM 角色具有許可，可在請求的存取期間過期後，繼續允許存取資源和動作。選擇檢視詳細資訊以檢閱許可界限，這會定義角色可擁有的最大許可。在定義其實際許可的建立期間，產品提供者會將其他政策套用至角色。這些政策看起來可能比界限更窄或更寬，具體取決於產品提供者如何定義它們。不過，許可界限保證角色的有效許可永遠不會超過您在請求核准期間看到的許可，無論哪些政策連接到角色。如需詳細資訊，請參閱許可界限。

5. 檢閱許可模擬結果。許可模擬功能會自動針對請求中包含的許可來評估您的身分許可。根據此分析，會顯示建議，指出要使用您目前的身分核准請求，還是將請求轉送給管理員。如需詳細資訊，請參閱許可模擬 Beta 版功能。
6. 在對話方塊中，選取您要繼續的方式。
   - 當您的身分具有足夠的許可，以允許產品提供者代表您執行加入程序時，請選取允許存取。當您選取此選項時，產品提供者的存取持續時間會在您提供存取權後開始。
   - 如果您的身分沒有足夠的許可，以允許產品提供者代表您執行加入程序，請選取請求核准。然後，選擇建立核准請求。當您選取此選項時，系統會建立暫時委派請求連結，供您與帳戶管理員共用。您的管理員可以存取 AWS 管理主控台，或使用存取連結來檢閱暫時委派請求，以核准請求並與請求者共用暫時存取權。

> **Note**
>
> 授予產品提供者存取權需要兩個動作：接受委派請求 (`AcceptDelegationRequest`) 和釋出交換字符 (`SendDelegationToken`)。當您核准請求時， AWS 管理主控台會自動執行這兩個步驟。如果您使用 AWS CLI 或 API，則必須分別執行這兩個步驟。

許可模擬功能 - Beta

當您收到暫時委派請求時，您可以自行核准，或將其轉送給帳戶管理員進行核准。只有在您擁有暫時委派請求中包含的服務和動作的許可時，才能將許可委派給產品提供者。如果您無法存取請求的服務和動作，即使這些許可包含在請求中，產品提供者也不會收到這些許可。

例如，暫時委派請求需要能夠建立 Amazon S3 儲存貯體、在 Amazon EC2 中啟動和停止執行個體，以及擔任 IAM 角色。核准請求的身分可以啟動和停止 Amazon EC2 中的執行個體，並擔任 IAM 角色，但沒有建立 Amazon S3 儲存貯體的許可。當此身分核准請求時，即使暫時委派請求中包含這些許可，產品提供者仍無法建立 Amazon S3 儲存貯體。

由於您只能委派您已擁有的許可，因此在核准之前評估您是否擁有請求的許可至關重要。許可模擬測試版功能透過將您的許可與請求中包含的許可進行比較，協助進行此評估。評估指出您是否可以使用目前的身分核准請求，或需要將其轉送給管理員。如果分析無法驗證您是否有足夠的許可，請將請求轉發給管理員以供檢閱。此評估是以模擬許可分析為基礎，可能與您的即時 AWS 環境不同，因此請在繼續之前仔細檢閱請求的許可。

後續步驟

啟動暫時委派請求後，您可以在請求的生命週期中管理和監控請求。下列程序可協助您追蹤、核准和控制暫時存取：

- 檢閱暫時委派請求 – 監控存取請求的狀態，並檢視核准或拒絕暫時委派請求的詳細資訊。
- 撤銷暫時委派存取權 – 在工作階段自然過期之前立即終止作用中的暫時委派工作階段。

檢閱暫時委派請求

啟動暫時委派請求後，您可以在 IAM 主控台中監控、核准和拒絕請求。暫時委派請求頁面提供所有請求的集中檢視，包括待核准、已完成或拒絕的請求。身為管理員，您可以檢閱這些請求，以授予產品提供者對 AWS 資源的存取權，或根據您組織的安全政策、業務需求或合規標準拒絕這些請求。此可見性可協助您追蹤產品提供者存取的生命週期，並維持暫時許可的監督。

> **Note**
>
> 您必須擁有 iam：AcceptDelegationRequest 許可，才能核准暫時委派請求。

核准暫時委派請求

1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇暫時委派請求。
3. 主頁面會顯示暫時委派請求的清單，其中包含下列資訊：
   - 請求 ID – 請求的唯一識別符
   - 狀態 – 目前狀態（待定、已核准、已拒絕、已共用、已過期）
   - 請求者 - 與請求相關聯的產品提供者
   - 由 - 帳戶中發起產品提供者請求的 IAM 主體啟動
   - 已建立請求 – 提交請求時
   - 請求過期 – 請求過期或即將過期時
4. （選用）使用篩選條件選項依狀態檢視請求：
   - 所有請求 – 檢視所有請求，無論狀態為何
   - 待處理 – 檢視等待管理員核准的請求

- 已核准 – 檢視已核准的請求
- 共用 – 檢視已共用存取權的請求
- 已拒絕 – 檢視具有拒絕原因的已拒絕請求

5. 若要檢視特定請求的詳細資訊或檢閱待核准請求，請選擇請求 ID。
6. 檢閱詳細的請求資訊：

   - 產品提供者資訊
   - 請求原因和理由
   - 請求的持續時間
   - 請求的 AWS 許可

7. 如果您是檢閱待定請求的管理員，請選擇下列其中一個選項：

   - 若要核准請求，請選擇核准。在核准對話方塊中，您可以檢視許可模擬的結果。如需詳細資訊，請參閱許可模擬 Beta 版功能。在確認存取持續時間和 AWS 您的身分後，選擇核准以授予存取權。如果產品提供者請求立即存取，他們會自動收到暫時許可，且存取持續時間開始。否則，請通知啟動請求的人員將存取權釋出給產品提供者。
   - 若要拒絕請求，請選擇拒絕。
     - 在拒絕對話方塊中，提供拒絕的明確原因，以協助請求者了解拒絕其請求的原因。
     - 選擇拒絕以拒絕存取。

8. 請求清單會自動重新整理以顯示最新的狀態資訊。您也可以手動重新整理頁面，以檢查狀態更新。

撤銷暫時委派存取權

雖然產品提供者存取工作階段的設計會在核准持續時間後自動過期，但在某些情況下，您可能需要立即終止存取。當出現安全問題、產品提供者的工作提早完成或業務需求變更時，撤銷作用中的產品提供者存取可提供緊急控制機制。請求啟動者和管理員都可以撤銷存取，以維護安全和操作控制。

撤銷暫時委派存取權

1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇暫時委派請求。
3. 找到您要撤銷之存取工作階段的請求 ID。
4. 選擇動作，然後選擇撤銷存取權。
5. 在對話方塊中，選擇撤銷存取權，以確認您想要立即終止存取工作階段。

撤銷存取權後，產品提供者將無法再存取您的 AWS 資源。撤銷會記錄在 AWS CloudTrail 中以供稽核之用。

> **Important**
>
> 撤銷存取權會立即終止產品提供者存取工作階段。任何使用 存取的進行中工作或程序都會中斷。確定撤銷不會中斷關鍵操作。

> **Note**
>
> 您無法撤銷使用根使用者核准之請求的存取權。 AWS 建議您避免使用根使用者核准委派請求。請改用具有適當許可的 IAM 角色。

管理委派請求的許可

管理員可以授予 IAM 主體許可，以管理來自產品提供者的委派請求。當您想要將核准授權委派給組織中的特定使用者或團隊，或當您需要控制誰可以對委派請求執行特定動作時，這會很有用。

下列 IAM 許可可用於管理委派請求：

權限	說明
iam：AssociateDelegationRequest	將未指派的委派請求與 AWS 您的帳戶建立關聯
iam：GetDelegationRequest	檢視委派請求的詳細資訊
iam：UpdateDelegationRequest	將委派請求轉送給管理員以進行核准
iam：AcceptDelegationRequest	核准委派請求
iam：SendDelegationToken	核准後將交換字符釋出給產品供應商

權限	說明
iam：RejectDelegationRequest	拒絕委派請求
iam：ListDelegationRequests	列出您帳戶的委派請求

> **Note**
>
> 根據預設，啟動委派請求的 IAM 主體會自動獲得管理該特定請求的許可。他們可以將其與其帳戶建立關聯、檢視請求詳細資訊、拒絕請求、將其轉送給管理員進行核准、在管理員核准後將交換字符釋出給產品提供者，以及列出他們擁有的委派請求。

通知

IAM 暫時委派與 AWS 使用者通知整合，協助您隨時掌握委派請求狀態變更的相關資訊。對於需要檢閱和核准委派請求的管理員來說，通知特別有用。

透過 AWS 使用者通知，您可以設定要透過多個管道傳送的提醒，包括電子郵件、Amazon Simple Notification Service (SNS)、適用於 Slack 或 Microsoft Teams 的 AWS Chatbot，以及 AWS 主控台行動應用程式。這可確保適當的人員在正確的時間收到通知，以便更快速回應待核准或了解存取變更。您也可以根據組織的需求和安全需求，自訂哪些事件觸發通知。

可用的通知事件

您可以訂閱以接收下列 IAM 暫時委派事件的通知：

- 已建立 IAM 暫時委派請求
- 指派的 IAM 暫時委派請求
- IAM 暫時委派請求待核准
- IAM 暫時委派請求遭拒
- 已接受 IAM 暫時委派請求
- IAM 暫時委派請求已完成
- IAM 暫時委派請求已過期

設定 通知

若要設定 IAM 暫時委派事件的通知：

1. 開啟 AWS 使用者通知主控台
2. 建立或更新通知組態
3. 選取 AWS IAM 做為服務
4. 選擇您要收到通知的委派請求事件
5. 設定您的交付管道（電子郵件、 AWS Chatbot 等）

如需設定 AWS 使用者通知的詳細指示，包括設定交付管道和管理通知規則，請參閱 AWS 使用者通知文件。

CloudTrail

產品提供者使用暫時委派存取執行的所有動作都會自動記錄在 AWS CloudTrail 中。這可讓您帳戶中產品提供者活動的完整可見性和可稽核性 AWS 。您可以識別產品提供者所採取的動作、發生的時間，以及執行這些動作的產品提供者帳戶。

為了協助您區分您自己的 IAM 主體所採取的動作，以及具有委派存取權的產品提供者所採取的動作，CloudTrail 事件會在 `userIdentity`元素`invokedByDelegate`下包含名為 的新欄位。此欄位包含產品提供者 AWS 的帳戶 ID，讓您輕鬆篩選和稽核所有委派的動作。

CloudTrail 事件結構

下列範例顯示產品提供者使用暫時委派存取所執行動作的 CloudTrail 事件：

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "[REDACTED:AWS_ACCESS_KEY]",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
```

```
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedByDelegate": {
            "accountId": "444455556666"
        }
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePolicyName"
    },
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

`invokedByDelegate` 欄位包含使用委派存取執行動作之產品提供者 AWS 的帳戶 ID。在此範例中，帳戶 444455556666（產品提供者）在帳戶 111122223333（客戶帳戶）中執行動作。

AWS 合作夥伴的 IAM 暫時委派

概觀

IAM 暫時委派可讓 AWS 客戶透過互動式引導式工作流程，順暢地將 AWS 合作夥伴產品加入和/或整合到其 AWS 環境中。客戶可以授予 AWS 合作夥伴有限的暫時存取權，以設定所需的 AWS 服務、減少加入摩擦並加速實現價值的時間。

若要將暫時委派整合到您的產品中，請了解如何了解您的整合、檢閱了解許可，並遵循政策評估準則。如需範例 IAM 政策，請參閱政策範本。當您準備好實作時，請參閱建置您的整合。

IAM 暫時委派可讓合作夥伴：

- 透過自動化資源佈建簡化客戶加入

- 消除手動組態步驟，降低整合複雜性
- 透過透明、客戶核准的許可建立信任
- 使用許可界限，以長期存取模式啟用持續操作

運作方式

1. 合作夥伴建立委派請求 - 合作夥伴建立請求，指定他們需要哪些許可以及需要多長時間
2. AWS 主控台中的客戶評論 - 客戶查看合作夥伴正在請求的許可以及原因
3. 客戶核准 - 客戶核准請求並釋出交換字符。權杖會在此指定的 SNS 主題上傳送給合作夥伴。
4. 合作夥伴收到臨時登入資料 - 合作夥伴將字符交換為臨時 AWS 登入資料
5. 合作夥伴設定資源 - 合作夥伴使用登入資料來設定客戶帳戶中的必要資源

合作夥伴資格

若要符合暫時委派整合的資格，合作夥伴必須符合下列要求：

- ISV Accelerate 參與 – 您必須註冊 ISV Accelerate (ISVA) 計畫。
- AWS Marketplace 清單 – 您的產品必須在 AWS Marketplace 中以「部署上 AWS」徽章列出。

加入程序

完成下列步驟，將暫時委派整合到您的產品中：

1. 步驟 1：檢閱需求

   檢閱本文件以了解資格要求，並完成下方的合作夥伴問卷。
2. 步驟 2：提交您的加入請求

   傳送電子郵件至 aws-iam-partner-onboarding@amazon.com 或聯絡您的 AWS 代表。將您完成的合作夥伴問卷包含下表中的所有必要欄位。
3. 步驟 3： AWS 驗證和檢閱

   AWS 將：

   - 驗證您是否符合資格條件
   - 檢閱您的政策範本和許可界限

- 針對您提交的成品提供意見回饋

4. 步驟 4：精簡您的政策

   回應 AWS 意見回饋，並視需要提交更新的政策範本或許可界限。

5. 步驟 5：完成註冊

   核准後， AWS 將：

   - 為指定的帳戶啟用 API 存取
   - 共用政策範本和許可界限ARNs （ 如適用 ）

   加入完成時，您會收到確認。然後，您可以從您的註冊帳戶存取暫時委派 APIs、CreateDelegationRequest 和 GetDelegatedAccessToken，並開始將委派請求工作流程整合至您的產品。

合作夥伴問卷

下表列出加入合作夥伴所需的資訊：

資訊	說明	必要
Partner Central AccountID	您在 AWS Partner Central 上註冊 AWS 帳戶的帳戶 ID。	是
PartnerId	Partner AWS Central 提供的合作夥伴 ID。	否
AWS Marketplace 產品 ID	AWS Partner Central 提供的產品 ID。	是
AWS accountIDs	您要用來呼叫暫時委派 APIs AWS 的帳戶 ID 清單。 IDs 這應該包括您的生產和非生產/測試帳戶。	是
合作夥伴名稱	當客戶檢閱您的暫時委派請求時，此名稱會顯示在 AWS 管理主控台中。	是
聯絡電子郵件 (s)	用於就整合與您聯絡的一或多個電子郵件地址。	是
請求者網域	您的網域 （ 例如 www.example.com)	是

資訊	說明	必要
整合描述	您想要使用此功能解決的使用案例簡短描述。您可以包含文件或其他公有資料的參考連結。	是
架構圖	架構圖說明您的整合使用案例 (s)。	否
政策範本	您必須為此功能註冊至少一個政策範本。政策範本會定義您想要在客戶 AWS 帳戶中請求的暫時許可。如需詳細資訊，請參閱政策範本一節。	是
政策範本名稱	您要註冊的政策範本名稱。	是
許可界限	如果您想要使用暫時許可在客戶的帳戶中建立 IAM 角色，則必須向 IAM 註冊許可界限。許可界限會連接到您建立的 IAM 角色，以限制角色的最大許可。您可以使用選取的 AWS 受管政策做為許可界限，或註冊新的自訂許可界限 (JSON)。如需詳細資訊，請參閱許可界限一節。	否
許可邊界名稱	許可界限的名稱。格式為：arn：aws：iam：：partner：policy/permission_boundary/<partner_domain>/<policy_name>_<date> 政策名稱必須包含建立日期做為尾碼。建立許可界限後，無法更新名稱。如果您使用的是現有的 AWS 受管政策，請改為提供 受管政策 ARN。	否
許可界限描述	許可界限的描述。建立許可界限後，無法更新此描述。	否

了解您的整合

完成加入程序後，您可以建立與 IAM 暫時委派的整合。完整的整合通常涉及三個主要的工作類別：

1. 使用者體驗和工作流程設計

在合作夥伴應用程式中建立前端體驗，引導客戶完成暫時委派工作流程。合作夥伴應用程式應該：

- 提供明確的加入或組態流程，讓客戶可以授予暫時存取權。清楚標記此動作，例如「使用 IAM 暫時委派進行部署」。
- 將客戶重新導向至 AWS 管理主控台，以使用 CreateDelegationRequest API 傳回的主控台連結來檢閱和核准委派請求

- 提供有關請求哪些許可及其原因的適當訊息。客戶可以在委派請求詳細資訊頁面上看到此訊息。
- 在客戶完成核准後，處理客戶傳回至您的應用程式 AWS。

暫時委派請求的最佳實務

為您的合作夥伴應用程式實作暫時委派時，請遵循下列定義的最佳實務，協助客戶驗證請求真實性和準確性。

1。在請求訊息中包含使用者可識別的內容

您應該在委派請求訊息中包含使用者特定的內容。此資訊可協助使用者從現有的合作夥伴工作流程中識別請求，並區分合法請求。請求訊息中建議的資訊：

- 服務中的客戶帳戶識別符或使用者名稱
- 工作區名稱、訂閱 ID 或組織識別符
- 正在存取的特定資源（叢集名稱、專案名稱、環境）
- 為此委派嘗試產生的唯一交易或請求識別符

請求訊息範例：

```
Request from Partner A workspace "production-analytics"
Account: user@example.com
Workspace ID: 1234ABCD
Cluster: ml-training-cluster-01
Request ID: 1111-2222-3333-4444
```

2. 選用：啟動委派時包含 AWS 帳戶 ID

如果客戶帳戶 ID 可用，請將其包含在委派請求中。此驗證步驟會在客戶的意圖和委派字符之間建立額外的繫結。

3. 安全驗證的設計請求訊息

建構您的請求訊息，讓客戶在授予存取權之前可放心地驗證合法性。使用者體驗需求：

- 在重新導向至 之前，在您的應用程式界面中顯著顯示請求訊息 AWS
- 使用直接連接到客戶目前動作的清晰、描述性語言

- 避免可能套用至任何委派請求的一般訊息
- 包含足夠的詳細資訊，讓客戶確認請求符合其預期的工作流程
- 顯示將接收委派存取權 AWS 的帳戶 ID
- 明確說明您的應用程式將請求哪些許可

將客戶重新導向至 AWS 同意畫面之前，請在應用程式中包含確認通知：

```
## Redirecting to IAM Temporary Delegation
You are about to grant [Your Service Name] temporary access to your AWS account.

Before clicking "Allow" on the AWS consent screen:
• Verify the request details match your current action
• Confirm the AWS account ID matches your intended account
• Make sure you initiated this request from [Your Service Name]
```

4. 工作階段繫結建議

當 AWS IAM 處理核心授權流程時，請在應用程式中實作這些實務，以強化工作階段完整性：

- 為每個委派嘗試產生唯一的單次使用請求識別符
- 將委派請求與客戶的作用中應用程式工作階段建立關聯
- 根據您的委派程序實作適當的請求過期
- 驗證委派回呼是否符合原始請求內容
- 記錄所有委派請求啟動和完成以進行安全性監控

遵循這些最佳實務，有助於建立更安全的暫時委派體驗。

2. API 整合

使用 IAM 暫時委派 APIs來傳送和管理委派請求。註冊 AWS 帳戶後，您可以存取下列 APIs：

- IAM CreateDelegationRequest – 為客戶 AWS 的帳戶建立委派請求。此 API 會傳回主控台連結，您可以將客戶重新導向至 ，以檢閱和核准請求。
- AWS STS GetDelegatedAccessToken – 在客戶核准委派請求後擷取臨時 AWS 憑證。使用這些登入資料在客戶的帳戶中執行動作。

您的整合應處理委派請求的完整生命週期，包括建立請求、監控其狀態，以及在核准時擷取臨時憑證。

3. 資源組態和協調

在您取得臨時登入資料後， 會協調必要的工作流程，以設定客戶 AWS 帳戶中的資源。這可能包括：

- 直接呼叫 AWS 服務 APIs以建立和設定資源
- 使用 AWS CloudFormation 範本部署基礎設施
- 建立持續存取的 IAM 角色 （需要使用許可界限）。作為defense-in-depth措施，我們建議您使用動態角色名稱，例如具有隨機產生尾碼的名稱。此方法可讓每個客戶帳戶中的角色名稱獨一無二且無法預測。

您的協同運作邏輯應該是等冪的，並正常處理失敗，因為客戶可能需要重試或修改其委派核准。

了解許可

作為功能加入程序的一部分，您需要向 IAM 註冊政策，以定義您想要在客戶 AWS 帳戶中請求的許可。註冊程序為客戶提供更一致的體驗，並有助於避免政策撰寫中的常見陷阱。

在註冊期間， 會根據一組驗證 AWS 來評估您的政策。這些驗證旨在標準化政策格式和結構，並針對已知的反模式提供基本保護。驗證也會降低提升權限、意外跨帳戶存取，以及廣泛存取客戶帳戶中高價值資源的風險。

許可類型

AWS 會考慮兩種類型的許可：暫時和長期。

暫時許可

暫時許可會限制指派給任何暫時委派存取工作階段的許可。暫時許可會在套用至委派工作階段的政策範本中說明。範本支援您在建立委派請求時提供的參數。這些參數值接著會繫結至工作階段。暫時許可的運作方式與 AWS STS 目前提供的工作階段政策相同：政策會限制基礎使用者的功能，但不授予任何其他存取權。如需詳細資訊，請參閱工作階段政策 AWS STS 的文件。

長期許可

長期許可會限制透過暫時存取建立或管理之任何角色的許可。長期許可會實作為 IAM 許可界限。您可以在加入 AWS 時向 提交一或多個許可界限。核准後， AWS 將與您共用政策 ARN，供您在政策中參考。

這些界限政策有兩個值得注意的功能。首先，它們是不可變的。如果您想要更新許可，您可以註冊新的許可界限。然後，您可以透過傳送新的委派請求，將新的許可界限附加到客戶的角色。其次，政策不會建立範本。由於相同的邊界政策是全域共用的，因此無法根據每位客戶修改。

> **Important**
>
> 許可界限的大小上限為 6，144 個字元。

> **Note**
>
> 如果您想要更新許可界限或政策範本，請透過 aws-iam-partner-onboarding@amazon.com 聯絡 IAM。註冊新的許可界限後，您可以傳送委派請求給客戶，以更新 IAM 角色並連接新註冊的許可界限。如需詳細資訊，請參閱範例一節。

範例使用案例：資料處理工作負載

考慮在客戶帳戶中執行資料處理工作負載的產品供應商。供應商需要在初始加入期間設定基礎設施，但也需要持續存取才能操作工作負載。

暫時許可（用於初始設定）：

- 建立 Amazon EC2 執行個體、VPC 和安全群組
- 為已處理的資料建立 Amazon S3 儲存貯體
- 建立持續操作的 IAM 角色
- 將許可界限連接至 IAM 角色

長期許可（具有持續操作許可界限的 IAM 角色）：

- 啟動和停止 Amazon EC2 執行個體以執行處理任務
- 從 Amazon S3 儲存貯體讀取輸入資料
- 將處理的結果寫入 Amazon S3 儲存貯體

臨時許可會在加入期間使用一次，以設定基礎設施。在此過程中建立的 IAM 角色具有許可界限，其最大許可僅限於持續工作負載管理所需的操作。這可確保即使修改角色的政策，也不能超過邊界中定義的許可。

政策評估準則

AWS 會根據一組準則來評估您提交的政策。相同的評估準則同時適用於政策範本和許可界限，並在適當時記下細微差異。

基於評估目的，服務會分成不同的群組。最重要的區別在於管理存取、登入資料和金鑰的安全敏感服務。授予這些服務存取權的政策需要專注於已完成的工作。安全敏感服務包括下列項目：AWS Identity and Access Management (IAM)、 AWS Key Management Service (KMS)、 AWS Resource Access Manager (RAM)、 AWS IAM Identity Center、 AWS Organizations 和 AWS Secrets Manager。

次要區別是可以跨帳戶邊界存取資料的 服務。這些服務的政策必須包含保護，以防止意外的跨帳戶存取。

常見驗證

所有政策陳述式都必須遵循下列準則：

- 所有陳述式都必須依該順序包含效果、動作 （ 或 NotAction)、資源和條件欄位
- 單一陳述式中的所有動作都必須依字母順序列出
- 政策中包含的所有 ARNs 必須遵循相關服務的公有文件中定義的語法
- NotAction 欄位只能用於拒絕陳述式
- 允許陳述式中的動作必須包含服務代碼。不允許一般萬用字元 ("*")

安全敏感的服務限制

下列限制適用於上述對安全敏感的服務：

- 允許陳述式中的動作必須比 【服務】 ：* 更具體
- 暫時存取政策範本的允許陳述式中的動作不得包含萬用字元
- 如 iam：PassRole 或 iam：CreateServiceLinkedRole 等敏感動作需要額外範圍，例如特定資源或條件式檢查。這些動作包括：
  - IAM 角色傳遞
  - IAM 角色修改動作
  - IAM 政策修改動作
  - AWS KMS 寫入或密碼編譯操作

  - AWS RAM 寫入或共用操作
  - AWS 擷取或修改秘密或修改資源政策的 Secrets Manager 操作
- 其他動作可能會使用萬用字元資源，例如 iam：ListUsers 或 iam：GetPolicy
- 會封鎖管理登入資料的動作，例如 iam：CreateAccessKey

IAM 特定限制

對於 IAM：

- IAM 角色和政策僅允許有限的寫入操作。您無法請求其他 IAM 資源的許可，例如使用者、群組和憑證。
- 政策連接或內嵌政策管理動作僅限於具有許可界限的角色。許可界限必須由合作夥伴提供，或在允許的 AWS 受管政策清單中。如果未授予高權限或管理許可，則可能允許受 AWS 管政策。例如，可以接受特定任務函數或 SecurityAudit 政策的 AWS 受管政策。 AWS 將在加入程序期間case-by-case每個 AWS 受管政策。
- 政策管理僅適用於具有合作夥伴特定路徑的政策：arn：aws：iam：：@{AccountId}：policy/partner_domain.com/【feature】*
- 標籤可能只會在資源建立期間套用，而且僅適用於角色和政策
- iam：PassRole 檢查必須符合特定名稱或路徑字首

AWS STS特定限制

對於 AWS STS：

- sts：AssumeRole 必須範圍限定為特定角色 ARN、角色 ARN 字首，或限制為一組帳戶或組織 ID/組織單位

IAM Identity Center 限制

對於 AWS IAM Identity Center，會封鎖下列動作：

- 處理許可管理的所有動作（例如，sso：AttachCustomerManagedPolicyReferenceToPermissionSet)
- AWS Identity Store 的使用者、群組和成員資格修改
- 標籤管理

AWS Organizations 限制

對於 AWS Organizations，僅允許讀取動作。

其他服務特定的驗證

- 取得秘密或登入資料的動作，例如 glue：GetConnection 或 redshift：GetClusterCredentials，必須具有符合完整 ARNs、ARN 字首或標籤的條件
- 對於 Amazon Redshift：僅允許在特定資料庫名稱上使用 redshift：GetClusterCredentials，且僅允許在特定工作群組名稱上使用 redshift：GetClusterCredentialsWithIAM

> **Note**
>
> 管理帳戶中的 IAM 資源時，建議使用指出您名稱的路徑，例如 arn：aws：iam：：111122223333：role/partner.com/rolename。這將有助於區分與您的整合相關聯的資源，讓客戶更容易探索、稽核和分析。

跨帳戶存取要求

可能允許跨帳戶存取的陳述式必須至少包含下列其中一項：

- 為資源指定帳戶或組織的條件（例如，aws：ResourceOrgId 符合一或多個預期值）
- 包含特定帳戶的資源欄位（例如 arn：aws：sqs：*：111122223333：*)
- 包含非萬用字元帳戶和完整資源名稱的資源欄位（例如 arn：aws：s3：：full-bucket-name)

> **Note**
>
> 跨帳戶存取是一種敏感功能，需要明確的業務理由。 AWS 將在加入過程中仔細審查跨帳戶存取的需求。

政策範本

政策範本是一種新的 IAM 建構，旨在定義合作夥伴在客戶帳戶中請求的暫時許可。與一般 IAM 政策一樣，它們使用具有效果、動作、資源和條件元素的陳述式來定義許可。關鍵差異在於政策範本包含參數（例如 @{bucketName}），這些參數會在您建立委派請求時以實際值取代。

政策範本的運作方式

在加入程序中，您會向 註冊政策範本 AWS。 AWS 會為每個範本指派唯一的 ARN，供您在建立委派請求時參考。

當您建立委派請求時，您可以指定：

- 政策範本 ARN
- 要取代為範本的參數值

AWS 會將 範本與您的參數值合併，以產生標準 IAM 政策。客戶在核准委派請求時檢閱此最終轉譯政策，並查看將授予哪些許可。

> **Note**
>
> 最終轉譯政策的大小上限為 2048 個字元。

以下是示範範本替換如何運作的簡單範例。

政策範本：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
}
```

委派請求中提供的參數：

```
{
    "Name": "bucketName",
```

```
    "Values": ["customer-data-bucket"],
    "Type": "String"
}
```

最終轉譯政策（客戶看到的內容）：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::customer-data-bucket/*"
        }
    ]
}
```

範本語法

政策範本使用兩個主要功能來提供彈性：參數替換和條件式陳述式。參數替換可讓您在範本中定義預留位置，這些預留位置會在建立委派請求時以實際值取代。條件式陳述式可讓您根據參數值包含或排除整個政策陳述式。

參數替代和類型

使用 @{parameterName} 語法來定義政策範本中的參數。建立委派請求時，您必須指定每個參數的類型。

String

直接替換為範本的單一值。

範本：

```
"Resource": "arn:aws:s3:::@{bucketName}/*"
```

參數：

```
{
    "Name": "bucketName",
    "Values": ["my-bucket"],
    "Type": "String"
}
```

轉譯的結果：

```
"Resource": "arn:aws:s3:::my-bucket/*"
```

StringList

產生多個資源項目的多個值。在資源 ARN 中使用 StringList 參數時，它會展開以為每個值建立個別的資源項目。

範本：

```
"Resource": "arn:aws:s3:::@{bucketNames}/*"
```

參數：

```
{
    "Name": "bucketNames",
    "Values": ["bucket-1", "bucket-2"],
    "Type": "StringList"
}
```

轉譯的結果：

```
"Resource": [
    "arn:aws:s3:::bucket-1/*",
    "arn:aws:s3:::bucket-2/*"
]
```

跨產品行為

在相同的資源 ARN 中使用多個參數時，StringList 參數會建立所有組合的跨產品。

範本：

```
"Resource": "arn:aws:s3:::@{bucketNames}/@{prefix}/*"
```

參數：

```
[
    {
        "Name": "bucketNames",
        "Values": ["bucket-1", "bucket-2"],
        "Type": "StringList"
    },
    {
        "Name": "prefix",
        "Values": ["data"],
        "Type": "String"
    }
]
```

轉譯的結果：

```
"Resource": [
    "arn:aws:s3:::bucket-1/data/*",
    "arn:aws:s3:::bucket-2/data/*"
]
```

條件式陳述式

使用 @Enabled 指令，根據參數值有條件地包含或排除整個陳述式。

語法：

- @Enabled： "parameterName" - 當參數值為 "True" 時包含陳述式
- @Enabled："！parameterName" - 當參數值不是 "True"（否定）時包含陳述式

範本：

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

```
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
            "@Enabled": "ENABLE_S3_WRITE",
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
}
```

參數 (ENABLE_S3_WRITE 為 "True" 時）：

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["True"],
        "Type": "String"
    }
]
```

轉譯的結果：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::my-bucket/*"
        }
    ]
```

```
}
```

參數（當 ENABLE_S3_WRITE 為 "False" 時）：

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["False"],
        "Type": "String"
    }
]
```

轉譯的結果：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        }
    ]
}
```

當 ENABLE_S3_WRITE 設定為 "True" 時，會包含條件式陳述式。設定為 "False" 時，陳述式會從轉譯政策中排除。

其他範例

下列範例示範在暫時委派中使用政策範本的常見模式。他們專注於建立具有長期存取許可界限的 IAM 角色，並顯示將許可範圍限定到特定資源的不同策略。這些範例說明如何使用 ARN 字首、資源標記和許可界限更新等技術來平衡彈性和安全性。

範例 1：授予特定資源的長期存取權

下列許可界限會提交為 "partner.com" 的 "SQSAccessorBoundary"：

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

> **Note**
>
> 這包括相同的帳戶條件，以避免將存取權授予具有開啟資源政策的其他帳戶中的佇列。無法包含客戶帳戶 ID 的直接參考，因為邊界會跨所有客戶共用，且無法進行範本化。

由於這是此政策的第一個版本，其 ARN 為 arn：aws：iam：：partner：policy/permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15

提交下列政策範本以取得臨時存取許可：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sqs:ListQueues"
            ],
            "Resource": "arn:aws:sqs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
```

```
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
                }
            }
        }
    ]
}
```

範例 2：使用 ARN 字首

許可界限可以指定資源 ARN 字首來限制存取：

```
"Resource": "arn:aws:sqs:*:@{AccountId}:PartnerPrefix*"
```

這只會限制存取具有該字首的資源，進而減少可存取資源的範圍。

範例 3：使用標籤進行資源存取控制

您可以在暫時委派存取期間標記資源，並依賴這些標籤進行長期存取控制。

允許存取已標記資源的許可界限：

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:ResourceTag/ManagedByPartnerDotCom": "false"
        },
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

在建立時標記新佇列的政策範本：

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:CreateQueue",
        "sqs:TagQueue"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:RequestTag/ManagedByPartnerDotCom": "false"
        }
    }
}
```

標記預先存在佇列並建立角色的政策範本：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sqs:TagQueue"
            ],
            "Resource": "arn:aws:sqs:*:@{AccountId}:@{QueueName}",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "ManagedByPartnerDotCom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
```

```
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
                }
            }
        }
    ]
}
```

此方法可讓客戶明確確認哪些特定資源可以長期存取。

範例 4：更新許可界限

若要更新許可界限，請使用新的日期尾碼註冊新版本，並請求替換許可。

使用其他許可更新許可界限：

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:PurgeQueue",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

做為第二個版本，此政策具有 ARN：arn：aws：iam：：partner：policy/permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_20

更新現有角色許可界限的政策範本：

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

```
            "Action": [
                "iam:PutRolePermissionsBoundary"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_20"
                }
            }
        }
    ]
}
```

客戶必須核准此委派請求，才能更新現有角色的許可界限。

建置您的整合

了解請求生命週期

在建置整合之前，請務必了解委派請求從建立到完成的進度。

請求狀態

委派請求會進行下列狀態：

State	說明
未指派	已建立請求，但尚未與客戶帳戶和 IAM 主體建立關聯。可能已建立請求，但未指定目標帳戶，或使用目標帳戶 ID，但尚未由帳戶擁有者宣告。
已指派	與客戶帳戶相關聯並等待審核的請求
待核准	客戶已將請求轉送給管理員進行核准
已接受	請求已由客戶核准，但交換字符尚未發佈
完成	發佈給產品提供者的 Exchange 權杖。委派期間（交換字符有效性）會在請求達到完成狀態時開始
已拒絕	客戶拒絕的請求

State	說明
已過期	請求因閒置或逾時而過期

狀態轉換

正常流程（核准路徑）

- 未指派 → 已指派：客戶將請求與其帳戶建立關聯
- 已指派 → 已接受或已指派 → 待核准：客戶直接核准請求，或轉送給管理員以供檢閱
- 待核准 → 已接受：管理員核准請求
- 已接受 → 已完成：客戶發行交換字符

拒絕路徑

- 已指派 → 已拒絕：客戶拒絕請求
- 待核准 → 已拒絕：管理員拒絕請求
- 已接受 → 已拒絕：客戶在釋出權杖之前撤銷核准

過期路徑

如果在指定的時間範圍內未採取任何動作，請求會自動過期：

- 未指派 → 已過期 (1 天)
- 指派 → 已過期 (7 天)
- 待核准 → 已過期 (7 天)
- 已接受 → 已過期 (7 天)
- 拒絕 → 已過期 (7 天)
- 完成 → 已過期 (7 天)

終端機狀態

下列狀態為終端機（沒有進一步轉換）：

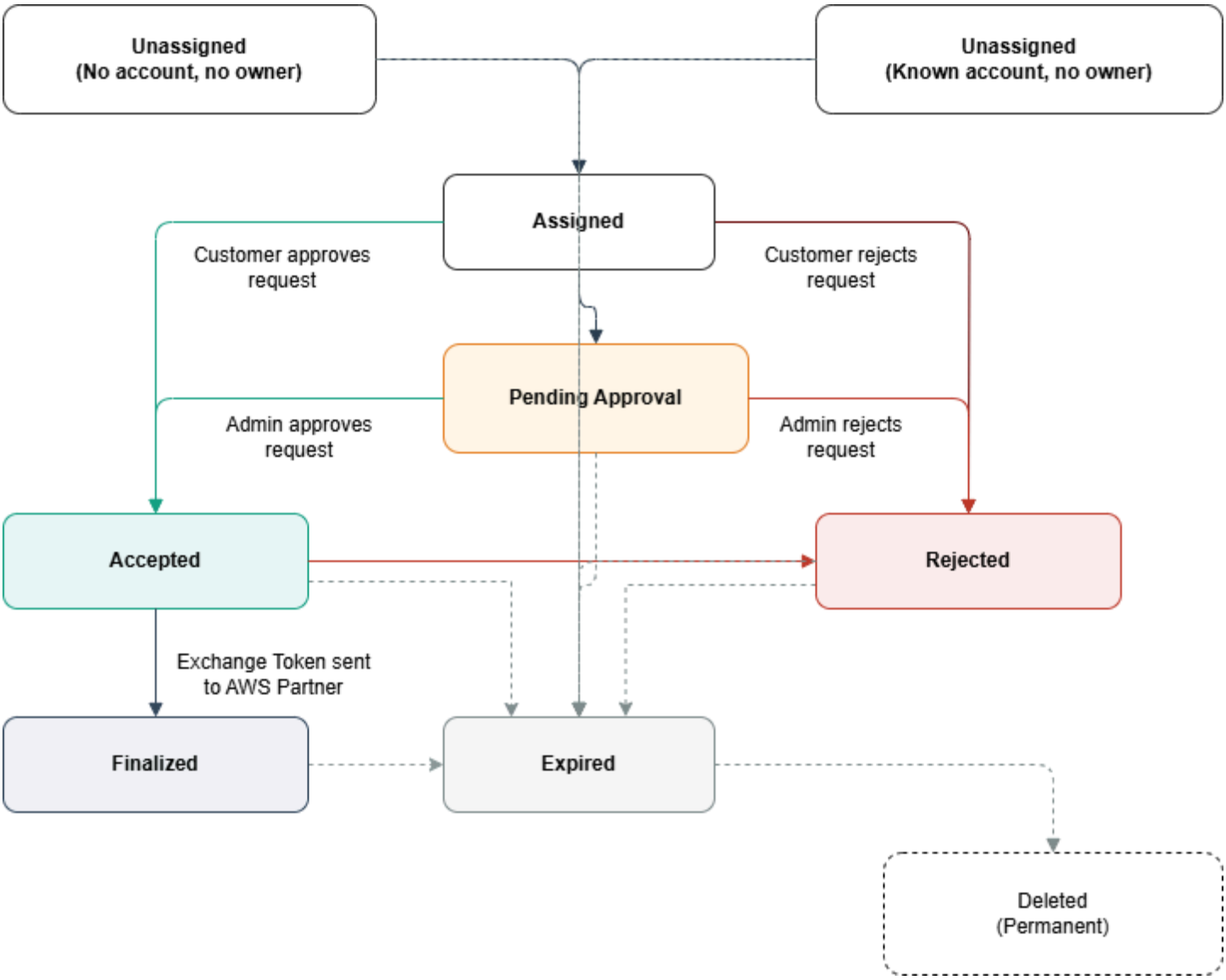
- 完成：已傳送交換字符

- 已拒絕：已拒絕請求
- 已過期：請求逾時或委派期間已結束

過期的請求最終會在保留期間之後從系統中刪除。

管理應用程式中的委派請求狀態

身為合作夥伴，您必須追蹤系統中的委派請求狀態，並將其呈現給客戶。當您收到狀態變更的 SNS 通知時，請將這些更新存放在後端，並在面向客戶的 UI 中反映這些更新。請特別注意待核准狀態 - 當客戶轉送請求給管理員以供檢閱時， 會 AWS 傳送待核准通知給您。請求可保持此狀態長達 7 天，同時等待管理員動作。在此期間，向客戶顯示他們的請求正在等待您應用程式中的管理員核准。考慮提供 AWS 主控台的深層連結，客戶可以在其中檢查請求狀態或追蹤管理員。正確處理後端中的狀態機器，並在每個階段向客戶提供正確的狀態資訊，對於良好的整合體驗至關重要。



設定 通知

IAM 使用 Amazon Simple Notification Service (SNS) 將委派請求狀態變更傳達給您。建立委派請求時，您必須從已註冊的 AWS 帳戶提供 SNS 主題 ARN。IAM 會將重要事件的訊息發佈至此主題，包括客戶核准或拒絕請求的時間，以及交換字符何時準備就緒。

> **Note**
>
> SNS 主題不能位於選擇加入 AWS 區域。您的 SNS 主題必須位於預設啟用 AWS 的區域。如需選擇加入區域的清單，請參閱 AWS 帳戶管理指南中的管理 AWS 區域。

SNS 主題組態

若要接收委派請求通知，您必須設定 SNS 主題，以授予 IAM 發佈訊息的許可。將下列政策陳述式新增至 SNS 主題政策：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIAMServiceToPublish",
            "Effect": "Allow",
            "Principal": {
                "Service": "iam.amazonaws.com"
            },
            "Action": "SNS:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:TOPIC-NAME"
        }
    ]
}
```

> **Important**
>
> SNS 主題必須位於其中一個已註冊 AWS 帳戶中。IAM 不接受來自其他帳戶的 SNS 主題。如果主題政策未正確設定，您將不會收到狀態變更通知或交換字符。

通知類型

IAM 傳送兩種類型的通知：

StateChange 通知

當委派請求轉換為新狀態（已指派、待核准、已接受、已完成、已拒絕、已過期）時傳送。

ExchangeToken 通知

當客戶釋出委派字符（狀態完成）時傳送。此通知包含您取得憑證所需的交換字符。

通知狀態

您將會收到下列委派請求狀態的通知：

State	通知類型	說明
已指派	狀態變更	請求已與客戶帳戶相關聯
待核准	狀態變更	客戶已將請求轉送給管理員進行核准
已接受	狀態變更	客戶已核准請求，但尚未釋出字符
已定案	狀態變更	客戶已釋出交換字符
已定案	ExchangeToken	此通知包含 Exchange Token
REJECTED	狀態變更	客戶已拒絕請求
已過期	狀態變更	請求在完成之前已過期

通知訊息格式

IAM 發佈標準 SNS 通知。委派請求資訊包含在訊息欄位中，做為 JSON 字串。

常見欄位（所有通知）

欄位	Type	說明
Type	String	"StateChange" 或 "ExchangeToken"
RequestId	String	IAM 委派請求 ID

欄位	Type	說明
RequestorWorkflowId	String	您在建立請求時提供的工作流程 ID
State	String	請求的目前狀態
OwnerAccountId	String	客戶的 AWS 帳戶 ID
UpdatedAt	String	狀態變更時的時間戳記 (ISO 8601 格式)

其他欄位 （ 僅限 ExchangeToken 通知 ）

欄位	Type	說明
ExchangeToken	String	使用 AWS STS GetDelegatedAccessToken API 交換登入資料的字符
ExpiresAt	String	委派存取過期時 (ISO 8601 格式)

範例通知

StateChange 通知

```
{
  "Type": "Notification",
  "MessageId": "61ee8ad4-6eec-56b5-8f3d-eba57556aa13",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestorWorkflowId\":\"workflow-12345\",\"Type\":\"StateChange\",
\"RequestId\":\"dr-abc123\",\"State\":\"ACCEPTED\",\"OwnerAccountId\":\"111122223333\",
\"UpdatedAt\":\"2025-01-15T10:30:00.123Z\"}",
  "Timestamp": "2025-01-15T10:30:00.456Z",
  "SignatureVersion": "1",
  "Signature": "...",
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
}
```

ExchangeToken 通知

```
{
  "Type": "Notification",
  "MessageId": "e44e5435-c72c-5333-aba3-354406782f5b",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestId\":\"dr-abc123\",\"RequestorWorkflowId\":\"workflow-12345\",
\"State\":\"FINALIZED\",\"OwnerAccountId\":\"111122223333\",\"ExchangeToken\":
\"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...\",\"ExpiresAt\":\"2025-01-15T18:30:00.123Z\",
\"UpdatedAt\":\"2025-01-15T10:30:00.456Z\",\"Type\":\"ExchangeToken\"}",
  "Timestamp": "2025-01-15T10:30:00.789Z",
  "SignatureVersion": "1",
  "Signature": "...",
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
}
```


交換字符

當客戶接受並完成委派請求時，IAM 會發出交換字符或權杖的交易。產品提供者使用此交換或權杖交易來呼叫 AWS STS GetDelegatedAccessToken API，以取得具有客戶核准許可的臨時 AWS 登入資料。交換字符本身不會授予 AWS 資源的存取權；必須透過 AWS STS 交換實際登入資料。

交換字符只能由建立委派請求的產品提供者帳戶兌換。請求帳戶內嵌在字符中，確保只有授權的產品提供者可以取得登入資料來存取客戶帳戶。

存取持續時間

委派期間會在客戶釋出交換字符時開始，而不是產品提供者兌換權杖時。在客戶釋出字符之後：

- 產品提供者透過 SNS 通知接收字符
- 他們可以立即將其交換為登入資料
- 登入資料過期時間：發行時間 + 核准的持續時間
- 產品提供者可以在過期前多次交換字符，以在需要時取得新的登入資料

多個兌換

產品提供者可以在有效期間多次交換字符，以取得新的登入資料。不過，從相同交換字符取得的所有登入資料都會同時過期，取決於您何時釋出字符。

範例：如果您核准 2 小時委派請求，並在上午 10：00 釋出權杖：

權杖發行時間	字符交換時間	登入資料過期	可用時間
上午 10：00	上午 10：00	12:00 PM	2 小時
上午 10：00	上午 10：20	12:00 PM	1 小時 40 分鐘
上午 10：00	上午 11：40	12:00 PM	20 分鐘
上午 10：00	中午 12：10	失敗（權杖過期）	0 分鐘

如表格所示，稍後在有效期間交換字符會導致產品供應商的可用時間變少。

帳戶存取管理員

什麼是帳戶存取管理員？

帳戶存取管理員可讓您將 AWS 帳戶 存取權指派給 IAM Identity Center 組織執行個體中的使用者和群組。您可以使用 中的 IAM 角色來指派存取權 AWS 帳戶。您可以搭配 IAM Identity Center 許可集或單獨使用帳戶存取管理員。帳戶存取管理員和 IAM Identity Center 許可集都會使用您從身分來源同步或直接在 IAM Identity Center 中建立的使用者和群組。

為什麼要使用帳戶存取管理員？

帳戶存取管理員可讓您將現有的 IAM 角色指派給您從公司身分來源同步至 IAM Identity Center 的使用者和群組。使用現有的 IAM 角色可讓您更靈活地自訂使用者存取權，包括：

- 設定 IAM 角色標籤，並針對屬性型存取控制 (ABAC) 使用工作階段標籤。
- 設定 IAM 角色信任政策以限制角色假設，或使用 IdP 宣告作為信任政策條件。
- 設定 IAM 角色路徑，以邏輯方式將角色分組並簡化管理。如需詳細資訊，請參閱設定 IAM 角色路徑以邏輯方式將角色分組並簡化管理。

下表比較三個帳戶存取選項：

	IAM 角色 + 帳戶存取管理員	IAM Identity Center 許可集	IAM 角色 + IAM 聯合（直接角色假設）
最適合	不同帳戶的自訂 IAM 角色	統一的基準人工存取，例如唯讀和佈建為不可變 IAM 角色的管理員	適合工作負載
Scope (範圍)	多個帳戶 (AWS 組織）	多個帳戶 (AWS 組織）	每個帳戶
IAM 角色佈建	需要基礎設施作為程式碼 (IaC) 或透過 IAM AWS 主控台手動佈建	自動佈建	需要基礎設施做為程式碼 (IaC)，或透過

	IAM 角色 + 帳戶存取管理員	IAM Identity Center 許可集	IAM 角色 + IAM 聯合（直接角色假設）
			AWS IAM 和外部 IdP 主控台手動佈建
Role-to-identity映射位於	帳戶存取管理員	IAM Identity Center	外部 IdP
最終使用者體驗	使用者以 AWS IAM Identity Center 使用者 AWS 身分登入後，透過 AWS 帳戶存取入口網站 URL 存取帳戶。	使用者以 IAM Identity Center 使用者 AWS 身分登入後，透過 AWS 存取入口網站存取 AWS 帳戶。	使用者透過 AWS 其 IdP 入口網站中的帳戶特定 SAML 應用程式（圖示）存取帳戶。
AWS CLI 使用者體驗	1. 使用者登入 AWS ，然後在 Web 瀏覽器中存取具有特定角色的帳戶。 2. 命令列aws login上的使用者類型，可從瀏覽器擷取 IAM 角色工作階段憑證。在 IAM 角色工作階段期間，使用者可以繼續使用 ，AWS CLI 而無需在步驟 1 中重新驗證身分。	使用者透過輸入 來設定包含所需帳戶和角色對的設定檔aws configure sso。使用設定檔aws sso login啟動新工作階段的使用者類型。如果沒有作用中的 IAM Identity Center 工作階段，使用者會在瀏覽器中驗證並授權存取。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南。	使用 AWS SDK 憑證提供者的自訂整合

關鍵功能

帳戶存取入口網站

帳戶存取管理員為使用者提供帳戶存取入口網站，他們可以在其中透過帳戶存取管理員檢視和存取 AWS 帳戶 指派給他們的 。

CLI 存取

在瀏覽器 AWS 中登入後 AWS CLI ，您的人力資源可以使用 aws login命令 AWS 帳戶 透過 存取 。

帳戶存取 APIs

帳戶存取管理員有自己的 API 命名空間 `account-access`，與 IAM API 分開。您可以在 AWS CLI 和 SDKs 中使用此命名空間。

帳戶存取管理員入門

以下概述如何開始使用帳戶存取管理員。這會假設您要指派的 IAM 角色已存在於各自的 中 AWS 帳戶。如果這些角色不存在，請參閱使用自訂信任政策建立角色，以取得如何建立新角色的資訊。

開始使用帳戶存取管理員

1. 請確定您已啟用 IAM Identity Center 的組織執行個體，並在那裡佈建您的人力資源使用者和群組。如需詳細資訊，請參閱 AWS IAM Identity Center 《 使用者指南》中的 IAM Identity Center 入門一節。如需詳細資訊，請參閱本主題the section called “先決條件和考量事項”稍後的 。

   > **Note**
   >
   > 如果您使用複寫至多個 AWS 區域的 IAM Identity Center，或搭配客戶受管金鑰 (CMK)，則必須新增帳戶存取管理員在連線至 IAM Identity Center 時使用此 CMK 的`kms:Decrypt`許可。如需詳細資訊，請參閱AWS IAM Identity Center 《 使用者指南》中的基準 KMS 金鑰政策。

2. 請確定您具有啟用帳戶存取管理員和管理其中帳戶指派的許可。如需詳細資訊，請參閱the section called “帳戶存取管理員的身分和存取管理”。
3. 透過主控台或 API 啟用帳戶存取管理員。
4. 更新您的防火牆和閘道以存取帳戶存取入口網站。
5. 透過更新帳戶存取管理員的信任政策，授予 IAM 角色的存取權。
6. 從 IAM Identity Center 開始將帳戶存取權指派給您的人力資源使用者和群組。
7. 讓您的使用者了解如何 AWS 帳戶使用帳戶存取管理員角色指派來存取 。

先決條件和考量事項

適用下列先決條件和考量事項：

- 帳戶存取管理員在預設啟用 AWS 的商業區域中可用。不支援選擇加入商業區域、 GovCloud (US)、中國和其他非商業區域。
- 在啟用時，帳戶存取管理員會連線至 IAM Identity Center 的單一組織執行個體。您必須事先啟用 AWS 組織和 IAM Identity Center 的組織執行個體。不支援 IAM Identity Center 的帳戶執行個體。
- 我們建議您在啟用 IAM Identity Center 執行個體的相同 AWS 區域中啟用帳戶存取管理員。如果您尚未完成，請參閱 AWS IAM Identity Center 《使用者指南》中的選擇 AWS 區域的考量事項。
- IAM Identity Center 的組織執行個體和帳戶存取管理員都必須在組織的管理帳戶中啟用。

啟用帳戶存取管理員

建立 IAM Identity Center 的組織執行個體之後，您可以透過其主控台或 API 啟用帳戶存取管理員。您必須在與 IAM Identity Center 執行個體相同的 AWS 帳戶 和 區域中執行此操作。

啟用帳戶存取管理員之前，請從 AWS 組織管理帳戶執行下列命令，以授予帳戶存取管理員存取您的組織：

```
aws organizations enable-aws-service-access --service-principal account-
access.amazonaws.com
```

> **Note**
>
> 雖然帳戶存取管理員主控台是 IAM 主控台的一部分，但其 API 位於單獨的命名空間中。

使用下列其中一種方法來啟用帳戶存取管理員：

Console

啟用帳戶存取管理員

1. 登入您組織的管理帳戶。
2. 在 AWS Identity and Access Management 主控台導覽窗格中，選擇帳戶存取管理員。
3. 確認主控台區域已設定為 IAM Identity Center 執行個體的主要區域。

4. 如果 IAM Identity Center 沒有組織執行個體，系統會提示您現在啟用一個執行個體，然後再繼續。否則，請選擇啟用帳戶存取管理員。

AWS CLI

首先尋找 IAM Identity Center 執行個體的 ARN，並確認區域碼符合 IAM Identity Center 的主要區域。然後，執行下列命令：

```
aws account-access create-application \
  --region <Region> \
  --identity-source '{
    "identityCenter": {
      "instanceArn": "<IAM_IDENTITY_CENTER_INSTANCE_ARN>"
    }
  }'
```

更新防火牆和閘道

帳戶存取入口網站為您的使用者提供透過帳戶存取管理員指派給他們的 的 AWS 帳戶 單一登入存取。

如果您使用新一代防火牆 (NGFW) 或安全 Web Gateway (SWG) 等 Web 內容篩選解決方案來篩選特定 AWS 網域或 URL 端點的存取，則必須允許列出與帳戶存取入口網站相關聯的網域和 URL 端點。

下列清單提供要新增至 Web 內容篩選解決方案允許清單的雙堆疊網域和 URL 端點。帳戶存取管理員沒有IPv4-only端點。

雙堆疊允許清單

帳戶存取入口網站 URL：

- `https://[Tenant-ID].account-access.[Region].app.aws`

帳戶存取入口網站呼叫APIs：

- `[Tenant-ID].account-access.[Region].app.aws/api/*`
- `[Tenant-ID].account-access.[Region].app.aws/auth/*`

租用戶 ID 是與帳戶存取管理員相關聯的唯一隨機識別符。若要尋找其值，請導覽至帳戶存取管理員主控台中的設定索引標籤，然後在執行個體詳細資訊區段中查詢應用程式 URL。Tenant-ID 是 URL 中的第一個子網域。

帳戶存取管理員服務端點：

- 控制平面： `account-access.[Region].api.aws`
- 執行時間：`account-access-runtime.[Region].api.aws`
- 主控台：`console.aws.amazon.com/account-access`

如果您使用 AWS CLI，您可以在組織的管理或委派管理員帳戶中執行下列 CLI 命令來尋找租戶 ID。

```
aws account-access list-applications
```

回應包含帳戶存取管理員的相關資訊，包括應用程式 ARN 和租戶 ID。

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

允許列出網域和 URL 端點的考量事項

除了帳戶存取入口網站的允許清單要求之外，您使用的其他服務和應用程式可能需要網域的允許清單。

- 您需要允許列出 AWS 存取入口網站及其相依性。如需詳細資訊，請參閱 AWS IAM Identity Center《使用者指南》中的更新防火牆和閘道以允許存取 AWS 入口網站。
- 若要存取 AWS 帳戶 AWS 管理主控台和帳戶存取管理員主控台，您必須允許列出其他網域。如需 AWS 管理主控台 網域清單，請參閱AWS 管理主控台 《入門指南》中的故障診斷。

刪除帳戶存取管理員

刪除帳戶存取管理員時，其中的所有資料都會遭到刪除且無法復原。

使用下列程序刪除帳戶存取管理員。

Console

刪除您的帳戶存取管理員執行個體

1. 登入您組織的管理帳戶。
2. 在 AWS Identity and Access Management 主控台導覽窗格中，選擇帳戶存取管理員。
3. 確認主控台區域已設定為帳戶存取管理員執行個體的主要區域。
4. 選擇 Settings (設定) 標籤。
5. 在刪除帳戶存取管理員區段中，選擇刪除。
6. 對話方塊會提示您確認。準備就緒時，選擇刪除。

AWS CLI

執行下列命令來刪除帳戶存取管理員：

```
aws account-access delete-application \
  --region "<Region>" \
  --application-arn "<account_access_manager_ARN>"
```

⚠ Warning

刪除帳戶存取管理員是不可復原的，並具有下列後果：

- 所有角色指派都會永久移除。您的使用者會立即失去透過帳戶存取管理員指派 AWS 帳戶 的 IAM 角色存取 的能力。
- 所有使用者的帳戶存取管理員都會從 AWS 存取入口網站中移除。
- 透過帳戶存取管理員指派的 IAM 角色作用中工作階段會繼續，直到過期，但使用者無法啟動新的工作階段。
- 如果您稍後重新啟用帳戶存取管理員，則必須從頭開始重新建立所有指派。先前的指派無法復原。

在刪除之前，請考慮在個別帳戶中停用使用者存取權，以暫時限制存取權。

管理對 的存取 AWS 帳戶

帳戶存取管理員與 IAM Identity Center 整合 AWS Organizations ，並集中管理多個 之間現有 IAM 角色的指派， AWS 帳戶 而無需個別設定每個帳戶。使用帳戶存取管理員，您可以將 中的現有 IAM 角色指派給 IAM Identity Center AWS 帳戶 使用者和群組，以控制其對特定 的存取 AWS 帳戶。

透過帳戶存取管理員，您的團隊可以在其中建立自己的自訂 IAM 角色 AWS 帳戶，而身為管理員的您，可以使用帳戶存取管理員集中管理 IAM Identity Center 使用者和群組對這些 IAM 角色的指派。

Note

在帳戶存取管理員的內容中，使用者和群組一詞僅指 IAM Identity Center 中的人力資源使用者和群組。

委派的管理

當您啟用帳戶存取管理員時， 會在 的管理帳戶中 AWS 建立它， AWS Organizations 因為帳戶存取管理員會管理整個 AWS Organization 的 IAM 角色指派。您可以選擇將帳戶存取管理員的管理委派給其中的成員帳戶 AWS Organizations。

如果您選擇將成員帳戶註冊為帳戶存取管理員的委派管理員，則此帳戶中的使用者可執行大多數帳戶存取管理員管理任務。下表指出組織管理帳戶、委派管理成員帳戶或其他成員帳戶中的使用者是否可以執行帳戶存取管理員管理任務。IAM 角色管理會從管理任務清單中排除，因為它發生在帳戶存取管理員之外。

帳戶存取管理員管理任務	管理帳戶	委派管理員帳戶	成員帳戶
管理 IAM 角色指派*	是	是	否
啟用帳戶存取管理員	是	否	否
刪除帳戶存取管理員	是	否	否

帳戶存取管理員管理任務	管理帳戶	委派管理員帳戶	成員帳戶
在管理帳戶中啟用或停用使用者存取	是	否	否
在成員帳戶中啟用或停用使用者存取	是	是	否
以委派管理員身分註冊或取消註冊成員帳戶	是	否	否

*請參閱有關將使用者和群組指派至管理帳戶之委派管理的最佳實務。

註冊成員帳戶

若要設定委派管理，您必須先將組織中的成員帳戶註冊為委派管理員。該成員帳戶中具有足夠許可的使用者將具有帳戶存取管理員的管理存取權。

帳戶存取管理員支援一次只將一個成員帳戶註冊為委派管理員。您只能在使用來自管理帳戶的登入資料登入時註冊成員帳戶。

使用下列程序，將 AWS 組織中的成員帳戶註冊為委派管理員。

註冊成員帳戶

1. AWS 管理主控台 使用 管理帳戶的登入資料登入 AWS Organizations。執行 RegisterDelegatedAdministrator API 需要管理帳戶登入資料。
2. 選取已啟用帳戶存取管理員的區域，然後開啟 AWS Identity and Access Management 主控台。
3. 選擇帳戶存取管理員，然後選擇設定索引標籤。
4. 在委派管理員區段中，選擇註冊。
5. 在註冊委派管理員頁面上，選取 AWS 帳戶 您要註冊的 ，然後選擇註冊。

取消註冊成員帳戶

您只能在使用來自管理帳戶的登入資料登入時取消註冊成員帳戶。

使用下列程序取消註冊 AWS 組織中先前指定為委派管理員的成員帳戶。

> **Important**
>
> 當您取消註冊帳戶時，您會移除所有管理員使用者從該帳戶管理帳戶存取管理員的能力。他們無法再從此帳戶管理 AWS 帳戶 存取權。
> 此操作不會影響帳戶存取管理員中設定的許可或指派。您的最終使用者會繼續 AWS 帳戶 從帳戶存取入口網站存取他們的 。

取消註冊成員帳戶

1. AWS 管理主控台 使用 管理帳戶的登入資料登入 AWS Organizations。執行 DeregisterDelegatedAdministrator API 需要管理帳戶登入資料。
2. 選取已啟用帳戶存取管理員的區域，然後開啟 AWS Identity and Access Management 主控台。
3. 選擇帳戶存取管理員，然後選擇設定索引標籤。
4. 在委派管理員區段中，選擇取消註冊帳戶。
5. 在取消註冊委派管理員對話方塊中，檢閱安全性影響，然後確認您了解。
6. 選擇 Deregister (取消註冊)。

檢視已註冊的委派管理員帳戶

使用下列程序來尋找 中哪個成員帳戶 AWS Organizations 已設定為帳戶存取管理員的委派管理員。

檢視已註冊的成員帳戶

1. AWS 管理主控台 使用 中管理帳戶的登入資料登入 AWS Organizations。
2. 選取已啟用帳戶存取管理員的區域，然後開啟 AWS Identity and Access Management 主控台。
3. 選擇帳戶存取管理員，然後選擇設定索引標籤。
4. 在委派管理員區段中，您可以檢視目前的委派管理員帳戶詳細資訊。

準備您的 IAM 角色

開始將 IAM 角色指派給 IAM Identity Center 使用者和群組之前，您需要：

1. 識別感興趣的帳戶中的相關 IAM 角色，以及

2. 請確定 IAM 角色的信任政策包含以下提供的陳述式。

如果角色具有現有的信任政策，請將此陳述式新增至其陳述式陣列。如果您要建立新的角色，您可以使用整個政策文件。此政策陳述式允許帳戶存取管理員代表使用者擔任角色並設定內容。Condition 區塊透過確保帳戶存取管理員只能代表您自己的帳戶和帳戶存取管理員應用程式擔任角色，提供混淆代理人保護。如需詳細資訊，請參閱更新角色信任政策。

<your-account-id> 將 取代為您的 AWS 帳戶 ID、將 <region>取代為您的 IAM Identity Center 主要區域，並將 <application-id>取代為您的 帳戶存取管理員應用程式 ID。您可以在帳戶存取管理員主控台的設定頁面上找到應用程式 ARN。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your-account-id>",
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

選用 — **sts:TagSession**。如果您希望身分提供者的上游聲明屬性（例如，群組成員資格、部門、成本中心）在帳戶存取管理員有問題的工作階段上傳播為主體標籤，請將 sts:TagSession新增至Action清單。如果沒有此動作，角色假設仍會成功；只會抑制工作階段標籤傳播。當目標帳戶中的下游 IAM 政策使用委託人標籤 (aws:PrincipalTag/<key>) 時新增它；否則請將其移出。保留相同的Condition區塊。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your-account-id>",
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

> **Note**
>
> 我們建議您更新 IAM 角色建立工作流程，以在需要時自動新增此信任政策陳述式。

指派和移除存取權

指派 的使用者或群組存取權 AWS 帳戶

您可以透過在使用者或群組、帳戶和使用者或群組可在帳戶中擔任的 IAM 角色之間建立指派， AWS 帳戶 將使用者或群組存取權指派給 。因此，單一指派是由使用者或群組 ID、帳戶 ID 和 IAM 角色 ARN 組成的三元組。

如果您想要執行大量指派，例如將相同的角色指派給多個使用者和群組，您必須個別建立每個指派。您可以使用帳戶存取 API 和 AWS SDK/CLI 編寫此程序的指令碼。

主控台

主控台提供數種不同的方式來指派新的存取權，取決於您想要先識別帳戶、使用者或群組。

> **Note**
>
> 帳戶存取管理員主控台為使用者、群組和帳戶提供搜尋功能。若要識別您要指派的角色，您必須事先找到其名稱（或 ARN)。帳戶存取管理員主控台無法偵測 中的 IAM 角色 AWS 帳戶。

對於本節中的所有程序，請登入管理帳戶（或委派管理員帳戶），選取啟用帳戶存取管理員的區域，然後在 IAM 主控台中導覽至帳戶存取管理員。

Accounts tab

從帳戶索引標籤指派使用者或群組存取權

1. 選擇 Accounts (帳戶) 標籤。
2. 如果您已經知道帳戶 ID，請選擇指派新的存取權。在指派帳戶存取頁面上，查詢所需的使用者或群組，輸入帳戶 ID 和 IAM 角色名稱，然後選擇指派存取。
3. 如果您需要先尋找帳戶，您可以 AWS 帳戶 依名稱、ID 或電子郵件地址搜尋所需的 。或者，您可以檢閱組織結構或帳戶清單來尋找相關的 AWS 帳戶。選擇帳戶前面的選項按鈕，然後指派新的存取權。在指派帳戶存取頁面上，查詢所需的使用者或群組，輸入 IAM 角色名稱，然後選擇指派存取。
4. 組織結構或帳戶清單可讓您選擇帳戶以檢視帳戶詳細資訊和帳戶存取指派。您也可以選擇從該處指派新的存取權。

Users tab

從使用者索引標籤指派使用者存取權

1. 選擇使用者索引標籤。
2. 您可以依使用者名稱和顯示名稱搜尋使用者，或在顯示的使用者清單中尋找使用者。
3. 選擇您要為其指派新存取權的使用者。
4. 下一頁顯示使用者屬性、帳戶指派和群組成員資格。
5. 選擇指派新的存取權。
6. 在指派帳戶存取頁面上，輸入帳戶 ID 和 IAM 角色名稱，然後選擇指派存取。

Groups tab

從群組索引標籤指派群組存取權

1. 在導覽窗格中，選擇帳戶存取管理員。
2. 選擇 Groups (群組) 標籤。
3. 您可以依群組名稱搜尋群組，或在顯示的群組清單中找到群組。
4. 選擇您要為其指派新存取權的群組。
5. 下一頁顯示群組屬性、帳戶指派和使用者成員。
6. 選擇指派新的存取權。
7. 在指派帳戶存取頁面上，輸入帳戶 ID 和 IAM 角色名稱，然後選擇指派存取。

AWS CLI

下列 AWS CLI 命令可用來為使用者建立帳戶指派：

```
aws account-access create-entitlement \
  --region <Region> \
  --application-arn "<Account_access_manager_ARN>" \
  --entitlement '{
    "principalRole": {
      "principal": {
        "identityCenter": {
          "userId": "<Identity_Center_User_ID>"
        }
      },
      "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
    }
  }'
```

下列 AWS CLI 命令可用來為群組建立帳戶指派：

```
aws account-access create-entitlement \
  --region <Region> \
  --application-arn "<Account_access_manager_ARN>" \
  --entitlement '{
    "principalRole": {
      "principal": {
        "identityCenter": {
          "groupId": "<IDC_GROUP_ID>"
```

```
        }
      },
      "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
    }
  }'
```

移除 的使用者或群組存取權 AWS 帳戶

您可以透過多種方式找到要移除的指派。

主控台

對於本節中的所有程序，登入管理帳戶 （或委派的管理員帳戶），選取啟用帳戶存取管理員的區域，然後在 IAM 主控台中導覽至帳戶存取管理員。

Accounts tab

從帳戶索引標籤移除使用者或群組存取權

1. 選擇 Accounts (帳戶) 標籤。
2. 尋找相關帳戶並選擇其名稱。
3. 在帳戶存取指派區段中，選擇您要移除存取權的使用者或群組，然後選擇取消指派。
4. 在模態中，選擇取消指派以確認。

Users tab

從使用者索引標籤移除使用者存取權

1. 選擇使用者索引標籤。
2. 選擇您要移除存取權的使用者。
3. 在帳戶存取指派區段中，選擇您要移除存取權的帳戶，然後選擇取消指派。
4. 在模態中，選擇取消指派以確認。

AWS CLI

若要使用 CLI 移除存取權，您必須先尋找帳戶指派 （授權），然後在下列命令中使用其 ID。您可以在本主題稍後看到如何搜尋特定帳戶、使用者、群組或角色的帳戶指派。CLI 將指派稱為權利。

```
aws account-access delete-entitlement \
```

```
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --entitlement-id "<ENTITLEMENT_ID>"
```

檢視或搜尋帳戶指派

您可以透過多種方式檢視帳戶指派。

主控台

對於本節中的所有程序，請登入管理帳戶 （或委派管理員帳戶），選取啟用帳戶存取管理員的區域，然後在 IAM 主控台中導覽至帳戶存取管理員。

Accounts tab

從帳戶索引標籤檢視使用者或群組存取權

1. 選擇 Accounts (帳戶) 標籤。
2. 根據預設，您會看到階層檢視。如有需要，您可以切換到清單檢視。您可以依帳戶的名稱、ID、電子郵件和 OU ID 來搜尋帳戶。當您找到所需的帳戶時，請選擇其名稱。
3. 在帳戶頁面上，您可以在帳戶存取指派區段中看到帳戶指派。您可以依使用者名稱、使用者 ID、群組名稱、群組 ID 和 IAM 角色 ARN 進一步篩選。

Users tab

從使用者索引標籤檢視使用者存取權

1. 選擇使用者索引標籤。
2. 依名稱或使用者 ID 搜尋使用者。當您找到所需的使用者時，請選擇使用者名稱。
3. 在使用者頁面上，您可以看到使用者的 帳戶指派。您可以依帳戶 ID、帳戶名稱和 IAM 角色 ARN 進一步篩選。

Groups tab

從群組索引標籤檢視群組存取權

1. 選擇 Groups (群組) 標籤。
2. 依名稱或群組 ID 搜尋群組。當您找到所需的群組時，請選擇群組名稱。

3. 在群組頁面上，您可以查看群組的帳戶指派。您可以依帳戶 ID、帳戶名稱和 IAM 角色 ARN 進一步篩選。

AWS CLI

下列命令說明如何使用不同的篩選條件擷取帳戶指派。CLI 將指派稱為權利。

擷取特定帳戶的前 100 個指派（權限）

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principalRole": {
        "account": "<ACCOUNT_ID>"
      }
    }' \
    --max-results 100
```

擷取特定使用者的所有指派（權限）

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principal": {
        "type": "USER",
        "id": "<USER_ID>"
      }
    }'
```

擷取特定群組的所有指派（權限）

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principal": {
        "type": "GROUP",
        "id": "<GROUP_ID>"
      }
```

```
    }'
```

工作階段和存取控制

了解工作階段

AWS 帳戶 使用帳戶存取管理員中的指派存取 的使用者會啟動下列工作階段類型：

1. IAM 角色工作階段 - 使用者擔任的 IAM 角色是臨時憑證，用於存取預設持續時間為 1 小時 AWS 的資源。
2. 帳戶存取入口網站工作階段 – 使用者在使用帳戶存取入口網站時啟動此工作階段。它以應用程式工作階段的形式由 IAM Identity Center 管理。
3. AWS 存取入口網站工作階段 （ 也稱為使用者互動式工作階段) – 此工作階段由 IAM Identity Center 管理。
4. 外部身分來源 (IdP 或 Microsoft AD) 工作階段 – 當 IAM Identity Center 執行個體連線至外部身分來源進行身分驗證時，此工作階段適用。

如需 IAM Identity Center 中使用者互動式和應用程式工作階段的詳細資訊，包括工作階段持續時間，請參閱AWS IAM Identity Center 《 使用者指南》中的了解 IAM Identity Center 中的身分驗證工作階段。

> **Important**
>
> 無論 IAM 角色中設定的工作階段持續時間上限、帳戶存取入口網站和 AWS 存取入口網站中的工作階段狀態為何，帳戶存取管理員指派的 IAM 角色工作階段都會根據預設工作階段持續時間 1 小時運作。

撤銷使用者存取權

若要撤銷透過帳戶存取管理員角色指派取得的使用者存取權，請參閱AWS IAM Identity Center 《 使用者指南》中的撤銷使用者存取權。

屬性型存取控制

您可以將屬性名稱設定為 ，其中 *TagKey* 是您要填入的工作階段標籤索引鍵`https://aws.amazon.com/SAML/Attributes/AccessControl:TagKey`，以設定身分提供者 (IdP) 透過

SAML 聲明傳送屬性做為工作階段標籤。IAM Identity Center 會將屬性名稱和值從 IdP 傳遞至 以進行政策評估。

人力存取

帳戶存取管理員可讓您將存取權映射至 中的 IAM 角色，以將 AWS 帳戶 存取權指派給 IAM Identity Center 中的人力資源使用者和群組 AWS 帳戶。如果您是初次使用 IAM Identity Center，請參閱AWS IAM Identity Center 《 使用者指南》中的設定 AWS 資源的人力資源存取權，以取得身分驗證工作階段、使用者存取管理、密碼管理和多重要素驗證的相關資訊。

使用者入口網站

若要存取指派給他們的 AWS 帳戶，您的人力資源使用者可以使用書籤直接導覽至帳戶存取入口網站，或從 AWS 存取入口網站中啟動。

您的使用者可能會與兩個使用者入口網站互動：

帳戶存取入口網站

這是入口網站 (URL)，可讓人力資源使用者使用帳戶存取管理員來存取 AWS 帳戶 指派給他們的 。此入口網站 (URL) 可以加入書籤以供直接存取，也可以在 AWS 存取入口網站中做為 AWS 受管應用程式使用。

AWS 存取入口網站

這是 AWS IAM Identity Center 使用者的主要進入點。此入口網站會顯示使用者可存取的所有 AWS 應用程式，包括帳戶存取入口網站。

> **Tip**
>
> 建議您將兩個入口網站加入書籤，以加快導覽速度。如果您使用外部身分提供者，也可以在那裡為兩個入口網站建立書籤應用程式。

帳戶存取入口網站入門

尋找帳戶存取入口網站 URL

若要尋找帳戶存取入口網站 URL，請選擇帳戶存取管理員主控台中的設定索引標籤。在執行個體詳細資訊區段中查詢應用程式 URL。URL 遵循模式 `https://[Tenant-ID].account-access.[Region].app.aws`。租用戶 ID 是與帳戶存取管理員相關聯的唯一隨機識別符。

如果您使用 AWS CLI，您可以在組織的管理或委派管理員帳戶中執行下列 CLI 命令來尋找租戶 ID。

```
aws account-access list-applications
```

回應包含帳戶存取管理員的相關資訊，包括應用程式 ARN 和租戶 ID。使用租戶 ID 和帳戶存取管理員的區域代碼來建構應用程式 URL。遵循本節稍早提供的模式。

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

設定虛設入口網站 URL

如需實作自訂虛設網域的指示，請參閱 AWS 安全部落格上的AWS 存取入口網站的區域路由：實作 IAM Identity Center 的自訂虛設網域。您可以根據帳戶存取入口網站的 URL 模式調整這些指示。

與您的使用者通訊

向您的人力資源使用者傳達以下內容：

- 帳戶存取入口網站 URL。
- 帳戶存取入口網站也可供您的使用者做為 AWS 存取入口網站的應用程式索引標籤中的應用程式使用。
- 如果您啟用從外部身分提供者入口網站或外部的其他入口網站啟動帳戶存取入口網站 AWS，請在通訊中包含此資訊。

使用者存取 AWS 帳戶

本主題說明使用者如何使用帳戶存取管理員來存取 AWS 帳戶 指派給他們的 。

取得您的入口網站 URL

您可以透過 AWS 帳戶 多種方式存取指派的 ：

1. 您的管理員可能會為您提供帳戶存取入口網站 URL （ 例如，`https://aa-gyxmap389.account-access.us-east-2.app.aws`或虛設 URL)。
2. 您的管理員可能會提供您主要 AWS 存取入口網站的 URL，其中會顯示指派給您的所有 AWS 應用程式，包括帳戶存取入口網站。
3. 您的組織可能已讓您從外部身分提供者入口網站或外部的其他入口網站啟動帳戶存取入口網站 AWS。

AWS 帳戶 使用 Web 瀏覽器存取

帳戶存取入口網站會顯示 AWS 帳戶 您有權存取的 ，以及您可以在每個帳戶中擔任的 IAM 角色。

在瀏覽器中存取 帳戶

1. 導覽至帳戶存取入口網站 URL，並在出現提示時，依照管理員的指示使用您的登入資料登入。

   > **Note**
   >
   > 若要改為從存取入口網站啟動帳戶 AWS 存取入口網站，請導覽至應用程式索引標籤。

2. 瀏覽器頁面會顯示 AWS 帳戶 您獲指派的 。您可以依帳戶名稱或 ID 搜尋帳戶。
3. 展開感興趣的帳戶，以查看您可以在帳戶中擔任的所有角色。
4. 選擇要使用 登入的 IAM 角色 （ 例如，開發人員 ）。系統會開啟新的瀏覽器索引標籤，讓您使用所選角色登入 帳戶。

AWS 帳戶 使用 AWS CLI AWS SDKs存取

您可以使用 AWS Command Line Interface 或 AWS 軟體開發套件 (SDKs) 搭配 IAM Identity Center 的使用者憑證，以程式設計方式存取 AWS 服務。

帳戶存取入口網站提供使用者單一登入存取權，以 AWS 帳戶 用於帳戶存取管理員中管理的帳戶指派。登入帳戶存取入口網站後，您可以取得臨時登入資料。然後，您可以使用 AWS CLI AWS SDKs中的登入資料來存取 中的資源 AWS 帳戶。

如果您使用 AWS CLI 以程式設計方式存取 AWS 服務，您可以使用本主題中的程序來啟動對 的存取 AWS CLI。如需 的相關資訊 AWS CLI，請參閱AWS Command Line Interface 《 使用者指南》。

如果您使用 AWS SDKs 以程式設計方式存取 AWS 服務，遵循本主題中的程序也會直接建立 AWS SDKs 的身分驗證。如需 AWS SDKs 的相關資訊，請參閱 AWS SDKs和工具參考指南。

AWS 帳戶 使用 存取 aws login（ 建議 ）

您可以使用 aws login命令來取得在帳戶存取管理員中指派給 AWS 帳戶 您的存取權。首先，安裝最新的 AWS CLI。然後，請遵循下列程序：

使用 存取 帳戶 AWS CLI

1. 在預設瀏覽器中， AWS 帳戶 使用帳戶其中一個指派的 IAM 角色，登入您的帳戶存取入口網站，然後從那裡登入目標。請遵循本主題稍早的程序。
2. 在命令列上執行 aws login命令。
3. 如果您尚未設定預設區域，則 會 AWS CLI 提示您指定您選擇的 AWS 區域 （ 例如 us-east-2)。AWS CLI 會記住您的選擇，以供日後使用。
4. 會在預設瀏覽器中 AWS CLI 開啟網頁，並提示您繼續執行您剛啟動或登入新工作階段的作用中工作階段。
5. 在網頁上，選擇具有您先前在瀏覽器中啟動之工作階段的圖磚。
6. 瀏覽器頁面會確認您的登入資料已成功共用，並可在工作階段過期前使用。您現在可以使用網頁關閉索引標籤，並 AWS CLI 使用此工作階段繼續使用 。

> **Note**
>
> IAM Identity Center 特定的 AWS CLI 命令，例如 aws sso configure和 aws sso login不適用於帳戶存取管理員中的帳戶指派。

AWS 帳戶 使用存取金鑰存取

您可以使用手動登入資料重新整理方法來取得特定 中指派給您的 IAM 角色的臨時登入資料 AWS 帳戶。若要這樣做，請複製並貼上臨時登入資料所需的命令。使用此方法，您必須手動重新整理暫時登入資料。

您可以執行 AWS CLI 命令，直到暫時登入資料過期為止。

若要取得手動重新整理的登入資料

1. 登入帳戶存取入口網站，如AWS 帳戶 使用 Web 瀏覽器存取中所述。
2. 找到您要 AWS 帳戶 從中擷取存取憑證的 ，並將其展開以顯示 IAM 角色名稱 （例如 Developer)。
3. 選擇所需角色旁的產生存取金鑰。

> **Note**
>
> 如果您未看到任何 AWS 帳戶 指派，則可能尚未指派給任何帳戶。在這種情況下，請聯絡您的管理員，並要求他們為您指派存取權。

4. 在取得登入資料對話方塊中，根據您安裝 的作業系統，選擇 MacOS 和 Linux、Windows 或 PowerShell 索引標籤 AWS CLI。
5. 選擇下列任一選項：

   - 選項 1：設定 AWS 環境變數

     選擇此選項可覆寫所有登入資料設定，包括登入資料檔案和組態檔案中的任何設定。如需詳細資訊，請參閱AWS Command Line Interface 《 使用者指南》中的設定 的環境變數 AWS CLI。

     若要使用此選項，請將命令複製到剪貼簿、將命令貼到您的 AWS CLI 終端機視窗，然後按 Enter 設定所需的環境變數。

   - 選項 2：在您的 AWS 服務用戶端中使用個別值

     選擇此選項可從 AWS 服務用戶端存取 AWS 資源。如需詳細資訊，請參閱要建置的工具 AWS。

     若要使用此選項，請將值複製到剪貼簿、將值貼到您的程式碼中，並將它們指派給 SDK 的適當變數。如需詳細資訊，請參閱特定 SDK API 的文件。

建立 AWS 管理主控台目的地的捷徑連結

帳戶存取入口網站提供建立可共用深層連結的選項。

建立捷徑連結

1. 登入帳戶存取入口網站時，請選擇建立捷徑連結按鈕。
2. 在對話方塊中：
   a. AWS 帳戶 使用帳戶 ID 或帳戶名稱選擇 。當您輸入 時，下拉式選單會顯示您可以存取的相符帳戶 IDs 和名稱。您只能選擇您有權存取的帳戶。
   b. （選用）從下拉式清單中選擇 IAM 角色。這些是為所選帳戶指派給您的 IAM 角色。如果您省略選擇角色，系統會提示使用者在使用捷徑連結時，為所選帳戶選取指派給他們的角色。

> **Note**
> 您無法透過捷徑連結授予新存取權。捷徑連結僅適用於已指派給使用者的帳戶和角色。

使用 URL 編碼建構安全 AWS 管理主控台捷徑連結

URL 的所有參數值，包括帳戶 ID、IAM 角色名稱和目的地 URL，都必須以 URL 編碼。

捷徑連結會以下列路徑延伸帳戶存取入口網站 URL：

```
/#/console?account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

商業區域中的完整 URL AWS 遵循此模式：

```
https://[Tenant-ID].account-access.[Region].app.aws/#/console?
account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

以下是範例捷徑連結，使用 S3FullAccess IAM 角色名稱將使用者登入帳戶 123456789012，並將其移至 S3 主控台首頁：

```
https://aa-gyxmap389.account-access.us-east-2.app.aws/#/console?
account_id=123456789012&role_name=S3FullAccess&destination=https%3A%2F
%2Fconsole.aws.amazon.com%2Fs3%2Fhome
```

考量事項

下列考量適用於捷徑連結的使用：

- 如果您與同事共用捷徑連結，他們需要擁有相同的帳戶指派，才能以相同的方式使用連結。
- 捷徑連結是區域連結。如果參考的區域發生中斷，連結可能無法運作。
- 捷徑連結包含純文字的帳戶 ID 和角色名稱。考慮您組織的政策是否限制存放此中繼資料的位置。

帳戶存取管理員的安全性

本節提供帳戶存取管理員特定的安全資訊。它補充了 中的資訊安全，涵蓋適用於整體 IAM 的安全主題。

帳戶存取管理員的身分和存取管理

下列各節說明您需要的許可：

- 在您的組織管理帳戶中管理帳戶存取管理員
- 從委派管理員帳戶管理帳戶存取管理員

從您的組織管理帳戶管理帳戶管理帳戶存取管理員

若要管理 AWS 組織管理帳戶中的帳戶存取管理員設定，您需要下列許可：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMCreateApplication",
      "Effect": "Allow",
      "Action": "account-access:CreateApplication",
      "Resource": "*"
    },
    {
      "Sid": "AAMApplicationAndEntitlementManagement",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteApplication",
        "account-access:DeleteEntitlement",
```

```
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:CreateApplication",
        "sso:DeleteApplication",
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances",
        "sso:PutApplicationAccessScope",
        "sso:PutApplicationAssignmentConfiguration",
        "sso:PutApplicationAuthenticationMethod",
        "sso:PutApplicationGrant"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
```

```
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentKMSActions",
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "*"
    },
    {
      "Sid": "CreateServiceLinkedRole",
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "arn:aws:iam::111122223333:role/aws-service-role/account-
access.amazonaws.com/AWSServiceRoleForAccountAccessManager",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "account-access.amazonaws.com"
        }
      }
    }
  ]
}
```

選用：若要account-access:CreateApplication依標籤限制，請將 AAMCreateApplication 陳述式取代為：

```
{
  "Sid": "AAMCreateApplicationConstrainedByTag",
  "Effect": "Allow",
  "Action": "account-access:CreateApplication",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:RequestTag/Environment": "Production"
    },
    "ForAllValues:StringEquals": {
      "aws:TagKeys": [
        "Environment"
      ]
    }
  }
}
```

選用：若要管理和註冊帳戶存取管理員的委派管理員，請新增：

```
{
  "Sid": "ManageDelegatedAdministrator",
  "Effect": "Allow",
  "Action": [
    "organizations:DeregisterDelegatedAdministrator",
    "organizations:RegisterDelegatedAdministrator"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "organizations:ServicePrincipal": "account-access.amazonaws.com"
    }
  }
}
```

若要從 AWS 組織管理帳戶將 IAM 角色指派給人力資源使用者和群組，而不管理帳戶存取管理員設定，您需要下列許可：

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

```
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
```

```
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

在委派的管理帳戶中管理帳戶存取管理員

若要從委派的管理帳戶管理帳戶存取管理員設定，您需要下列許可：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMDelegatedAdminManagement",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
```

```
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
```

```
      "Resource": "*"
    }
  ]
}
```

若要從委派管理帳戶將 IAM 角色指派給人力資源使用者和群組，而不管理帳戶存取管理員設定，您需要下列許可：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
```

```
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

帳戶存取管理員中的資料保護

本節補充 中的資料保護 AWS Identity and Access Management所提供的資訊，以及帳戶存取管理員特有的其他詳細資訊。

使用 IAM Identity Center 使用者和群組資料

帳戶存取管理員使用來自 IAM Identity Center 的使用者和群組資料來管理帳戶指派。不過，帳戶存取管理員只會存放和記錄 AWS產生的識別符 （使用者 ID 和群組 ID)，不會存放或記錄使用者名稱、電子郵件地址或其他個人屬性。

在帳戶存取管理員中記錄和監控

帳戶存取管理員使用自己的 API 命名空間 (`account-access`)，並透過 CloudTrail 記錄 API 呼叫。

如需有關 CloudTrail 的相關資訊，請參閱 AWS CloudTrail 使用者指南。

帳戶存取管理員中的配額和限制

下表說明帳戶存取管理員中的配額。配額增加請求必須來自管理或委派的管理員帳戶。若要增加配額，請參閱請求增加配額。

群組指派配額

資源	預設配額	可以提高
指派給 IAM 角色的群組數量上限	20	是

調節限制

資源	預設配額	可以提高
帳戶存取 API	帳戶存取 API 的集體限流限制為每秒 20 筆交易 (TPS)。對於讀取 APIs，您可以開啟支援案例來請求提高限制。寫入 API 操作限制為 15 個未完成的非同步呼叫。此限制無法提高。	是

使用 AWS SDKs IAM 程式碼範例

下列程式碼範例示範如何搭配 AWS 軟體開發套件 (SDK) 使用 IAM。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- 使用 AWS SDKs IAM 程式碼範例
  - 使用 AWS SDKs的 IAM 基本範例
    - Hello IAM
    - 使用 AWS SDK 了解 IAM 的基本概念
    - 使用 AWS SDKs的 IAM 動作
      - 搭配使用 AddClientIdToOpenIdConnectProvider 與 CLI
      - 搭配使用 AddRoleToInstanceProfile 與 CLI
      - 搭配使用 AddUserToGroup 與 CLI
      - 搭配使用 AttachGroupPolicy 與 CLI
      - AttachRolePolicy 搭配 AWS SDK 或 CLI 使用
      - AttachUserPolicy 搭配 AWS SDK 或 CLI 使用
      - ChangePassword 搭配 CLI 使用
      - CreateAccessKey 搭配 AWS SDK 或 CLI 使用
      - CreateAccountAlias 搭配 AWS SDK 或 CLI 使用
      - CreateGroup 搭配 AWS SDK 或 CLI 使用
      - CreateInstanceProfile 搭配 AWS SDK 或 CLI 使用
      - CreateLoginProfile 搭配 CLI 使用
      - CreateOpenIdConnectProvider 搭配 CLI 使用
      - CreatePolicy 搭配 AWS SDK 或 CLI 使用
      - CreatePolicyVersion 搭配 AWS SDK 或 CLI 使用
      - CreateRole 搭配 AWS SDK 或 CLI 使用
      - CreateSAMLProvider 搭配 AWS SDK 或 CLI 使用
      - CreateServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
      - CreateUser 搭配 AWS SDK 或 CLI 使用

- 搭配使用 CreateVirtualMfaDevice 與 CLI
- 搭配使用 DeactivateMfaDevice 與 CLI
- DeleteAccessKey 搭配 AWS SDK 或 CLI 使用
- DeleteAccountAlias 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteAccountPasswordPolicy 與 CLI
- DeleteGroup 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteGroupPolicy 與 CLI
- DeleteInstanceProfile 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteLoginProfile 與 CLI
- 搭配使用 DeleteOpenIdConnectProvider 與 CLI
- DeletePolicy 搭配 AWS SDK 或 CLI 使用
- DeletePolicyVersion 搭配 AWS SDK 或 CLI 使用
- DeleteRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteRolePermissionsBoundary 與 CLI
- DeleteRolePolicy 搭配 AWS SDK 或 CLI 使用
- DeleteSAMLProvider 搭配 AWS SDK 或 CLI 使用
- DeleteServerCertificate 搭配 AWS SDK 或 CLI 使用
- DeleteServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteSigningCertificate 與 CLI
- DeleteUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteUserPermissionsBoundary 與 CLI
- DeleteUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteVirtualMfaDevice 與 CLI
- 搭配使用 DetachGroupPolicy 與 CLI
- DetachRolePolicy 搭配 AWS SDK 或 CLI 使用
- DetachUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 EnableMfaDevice 與 CLI
- GenerateCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GenerateServiceLastAccessedDetails 與 CLI
- GetAccessKeyLastUsed 搭配 AWS SDK 或 CLI 使用

- GetAccountAuthorizationDetails 搭配 AWS SDK 或 CLI 使用
- GetAccountPasswordPolicy 搭配 AWS SDK 或 CLI 使用
- GetAccountSummary 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetContextKeysForCustomPolicy 與 CLI
- 搭配使用 GetContextKeysForPrincipalPolicy 與 CLI
- GetCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetGroup 與 CLI
- 搭配使用 GetGroupPolicy 與 CLI
- 搭配使用 GetInstanceProfile 與 CLI
- 搭配使用 GetLoginProfile 與 CLI
- 搭配使用 GetOpenIdConnectProvider 與 CLI
- GetPolicy 搭配 AWS SDK 或 CLI 使用
- GetPolicyVersion 搭配 AWS SDK 或 CLI 使用
- GetRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetRolePolicy 與 CLI
- 搭配使用 GetSamlProvider 與 CLI
- GetServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetServiceLastAccessedDetails 與 CLI
- 搭配使用 GetServiceLastAccessedDetailsWithEntities 與 CLI
- GetServiceLinkedRoleDeletionStatus 搭配 AWS SDK 或 CLI 使用
- GetUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetUserPolicy 與 CLI
- ListAccessKeys 搭配 AWS SDK 或 CLI 使用
- ListAccountAliases 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedGroupPolicies 與 CLI
- ListAttachedRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedUserPolicies 與 CLI
- 搭配使用 ListEntitiesForPolicy 與 CLI
- 搭配使用 ListGroupPolicies 與 CLI
- ListGroups 搭配 AWS SDK 或 CLI 使用

- 搭配使用 ListGroupsForUser 與 CLI
- 搭配使用 ListInstanceProfiles 與 CLI
- 搭配使用 ListInstanceProfilesForRole 與 CLI
- 搭配使用 ListMfaDevices 與 CLI
- 搭配使用 ListOpenIdConnectProviders 與 CLI
- ListPolicies 搭配 AWS SDK 或 CLI 使用
- ListPolicyVersions 搭配 AWS SDK 或 CLI 使用
- ListRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListRoleTags 與 CLI
- ListRoles 搭配 AWS SDK 或 CLI 使用
- ListSAMLProviders 搭配 AWS SDK 或 CLI 使用
- ListServerCertificates 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListSigningCertificates 與 CLI
- ListUserPolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListUserTags 與 CLI
- ListUsers 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListVirtualMfaDevices 與 CLI
- 搭配使用 PutGroupPolicy 與 CLI
- 搭配使用 PutRolePermissionsBoundary 與 CLI
- PutRolePolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 PutUserPermissionsBoundary 與 CLI
- PutUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 RemoveClientIdFromOpenIdConnectProvider 與 CLI
- 搭配使用 RemoveRoleFromInstanceProfile 與 CLI
- 搭配使用 RemoveUserFromGroup 與 CLI
- 搭配使用 ResyncMfaDevice 與 CLI
- SetDefaultPolicyVersion 搭配 AWS SDK 或 CLI 使用
- 搭配使用 TagRole 與 CLI
- 搭配使用 TagUser 與 CLI
- 搭配使用 UntagRole 與 CLI

- 搭配使用 UntagUser 與 CLI
- UpdateAccessKey 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateAccountPasswordPolicy 與 CLI
- 搭配使用 UpdateAssumeRolePolicy 與 CLI
- 搭配使用 UpdateGroup 與 CLI
- 搭配使用 UpdateLoginProfile 與 CLI
- 搭配使用 UpdateOpenIdConnectProviderThumbprint 與 CLI
- 搭配使用 UpdateRole 與 CLI
- 搭配使用 UpdateRoleDescription 與 CLI
- 搭配使用 UpdateSamlProvider 與 CLI
- UpdateServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateSigningCertificate 與 CLI
- UpdateUser 搭配 AWS SDK 或 CLI 使用
- UploadServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UploadSigningCertificate 與 CLI

- 使用 AWS SDKs IAM 案例
  - 使用 AWS SDK 建置和管理彈性服務
  - 設定容器服務連線
  - 為無伺服器啟動類型建立容器任務
  - 使用函數代理整合建立靜態 API
  - 使用 AWS SDK 建立唯讀和讀寫 IAM 使用者
  - 為虛擬機器執行個體建立容器服務
  - 建立受管監控工作區
  -
  - 建立您的第一個無伺服器函數
  - 使用 CLI 開始使用無伺服器資料倉儲
  - 組態管理入門
  - 開始使用物聯網裝置保護
  - 機器學習功能存放區的入門
  - 受管 kubernetes 叢集入門

- 受管串流入門
- 佈建資料倉儲叢集入門
- 工作流程協同運作入門
- 使用 SDK 管理 IAM 存取金鑰 AWS
- 使用 SDK 管理 IAM 政策 AWS
- 使用 SDK 管理 IAM 角色 AWS
- 使用 SDK 管理您的 IAM 帳戶 AWS
- 將硬式編碼秘密移至安全秘密儲存
- 允許 AWS Compute Optimizer 自動化功能套用建議的動作
- 在整個組織中啟用自動化的政策
- 為您的帳戶啟用自動化的政策
- 授予組織管理帳戶 Compute Optimizer Automation 完整存取權的政策
- 授予獨立 AWS 帳戶 Compute Optimizer Automation 完整存取權的政策
- 為組織的管理帳戶授予 Compute Optimizer Automation 唯讀存取權的政策
- 授予獨立 AWS 帳戶 Compute Optimizer Automation 唯讀存取權的政策
- 授予運算最佳化自動化服務連結角色許可的政策
- 使用 SDK 轉返 IAM 政策版本 AWS
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 使用 AWS Command Line Interface v2 設定 DynamoDB 的屬性型存取控制
- 設定系統管理
- 在監控儀表板中使用屬性變數來監控多個無伺服器函數
- 使用 AWS Command Line Interface v2 處理 DynamoDB 串流和Time-to-Live
- 使用 SDK 搭配 IAM 政策建置器 API AWS

- AWS STS 使用 AWS SDKs程式碼範例
  - AWS STS 使用 AWS SDKs的基本範例
    - AWS STS 使用 AWS SDKs的動作
      - AssumeRole 搭配 AWS SDK 或 CLI 使用
      - 搭配使用 AssumeRoleWithWebIdentity 與 CLI
      - 搭配使用 DecodeAuthorizationMessage 與 CLI
      - 搭配使用 GetFederationToken 與 CLI

    - GetSessionToken 搭配 AWS SDK 或 CLI 使用
- AWS STS 使用 AWS SDKs案例
    - 使用 SDK 搭配 擔任需要 MFA 字符 AWS STS 的 IAM 角色 AWS
    - 設定容器服務連線
    - 使用 AWS SDK AWS STS 為聯合身分使用者使用 建構 URL
    - 為無伺服器啟動類型建立容器任務
    - 使用函數代理整合建立靜態 API
    - 為虛擬機器執行個體建立容器服務
    - 建立受管監控工作區
    - 建立以函數名稱做為變數的監控儀表板
    - AWS STS 使用 AWS SDK 透過 取得需要 MFA 字符的工作階段字符
    - 組態管理入門
    - 容器登錄檔入門
    - 開始使用物聯網簡訊
    - 機器學習功能存放區的入門
    - 受管 kubernetes 叢集入門
    - 受管串流入門
    - 推送通知入門
    - 搜尋和分析引擎入門
    - Web 應用程式防火牆入門
    - 將硬式編碼秘密移至安全秘密儲存
    - 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
    - 設定系統管理

使用 AWS SDKs IAM 程式碼範例

下列程式碼範例示範如何搭配 AWS 軟體開發套件 (SDK) 使用 IAM。

基本概念是程式碼範例，這些範例說明如何在服務內執行基本操作。

~~Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但~~
您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

IAM

- 使用 AWS SDKs的 IAM 基本範例
  - Hello IAM
  - 使用 AWS SDK 了解 IAM 的基本概念
  - 使用 AWS SDKs的 IAM 動作
    - 搭配使用 AddClientIdToOpenIdConnectProvider 與 CLI
    - 搭配使用 AddRoleToInstanceProfile 與 CLI
    - 搭配使用 AddUserToGroup 與 CLI
    - 搭配使用 AttachGroupPolicy 與 CLI
    - AttachRolePolicy 搭配 AWS SDK 或 CLI 使用
    - AttachUserPolicy 搭配 AWS SDK 或 CLI 使用
    - ChangePassword 搭配 CLI 使用
    - CreateAccessKey 搭配 AWS SDK 或 CLI 使用
    - CreateAccountAlias 搭配 AWS SDK 或 CLI 使用
    - CreateGroup 搭配 AWS SDK 或 CLI 使用
    - CreateInstanceProfile 搭配 AWS SDK 或 CLI 使用
    - CreateLoginProfile 搭配 CLI 使用
    - CreateOpenIdConnectProvider 搭配 CLI 使用
    - CreatePolicy 搭配 AWS SDK 或 CLI 使用
    - CreatePolicyVersion 搭配 AWS SDK 或 CLI 使用
    - CreateRole 搭配 AWS SDK 或 CLI 使用
    - CreateSAMLProvider 搭配 AWS SDK 或 CLI 使用
    - CreateServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
    - CreateUser 搭配 AWS SDK 或 CLI 使用
    - 搭配使用 CreateVirtualMfaDevice 與 CLI
    - 搭配使用 DeactivateMfaDevice 與 CLI
    - DeleteAccessKey 搭配 AWS SDK 或 CLI 使用

- DeleteAccountAlias 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteAccountPasswordPolicy 與 CLI
- DeleteGroup 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteGroupPolicy 與 CLI
- DeleteInstanceProfile 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteLoginProfile 與 CLI
- 搭配使用 DeleteOpenIdConnectProvider 與 CLI
- DeletePolicy 搭配 AWS SDK 或 CLI 使用
- DeletePolicyVersion 搭配 AWS SDK 或 CLI 使用
- DeleteRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteRolePermissionsBoundary 與 CLI
- DeleteRolePolicy 搭配 AWS SDK 或 CLI 使用
- DeleteSAMLProvider 搭配 AWS SDK 或 CLI 使用
- DeleteServerCertificate 搭配 AWS SDK 或 CLI 使用
- DeleteServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteSigningCertificate 與 CLI
- DeleteUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteUserPermissionsBoundary 與 CLI
- DeleteUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteVirtualMfaDevice 與 CLI
- 搭配使用 DetachGroupPolicy 與 CLI
- DetachRolePolicy 搭配 AWS SDK 或 CLI 使用
- DetachUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 EnableMfaDevice 與 CLI
- GenerateCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GenerateServiceLastAccessedDetails 與 CLI
- GetAccessKeyLastUsed 搭配 AWS SDK 或 CLI 使用
- GetAccountAuthorizationDetails 搭配 AWS SDK 或 CLI 使用
- GetAccountPasswordPolicy 搭配 AWS SDK 或 CLI 使用
- GetAccountSummary 搭配 AWS SDK 或 CLI 使用

- 搭配使用 GetContextKeysForCustomPolicy 與 CLI
- 搭配使用 GetContextKeysForPrincipalPolicy 與 CLI
- GetCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetGroup 與 CLI
- 搭配使用 GetGroupPolicy 與 CLI
- 搭配使用 GetInstanceProfile 與 CLI
- 搭配使用 GetLoginProfile 與 CLI
- 搭配使用 GetOpenIdConnectProvider 與 CLI
- GetPolicy 搭配 AWS SDK 或 CLI 使用
- GetPolicyVersion 搭配 AWS SDK 或 CLI 使用
- GetRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetRolePolicy 與 CLI
- 搭配使用 GetSamlProvider 與 CLI
- GetServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetServiceLastAccessedDetails 與 CLI
- 搭配使用 GetServiceLastAccessedDetailsWithEntities 與 CLI
- GetServiceLinkedRoleDeletionStatus 搭配 AWS SDK 或 CLI 使用
- GetUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetUserPolicy 與 CLI
- ListAccessKeys 搭配 AWS SDK 或 CLI 使用
- ListAccountAliases 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedGroupPolicies 與 CLI
- ListAttachedRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedUserPolicies 與 CLI
- 搭配使用 ListEntitiesForPolicy 與 CLI
- 搭配使用 ListGroupPolicies 與 CLI
- ListGroups 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListGroupsForUser 與 CLI
- 搭配使用 ListInstanceProfiles 與 CLI
- 搭配使用 ListInstanceProfilesForRole 與 CLI

- 搭配使用 ListMfaDevices 與 CLI
- 搭配使用 ListOpenIdConnectProviders 與 CLI
- ListPolicies 搭配 AWS SDK 或 CLI 使用
- ListPolicyVersions 搭配 AWS SDK 或 CLI 使用
- ListRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListRoleTags 與 CLI
- ListRoles 搭配 AWS SDK 或 CLI 使用
- ListSAMLProviders 搭配 AWS SDK 或 CLI 使用
- ListServerCertificates 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListSigningCertificates 與 CLI
- ListUserPolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListUserTags 與 CLI
- ListUsers 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListVirtualMfaDevices 與 CLI
- 搭配使用 PutGroupPolicy 與 CLI
- 搭配使用 PutRolePermissionsBoundary 與 CLI
- PutRolePolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 PutUserPermissionsBoundary 與 CLI
- PutUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 RemoveClientIdFromOpenIdConnectProvider 與 CLI
- 搭配使用 RemoveRoleFromInstanceProfile 與 CLI
- 搭配使用 RemoveUserFromGroup 與 CLI
- 搭配使用 ResyncMfaDevice 與 CLI
- SetDefaultPolicyVersion 搭配 AWS SDK 或 CLI 使用
- 搭配使用 TagRole 與 CLI
- 搭配使用 TagUser 與 CLI
- 搭配使用 UntagRole 與 CLI
- 搭配使用 UntagUser 與 CLI
- UpdateAccessKey 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateAccountPasswordPolicy 與 CLI

- 搭配使用 UpdateAssumeRolePolicy 與 CLI
- 搭配使用 UpdateGroup 與 CLI
- 搭配使用 UpdateLoginProfile 與 CLI
- 搭配使用 UpdateOpenIdConnectProviderThumbprint 與 CLI
- 搭配使用 UpdateRole 與 CLI
- 搭配使用 UpdateRoleDescription 與 CLI
- 搭配使用 UpdateSamlProvider 與 CLI
- UpdateServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateSigningCertificate 與 CLI
- UpdateUser 搭配 AWS SDK 或 CLI 使用
- UploadServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UploadSigningCertificate 與 CLI

- 使用 AWS SDKs IAM 案例
  - 使用 AWS SDK 建置和管理彈性服務
  - 設定容器服務連線
  - 為無伺服器啟動類型建立容器任務
  - 使用函數代理整合建立靜態 API
  - 使用 AWS SDK 建立唯讀和讀寫 IAM 使用者
  - 為虛擬機器執行個體建立容器服務
  - 建立受管監控工作區
  -
  - 建立您的第一個無伺服器函數
  - 使用 CLI 開始使用無伺服器資料倉儲
  - 組態管理入門
  - 開始使用物聯網裝置保護
  - 機器學習功能存放區的入門
  - 受管 kubernetes 叢集入門
  - 受管串流入門
  - 佈建資料倉儲叢集入門
  - 工作流程協同運作入門

- 使用 SDK 管理 IAM 存取金鑰 AWS
- 使用 SDK 管理 IAM 政策 AWS
- 使用 SDK 管理 IAM 角色 AWS
- 使用 SDK 管理您的 IAM 帳戶 AWS
- 將硬式編碼秘密移至安全秘密儲存
- 允許 AWS Compute Optimizer 自動化功能套用建議的動作
- 在整個組織中啟用自動化的政策
- 為您的帳戶啟用自動化的政策
- 授予組織管理帳戶 Compute Optimizer Automation 完整存取權的政策
- 授予獨立 AWS 帳戶 Compute Optimizer Automation 完整存取權的政策
- 為組織的管理帳戶授予 Compute Optimizer Automation 唯讀存取權的政策
- 授予獨立 AWS 帳戶 Compute Optimizer Automation 唯讀存取權的政策
- 授予運算最佳化自動化服務連結角色許可的政策
- 使用 SDK 轉返 IAM 政策版本 AWS
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 使用 AWS Command Line Interface v2 設定 DynamoDB 的屬性型存取控制
- 設定系統管理
- 在監控儀表板中使用屬性變數來監控多個無伺服器函數
- 使用 AWS Command Line Interface v2 處理 DynamoDB 串流和Time-to-Live
- 使用 SDK 搭配 IAM 政策建置器 API AWS

使用 AWS SDKs的 IAM 基本範例

下列程式碼範例示範如何 AWS Identity and Access Management 搭配 AWS SDKs 使用 的基本概念。

範例

- Hello IAM
- 使用 AWS SDK 了解 IAM 的基本概念
- 使用 AWS SDKs的 IAM 動作
  - 搭配使用 AddClientIdToOpenIdConnectProvider 與 CLI
  - 搭配使用 AddRoleToInstanceProfile 與 CLI

- 搭配使用 AddUserToGroup 與 CLI
- 搭配使用 AttachGroupPolicy 與 CLI
- AttachRolePolicy 搭配 AWS SDK 或 CLI 使用
- AttachUserPolicy 搭配 AWS SDK 或 CLI 使用
- ChangePassword 搭配 CLI 使用
- CreateAccessKey 搭配 AWS SDK 或 CLI 使用
- CreateAccountAlias 搭配 AWS SDK 或 CLI 使用
- CreateGroup 搭配 AWS SDK 或 CLI 使用
- CreateInstanceProfile 搭配 AWS SDK 或 CLI 使用
- CreateLoginProfile 搭配 CLI 使用
- CreateOpenIdConnectProvider 搭配 CLI 使用
- CreatePolicy 搭配 AWS SDK 或 CLI 使用
- CreatePolicyVersion 搭配 AWS SDK 或 CLI 使用
- CreateRole 搭配 AWS SDK 或 CLI 使用
- CreateSAMLProvider 搭配 AWS SDK 或 CLI 使用
- CreateServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
- CreateUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 CreateVirtualMfaDevice 與 CLI
- 搭配使用 DeactivateMfaDevice 與 CLI
- DeleteAccessKey 搭配 AWS SDK 或 CLI 使用
- DeleteAccountAlias 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteAccountPasswordPolicy 與 CLI
- DeleteGroup 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteGroupPolicy 與 CLI
- DeleteInstanceProfile 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteLoginProfile 與 CLI
- 搭配使用 DeleteOpenIdConnectProvider 與 CLI
- DeletePolicy 搭配 AWS SDK 或 CLI 使用
- DeletePolicyVersion 搭配 AWS SDK 或 CLI 使用
- DeleteRole 搭配 AWS SDK 或 CLI 使用

- 搭配使用 DeleteRolePermissionsBoundary 與 CLI
- DeleteRolePolicy 搭配 AWS SDK 或 CLI 使用
- DeleteSAMLProvider 搭配 AWS SDK 或 CLI 使用
- DeleteServerCertificate 搭配 AWS SDK 或 CLI 使用
- DeleteServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteSigningCertificate 與 CLI
- DeleteUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteUserPermissionsBoundary 與 CLI
- DeleteUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteVirtualMfaDevice 與 CLI
- 搭配使用 DetachGroupPolicy 與 CLI
- DetachRolePolicy 搭配 AWS SDK 或 CLI 使用
- DetachUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 EnableMfaDevice 與 CLI
- GenerateCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GenerateServiceLastAccessedDetails 與 CLI
- GetAccessKeyLastUsed 搭配 AWS SDK 或 CLI 使用
- GetAccountAuthorizationDetails 搭配 AWS SDK 或 CLI 使用
- GetAccountPasswordPolicy 搭配 AWS SDK 或 CLI 使用
- GetAccountSummary 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetContextKeysForCustomPolicy 與 CLI
- 搭配使用 GetContextKeysForPrincipalPolicy 與 CLI
- GetCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetGroup 與 CLI
- 搭配使用 GetGroupPolicy 與 CLI
- 搭配使用 GetInstanceProfile 與 CLI
- 搭配使用 GetLoginProfile 與 CLI
- 搭配使用 GetOpenIdConnectProvider 與 CLI
- GetPolicy 搭配 AWS SDK 或 CLI 使用
- GetPolicyVersion 搭配 AWS SDK 或 CLI 使用

- GetRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetRolePolicy 與 CLI
- 搭配使用 GetSamlProvider 與 CLI
- GetServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetServiceLastAccessedDetails 與 CLI
- 搭配使用 GetServiceLastAccessedDetailsWithEntities 與 CLI
- GetServiceLinkedRoleDeletionStatus 搭配 AWS SDK 或 CLI 使用
- GetUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetUserPolicy 與 CLI
- ListAccessKeys 搭配 AWS SDK 或 CLI 使用
- ListAccountAliases 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedGroupPolicies 與 CLI
- ListAttachedRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedUserPolicies 與 CLI
- 搭配使用 ListEntitiesForPolicy 與 CLI
- 搭配使用 ListGroupPolicies 與 CLI
- ListGroups 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListGroupsForUser 與 CLI
- 搭配使用 ListInstanceProfiles 與 CLI
- 搭配使用 ListInstanceProfilesForRole 與 CLI
- 搭配使用 ListMfaDevices 與 CLI
- 搭配使用 ListOpenIdConnectProviders 與 CLI
- ListPolicies 搭配 AWS SDK 或 CLI 使用
- ListPolicyVersions 搭配 AWS SDK 或 CLI 使用
- ListRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListRoleTags 與 CLI
- ListRoles 搭配 AWS SDK 或 CLI 使用
- ListSAMLProviders 搭配 AWS SDK 或 CLI 使用
- ListServerCertificates 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListSigningCertificates 與 CLI

- ListUserPolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListUserTags 與 CLI
- ListUsers 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListVirtualMfaDevices 與 CLI
- 搭配使用 PutGroupPolicy 與 CLI
- 搭配使用 PutRolePermissionsBoundary 與 CLI
- PutRolePolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 PutUserPermissionsBoundary 與 CLI
- PutUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 RemoveClientIdFromOpenIdConnectProvider 與 CLI
- 搭配使用 RemoveRoleFromInstanceProfile 與 CLI
- 搭配使用 RemoveUserFromGroup 與 CLI
- 搭配使用 ResyncMfaDevice 與 CLI
- SetDefaultPolicyVersion 搭配 AWS SDK 或 CLI 使用
- 搭配使用 TagRole 與 CLI
- 搭配使用 TagUser 與 CLI
- 搭配使用 UntagRole 與 CLI
- 搭配使用 UntagUser 與 CLI
- UpdateAccessKey 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateAccountPasswordPolicy 與 CLI
- 搭配使用 UpdateAssumeRolePolicy 與 CLI
- 搭配使用 UpdateGroup 與 CLI
- 搭配使用 UpdateLoginProfile 與 CLI
- 搭配使用 UpdateOpenIdConnectProviderThumbprint 與 CLI
- 搭配使用 UpdateRole 與 CLI
- 搭配使用 UpdateRoleDescription 與 CLI
- 搭配使用 UpdateSamlProvider 與 CLI
- UpdateServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateSigningCertificate 與 CLI
- UpdateUser 搭配 AWS SDK 或 CLI 使用

- UploadServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UploadSigningCertificate 與 CLI

Hello IAM

下列程式碼範例示範如何開始使用 IAM。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
```

```
        });
    }
}
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListPolicies。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

CMakeLists.txt CMake 檔案的程式碼。

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

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

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

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

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

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

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

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

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

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

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

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

iam.cpp 來源檔案的程式碼。

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
```

```
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
 policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
 std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
 <<
                          std::endl;
```

```
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
    }


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

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API 參考中的 ListPolicies。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
package main

import (
 "context"
 "fmt"

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

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
```

```
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListPolicies。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
import java.util.List;

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

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
```

```
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 ListPolicies。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      for (const policy of page.Policies) {
        console.log(`${policy.PolicyName}`);
        policyCount++;
      }
    }
  }
  console.log(`Found ${policyCount} policies.`);
```

```
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 ListPolicies。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import boto3


def main():
    """
    Lists the managed policies in your AWS account using the AWS SDK for Python
 (Boto3).
    """
    iam = boto3.client("iam")

    try:
        # Get a paginator for the list_policies operation
        paginator = iam.get_paginator("list_policies")

        # Iterate through the pages of results
        for page in paginator.paginate(Scope="All", OnlyAttached=False):
            for policy in page["Policies"]:
                print(f"Policy name: {policy['PolicyName']}")
                print(f"  Policy ARN: {policy['Arn']}")
    except boto3.exceptions.BotoCoreError as e:
        print(f"Encountered an error while listing policies: {e}")


if __name__ == "__main__":
```

```
    main()
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListPolicies。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
require 'aws-sdk-iam'
require 'logger'

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

  # Lists and prints all IAM policies in the current AWS account.
  def list_policies
    @logger.info('Here are the IAM policies in your account:')

    paginator = @client.list_policies
    policies = []

    paginator.each_page do |page|
      policies.concat(page.policies)
    end

    if policies.empty?
      @logger.info("You don't have any IAM policies.")
    else
```

```
      policies.each do |policy|
        @logger.info("- #{policy.policy_name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  iam_client = Aws::IAM::Client.new
  manager = IAMManager.new(iam_client)
  manager.list_policies
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListPolicies。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

來自 src/bin/hello.rs。

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}
```

```
#[tokio::main]
async fn main() -> Result<(), Box<SdkError<ListPoliciesError>>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

來自 src/iam-service-lib.rs。

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API 參考中的 ListPolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 了解 IAM 的基本概念

下列程式碼範例示範如何建立使用者並擔任角色。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

- 建立沒有許可的使用者。
- 建立一個可授予許可的角色，以列出帳戶的 Amazon S3 儲存貯體。
- 新增政策，讓使用者擔任該角色。
- 使用暫時憑證，擔任角色並列出 Amazon S3 儲存貯體，然後清理資源。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
```

```
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
```

```
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
```

```
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
```

```
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
```

```
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
   /// <summary>
   /// Delete an IAM user policy.
   /// </summary>
   /// <param name="policyName">The name of the IAM policy to delete.</param>
   /// <param name="userName">The username of the IAM user.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
   {
       var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
       {
           PolicyArn = policyArn,
           RoleName = roleName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Gets the IAM password policy for an AWS account.
   /// </summary>
   /// <returns>The PasswordPolicy for the AWS account.</returns>
   public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
   {
```

```
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
```

```
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }
```

```
        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
```

```
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
```

```
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;
```

```
public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();
```

```
        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;
```

```
        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
```

```
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }
```

```
        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
```

```
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
stsService)
```

```
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
```

```
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
```

```
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
##############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
```

```
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
```

```
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
```

```
 fi

 local assume_role_policy_document="{
               \"Version\": \"2012-10-17\",
               \"Statement\": [{
                   \"Effect\": \"Allow\",
                   \"Action\": \"sts:AssumeRole\",
                   \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
```

```
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

此案例中使用的 IAM 函數。

```
###############################################################################
# function iam_user_exists
#
```

```
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
```

```
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
```

```
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

#############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_user_access_key"
  echo "Creates an AWS Identity and Access Management (IAM) key pair."
  echo "  -u user_name   The name of the IAM user."
  echo "  [-f file_name]   Optional file name for the access key output."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    f) file_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
```

```
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }
```

```
# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
  return 1
fi
```

```
  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
```

```
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
```

```
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
```

```
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi

iecho "delete-access-key response:$response"
iecho
```

```
  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;
```

```
    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }

        user = outcome.GetResult().GetUser();
    }
```

```
    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
```

```
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
```

```
                        listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
```

```
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
```

```
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }
```

```
    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- 如需 API 詳細資訊，請參閱《適用於 C++ 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy

- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在命令提示中執行互動式案例。

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/aws-sdk-go-v2/service/s3"
```

```
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
```

```
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

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

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
```

```
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
```

```
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
```

```
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
```

```
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
```

```
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
  if err != nil {
```

```
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
 *key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
  log.Println(strings.Repeat("-", 88))
 }

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
 Prefix string
```

```
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

定義包裝帳號動作的結構。

```
import (
 "context"
 "log"

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

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
```

```
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

定義包裝政策動作的結構。

```
import (
 "context"
 "encoding/json"
 "log"

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

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
```

```
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
```

```
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
```

```
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

定義包裝角色動作的結構。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
```

```
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

```
// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

```
// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

定義包裝使用者動作的結構。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
```

```
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
```

```
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

```
// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
```

```
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 使用者動作的函數。

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
```

```
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

   public static String userArn;

   public static void main(String[] args) throws Exception {

       final String usage = """

               Usage:
                   <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

               Where:
                   username - The name of the IAM user to create.\s
                   policyName - The name of the policy to create.\s
                   roleName - The name of the role to create.\s
                   roleSessionName - The name of the session required for the
assumeRole operation.\s
                   bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
               """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
       String policyName = args[1];
       String roleName = args[2];
       String roleSessionName = args[3];
       String bucketName = args[4];

       Region region = Region.AWS_GLOBAL;
       IamClient iam = IamClient.builder()
               .region(region)
               .build();

       System.out.println(DASHES);
       System.out.println("Welcome to the AWS IAM example scenario.");
       System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();
```

```
            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

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

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
```

```
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

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

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }
```

```
            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

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

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();
```

```
            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);
```

```
            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

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

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

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

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
```

```
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

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

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立一個可授予許可的 IAM 使用者和角色，以列出 Amazon S3 儲存貯體。使用者只有擔任該角色的權利。擔任角色後，請使用暫時性憑證列出該帳戶的儲存貯體。

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
```

```
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
```

```
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
```

```
                // Allow the previously created user to assume this role.
                AWS: User.Arn,
              },
              Action: "sts:AssumeRole",
            },
          ],
        }),
        RoleName: roleName,
      }),
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);
```

```
// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
  listBuckets(s3Client),
);

// Clean up.
```

```
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }
```

```
  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 使用者動作的函數。

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>
```

```
    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}
```

```
suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
```

```
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
```

```
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }
```

```
    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")
```

```
    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser

- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
```

```
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
```

```
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立一個可授予許可的 IAM 使用者和角色，以列出 Amazon S3 儲存貯體。使用者只有擔任該角色的權利。擔任角色後，請使用暫時性憑證列出該帳戶的儲存貯體。

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
```

```
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
```

```
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
```

```
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
 "
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
```

```
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
```

```
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
```

```
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- 如需 API 的詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立一個可授予許可的 IAM 使用者和角色，以列出 Amazon S3 儲存貯體。使用者只有擔任該角色的權利。擔任角色後，請使用暫時性憑證列出該帳戶的儲存貯體。

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end
```

```
  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
```

```
 # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
 # @return [Aws::IAM::Role] The newly created role.
 def create_role(role_name, user)
   trust_policy = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Principal: { 'AWS': user.arn },
       Action: 'sts:AssumeRole'
     }]
   }.to_json
   role = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: trust_policy
   ).role
   @logger.info("Created role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create a role for the demo. Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 else
   role
 end

 # Creates a policy that grants permission to list S3 buckets in the account,
and
 # then attaches the policy to a role.
 #
 # @param policy_name [String] The name to give the policy.
 # @param role [Aws::IAM::Role] The role that the policy is attached to.
 # @return [Aws::IAM::Policy] The newly created policy.
 def create_and_attach_role_policy(policy_name, role)
   policy_document = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Action: 's3:ListAllMyBuckets',
       Resource: 'arn:aws:s3:::*'
     }]
   }.to_json
   policy = @iam_client.create_policy(
     policy_name: policy_name,
     policy_document: policy_document
```

```
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 'sts:AssumeRole',
        Resource: role.arn
      }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end
```

```
 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
     count -= 1
     break if count.zero?
   end
 rescue Aws::Errors::ServiceError => e
   if e.code == 'AccessDenied'
     puts('Attempt to list buckets with no permissions: AccessDenied.')
   else
     @logger.info("Couldn't list buckets for the account. Here's why: ")
     @logger.info("\t#{e.code}: #{e.message}")
     raise
   end
 end

 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end
```

```
 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Deletes a user. If the user has inline policies or access keys, they are
deleted
 # before the user is deleted.
 #
```

```
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
```

```
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
```

```
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;
```

```
    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
```

```
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
```

```
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
```

```
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的下列主題。
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDKs的 IAM 動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 IAM 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 IAM API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 使用 AWS SDKs IAM 案例 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《AWS Identity and Access Management API 參考》。

範例

- 搭配使用 AddClientIdToOpenIdConnectProvider 與 CLI
- 搭配使用 AddRoleToInstanceProfile 與 CLI
- 搭配使用 AddUserToGroup 與 CLI
- 搭配使用 AttachGroupPolicy 與 CLI
- AttachRolePolicy 搭配 AWS SDK 或 CLI 使用
- AttachUserPolicy 搭配 AWS SDK 或 CLI 使用
- ChangePassword 搭配 CLI 使用
- CreateAccessKey 搭配 AWS SDK 或 CLI 使用
- CreateAccountAlias 搭配 AWS SDK 或 CLI 使用
- CreateGroup 搭配 AWS SDK 或 CLI 使用
- CreateInstanceProfile 搭配 AWS SDK 或 CLI 使用
- CreateLoginProfile 搭配 CLI 使用
- CreateOpenIdConnectProvider 搭配 CLI 使用
- CreatePolicy 搭配 AWS SDK 或 CLI 使用
- CreatePolicyVersion 搭配 AWS SDK 或 CLI 使用
- CreateRole 搭配 AWS SDK 或 CLI 使用
- CreateSAMLProvider 搭配 AWS SDK 或 CLI 使用
- CreateServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
- CreateUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 CreateVirtualMfaDevice 與 CLI
- 搭配使用 DeactivateMfaDevice 與 CLI
- DeleteAccessKey 搭配 AWS SDK 或 CLI 使用
- DeleteAccountAlias 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteAccountPasswordPolicy 與 CLI
- DeleteGroup 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteGroupPolicy 與 CLI

- DeleteInstanceProfile 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteLoginProfile 與 CLI
- 搭配使用 DeleteOpenIdConnectProvider 與 CLI
- DeletePolicy 搭配 AWS SDK 或 CLI 使用
- DeletePolicyVersion 搭配 AWS SDK 或 CLI 使用
- DeleteRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteRolePermissionsBoundary 與 CLI
- DeleteRolePolicy 搭配 AWS SDK 或 CLI 使用
- DeleteSAMLProvider 搭配 AWS SDK 或 CLI 使用
- DeleteServerCertificate 搭配 AWS SDK 或 CLI 使用
- DeleteServiceLinkedRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteSigningCertificate 與 CLI
- DeleteUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteUserPermissionsBoundary 與 CLI
- DeleteUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteVirtualMfaDevice 與 CLI
- 搭配使用 DetachGroupPolicy 與 CLI
- DetachRolePolicy 搭配 AWS SDK 或 CLI 使用
- DetachUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 EnableMfaDevice 與 CLI
- GenerateCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GenerateServiceLastAccessedDetails 與 CLI
- GetAccessKeyLastUsed 搭配 AWS SDK 或 CLI 使用
- GetAccountAuthorizationDetails 搭配 AWS SDK 或 CLI 使用
- GetAccountPasswordPolicy 搭配 AWS SDK 或 CLI 使用
- GetAccountSummary 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetContextKeysForCustomPolicy 與 CLI
- 搭配使用 GetContextKeysForPrincipalPolicy 與 CLI

- GetCredentialReport 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetGroup 與 CLI
- 搭配使用 GetGroupPolicy 與 CLI
- 搭配使用 GetInstanceProfile 與 CLI
- 搭配使用 GetLoginProfile 與 CLI
- 搭配使用 GetOpenIdConnectProvider 與 CLI
- GetPolicy 搭配 AWS SDK 或 CLI 使用
- GetPolicyVersion 搭配 AWS SDK 或 CLI 使用
- GetRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetRolePolicy 與 CLI
- 搭配使用 GetSamlProvider 與 CLI
- GetServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetServiceLastAccessedDetails 與 CLI
- 搭配使用 GetServiceLastAccessedDetailsWithEntities 與 CLI
- GetServiceLinkedRoleDeletionStatus 搭配 AWS SDK 或 CLI 使用
- GetUser 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetUserPolicy 與 CLI
- ListAccessKeys 搭配 AWS SDK 或 CLI 使用
- ListAccountAliases 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedGroupPolicies 與 CLI
- ListAttachedRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListAttachedUserPolicies 與 CLI
- 搭配使用 ListEntitiesForPolicy 與 CLI
- 搭配使用 ListGroupPolicies 與 CLI
- ListGroups 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListGroupsForUser 與 CLI
- 搭配使用 ListInstanceProfiles 與 CLI
- 搭配使用 ListInstanceProfilesForRole 與 CLI

- 搭配使用 ListMfaDevices 與 CLI
- 搭配使用 ListOpenIdConnectProviders 與 CLI
- ListPolicies 搭配 AWS SDK 或 CLI 使用
- ListPolicyVersions 搭配 AWS SDK 或 CLI 使用
- ListRolePolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListRoleTags 與 CLI
- ListRoles 搭配 AWS SDK 或 CLI 使用
- ListSAMLProviders 搭配 AWS SDK 或 CLI 使用
- ListServerCertificates 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListSigningCertificates 與 CLI
- ListUserPolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListUserTags 與 CLI
- ListUsers 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListVirtualMfaDevices 與 CLI
- 搭配使用 PutGroupPolicy 與 CLI
- 搭配使用 PutRolePermissionsBoundary 與 CLI
- PutRolePolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 PutUserPermissionsBoundary 與 CLI
- PutUserPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 RemoveClientIdFromOpenIdConnectProvider 與 CLI
- 搭配使用 RemoveRoleFromInstanceProfile 與 CLI
- 搭配使用 RemoveUserFromGroup 與 CLI
- 搭配使用 ResyncMfaDevice 與 CLI
- SetDefaultPolicyVersion 搭配 AWS SDK 或 CLI 使用
- 搭配使用 TagRole 與 CLI
- 搭配使用 TagUser 與 CLI
- 搭配使用 UntagRole 與 CLI
- 搭配使用 UntagUser 與 CLI
- UpdateAccessKey 搭配 AWS SDK 或 CLI 使用

- 搭配使用 UpdateAccountPasswordPolicy 與 CLI
- 搭配使用 UpdateAssumeRolePolicy 與 CLI
- 搭配使用 UpdateGroup 與 CLI
- 搭配使用 UpdateLoginProfile 與 CLI
- 搭配使用 UpdateOpenIdConnectProviderThumbprint 與 CLI
- 搭配使用 UpdateRole 與 CLI
- 搭配使用 UpdateRoleDescription 與 CLI
- 搭配使用 UpdateSamlProvider 與 CLI
- UpdateServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateSigningCertificate 與 CLI
- UpdateUser 搭配 AWS SDK 或 CLI 使用
- UploadServerCertificate 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UploadSigningCertificate 與 CLI

搭配使用 **AddClientIdToOpenIdConnectProvider** 與 CLI

下列程式碼範例示範如何使用 AddClientIdToOpenIdConnectProvider。

CLI

AWS CLI

若要將用戶端 ID (對象) 新增至 Open-ID Connect (OIDC) 提供者

下列 add-client-id-to-open-id-connect-provider 命令會將用戶端 ID my-application-ID 新增至名為 server.example.com 的 OIDC 提供者。

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

此命令不會產生輸出。

若要建立 OIDC 提供者，請使用 create-open-id-connect-provider 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 Creating OpenID Connect (OIDC) identity providers。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AddClientIdToOpenIdConnectProvider。

PowerShell

Tools for PowerShell V4

範例 1：此命令會將用戶端 ID (或對象) **my-application-ID** 新增至名為 **server.example.com** 的現有 OIDC 提供者。

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AddClientIdToOpenIdConnectProvider。

Tools for PowerShell V5

範例 1：此命令會將用戶端 ID (或對象) **my-application-ID** 新增至名為 **server.example.com** 的現有 OIDC 提供者。

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AddClientIdToOpenIdConnectProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **AddRoleToInstanceProfile** 與 CLI

下列程式碼範例示範如何使用 AddRoleToInstanceProfile。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 為虛擬機器執行個體建立容器服務
- 受管串流入門
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

CLI

AWS CLI

若要將角色新增至執行個體設定檔

下列 add-role-to-instance-profile 命令會將名為 S3Access 的角色新增至名為 Webserver 的執行個體設定檔。

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

此命令不會產生輸出。

若要建立執行個體設定檔，請使用 create-instance-profile 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用 IAM 角色為在 Amazon EC2 執行個體上執行的應用程式授予許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AddRoleToInstanceProfile。

PowerShell

Tools for PowerShell V4

範例 1：此命令會將名為 **S3Access** 的角色新增至名為 **webserver** 的現有執行個體設定檔。若要建立執行個體設定檔，請使用 **New-IAMInstanceProfile** 命令。使用此命令建立執行個體設定檔並將其與角色關聯起來之後，可以將其連接至 EC2 執行個體。為此，請使用 **New-EC2Instance** cmdlet 搭配 **InstanceProfile_Arn** 或 **InstanceProfile-Name** 參數啟動新的執行個體。

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AddRoleToInstanceProfile。

Tools for PowerShell V5

範例 1：此命令會將名為 **S3Access** 的角色新增至名為 **webserver** 的現有執行個體設定檔。若要建立執行個體設定檔，請使用 **New-IAMInstanceProfile** 命令。使用此命令建立執行個體設定檔並將其與角色關聯起來之後，可以將其連接至 EC2 執行個體。為此，請使用 **New-EC2Instance** cmdlet 搭配 **InstanceProfile_Arn** 或 **InstanceProfile-Name** 參數啟動新的執行個體。

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AddRoleToInstanceProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **AddUserToGroup** 與 CLI

下列程式碼範例示範如何使用 AddUserToGroup。

CLI

AWS CLI

將使用者新增至 IAM 群組

下列 add-user-to-group 命令會將名為 Bob 的 IAM 使用者新增至名為 Admins 的 IAM 群組。

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 IAM 使用者群組中新增和移除使用者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AddUserToGroup。

PowerShell

Tools for PowerShell V4

範例 1：此命令會將名為 **Bob** 的使用者新增至名為 **Admins** 的群組。

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AddUserToGroup。

Tools for PowerShell V5

範例 1：此命令會將名為 **Bob** 的使用者新增至名為 **Admins** 的群組。

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AddUserToGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **AttachGroupPolicy** 與 CLI

下列程式碼範例示範如何使用 AttachGroupPolicy。

CLI

AWS CLI

若要將受管政策連接至 IAM 群組

下列attach-group-policy命令會將名為 的 AWS 受管政策連接至名為 ReadOnlyAccess的 IAM 群組Finance。

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
```

```
    --group-name Finance
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的受管政策和內嵌政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AttachGroupPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將名為 **TesterPolicy** 的客戶管理政策連接到 IAM 群組 **Testers**。該群組中的使用者會立即受到該政策預設版本中定義之許可影響。

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

範例 2：此範例會將名為 的 AWS 受管政策連接至 **AdministratorAccess** IAM 群組 **Admins**。該群組中的使用者會立即受到該政策最新版本中定義之許可影響。

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AttachGroupPolicy。

Tools for PowerShell V5

範例 1：此範例會將名為 **TesterPolicy** 的客戶管理政策連接到 IAM 群組 **Testers**。該群組中的使用者會立即受到該政策預設版本中定義之許可影響。

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

範例 2：此範例會將名為 的 AWS 受管政策連接至 **AdministratorAccess** IAM 群組 **Admins**。該群組中的使用者會立即受到該政策最新版本中定義之許可影響。

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AttachGroupPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`AttachRolePolicy` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `AttachRolePolicy`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 設定容器服務連線
- 為無伺服器啟動類型建立容器任務
- 使用函數代理整合建立靜態 API
- 為虛擬機器執行個體建立容器服務
- 建立受管監控工作區
-
- 建立您的第一個無伺服器函數
- 組態管理入門
- 開始使用物聯網裝置保護
- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門
- 受管串流入門
- 工作流程協同運作入門
- 管理角色
- 將硬式編碼秘密移至安全秘密儲存
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 設定系統管理
- 在監控儀表板中使用屬性變數來監控多個無伺服器函數
- 使用串流和存留時間

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 AttachRolePolicy。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam attach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AttachRolePolicy。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
            return false;
        }

        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
```

```
                                return policy.GetPolicyArn() == policyArn;
                            })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
 iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
 <<
                  roleName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 AttachRolePolicy。

CLI

AWS CLI

將受管政策連接至 IAM 角色

下列attach-role-policy命令會將名為 的 AWS 受管政策連接至名為 ReadOnlyAccess的 IAM 角色ReadOnlyRole。

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的受管政策和內嵌政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AttachRolePolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// AttachRolePolicy attaches a policy to a role.
```

```
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 AttachRolePolicy。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
```

```
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
 ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            ListAttachedRolePoliciesResponse response =
 iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

            // Ensure that the policy is not attached to this role
            String polArn = "";
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
 this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);

            System.out.println("Successfully attached policy " + policyArn +
                    " to role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 AttachRolePolicy。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

連接政策。

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 AttachRolePolicy。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
      }
    });
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 AttachRolePolicy。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun attachIAMRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
```

```
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 AttachRolePolicy。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
               \"Version\": \"2012-10-17\",
               \"Statement\": [{
                   \"Effect\": \"Allow\",
                   \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                   \"Action\": \"sts:AssumeRole\"
```

```
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 AttachRolePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將名為 的 AWS 受管政策連接至 **SecurityAudit** IAM 角色 **CoSecurityAuditors**。該最新版本政策中所定義的許可會立即影響擔任該角色的使用者。

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AttachRolePolicy。

Tools for PowerShell V5

範例 1：此範例會將名為 的 AWS 受管政策連接至 **SecurityAudit** IAM 角色 **CoSecurityAuditors**。該最新版本政策中所定義的許可會立即影響擔任該角色的使用者。

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AttachRolePolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

使用 Boto3 Policy 物件將政策連接至角色。

```
def attach_to_role(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
```

```
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
role_name)
        raise
```

使用 Boto3 Role 物件將政策連接至角色。

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
role_name)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 AttachRolePolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、連接和分離角色政策。

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 AttachRolePolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 AttachRolePolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->attachrolepolicy(
          iv_rolename = iv_role_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy attached to role successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role or policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 AttachRolePolicy。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func attachRolePolicy(role: String, policyArn: String) async throws {
```

```
        let input = AttachRolePolicyInput(
            policyArn: policyArn,
            roleName: role
        )
        do {
            _ = try await client.attachRolePolicy(input: input)
        } catch {
            print("ERROR: Attaching a role policy:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 AttachRolePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AttachUserPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 AttachUserPolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建立唯讀和讀寫的使用者

CLI

AWS CLI

將受管政策連接至 IAM 使用者

下列attach-user-policy命令會將名為 的 AWS 受管政策連接至名為 AdministratorAccess的 IAM 使用者Alice。

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的受管政策和內嵌政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AttachUserPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將名為 **AmazonCognitoPowerUser**的 AWS 受管政策連接至 IAM 使用者 **Bob**。該最新版本政策中所定義的許可會立即影響使用者。

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AttachUserPolicy。

Tools for PowerShell V5

範例 1：此範例會將名為 **AmazonCognitoPowerUser**的 AWS 受管政策連接至 IAM 使用者 **Bob**。該最新版本政策中所定義的許可會立即影響使用者。

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AttachUserPolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.
```

```
    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
user_name)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 AttachUserPolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Attaches a policy to a user
#
# @param user_name [String] The name of the user
# @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_user(user_name, policy_arn)
  @iam_client.attach_user_policy(
    user_name: user_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to user: #{e.message}")
```

```
    false
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 AttachUserPolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 AttachUserPolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
TRY.
    lo_iam->attachuserpolicy(
      iv_username = iv_user_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy attached to user successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User or policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 AttachUserPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ChangePassword` 搭配 CLI 使用

下列程式碼範例示範如何使用 `ChangePassword`。

CLI

AWS CLI

若要變更您的 IAM 使用者的密碼

若要變更 IAM 使用者的密碼，建議使用 `--cli-input-json` 參數來傳遞包含舊密碼和新密碼的 JSON 檔案。採用此方法時，可以使用含非英數字元的高保護性密碼。當您以命令列參數形式傳遞密碼時，使用含非英數字元的密碼可能會有困難。若要使用 `--cli-input-json` 參

數，請先從使用含 --generate-cli-skeleton 參數的 change-password 命令開始，範例如下所示。

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

先前的命令會建立名為 change-password.json 的 JSON 檔案，可用來填入舊密碼與新密碼。例如，此檔案可能如下所示：

```
{
    "OldPassword": "3s0K_;xh4~8XXI",
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

接下來，若要變更密碼，請再次使用 change-password 命令，這次傳遞 --cli-input-json 參數以指定 JSON 檔案。下列 change-password 命令會將 --cli-input-json 參數與名為 change-password.json 的 JSON 檔案搭配使用。

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

此命令不會產生輸出。

此命令只能由 IAM 使用者呼叫。如果使用 AWS 帳戶（根）登入資料呼叫此命令，則命令會傳回InvalidUserType錯誤。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 使用者如何變更自己的密碼。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ChangePassword。

PowerShell

Tools for PowerShell V4

範例 1：此命令會變更執行命令的使用者密碼。此命令只能由 IAM 使用者呼叫。如果您在使用 AWS 帳戶（根）登入資料登入時呼叫此命令，命令會傳回**InvalidUserType**錯誤。

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ChangePassword。

Tools for PowerShell V5

範例 1：此命令會變更執行命令的使用者密碼。此命令只能由 IAM 使用者呼叫。如果您在使用 AWS 帳戶（根）登入資料登入時呼叫此命令，命令會傳回**InvalidUserType**錯誤。

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ChangePassword。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateAccessKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateAccessKey。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 建立唯讀和讀寫的使用者
- 管理存取金鑰

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
```

```
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 CreateAccessKey。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
```

```
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
```

```
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateAccessKey。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
    request.SetUserName(userName);

    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 CreateAccessKey。

CLI

AWS CLI

為 IAM 使用者建立存取金鑰

下列 `create-access-key` 命令會為名為 Bob 的 IAM 使用者建立存取金鑰 (存取金鑰 ID 與私密存取金鑰)。

```
aws iam create-access-key \
    --user-name Bob
```

輸出：

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

請將私密存取金鑰存放在安全之處。遺失的金鑰無法復原，您必須建立新的存取金鑰。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的存取金鑰。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateAccessKey。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

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

```
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 CreateAccessKey。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

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

                Usage:
                   <user>\s

                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

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

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }
```

```
    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 CreateAccessKey。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立存取金鑰。

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
```

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateAccessKey。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateAccessKey。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun createIAMAccessKey(user: String?): String {
    val request =
        CreateAccessKeyRequest {
            userName = user
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 CreateAccessKey。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立新的存取金鑰和私密存取金鑰對，並將其指派給使用者 David。請確定將 AccessKeyId 和 SecretAccessKey 值儲存到檔案中，因為這是唯一可以取得 SecretAccessKey 的時機。您稍後便無法擷取它。若您遺失了密碼金鑰，必須建立新的存取金鑰對。

```
New-IAMAccessKey -UserName David
```

輸出：

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
```

```
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateAccessKey。

Tools for PowerShell V5

範例 1：此範例會建立新的存取金鑰和私密存取金鑰對，並將其指派給使用者 David。請確定將 AccessKeyId 和 SecretAccessKey 值儲存到檔案中，因為這是唯一可以取得 SecretAccessKey 的時機。您稍後便無法擷取它。若您遺失了密碼金鑰，必須建立新的存取金鑰對。

```
New-IAMAccessKey -UserName David
```

輸出：

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateAccessKey。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
```

```
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 CreateAccessKey。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、停用和刪除存取金鑰。

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
```

```
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
```

```
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 CreateAccessKey。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 CreateAccessKey。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->createaccesskey(
          iv_username = iv_user_name ).
        MESSAGE 'Access key created successfully.' TYPE 'I'.
```

```
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Maximum number of access keys reached.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 CreateAccessKey。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func createAccessKey(userName: String) async throws ->
 IAMClientTypes.AccessKey {
        let input = CreateAccessKeyInput(
            userName: userName
        )
        do {
            let output = try await iamClient.createAccessKey(input: input)
            guard let accessKey = output.accessKey else {
                throw ServiceHandlerError.keyError
            }
            return accessKey
        } catch {
            print("ERROR: createAccessKey:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 CreateAccessKey。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateAccountAlias` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateAccountAlias`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理您的帳戶

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created account alias " << aliasName <<
```

```
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 CreateAccountAlias。

CLI

AWS CLI

建立帳戶別名

下列create-account-alias命令會examplecorp為您的 AWS 帳戶建立別名。

```
aws iam create-account-alias \
    --account-alias examplecorp
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的AWS 您的帳戶 ID 及其別名。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateAccountAlias。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
```

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

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

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

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();
```

```
            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 CreateAccountAlias。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立帳戶別名。

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
```

```
};
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateAccountAlias。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateAccountAlias。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun createIAMAccountAlias(alias: String) {
    val request =
        CreateAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 CreateAccountAlias。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將 AWS 您帳戶的帳戶別名變更為 **mycompanyaws**。使用者登入頁面的網址會變更為 https://mycompanyaws.signin.aws.amazon.com/console。使用您的帳戶 ID 號碼而非別名的原始 URL (https://<accountidnumber>.signin.aws.amazon.com/console) 繼續有效。但是，任何先前定義的別名型 URL 已失效。

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateAccountAlias。

Tools for PowerShell V5

範例 1：此範例會將 AWS 您帳戶的帳戶別名變更為 **mycompanyaws**。使用者登入頁面的網址會變更為 https://mycompanyaws.signin.aws.amazon.com/console。使用您的帳戶 ID 號碼而非別名的原始 URL (https://<accountidnumber>.signin.aws.amazon.com/console) 繼續有效。但是，任何先前定義的別名型 URL 已失效。

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateAccountAlias。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 CreateAccountAlias。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出、建立和刪除帳戶別名。

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
```

```
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 CreateAccountAlias。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->createaccountalias(
          iv_accountalias = iv_account_alias ).
```

```
        MESSAGE 'Account alias created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Account alias already exists.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Account alias limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 CreateAccountAlias。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateGroup 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateGroup。

CLI

AWS CLI

建立 IAM 群組

下列 create-group 命令會建立名為 Admins 的 IAM 群組。

```
aws iam create-group \
    --group-name Admins
```

輸出：

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 使用者群組。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateGroup。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateGroup。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立名為 **Developers** 的新 IAM 群組。

```
New-IAMGroup -GroupName Developers
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateGroup。

Tools for PowerShell V5

範例 1：此範例會建立名為 **Developers** 的新 IAM 群組。

```
New-IAMGroup -GroupName Developers
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateInstanceProfile 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateInstanceProfile。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建置及管理彈性服務
- 為虛擬機器執行個體建立容器服務
- 受管串流入門

- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {

       var assumeRoleDoc = "{" +
                                   "\"Version\": \"2012-10-17\"," +
                                   "\"Statement\": [{" +
                                        "\"Effect\": \"Allow\"," +
                                        "\"Principal\": {" +
                                        "\"Service\": [" +
                                            "\"ec2.amazonaws.com\"" +
```

```
                                    "]" +
                                    "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
```

```
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
```

```
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 CreateInstanceProfile。

CLI

AWS CLI

建立執行個體設定檔

下列 `create-instance-profile` 命令會建立名為 `Webserver` 的執行個體設定檔。

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

輸出：

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
```

```
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

若要將角色新增至執行個體設定檔，請使用 `add-role-to-instance-profile` 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用 IAM 角色為在 Amazon EC2 執行個體上執行的應用程式授予許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateInstanceProfile。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
const { InstanceProfile } = await iamClient.send(
  new CreateInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
  }),
);
await waitUntilInstanceProfileExists(
  { client: iamClient },
  { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
);
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 CreateInstanceProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立名為 **ProfileForDevEC2Instance** 的新 IAM 執行個體設定檔。必須單獨執行 **Add-IAMRoleToInstanceProfile** 命令，將執行個體設定檔與為執行個體提供許

可的現有 IAM 角色關聯起來。最後，在啟動執行個體時，將執行個體設定檔連接到 EC2 執行個體。為此，請使用含 **InstanceProfile_Arn** 或 **InstanceProfile_Name** 參數的 **New-EC2Instance** cmdlet。

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

輸出：

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateInstanceProfile。

Tools for PowerShell V5

範例 1：此範例會建立名為 **ProfileForDevEC2Instance** 的新 IAM 執行個體設定檔。必須單獨執行 **Add-IAMRoleToInstanceProfile** 命令，將執行個體設定檔與為執行個體提供許可的現有 IAM 角色關聯起來。最後，在啟動執行個體時，將執行個體設定檔連接到 EC2 執行個體。為此，請使用含 **InstanceProfile_Arn** 或 **InstanceProfile_Name** 參數的 **New-EC2Instance** cmdlet。

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

輸出：

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateInstanceProfile。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例會建立政策、角色和執行個體設定檔，並將它們全部連結在一起。

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
```

```
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
```

```
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
```

```
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 CreateInstanceProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateLoginProfile` 搭配 CLI 使用

下列程式碼範例示範如何使用 `CreateLoginProfile`。

CLI

AWS CLI

為 IAM 使用者建立密碼

若要為 IAM 使用者建立密碼，建議使用 `--cli-input-json` 參數傳遞包含密碼的 JSON 檔案。採用此方法時，可以建立含非英數字元的強式密碼。當您以命令列參數形式傳遞密碼時，建立含非英數字元的密碼可能會很困難。

若要使用 `--cli-input-json` 參數，請先使用 `create-login-profile` 命令搭配 `--generate-cli-skeleton` 參數，如下列範例中所示。

```
aws iam create-login-profile \
    --generate-cli-skeleton > create-login-profile.json
```

先前的命令會建立名為 create-login-profile.json 的 JSON 檔案，可用來填入後續 `create-login-profile` 命令的資訊。例如：

```
{
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

接下來，若要為 IAM 使用者建立密碼，請再次使用 create-login-profile 命令，這次傳遞 --cli-input-json 參數來指定 JSON 檔案。下列 create-login-profile 命令會將 --cli-input-json 參數搭配名為 create-login-profile.json 的 JSON 檔案使用。

```
aws iam create-login-profile \
    --cli-input-json file://create-login-profile.json
```

輸出：

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

如果新密碼違反帳戶密碼政策，則命令會傳回 PasswordPolicyViolation 錯誤。

若要為已有密碼的使用者變更密碼，請使用 update-login-profile。若要設定帳戶的密碼政策，請使用 update-account-password-policy 命令。

如果帳戶密碼政策允許，IAM 使用者可以使用 change-password 命令變更自己的密碼。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的密碼。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateLoginProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會為名為 Bob 的 IAM 使用者建立 (臨時) 密碼並設定在 **Bob** 下次登入時要求使用者變更密碼的旗標。

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

輸出：

```
CreateDate                  PasswordResetRequired              UserName
----------                  ---------------------              --------
```

```
4/14/2015 12:26:30 PM         True                                  Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateLoginProfile。

Tools for PowerShell V5

範例 1：此範例會為名為 Bob 的 IAM 使用者建立 (臨時) 密碼並設定在 **Bob** 下次登入時要求使用者變更密碼的旗標。

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

輸出：

```
CreateDate                    PasswordResetRequired                 UserName
----------                    ---------------------                 --------
4/14/2015 12:26:30 PM         True                                  Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateLoginProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateOpenIdConnectProvider 搭配 CLI 使用

下列程式碼範例示範如何使用 `CreateOpenIdConnectProvider`。

CLI

AWS CLI

若要建立 IAM OpenID Connect (OIDC) 提供者

若要建立 OpenID Connect (OIDC) 提供者，建議使用 `--cli-input-json` 參數來傳遞包含必要參數的 JSON 檔案。建立 OIDC 提供者時，必須傳遞提供者的 URL，且 URL 必須以 `https://` 開頭。以命令列參數形式傳遞 URL 可能會很困難，因為在某些命令列環境中，冒號 (:) 和正斜線 (/) 字元有特殊含義。使用 `--cli-input-json` 參數可以避開這個限制。

若要使用 `--cli-input-json` 參數，請先使用 `create-open-id-connect-provider` 命令搭配 `--generate-cli-skeleton` 參數，如下列範例中所示。

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

上一個命令會建立名為 create-open-id-connect-provider.json 的 JSON 檔案，用來填入後續 `create-open-id-connect-provider` 命令的資訊。例如：

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

如果您 OIDC 提供者的探索和 JWKS 端點使用不同的憑證，請在 中包含兩個端點的指紋ThumbprintList。

接下來，若要建立 OpenID Connect (OIDC) 提供者，請再次使用 `create-open-id-connect-provider` 命令，這次傳遞 `--cli-input-json` 參數來指定 JSON 檔案。下列 `create-open-id-connect-provider` 命令會將 `--cli-input-json` 參數與名為 create-open-id-connect-provider.json 的 JSON 檔案搭配使用。

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

輸出：

```
{
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

如需有關 OIDC 提供者的詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 OpenID Connect (OIDC) 身分提供者。

如需如何取得 OIDC 提供者指紋的詳細資訊，請參閱《AWS IAM 使用者指南》中的取得 OpenID Connect 身分提供者的指紋。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateOpenIdConnectProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立與 URL **https://example.oidcprovider.com** 和用戶端 ID **my-testapp-1** 中的 OIDC 相容提供者服務關聯的 IAM OIDC 提供者。OIDC 提供者會提供指紋。若要驗證指紋，請依照 http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html 中的步驟進行。

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

輸出：

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateOpenIdConnectProvider。

Tools for PowerShell V5

範例 1：此範例會建立與 URL **https://example.oidcprovider.com** 和用戶端 ID **my-testapp-1** 中的 OIDC 相容提供者服務關聯的 IAM OIDC 提供者。OIDC 提供者會提供指紋。若要驗證指紋，請依照 http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html 中的步驟進行。

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

輸出：

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateOpenIdConnectProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreatePolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreatePolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 建立唯讀和讀寫的使用者
- 建立受管監控工作區
- 受管串流入門
- 工作流程協同運作入門
- 管理政策
- 設定屬性型存取控制
- 設定系統管理
- 使用 IAM 政策產生器 API

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
```

```
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 CreatePolicy。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
```

```
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreatePolicy。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
                                      const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));
```

```
    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }

    return result;
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
                 << "       \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    }"
                 << "   ]"
                 << "}";
```

```
    return stringStream.str();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 CreatePolicy。

CLI

AWS CLI

範例 1：建立客戶管理政策

下列命令會建立名為 my-policy 的客戶管理政策。檔案 policy.json 是目前資料夾中的 JSON 文件，在名為 amzn-s3-demo-bucket 的 Amazon S3 儲存貯體中授予 shared 資料夾的唯讀存取權限。

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json
```

policy.json 的內容：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/shared/*"
            ]
        }
    ]
}
```

輸出：

```
{
```

```
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

如需使用檔案做為字串參數輸入的詳細資訊，請參閱《CLI AWS 使用者指南》中的指定 CLI 的參數值。 AWS

範例 2：建立內含描述的客戶管理政策

下列命令會建立名為 my-policy 的客戶管理政策，其中包含不可變的描述。

檔案 policy.json 是目前資料夾中的 JSON 文件，可針對名為 amzn-s3-demo-bucket 的 Amazon S3 儲存貯體，授予所有 Put、List 和 Get 動作的存取權限。

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for amzn-s3-demo-bucket"
```

policy.json 的內容：

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
           "Effect": "Allow",
           "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
```

```
            ]
        }
    ]
}
```

輸出：

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

如需有關以身分為基礎之政策的詳細資訊，請參閱《AWS IAM 使用者指南》中的以身分為基礎和以資源為基礎的政策。

範例 3：建立內含標籤的客戶管理政策

下列命令會建立名為 my-policy 的客戶管理政策，其中包含標籤。此範例使用具有下列 JSON 格式標籤的 --tags 參數：'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'。或者，--tags 旗標可以與速記格式的標籤一起使用：'Key=Department,Value=Accounting Key=Location,Value=Seattle'。

檔案 policy.json 是目前資料夾中的 JSON 文件，可針對名為 amzn-s3-demo-bucket 的 Amazon S3 儲存貯體，授予所有 Put、List 和 Get 動作的存取權限。

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

policy.json 的內容：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

輸出：

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {
        ]
    }
```

```
}
```

如需有關標記政策的詳細資訊，請參閱《AWS IAM 使用者指南》中的標記客戶管理政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreatePolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
```

```
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 CreatePolicy。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

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

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
```

```
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

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

        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
+ result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
```

```
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);

            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 CreatePolicy。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立政策。

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} policyName
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreatePolicy。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });
```

```
// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var myManagedPolicy = {
  Version: "2012-10-17",
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
  PolicyName: "myDynamoDBPolicy",
};

iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreatePolicy。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun createIAMPolicy(policyNameVal: String?): String {
    val policyDocumentVal = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "dynamodb:DeleteItem",
                    "dynamodb:GetItem",
                    "dynamodb:PutItem",
                    "dynamodb:Scan",
                    "dynamodb:UpdateItem"
                ],
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 CreatePolicy。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    /**
     * @param string $policyName
     * @param string $policyDocument
     * @return array
     */
    public function createPolicy(string $policyName, string $policyDocument)
    {
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 CreatePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會在名為 的目前 AWS 帳戶中建立新的 IAM MySamplePolicy 政策。 檔案MySamplePolicy.json會提供政策內容。請注意，必須使用 -Raw 切換參數，才能成功處理 JSON 政策檔案。

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

輸出：

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreatePolicy。

Tools for PowerShell V5

範例 1：此範例會在名為 的目前 AWS 帳戶中建立新的 IAM MySamplePolicy 政策。 檔案MySamplePolicy.json會提供政策內容。請注意，必須使用 -Raw 切換參數，才能成功處理 JSON 政策檔案。

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

輸出：

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreatePolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
```

```
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 CreatePolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、連接和分離角色政策。

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
```

```
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
```

```
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 CreatePolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
) -> Result<Policy, iamError> {
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
    Ok(policy.policy.unwrap())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 CreatePolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->createpolicy(
          iv_policyname = iv_policy_name
```

```
          iv_policydocument = iv_policy_document
          iv_description = iv_description ).
        MESSAGE 'Policy created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Policy already exists.' TYPE 'E'.
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Policy document is malformed.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 CreatePolicy。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
```

```
        } catch {
            print("ERROR: createPolicy:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 CreatePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreatePolicyVersion 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreatePolicyVersion。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理政策

CLI

AWS CLI

建立新版本的受管政策

此範例會建立新 v2 版的 IAM 政策 (其 ARN 為 arn:aws:iam::123456789012:policy/MyPolicy)，並將該版本設為預設版本。

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

輸出：

```
{
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
```

```
        "IsDefaultVersion": true
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 政策的版本控制。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreatePolicyVersion。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立新 v2 版的 IAM 政策 (其 ARN 為 **arn:aws:iam::123456789012:policy/MyPolicy**)，並將該版本設為預設版本。**NewPolicyVersion.json** 檔案會提供政策內容。請注意，必須使用 **-Raw** 切換參數，才能成功處理 JSON 政策檔案。

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

輸出：

```
CreateDate                          Document                IsDefaultVersion
          VersionId
----------                          --------                ----------------
          ---------
4/15/2015 10:54:54 AM                                       True
          v2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreatePolicyVersion。

Tools for PowerShell V5

範例 1：此範例會建立新 v2 版的 IAM 政策 (其 ARN 為 **arn:aws:iam::123456789012:policy/MyPolicy**)，並將該版本設為預設版本。**NewPolicyVersion.json** 檔案會提供政策內容。請注意，必須使用 **-Raw** 切換參數，才能成功處理 JSON 政策檔案。

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

輸出：

```
CreateDate                              Document                IsDefaultVersion
           VersionId
----------                              --------                ---------------
           ---------
4/15/2015 10:54:54 AM                                           True
           v2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreatePolicyVersion。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
```

```
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 CreatePolicyVersion。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->createpolicyversion(
          iv_policyarn = iv_policy_arn
          iv_policydocument = iv_policy_document
          iv_setasdefault = iv_set_as_default ).
        MESSAGE 'Policy version created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Policy document is malformed.' TYPE 'E'.
```

```
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy version limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 CreatePolicyVersion。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateRole` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateRole`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 設定容器服務連線
- 為無伺服器啟動類型建立容器任務
- 使用函數代理整合建立靜態 API
- 為虛擬機器執行個體建立容器服務
- 建立受管監控工作區
-
- 建立您的第一個無伺服器函數
- 開始使用無伺服器資料倉儲
- 組態管理入門
- 開始使用物聯網裝置保護
- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門
- 受管串流入門
- 佈建資料倉儲叢集入門
- 工作流程協同運作入門
- 管理角色

- 將硬式編碼秘密移至安全秘密儲存
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 設定系統管理
- 在監控儀表板中使用屬性變數來監控多個無伺服器函數
- 使用串流和存留時間

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 CreateRole。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
```

```
  echo "  -n role_name   The name of the IAM role."
  echo "  -p policy_json -- The assume role policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateRole。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::createIamRole(
        const Aws::String &roleName,
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
```

```
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 CreateRole。

CLI

AWS CLI

範例 1：建立 IAM 角色

下列 `create-role` 命令會建立名為 `Test-Role` 的角色，並將信任政策連接至該角色。

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

輸出：

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

在 Test-Role-Trust-Policy.json 檔案中，將信任政策定義為 JSON 文件。(檔案名稱和副檔名沒有意義。) 信任政策必須指定主體。

若要將許可政策連接至角色，請使用 `put-role-policy` 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 角色。

範例 2：建立具有指定最長工作階段持續時間的 IAM 角色

下列 create-role 命令會建立名為 Test-Role 的角色，並設定 7200 秒 (2 小時) 的最長工作階段持續時間。

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

輸出：

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的修改角色最長工作階段持續時間 (AWS API)。

範例 3：建立內含標籤的 IAM 角色

下列命令會建立內含標籤的 IAM 角色 Test-Role。此範例使用具有下列 JSON 格式標記的 --tags 參數旗標：'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'。或者，--tags 旗標可以與速記格式的標籤一起使用：'Key=Department,Value=Accounting Key=Location,Value=Seattle'。

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

輸出：

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的標記 IAM 角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateRole。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
```

```
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 CreateRole。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
```

```
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s

                Where:
                    rolename - The name of the role to create.\s
                    fileLocation - The location of the JSON document that
 represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
```

```
        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
 fileLocation) throws Exception {
        try {
            JSONObject jsonObject = (JSONObject)
 readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
 response.role().arn());

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

    public static Object readJsonSimpleDemo(String filename) throws Exception {
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 CreateRole。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立角色。

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
          },
          Action: "sts:AssumeRole",
        },
      ],
    }),
    RoleName: roleName,
  });

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

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateRole。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
     * @throws AwsException
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 CreateRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立名為 MyNewRole 的新角色，並將其連接至在 NewRoleTrustPolicy.json 檔案中找到的政策。請注意，必須使用 -Raw 切換參數，才能成功處理 JSON 政策檔案。輸出中顯示的政策文件是以 URL 編碼。在此範例中，它會使用 UrlDecode .NET 方法解碼。

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

輸出：

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument  : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                            %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                            %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                            %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                            %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                            %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                            %20%5D%0D%0A%7D
CreateDate                : 4/15/2015 11:04:23 AM
Path                      : /
RoleId                    : V5PAJI2KPN4EAEXAMPLE1
RoleName                  : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateRole。

Tools for PowerShell V5

範例 1：此範例會建立名為 **MyNewRole** 的新角色，並將其連接至在 **NewRoleTrustPolicy.json** 檔案中找到的政策。請注意，必須使用 **-Raw** 切換參數，才能成功處理 JSON 政策檔案。輸出中顯示的政策文件是以 URL 編碼。在此範例中，它會使用 **UrlDecode** .NET 方法解碼。

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

輸出：

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                           %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                           %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                           %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                           %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                           %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                           %20%5D%0D%0A%7D
```

```
CreateDate                : 4/15/2015 11:04:23 AM
Path                      : /
RoleId                    : V5PAJI2KPN4EAEXAMPLE1
RoleName                  : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateRole。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
```

```
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 CreateRole。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Creates a role and attaches policies to it.
#
# @param role_name [String] The name of the role.
```

```
  # @param assume_role_policy_document [Hash] The trust relationship policy
document.
  # @param policy_arns [Array<String>] The ARNs of the policies to attach.
  # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
  def create_role(role_name, assume_role_policy_document, policy_arns)
    response = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: assume_role_policy_document.to_json
    )
    role_arn = response.role.arn

    policy_arns.each do |policy_arn|
      @iam_client.attach_role_policy(
        role_name: role_name,
        policy_arn: policy_arn
      )
    end

    role_arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating role: #{e.message}")
    nil
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 CreateRole。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
    role_policy_document: &str,
```

```
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 CreateRole。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->createrole(
          iv_rolename = iv_role_name
          iv_assumerolepolicydocument = iv_assume_role_policy_document ).
        MESSAGE 'Role created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Role already exists.' TYPE 'E'.
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Assume role policy document is malformed.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Role limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 CreateRole。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func createRole(name: String, policyDocument: String) async throws ->
 String {
        let input = CreateRoleInput(
            assumeRolePolicyDocument: policyDocument,
            roleName: name
        )
        do {
            let output = try await client.createRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            guard let id = role.roleId else {
                throw ServiceHandlerError.noSuchRole
            }
            return id
        } catch {
            print("ERROR: createRole:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 CreateRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateSAMLProvider` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateSAMLProvider`。

CLI

AWS CLI

建立 SAML 提供者

此範例會在 IAM 中建立名為 `MySAMLProvider` 的新 SAML 提供者。它會由在檔案 `SAMLMetaData.xml` 中找到的 SAML 中繼資料文件進行描述。

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

輸出：

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM SAML 身分提供者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateSAMLProvider。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import * as path from "node:path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
 */
const sampleMetadataDocument = readFileSync(
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
    Name: providerName,
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 CreateSAMLProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會在 IAM 中建立新的 SAML 提供者實體。它被命名為 **MySAMLProvider**，由在檔案 **SAMLMetaData.xml** 中找到的 SAML 中繼資料文件描述，該檔案需從 SAML 服務提供者的網站單獨下載。

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

輸出：

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateSAMLProvider。

Tools for PowerShell V5

範例 1：此範例會在 IAM 中建立新的 SAML 提供者實體。它被命名為 **MySAMLProvider**，由在檔案 **SAMLMetaData.xml** 中找到的 SAML 中繼資料文件描述，該檔案需從 SAML 服務提供者的網站單獨下載。

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

輸出：

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateSAMLProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateServiceLinkedRole 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateServiceLinkedRole。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 CreateServiceLinkedRole。

CLI

AWS CLI

建立服務連結角色

下列create-service-linked-role範例會為指定的服務建立 AWS 服務連結角色，並連接指定的描述。

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

輸出：

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用服務連結角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateServiceLinkedRole。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 CreateServiceLinkedRole。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立服務連結角色。

```
import {
  CreateServiceLinkedRoleCommand,
  GetRoleCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
```

```
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });
  try {
    const response = await client.send(command);
    console.log(response);
    return response;
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInputException" &&
      caught.message.includes(
        "Service role name AWSServiceRoleForElasticBeanstalk has been taken in
 this account",
      )
    ) {
      console.warn(caught.message);
      return client.send(
        new GetRoleCommand({ RoleName: "AWSServiceRoleForElasticBeanstalk" }),
      );
    }
    throw caught;
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateServiceLinkedRole。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 CreateServiceLinkedRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立用於自動擴展服務的服務連結角色。

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateServiceLinkedRole。

Tools for PowerShell V5

範例 1：此範例會建立用於自動擴展服務的服務連結角色。

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateServiceLinkedRole。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
 service_name)
        raise
    else:
        return role
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 CreateServiceLinkedRole。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
 # Creates a service-linked role
 #
 # @param service_name [String] The service name to create the role for.
 # @param description [String] The description of the service-linked role.
 # @param suffix [String] Suffix for customizing role name.
 # @return [String] The name of the created role
 def create_service_linked_role(service_name, description, suffix)
   response = @iam_client.create_service_linked_role(
     aws_service_name: service_name, description: description, custom_suffix:
suffix
   )
   role_name = response.role.role_name
   @logger.info("Created service-linked role #{role_name}.")
   role_name
 rescue Aws::Errors::ServiceError => e
   @logger.error("Couldn't create service-linked role for #{service_name}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   raise
 end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 CreateServiceLinkedRole。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
        .send()
        .await?;

    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 CreateServiceLinkedRole。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listpolicyversions(
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 CreateServiceLinkedRole。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func createServiceLinkedRole(service: String, suffix: String? = nil,
 description: String?)
                    async throws -> IAMClientTypes.Role {
        let input = CreateServiceLinkedRoleInput(
            awsServiceName: service,
            customSuffix: suffix,
            description: description
        )
        do {
            let output = try await client.createServiceLinkedRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
```

```
            return role
        } catch {
            print("ERROR: createServiceLinkedRole:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 CreateServiceLinkedRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateUser` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateUser`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 建立唯讀和讀寫的使用者

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
/// <summary>
/// Create an IAM user.
/// </summary>
/// <param name="userName">The username for the new IAM user.</param>
/// <returns>The IAM user that was created.</returns>
public async Task<User> CreateUserAsync(string userName)
```

```
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 CreateUser。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
```

```
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateUser。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
Aws::IAM::IAMClient iam(clientConfig);

Aws::IAM::Model::CreateUserRequest create_request;
create_request.SetUserName(userName);

auto create_outcome = iam.CreateUser(create_request);
if (!create_outcome.IsSuccess()) {
    std::cerr << "Error creating IAM user " << userName << ":" <<
              create_outcome.GetError().GetMessage() << std::endl;
}
else {
    std::cout << "Successfully created IAM user " << userName << std::endl;
}

return create_outcome.IsSuccess();
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 CreateUser。

CLI

AWS CLI

範例 1：建立 IAM 使用者

下列 `create-user` 命令會建立目前帳戶中名為 Bob 的 IAM 使用者。

```
aws iam create-user \
    --user-name Bob
```

輸出：

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
}
```

如需詳細資訊，請參閱《IAM 使用者指南》中的在 AWS 帳戶中建立 AWS IAM 使用者。

範例 2：在指定路徑建立 IAM 使用者

下列 `create-user` 命令會在指定路徑建立名為 Bob 的 IAM 使用者。

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

輸出：

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 識別符。

範例 3：建立內含標籤的 IAM 使用者

下列 `create-user` 命令會建立內含標籤、名為 Bob 的 IAM 使用者。此範例使用具有下列 JSON 格式標記的 `--tags` 參數旗標：`'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`。或者，`--tags` 旗標可以與速記格式的標籤一起使用：`'Key=Department,Value=Accounting Key=Location,Value=Seattle'`。

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

輸出：

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的標記 IAM 使用者。

範例 3：建立具有設定許可界限的 IAM 使用者

下列 `create-user` 命令會使用 AmazonS3FullAccess 許可界限，建立名為 Bob 的 IAM 使用者。

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

輸出：

```
{
    "User": {
```

```
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 實體許可界限。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateUser。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
```

```
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 CreateUser。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
```

```
import software.amazon.awssdk.services.iam.model.GetUserResponse;

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

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
                """;

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

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
```

```
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
 iamWaiter.waitUntilUserExists(userRequest);

 waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 CreateUser。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立使用者。

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
  return client.send(command);
};
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateUser。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
```

```
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
    );
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateUser。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun createIAMUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 CreateUser。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";


    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
            'UserName' => $name,
        ]);

        return $result['User'];
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 CreateUser。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立名為 **Bob** 的 IAM 使用者。如果 Bob 需要登入 AWS 主控台，則您必須分別執行 命令**New-IAMLoginProfile**，以使用密碼建立登入設定檔。如果 Bob 需要執行 AWS

PowerShell 或跨平台 CLI 命令或進行 AWS API 呼叫，則必須單獨執行**New-IAMAccessKey**命令來建立存取金鑰。

```
New-IAMUser -UserName Bob
```

輸出：

```
Arn               : arn:aws:iam::123456789012:user/Bob
CreateDate        : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path              : /
UserId            : AIDAJWGEFDMEMEXAMPLE1
UserName          : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateUser。

Tools for PowerShell V5

範例 1：此範例會建立名為 **Bob** 的 IAM 使用者。如果 Bob 需要登入 AWS 主控台，則您必須分別執行 命令**New-IAMLoginProfile**，以使用密碼建立登入設定檔。如果 Bob 需要執行 AWS PowerShell 或跨平台 CLI 命令或進行 AWS API 呼叫，則必須單獨執行**New-IAMAccessKey**命令來建立存取金鑰。

```
New-IAMUser -UserName Bob
```

輸出：

```
Arn               : arn:aws:iam::123456789012:user/Bob
CreateDate        : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path              : /
UserId            : AIDAJWGEFDMEMEXAMPLE1
UserName          : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateUser。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 CreateUser。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
  # Creates a user and their login profile
  #
  # @param user_name [String] The name of the user
  # @param initial_password [String] The initial password for the user
  # @return [String, nil] The ID of the user if created, or nil if an error
 occurred
  def create_user(user_name, initial_password)
    response = @iam_client.create_user(user_name: user_name)
    @iam_client.wait_until(:user_exists, user_name: user_name)
    @iam_client.create_login_profile(
      user_name: user_name,
      password: initial_password,
      password_reset_required: true
    )
    @logger.info("User '#{user_name}' created successfully.")
    response.user.user_id
  rescue Aws::IAM::Errors::EntityAlreadyExists
    @logger.error("Error creating user '#{user_name}': user already exists.")
    nil
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating user '#{user_name}': #{e.message}")
    nil
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 CreateUser。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;
```

```
    Ok(response.user.unwrap())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 CreateUser。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
TRY.
    oo_result = lo_iam->createuser(
      iv_username = iv_user_name ).
    MESSAGE 'User created successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamentityalrdyexex.
    MESSAGE 'User already exists.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Limit exceeded for IAM users.' TYPE 'E'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Entity does not exist.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 CreateUser。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func createUser(name: String) async throws -> String {
        let input = CreateUserInput(
            userName: name
        )
        do {
            let output = try await client.createUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            guard let id = user.userId else {
                throw ServiceHandlerError.noSuchUser
            }
            return id
        } catch {
            print("ERROR: createUser:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 CreateUser。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **CreateVirtualMfaDevice** 與 CLI

下列程式碼範例示範如何使用 CreateVirtualMfaDevice。

CLI

AWS CLI

若要建立虛擬 MFA 裝置

此範例會建立名為 BobsMFADevice 的新虛擬 MFA 裝置。它會建立包含名為 QRCode.png 的引導資訊的檔案，並將其放至 C:/ 目錄中。此範例中使用的引導方法為 QRCodePNG。

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
    --bootstrap-method QRCodePNG
```

輸出：

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 AWS中使用多重要素驗證 (MFA)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateVirtualMfaDevice。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立新的虛擬 MFA 裝置。第 2 行和第 3 行擷取虛擬 MFA 軟體程式建立帳戶所需的 **Base32StringSeed** 值 (做為 QR 程式碼的替代方案)。使用值設定程式後，依序從程式取得兩個驗證碼。最後，使用最後一個命令將虛擬 MFA 裝置連結至 IAM 使用者 **Bob**，並使用兩個驗證碼同步帳戶。

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
```

```
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

輸出：

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

範例 2：此範例會建立新的虛擬 MFA 裝置。第 2 行和第 3 行會擷取 **QRCodePNG** 值，並將其寫入檔案。此映像可由虛擬 MFA 軟體程式掃描，以建立帳戶 (做為手動輸入 Base32StringSeed 值的替代方案)。在虛擬 MFA 程式中建立帳戶後，請依序取得兩個驗證碼，並在最後一個命令中輸入它們，將虛擬 MFA 裝置連結至 IAM 使用者 **Bob** 並同步帳戶。

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

輸出：

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateVirtualMfaDevice。

Tools for PowerShell V5

範例 1：此範例會建立新的虛擬 MFA 裝置。第 2 行和第 3 行擷取虛擬 MFA 軟體程式建立帳戶所需的 **Base32StringSeed** 值 (做為 QR 程式碼的替代方案)。使用值設定程式後，依序從程式取得兩個驗證碼。最後，使用最後一個命令將虛擬 MFA 裝置連結至 IAM 使用者 **Bob**，並使用兩個驗證碼同步帳戶。

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
```

```
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

輸出：

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

範例 2：此範例會建立新的虛擬 MFA 裝置。第 2 行和第 3 行會擷取 **QRCodePNG** 值，並將其寫入檔案。此映像可由虛擬 MFA 軟體程式掃描，以建立帳戶 (做為手動輸入 Base32StringSeed 值的替代方案)。在虛擬 MFA 程式中建立帳戶後，請依序取得兩個驗證碼，並在最後一個命令中輸入它們，將虛擬 MFA 裝置連結至 IAM 使用者 **Bob** 並同步帳戶。

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

輸出：

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateVirtualMfaDevice。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeactivateMfaDevice** 與 CLI

下列程式碼範例示範如何使用 DeactivateMfaDevice。

CLI

AWS CLI

若要停用 MFA 裝置

此命令會使用與使用者 Bob 關聯的 ARN arn:aws:iam::210987654321:mfa/BobsMFADevice 停用虛擬 MFA 裝置。

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 AWS中使用多重要素驗證 (MFA)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeactivateMfaDevice。

PowerShell

Tools for PowerShell V4

範例 1：此命令會停用與序號為 **123456789012** 的使用者 **Bob** 關聯的硬體 MFA 裝置。

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

範例 2：此命令會停用與 ARN 為 **arn:aws:iam::210987654321:mfa/David** 的使用者 **David** 關聯的虛擬 MFA 裝置。請注意，虛擬 MFA 裝置不會從帳戶中刪除。虛擬裝置仍然存在，並顯示在 **Get-IAMVirtualMFADevice** 命令的輸出中。必須先使用 **Remove-IAMVirtualMFADevice** 命令刪除舊的虛擬 MFA 裝置，才能為同一使用者建立新裝置。

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeactivateMfaDevice。

Tools for PowerShell V5

範例 1：此命令會停用與序號為 **123456789012** 的使用者 **Bob** 關聯的硬體 MFA 裝置。

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

範例 2：此命令會停用與 ARN 為 **arn:aws:iam::210987654321:mfa/David** 的使用者 **David** 關聯的虛擬 MFA 裝置。請注意，虛擬 MFA 裝置不會從帳戶中刪除。虛擬裝置仍然存在，並顯示在 **Get-IAMVirtualMFADevice** 命令的輸出中。必須先使用 **Remove-IAMVirtualMFADevice** 命令刪除舊的虛擬 MFA 裝置，才能為同一使用者建立新裝置。

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeactivateMfaDevice。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteAccessKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteAccessKey。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 建立唯讀和讀寫的使用者
- 管理存取金鑰

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
```

```
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 DeleteAccessKey。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
```

```
}

########################################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
```

```
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteAccessKey。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 DeleteAccessKey。

CLI

AWS CLI

刪除 IAM 使用者的存取金鑰

下列 `delete-access-key` 命令會為名為 Bob 的 IAM 使用者刪除指定的存取金鑰 (存取金鑰 ID 與私密存取金鑰)。

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

此命令不會產生輸出。

若要列出為 IAM 使用者定義的存取金鑰，請使用 `list-access-keys` 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的存取金鑰。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteAccessKey。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
```

```
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 DeleteAccessKey。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
```

```
import software.amazon.awssdk.services.iam.model.IamException;

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

                Usage:
                    <username> <accessKey>\s

                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
```

```
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 DeleteAccessKey。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

刪除存取金鑰。

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
```

```
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteAccessKey。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteAccessKey。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun deleteKey(
    userNameVal: String,
    accessKey: String,
) {
    val request =
        DeleteAccessKeyRequest {
            accessKeyId = accessKey
            userName = userNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 DeleteAccessKey。

PowerShell

Tools for PowerShell V4

範例 1：此範例AKIAIOSFODNN7EXAMPLE會從名為 的使用者中刪除具有金鑰 ID 的 AWS 存取金鑰對Bob。

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteAccessKey。

Tools for PowerShell V5

範例 1：此範例**AKIAIOSFODNN7EXAMPLE**會從名為 的使用者中刪除具有金鑰 ID 的 AWS 存取金鑰對**Bob**。

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteAccessKey。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DeleteAccessKey。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、停用和刪除存取金鑰。

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
```

```
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
```

```
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 DeleteAccessKey。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
```

```
            }
        }
    }
    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DeleteAccessKey。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
TRY.
    lo_iam->deleteaccesskey(
      iv_accesskeyid = iv_access_key_id
      iv_username = iv_user_name ).
    MESSAGE 'Access key deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Access key or user does not exist.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DeleteAccessKey。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                                key: IAMClientTypes.AccessKey) async throws
    {
        let userName: String?

        if user != nil {
            userName = user!.userName
        } else {
            userName = nil
        }

        let input = DeleteAccessKeyInput(
            accessKeyId: key.accessKeyId,
            userName: userName
        )
        do {
            _ = try await iamClient.deleteAccessKey(input: input)
        } catch {
            print("ERROR: deleteAccessKey:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 DeleteAccessKey。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteAccountAlias 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteAccountAlias。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理您的帳戶

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 DeleteAccountAlias。

CLI

AWS CLI

刪除帳戶別名

下列 delete-account-alias 命令會移除目前帳戶的別名 mycompany。

```
aws iam delete-account-alias \
    --account-alias mycompany
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的AWS 您的帳戶 ID 及其別名。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteAccountAlias。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to delete.\s
                """;

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

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

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

```
        }
        System.out.println("Done");
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 DeleteAccountAlias。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

刪除帳戶別名。

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteAccountAlias。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteAccountAlias。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun deleteIAMAccountAlias(alias: String) {
    val request =
        DeleteAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 DeleteAccountAlias。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從 中移除帳戶別名 AWS 帳戶。在 https://mycompanyaws.signin.aws.amazon.com/console，使用者無法再使用別名登入頁面。您必須改為在 https：//https://<accountidnumber>.signin.aws.amazon.com/console 使用原始 URL 搭配您的 AWS 帳戶 ID 號碼。

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteAccountAlias。

Tools for PowerShell V5

範例 1：此範例會從 中移除帳戶別名 AWS 帳戶。在 https://mycompanyaws.signin.aws.amazon.com/console，使用者無法再使用別名登入頁面。您必須改為在 https：//https://<accountidnumber>.signin.aws.amazon.com/console 使用原始 URL 搭配您的 AWS 帳戶 ID 號碼。

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteAccountAlias。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DeleteAccountAlias。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出、建立和刪除帳戶別名。

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 DeleteAccountAlias。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->deleteaccountalias(
          iv_accountalias = iv_account_alias ).
        MESSAGE 'Account alias deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Account alias does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DeleteAccountAlias。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`DeleteAccountPasswordPolicy`** 與 CLI

下列程式碼範例示範如何使用 `DeleteAccountPasswordPolicy`。

CLI

AWS CLI

若要刪除目前的帳戶密碼政策

下列 `delete-account-password-policy` 命令會移除目前帳戶的密碼政策。

```
aws iam delete-account-password-policy
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的設定 IAM 使用者的帳戶密碼政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteAccountPasswordPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除 的密碼政策， AWS 帳戶 並將所有值重設為其原始預設值。如果密碼政策目前不存在，則會出現下列錯誤訊息：找不到名為 PasswordPolicy 的帳戶政策。

```
Remove-IAMAccountPasswordPolicy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteAccountPasswordPolicy。

Tools for PowerShell V5

範例 1：此範例會刪除 的密碼政策， AWS 帳戶 並將所有值重設為其原始預設值。如果密碼政策目前不存在，則會出現下列錯誤訊息：找不到名為 PasswordPolicy 的帳戶政策。

```
Remove-IAMAccountPasswordPolicy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteAccountPasswordPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteGroup 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteGroup。

CLI

AWS CLI

刪除 IAM 群組

下列 delete-group 命令會刪除名為 MyTestGroup 的 IAM 群組。

```
aws iam delete-group \
    --group-name MyTestGroup
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的刪除 IAM 使用者群組。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteGroup。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
```

```
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteGroup。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除名為 **MyTestGroup** 的 IAM 群組。第一個命令會移除所有屬於群組的 IAM 使用者，第二個命令則會刪除 IAM 群組。這兩個命令皆可在沒有任何確認提示的情況下運作。

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteGroup。

Tools for PowerShell V5

範例 1：此範例會刪除名為 **MyTestGroup** 的 IAM 群組。第一個命令會移除所有屬於群組的 IAM 使用者，第二個命令則會刪除 IAM 群組。這兩個命令皆可在沒有任何確認提示的情況下運作。

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteGroupPolicy** 與 CLI

下列程式碼範例示範如何使用 DeleteGroupPolicy。

CLI

AWS CLI

從 IAM 群組中刪除政策

下列 delete-group-policy 命令會將名為 ExamplePolicy 的政策從名為 Admins 的群組中刪除。

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

此命令不會產生輸出。

若要查看連接至群組的政策，請使用 list-group-policies 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteGroupPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從 IAM 群組 **Testers** 中移除名為 **TesterPolicy** 的內嵌政策。該群組中的使用者會立即失去此政策中定義的許可。

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteGroupPolicy。

Tools for PowerShell V5

範例 1：此範例會從 IAM 群組 **Testers** 中移除名為 **TesterPolicy** 的內嵌政策。該群組中的使用者會立即失去此政策中定義的許可。

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteGroupPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteInstanceProfile` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeleteInstanceProfile`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建置及管理彈性服務
- 受管串流入門
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
```

```
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 DeleteInstanceProfile。

CLI

AWS CLI

刪除執行個體設定檔

下列 `delete-instance-profile` 命令會刪除名為 `ExampleInstanceProfile` 的執行個體設定檔。

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用執行個體設定檔。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteInstanceProfile。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
const client = new IAMClient({});
await client.send(
  new DeleteInstanceProfileCommand({
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteInstanceProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除名為 **MyAppInstanceProfile** 的 EC2 執行個體設定檔。第一個命令將從執行個體設定檔分離任何角色，然後第二個命令將刪除執行個體設定檔。

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteInstanceProfile。

Tools for PowerShell V5

範例 1：此範例會刪除名為 **MyAppInstanceProfile** 的 EC2 執行個體設定檔。第一個命令將從執行個體設定檔分離任何角色，然後第二個命令將刪除執行個體設定檔。

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteInstanceProfile。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例會從執行個體設定檔中移除角色、分離所有附加到該角色的政策並刪除全部資源。

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
```

```
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
```

```
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DeleteInstanceProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteLoginProfile** 與 CLI

下列程式碼範例示範如何使用 DeleteLoginProfile。

CLI

AWS CLI

若要為 IAM 使用者刪除密碼

下列 delete-login-profile 命令會刪除名為 Bob 之 IAM 使用者的密碼。

```
aws iam delete-login-profile \
    --user-name Bob
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的密碼。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteLoginProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從名為 **Bob** 的 IAM 使用者中刪除登入設定檔。這可防止使用者登入 AWS 主控台。它不會阻止使用者使用可能仍連接至使用者帳戶的 AWS 存取金鑰來執行任何 AWS CLI、PowerShell 或 API 呼叫。

```
Remove-IAMLoginProfile -UserName Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteLoginProfile。

Tools for PowerShell V5

範例 1：此範例會從名為 **Bob** 的 IAM 使用者中刪除登入設定檔。這可防止使用者登入 AWS 主控台。它不會阻止使用者使用可能仍連接至使用者帳戶的 AWS 存取金鑰來執行任何 AWS CLI、PowerShell 或 API 呼叫。

```
Remove-IAMLoginProfile -UserName Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteLoginProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteOpenIdConnectProvider** 與 CLI

下列程式碼範例示範如何使用 DeleteOpenIdConnectProvider。

CLI

AWS CLI

若要刪除 IAM OpenID Connect 身分提供者

此範例會刪除連線至提供者 example.oidcprovider.com 的 IAM OIDC 提供者。

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Creating OpenID Connect (OIDC) identity providers。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteOpenIdConnectProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除連線至提供者 **example.oidcprovider.com** 的 IAM OIDC 提供者。請確定更新或刪除在角色信任政策的 **Principal** 元素中引用此提供者的任何角色。

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteOpenIdConnectProvider。

Tools for PowerShell V5

範例 1：此範例會刪除連線至提供者 **example.oidcprovider.com** 的 IAM OIDC 提供者。請確定更新或刪除在角色信任政策的 **Principal** 元素中引用此提供者的任何角色。

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteOpenIdConnectProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeletePolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeletePolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 建立唯讀和讀寫的使用者
- 建立受管監控工作區
- 受管串流入門
- 工作流程協同運作入門

- 管理政策
- 設定系統管理

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 DeletePolicy。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
```

```
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
  return 1
fi

iecho "delete-policy response:$response"
iecho

return 0
```

```
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeletePolicy。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 DeletePolicy。

CLI

AWS CLI

刪除 IAM 政策

此範例會刪除 ARN 為 arn:aws:iam::123456789012:policy/MySamplePolicy 的政策。

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeletePolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
```

```
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 DeletePolicy。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

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

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");

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

```
        System.out.println("Done");
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 DeletePolicy。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

刪除政策。

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeletePolicy。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
    val request =
        DeletePolicyRequest {
            policyArn = policyARNVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 DeletePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除 ARN 為 **arn:aws:iam::123456789012:policy/MySamplePolicy** 的政策。在刪除政策之前，必須先執行 **Remove-IAMPolicyVersion** 刪除除預設版本之外的所有版本。還必須將政策與任何 IAM 使用者、群組或角色分離。

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

範例 2：此範例會刪除政策，方法是先刪除所有非預設政策版本，將其與所有連接的 IAM 實體分離，最後刪除政策本身。第一行擷取政策物件。第二行擷取未標記為集合預設版本的所有政策版本，然後刪除集合中的每個政策。第三行擷取連接政策的所有 IAM 使用者、群組和角色。第四行到第六行將政策與每個連接的實體分離。最後一行使用此命令移除受管政策以及剩餘的預設版本。此範例會在任何需要 **-Force** 切換參數的行中加入該參數，以抑制確認提示。

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeletePolicy。

Tools for PowerShell V5

範例 1：此範例會刪除 ARN 為 **arn:aws:iam::123456789012:policy/MySamplePolicy** 的政策。在刪除政策之前，必須先執行 **Remove-IAMPolicyVersion** 刪除除預設版本之外的所有版本。還必須將政策與任何 IAM 使用者、群組或角色分離。

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

範例 2：此範例會刪除政策，方法是先刪除所有非預設政策版本，將其與所有連接的 IAM 實體分離，最後刪除政策本身。第一行擷取政策物件。第二行擷取未標記為集合預設版本的所有政策版本，然後刪除集合中的每個政策。第三行擷取連接政策的所有 IAM 使用者、群組和角色。第四行到第六行將政策與每個連接的實體分離。最後一行使用此命令移除受管政策以及剩餘的預設版本。此範例會在任何需要 **-Force** 切換參數的行中加入該參數，以抑制確認提示。

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeletePolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DeletePolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DeletePolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->deletepolicy( iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Policy cannot be deleted due to attachments.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DeletePolicy。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
        let input = DeletePolicyInput(
            policyArn: policy.arn
        )
        do {
            _ = try await iamClient.deletePolicy(input: input)
        } catch {
            print("ERROR: deletePolicy:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 DeletePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeletePolicyVersion` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeletePolicyVersion`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理政策

- 復原政策版本

CLI

AWS CLI

若要刪除受管政策的版本

此範例會從 ARN 為 arn:aws:iam::123456789012:policy/MySamplePolicy 的政策中刪除標識為 v2 的版本。

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱 AWS CLI Command Reference 中的 DeletePolicyVersion。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從 ARN 為 **arn:aws:iam::123456789012:policy/MySamplePolicy** 的政策中刪除標識為 **v2** 的版本。

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

範例 2：此範例會先刪除所有非預設政策版本，然後刪除政策本身，從而刪除政策。第一行擷取政策物件。第二行擷取未標記為集合預設值的所有政策版本，然後使用此命令刪除集合中的每個政策。最後一行移除政策本身以及剩餘的預設版本。請注意，若要成功刪除受管政策，還必須使用 **Unregister-IAMUserPolicy**、**Unregister-IAMGroupPolicy** 和 **Unregister-IAMRolePolicy** 命令，將政策與任何使用者、群組或角色分離。請參閱 **Remove-IAMPolicy** cmdlet 的範例。

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
```

```
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeletePolicyVersion。

Tools for PowerShell V5

範例 1：此範例會從 ARN 為 **arn:aws:iam::123456789012:policy/MySamplePolicy** 的政策中刪除標識為 **v2** 的版本。

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

範例 2：此範例會先刪除所有非預設政策版本，然後刪除政策本身，從而刪除政策。第一行擷取政策物件。第二行擷取未標記為集合預設值的所有政策版本，然後使用此命令刪除集合中的每個政策。最後一行移除政策本身以及剩餘的預設版本。請注意，若要成功刪除受管政策，還必須使用 **Unregister-IAMUserPolicy**、**Unregister-IAMGroupPolicy** 和 **Unregister-IAMRolePolicy** 命令，將政策與任何使用者、群組或角色分離。請參閱 **Remove-IAMPolicy** cmdlet 的範例。

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeletePolicyVersion。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
```

```
        lo_iam->deletepolicyversion(
          iv_policyarn = iv_policy_arn
          iv_versionid = iv_version_id ).
        MESSAGE 'Policy version deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy or version does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Cannot delete default policy version.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DeletePolicyVersion。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteRole 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteRole。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 設定容器服務連線
- 使用函數代理整合建立靜態 API
- 建立受管監控工作區
-
- 建立您的第一個無伺服器函數
- 開始使用無伺服器資料倉儲
- 組態管理入門
- 開始使用物聯網裝置保護
- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門

- 受管串流入門
- 佈建資料倉儲叢集入門
- 工作流程協同運作入門
- 管理角色
- 將硬式編碼秘密移至安全秘密儲存
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 設定系統管理
- 在監控儀表板中使用屬性變數來監控多個無伺服器函數

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 DeleteRole。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
################################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteRole。

CLI

AWS CLI

刪除 IAM 角色

下列 delete-role 命令會將名為 Test-Role 的角色移除。

```
aws iam delete-role \
    --role-name Test-Role
```

此命令不會產生輸出。

刪除角色之前，您必須先從任何執行個體設定檔 (remove-role-from-instance-profile) 移除該角色、分離任何受管政策 (detach-role-policy)，並刪除任何連接至該角色的內嵌政策 (delete-role-policy)。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 角色和使用執行個體設定檔。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteRole。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 DeleteRole。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

刪除角色。

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從目前的 IAM 帳戶刪除名為 **MyNewRole** 的角色。在刪除角色之前，必須先使用 **Unregister-IAMRolePolicy** 命令來分離任何受管政策。內嵌政策會與角色一起刪除。

```
Remove-IAMRole -RoleName MyNewRole
```

範例 2：此範例會從名為 **MyNewRole** 的角色分離任何受管政策，然後刪除該角色。第一行將連接至角色的任何受管政策擷取為集合，然後讓集合中的每個政策與角色分離。第二行刪除角色本身。內嵌政策會與角色一起刪除。

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteRole。

Tools for PowerShell V5

範例 1：此範例會從目前的 IAM 帳戶刪除名為 **MyNewRole** 的角色。在刪除角色之前，必須先使用 **Unregister-IAMRolePolicy** 命令來分離任何受管政策。內嵌政策會與角色一起刪除。

```
Remove-IAMRole -RoleName MyNewRole
```

範例 2：此範例會從名為 **MyNewRole** 的角色分離任何受管政策，然後刪除該角色。第一行將連接至角色的任何受管政策擷取為集合，然後讓集合中的每個政策與角色分離。第二行刪除角色本身。內嵌政策會與角色一起刪除。

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteRole。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DeleteRole。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    # Detach and delete attached policies
    @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
      response.attached_policies.each do |policy|
        @iam_client.detach_role_policy({
                                         role_name: role_name,
                                         policy_arn: policy.policy_arn
                                       })
        # Check if the policy is a customer managed policy (not AWS managed)
```

```
        unless policy.policy_arn.include?('aws:policy/')
          @iam_client.delete_policy({ policy_arn: policy.policy_arn })
          @logger.info("Deleted customer managed policy #{policy.policy_name}.")
        end
      end
    end

    # Delete the role
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Deleted role #{role_name}.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't detach policies and delete role #{role_name}. Here's
 why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 DeleteRole。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
        .await
        .is_err()
    {
        sleep(Duration::from_secs(2)).await;
    }
```

```
    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DeleteRole。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->deleterole( iv_rolename = iv_role_name ).
        MESSAGE 'Role deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Role cannot be deleted due to attached resources.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DeleteRole。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
```

```
import AWSS3


    public func deleteRole(role: IAMClientTypes.Role) async throws {
        let input = DeleteRoleInput(
            roleName: role.roleName
        )
        do {
            _ = try await iamClient.deleteRole(input: input)
        } catch {
            print("ERROR: deleteRole:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 DeleteRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`DeleteRolePermissionsBoundary`** 與 CLI

下列程式碼範例示範如何使用 `DeleteRolePermissionsBoundary`。

CLI

AWS CLI

若要從 IAM 角色刪除許可界限

下列 `delete-role-permissions-boundary` 範例將刪除指定 IAM 角色的許可界限。若要將許可界限套用至角色，請使用 `put-role-permissions-boundary` 命令。

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteRolePermissionsBoundary。

PowerShell

Tools for PowerShell V4

範例 1：此範例示範如何移除連接至 IAM 角色的許可界限。

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteRolePermissionsBoundary。

Tools for PowerShell V5

範例 1：此範例示範如何移除連接至 IAM 角色的許可界限。

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteRolePermissionsBoundary。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteRolePolicy` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeleteRolePolicy`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 開始使用無伺服器資料倉儲
- 組態管理入門
- 開始使用物聯網裝置保護
- 佈建資料倉儲叢集入門
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 DeleteRolePolicy。

CLI

AWS CLI

從 IAM 角色中移除政策

下列 delete-role-policy 命令會將名為 ExamplePolicy 的政策從名為 Test-Role 的角色中移除。

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的修改角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteRolePolicy。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
  });
  return client.send(command);
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 DeleteRolePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除內嵌在 IAM 角色 **S3BackupRole** 中的內嵌政策 **S3AccessPolicy**。

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteRolePolicy。

Tools for PowerShell V5

範例 1：此範例會刪除內嵌在 IAM 角色 **S3BackupRole** 中的內嵌政策 **S3AccessPolicy**。

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteRolePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteSAMLProvider 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteSAMLProvider。

CLI

AWS CLI

刪除 SAML 提供者

此範例會刪除 ARN 為 arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider 的 IAM SAML 2.0 提供者。

```
aws iam delete-saml-provider \
```

```
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM SAML 身分提供者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteSAMLProvider。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 DeleteSAMLProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除 ARN 為 **arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider** 的 IAM SAML 2.0 提供者。

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteSAMLProvider。

Tools for PowerShell V5

範例 1：此範例會刪除 ARN 為 **arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider** 的 IAM SAML 2.0 提供者。

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteSAMLProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteServerCertificate 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteServerCertificate。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeleteServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    const auto outcome = iam.DeleteServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error deleting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        std::cout << "Successfully deleted server certificate " <<
 certificateName
                  << std::endl;
    }

    return result;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 DeleteServerCertificate。

CLI

AWS CLI

從 AWS 您的帳戶刪除伺服器憑證

下列delete-server-certificate命令會從 AWS 您的帳戶移除指定的伺服器憑證。

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

此命令不會產生輸出。

若要列出您 AWS 帳戶中可用的伺服器憑證，請使用 `list-server-certificates`命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 IAM 中管理伺服器憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteServerCertificate。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

刪除伺服器憑證。

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v3/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-deleting。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteServerCertificate。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-deleting。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteServerCertificate。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除名為 **MyServerCert** 的伺服器憑證。

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteServerCertificate。

Tools for PowerShell V5

範例 1：此範例會刪除名為 **MyServerCert** 的伺服器憑證。

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteServerCertificate。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出、更新和刪除伺服器憑證。

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
```

```
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end
```

```
  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 DeleteServerCertificate。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteServiceLinkedRole` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeleteServiceLinkedRole`。

CLI

AWS CLI

刪除服務連結角色

下列 `delete-service-linked-role` 範例會刪除您不再需要的指定服務連結角色。刪除會以非同步方式發生。您可以使用 `get-service-linked-role-deletion-status` 命令，檢查刪除狀態並確認刪除的時間。

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

輸出：

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
```

```
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用服務連結角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteServiceLinkedRole。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
```

```
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 DeleteServiceLinkedRole。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 DeleteServiceLinkedRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例已經刪除服務連結的角色。請注意，如果服務仍在使用該角色，則此命令會導致失敗。

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteServiceLinkedRole。

Tools for PowerShell V5

範例 1：此範例已經刪除服務連結的角色。請注意，如果服務仍在使用該角色，則此命令會導致失敗。

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteServiceLinkedRole。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Deletes a service-linked role.
#
# @param role_name [String] The name of the role to delete.
def delete_service_linked_role(role_name)
  response = @iam_client.delete_service_linked_role(role_name: role_name)
```

```
    task_id = response.deletion_task_id
    check_deletion_status(role_name, task_id)
  rescue Aws::Errors::ServiceError => e
    handle_deletion_error(e, role_name)
  end

  private

  # Checks the deletion status of a service-linked role
  #
  # @param role_name [String] The name of the role being deleted
  # @param task_id [String] The task ID for the deletion process
  def check_deletion_status(role_name, task_id)
    loop do
      response = @iam_client.get_service_linked_role_deletion_status(
        deletion_task_id: task_id
      )
      status = response.status
      @logger.info("Deletion of #{role_name} #{status}.")
      break if %w[SUCCEEDED FAILED].include?(status)

      sleep(3)
    end
  end

  # Handles deletion error
  #
  # @param e [Aws::Errors::ServiceError] The error encountered during deletion
  # @param role_name [String] The name of the role attempted to delete
  def handle_deletion_error(e, role_name)
    return if e.code == 'NoSuchEntity'

    @logger.error("Couldn't delete #{role_name}. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 DeleteServiceLinkedRole。

Rust

適用於 Rust 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DeleteServiceLinkedRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`DeleteSigningCertificate`** 與 CLI

下列程式碼範例示範如何使用 `DeleteSigningCertificate`。

CLI

AWS CLI

若要刪除 IAM 使用者的簽署憑證

下列 `delete-signing-certificate` 命令將刪除名為 Bob 之 IAM 使用者的指定簽署憑證。

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

此命令不會產生輸出。

若要取得簽署憑證的 ID，請使用 `list-signing-certificates` 命令。

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的管理簽署憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteSigningCertificate。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從名為 **Bob** 的 IAM 使用者中刪除 ID 為 **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** 的簽署憑證。

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteSigningCertificate。

Tools for PowerShell V5

範例 1：此範例會從名為 **Bob** 的 IAM 使用者中刪除 ID 為 **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** 的簽署憑證。

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteSigningCertificate。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteUser` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeleteUser`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 建立唯讀和讀寫的使用者

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 DeleteUser。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
########################################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
```

```
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteUser。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
```

```
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 DeleteUser。

CLI

AWS CLI

刪除 IAM 使用者

下列 `delete-user` 命令會將名為 Bob 的 IAM 使用者從目前帳戶中移除。

```
aws iam delete-user \
    --user-name Bob
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的刪除 IAM 使用者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteUser。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
```

```
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 DeleteUser。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

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

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;

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

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
```

```
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 DeleteUser。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

刪除使用者。

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteUser。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteUser。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun deleteIAMUser(userNameVal: String) {
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 DeleteUser。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除名為 **Bob** 的 IAM 使用者。

```
Remove-IAMUser -UserName Bob
```

範例 2：此範例會刪除名為 **Theresa** 的 IAM 使用者以及任何必須先刪除的元素。

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
```

```
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteUser。

Tools for PowerShell V5

範例 1：此範例會刪除名為 Bob 的 IAM 使用者。

```
Remove-IAMUser -UserName Bob
```

範例 2：此範例會刪除名為 Theresa 的 IAM 使用者以及任何必須先刪除的元素。

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
```

```
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteUser。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DeleteUser。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 DeleteUser。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DeleteUser。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->deleteuser( iv_username = iv_user_name ).
        MESSAGE 'User deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'User cannot be deleted due to attached resources.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DeleteUser。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func deleteUser(user: IAMClientTypes.User) async throws {
        let input = DeleteUserInput(
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUser(input: input)
        } catch {
            print("ERROR: deleteUser:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 DeleteUser。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteUserPermissionsBoundary** 與 CLI

下列程式碼範例示範如何使用 DeleteUserPermissionsBoundary。

CLI

AWS CLI

若要從 IAM 使用者刪除許可界限

下列 delete-user-permissions-boundary 範例將刪除連接到名為 intern 之 IAM 使用者的許可界限。若要將許可界限套用至使用者，請使用 put-user-permissions-boundary 命令。

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteUserPermissionsBoundary。

PowerShell

Tools for PowerShell V4

範例 1：此範例示範如何移除連接到 IAM 使用者的許可界限。

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteUserPermissionsBoundary。

Tools for PowerShell V5

範例 1：此範例示範如何移除連接到 IAM 使用者的許可界限。

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteUserPermissionsBoundary。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteUserPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteUserPolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 DeleteUserPolicy。

CLI

AWS CLI

從 IAM 使用者中移除政策

下列 `delete-user-policy` 命令會將指定的政策從名為 Bob 的 IAM 使用者中移除。

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

此命令不會產生輸出。

若要取得 IAM 使用者的政策清單，請使用 `list-user-policies` 命令。

如需詳細資訊，請參閱《IAM 使用者指南》中的在 AWS 帳戶中建立 AWS IAM 使用者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteUserPolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"
```

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 DeleteUserPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除內嵌在名為 Bob 之 IAM 使用者中的、名為 AccessToEC2Policy 的內嵌政策。

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

範例 2：此範例會尋找內嵌在名為 **Theresa** 的 IAM 使用者中的所有內嵌政策，然後將其刪除。

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteUserPolicy。

Tools for PowerShell V5

範例 1：此範例會刪除內嵌在名為 **Bob** 之 IAM 使用者中的、名為 **AccessToEC2Policy** 的內嵌政策。

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

範例 2：此範例會尋找內嵌在名為 **Theresa** 的 IAM 使用者中的所有內嵌政策，然後將其刪除。

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteUserPolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
  # Deletes a user and their associated resources
  #
  # @param user_name [String] The name of the user to delete
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
```

```
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 DeleteUserPolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
        .await?;

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DeleteUserPolicy。

Swift

SDK for Swift

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
 throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            print("ERROR: deleteUserPolicy:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 DeleteUserPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`DeleteVirtualMfaDevice`** 與 CLI

下列程式碼範例示範如何使用 `DeleteVirtualMfaDevice`。

CLI

AWS CLI

若要移除虛擬 MFA 裝置

下列 `delete-virtual-mfa-device` 命令會從目前帳戶移除指定的 MFA 裝置。

```
aws iam delete-virtual-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 Deactivating MFA devices。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteVirtualMfaDevice。

PowerShell

Tools for PowerShell V4

範例 1：此範例會刪除 ARN 為 **arn:aws:iam::123456789012:mfa/bob** 的 IAM 虛擬 MFA 裝置。

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

範例 2：此範例會檢查 IAM 使用者 Theresa 是否已獲分派 MFA 裝置。如果找到一個，系統會為 IAM 使用者停用裝置。如果裝置是虛擬裝置，亦將刪除。

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DeleteVirtualMfaDevice。

Tools for PowerShell V5

範例 1：此範例會刪除 ARN 為 **arn:aws:iam::123456789012:mfa/bob** 的 IAM 虛擬 MFA 裝置。

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

範例 2：此範例會檢查 IAM 使用者 Theresa 是否已獲分派 MFA 裝置。如果找到一個，系統會為 IAM 使用者停用裝置。如果裝置是虛擬裝置，亦將刪除。

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DeleteVirtualMfaDevice。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DetachGroupPolicy** 與 CLI

下列程式碼範例示範如何使用 DetachGroupPolicy。

CLI

AWS CLI

從群組分離政策

此範例會從名為 Testers 的群組中移除 ARN 為 arn:aws:iam::123456789012:policy/TesterAccessPolicy 的受管政策。

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者群組。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DetachGroupPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將 ARN 為 **arn:aws:iam::123456789012:policy/TesterAccessPolicy** 的受管群組政策與名為 **Testers** 的群組分離。

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

範例 2：此範例會找出連接到名為 **Testers** 的群組的所有受管政策，並將其與群組分離。

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DetachGroupPolicy。

Tools for PowerShell V5

範例 1：此範例會將 ARN 為 **arn:aws:iam::123456789012:policy/TesterAccessPolicy** 的受管群組政策與名為 **Testers** 的群組分離。

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

範例 2：此範例會找出連接到名為 **Testers** 的群組的所有受管政策，並將其與群組分離。

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DetachGroupPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DetachRolePolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DetachRolePolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念
- 設定容器服務連線
- 使用函數代理整合建立靜態 API
- 建立受管監控工作區
-
- 建立您的第一個無伺服器函數
- 組態管理入門
- 開始使用物聯網裝置保護
- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門
- 受管串流入門
- 工作流程協同運作入門
- 管理角色
- 將硬式編碼秘密移至安全秘密儲存
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 設定系統管理
- 在監控儀表板中使用屬性變數來監控多個無伺服器函數

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
/// <summary>
/// Detach an IAM policy from an IAM role.
/// </summary>
```

```
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 DetachRolePolicy。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
##############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DetachRolePolicy。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
"
                  << roleName << std::endl;
    }

    return detachOutcome.IsSuccess();
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 DetachRolePolicy。

CLI

AWS CLI

將政策與角色分離

此範例會將具有 ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` 的受管政策從名為 `FedTesterRole` 的角色中移除。

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的修改角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DetachRolePolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
```

```
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 DetachRolePolicy。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

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

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
```

```
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }

    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 DetachRolePolicy。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

分離政策。

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DetachRolePolicy。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DetachRolePolicy。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun detachPolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        DetachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 DetachRolePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將 ARN 為 `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` 的受管群組政策與名為 `FedTesterRole` 的角色分離。

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

範例 2：此範例會找出連接到名為 **FedTesterRole** 的角色的所有受管政策，並將其與角色分離。

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DetachRolePolicy。

Tools for PowerShell V5

範例 1：此範例會將 ARN 為 **arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy** 的受管群組政策與名為 **FedTesterRole** 的角色分離。

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

範例 2：此範例會找出連接到名為 **FedTesterRole** 的角色的所有受管政策，並將其與角色分離。

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DetachRolePolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

使用 Boto3 Policy 物件將政策與角色分離。

```
def detach_from_role(role_name, policy_arn):
```

```
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

使用 Boto3 Role 物件將政策與角色分離。

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DetachRolePolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、連接和分離角色政策。

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 DetachRolePolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;
```

```
    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DetachRolePolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->detachrolepolicy(
          iv_rolename = iv_role_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy detached from role successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role or policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DetachRolePolicy。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
 IAMClientTypes.Role) async throws {
        let input = DetachRolePolicyInput(
            policyArn: policy.arn,
            roleName: role.roleName
        )

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

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 DetachRolePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DetachUserPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DetachUserPolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建立唯讀和讀寫的使用者

CLI

AWS CLI

將政策與使用者分離

此範例會將具有 ARN arn:aws:iam::123456789012:policy/TesterPolicy 的受管政策從使用者 Bob 中移除。

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的變更 IAM 使用者的許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DetachUserPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將 ARN 為 **arn:aws:iam::123456789012:policy/TesterPolicy** 的受管政策與名為 **Bob** 的 IAM 使用者分離。

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

範例 2：此範例會找出連接到名為 **Theresa** 的 IAM 使用者的所有受管政策，並將這些政策與使用者分離。

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DetachUserPolicy。

Tools for PowerShell V5

範例 1：此範例會將 ARN 為 **arn:aws:iam::123456789012:policy/TesterPolicy** 的受管政策與名為 **Bob** 的 IAM 使用者分離。

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

範例 2：此範例會找出連接到名為 **Theresa** 的 IAM 使用者的所有受管政策，並將這些政策與使用者分離。

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DetachUserPolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 DetachUserPolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
 # Detaches a policy from a user
 #
 # @param user_name [String] The name of the user
 # @param policy_arn [String] The ARN of the policy to detach
 # @return [Boolean] true if the policy was successfully detached, false
otherwise
 def detach_user_policy(user_name, policy_arn)
   @iam_client.detach_user_policy(
     user_name: user_name,
     policy_arn: policy_arn
   )
   @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
   true
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error('Error detaching policy: Policy or user does not exist.')
   false
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error detaching policy from user '#{user_name}':
#{e.message}")
   false
 end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 DetachUserPolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 DetachUserPolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->detachuserpolicy(
```

```
          iv_username = iv_user_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy detached from user successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User or policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 DetachUserPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **EnableMfaDevice** 與 CLI

下列程式碼範例示範如何使用 EnableMfaDevice。

CLI

AWS CLI

若要啟用 MFA 裝置

使用 create-virtual-mfa-device 命令建立新的虛擬 MFA 裝置後，可以將 MFA 裝置指派給使用者。下列 enable-mfa-device 範例會將序號為 arn:aws:iam::210987654321:mfa/BobsMFADevice 的 MFA 裝置指派給使用者 Bob。命令也會依序包含來自虛擬 MFA 裝置的前兩個代碼， AWS 以將裝置與 同步。

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的啟用虛擬多重要素驗證 (MFA) 裝置。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 EnableMfaDevice。

PowerShell

Tools for PowerShell V4

範例 1：此命令會啟用序號為 987654321098 的硬體 MFA 裝置，並將裝置與使用者 Bob 關聯起來。它包含裝置中依序排列的前兩個代碼。

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

範例 2：此範例會建立並啟用虛擬 MFA 裝置。第一個命令會建立虛擬裝置，並在變數 $MFADevice 中傳回裝置的物件表示。可以使用 .Base32StringSeed 或 QRCodePng 屬性來設定使用者的軟體應用程式。最後一個命令會將裝置指派給使用者 David，依裝置序號標識裝置。命令也會依序包含來自虛擬 MFA 裝置的前兩個代碼， AWS 以將裝置與 同步。

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 EnableMfaDevice。

Tools for PowerShell V5

範例 1：此命令會啟用序號為 987654321098 的硬體 MFA 裝置，並將裝置與使用者 Bob 關聯起來。它包含裝置中依序排列的前兩個代碼。

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

範例 2：此範例會建立並啟用虛擬 MFA 裝置。第一個命令會建立虛擬裝置，並在變數 $MFADevice 中傳回裝置的物件表示。可以使用 .Base32StringSeed 或 QRCodePng 屬性來設定使用者的軟體應用程式。最後一個命令會將裝置指派給使用者 David，依裝置序號標識裝置。命令也會依序包含來自虛擬 MFA 裝置的前兩個代碼， AWS 以將裝置與 同步。

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
```

```
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 EnableMfaDevice。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GenerateCredentialReport` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GenerateCredentialReport`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理您的帳戶

CLI

AWS CLI

產生憑證報告

下列範例會嘗試產生 AWS 帳戶的登入資料報告。

```
aws iam generate-credential-report
```

輸出：

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的取得 AWS 帳戶的登入資料報告。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GenerateCredentialReport。

PowerShell

Tools for PowerShell V4

範例 1：此範例會請求產生新的報告，這可以每四小時執行一次。如果最後一個報告仍然是最新的，「狀態」欄位會顯示：**COMPLETE**。**Get-IAMCredentialReport** 可用於檢視已完成的報告。

```
Request-IAMCredentialReport
```

輸出：

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GenerateCredentialReport。

Tools for PowerShell V5

範例 1：此範例會請求產生新的報告，這可以每四小時執行一次。如果最後一個報告仍然是最新的，「狀態」欄位會顯示：**COMPLETE**。**Get-IAMCredentialReport** 可用於檢視已完成的報告。

```
Request-IAMCredentialReport
```

輸出：

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GenerateCredentialReport。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GenerateCredentialReport。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
TRY.
    oo_result = lo_iam->generatecredentialreport( ).
    MESSAGE 'Credential report generation started.' TYPE 'I'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Report generation limit exceeded.' TYPE 'E'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when generating credential report.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GenerateCredentialReport。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GenerateServiceLastAccessedDetails** 與 CLI

下列程式碼範例示範如何使用 GenerateServiceLastAccessedDetails。

CLI

AWS CLI

範例 1：產生自訂政策的服務存取報告

下列 generate-service-last-accessed-details 範例會啟動背景任務以產生報告，列出 IAM 使用者以及採用名為 intern-boundary 之自訂政策的其他實體存取的服務。您可以執行 get-service-last-accessed-details 命令，讓報告在建立後顯示出來。

```
aws iam generate-service-last-accessed-details \
```

```
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

輸出：

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

範例 2：產生 AWS 受管 AdministratorAccess 政策的服務存取報告

下列generate-service-last-accessed-details範例會啟動背景任務，以產生報告，列出 IAM 使用者和其他實體使用 AWS 受管AdministratorAccess政策存取的服務。您可以執行 get-service-last-accessed-details 命令，讓報告在建立後顯示出來。

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

輸出：

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的AWS 使用上次存取資訊在 中精簡許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考i中的 GenerateServiceLastAccessedDetails。

PowerShell

Tools for PowerShell V4

範例 1：此範例等同於 GenerateServiceLastAccessedDetails API 的 cmdlet。這提供了任務 ID，可用於 Get-IAMServiceLastAccessedDetail 和 Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GenerateServiceLastAccessedDetails。

Tools for PowerShell V5

範例 1：此範例等同於 GenerateServiceLastAccessedDetails API 的 cmdlet。這提供了任務 ID，可用於 Get-IAMServiceLastAccessedDetail 和 Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GenerateServiceLastAccessedDetails。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetAccessKeyLastUsed 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetAccessKeyLastUsed。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理存取金鑰

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);
```

```
    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
                  secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 GetAccessKeyLastUsed。

CLI

AWS CLI

擷取最後一次使用指定存取金鑰之時機的相關資訊

下列範例會擷取最後一次使用存取金鑰 ABCDEXAMPLE 之時機的相關資訊。

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

輸出：

```
{
    "UserName":  "Bob",
```

```
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的存取金鑰。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetAccessKeyLastUsed。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

獲取存取金鑰。

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);

  if (response.AccessKeyLastUsed?.LastUsedDate) {
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
    the ${response.AccessKeyLastUsed.ServiceName} service on
```

```
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetAccessKeyLastUsed。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
  }
);
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/iam-examples-managing-access-keys.html#iam-examples-managing-access-keys-last-used。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetAccessKeyLastUsed。

PowerShell

Tools for PowerShell V4

範例 1：傳回所提供存取金鑰的擁有使用者名稱和上次使用資訊。

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetAccessKeyLastUsed。

Tools for PowerShell V5

範例 1：傳回所提供存取金鑰的擁有使用者名稱和上次使用資訊。

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetAccessKeyLastUsed。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.
```

```
    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetAccessKeyLastUsed。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->getaccesskeylastused(
          iv_accesskeyid = iv_access_key_id ).
        MESSAGE 'Retrieved access key last used information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
```

```
        MESSAGE 'Access key does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetAccessKeyLastUsed。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetAccountAuthorizationDetails 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetAccountAuthorizationDetails。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理您的帳戶

CLI

AWS CLI

列出 AWS 帳戶的 IAM 使用者、群組、角色和政策

下列get-account-authorization-details命令會傳回 AWS 帳戶中所有 IAM 使用者、群組、角色和政策的相關資訊。

```
aws iam get-account-authorization-details
```

輸出：

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
```

```
                    "Principal": {
                        "Service": "ec2.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2014-07-30T17:09:20Z",
        "InstanceProfileList": [
            {
                "InstanceProfileId": "AIPA1234567890EXAMPLE",
                "Roles": [
                    {
                        "AssumeRolePolicyDocument": {
                            "Version":"2012-10-17",
                            "Statement": [
                                {
                                    "Sid": "",
                                    "Effect": "Allow",
                                    "Principal": {
                                        "Service": "ec2.amazonaws.com"
                                    },
                                    "Action": "sts:AssumeRole"
                                }
                            ]
                        },
                        "RoleId": "AROA1234567890EXAMPLE",
                        "CreateDate": "2014-07-30T17:09:20Z",
                        "RoleName": "EC2role",
                        "Path": "/",
                        "Arn": "arn:aws:iam::123456789012:role/EC2role"
                    }
                ],
                "CreateDate": "2014-07-30T17:09:20Z",
                "InstanceProfileName": "EC2role",
                "Path": "/",
                "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
            }
        ],
        "RoleName": "EC2role",
        "Path": "/",
        "AttachedManagedPolicies": [
            {
```

```
                    "PolicyName": "AmazonS3FullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
                },
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
```

```
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
                                "Effect": "Allow"
                            }
                        ]
                    }
                }
            ]
        }
    ],
    "UserDetailList": [
        {
            "UserName": "Alice",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:24Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Alice"
        },
        {
            "UserName": "Bob",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:25Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [
```

```
                {
                    "PolicyName": "DenyBillingAndIAMPolicy",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": {
                            "Effect": "Deny",
                            "Action": [
                                "aws-portal:*",
                                "iam:*"
                            ],
                            "Resource": "*"
                        }
                    }
                }
            ],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        },
        {
            "UserName": "Charlie",
            "GroupList": [
                "Dev"
            ],
            "CreateDate": "2013-10-14T18:33:56Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Charlie"
        }
    ],
    "Policies": [
        {
            "PolicyName": "create-update-delete-set-managed-policies",
            "CreateDate": "2015-02-06T19:58:34Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-02-06T19:58:34Z",
                    "VersionId": "v1",
```

```
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": {
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
```

```
                                    "s3:List*"
                                ],
                                "Resource": [
                                    "arn:aws:s3:::amzn-s3-demo-bucket",
                                    "arn:aws:s3:::amzn-s3-demo-bucket/*"
                                ]
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
            "UpdateDate": "2015-01-21T23:39:41Z"
        },
        {
            "PolicyName": "AmazonEC2FullAccess",
            "CreateDate": "2015-02-06T18:40:15Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2014-10-30T20:59:46Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "ec2:*",
                                "Effect": "Allow",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "elasticloadbalancing:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "cloudwatch:*",
```

```
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 AWS 安全性稽核指南。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetAccountAuthorizationDetails。

PowerShell

Tools for PowerShell V4

範例 1：此範例會取得 AWS 帳戶中身分的授權詳細資訊，並顯示傳回物件的元素清單，包括使用者、群組和角色。例如，`UserDetailList` 屬性會顯示使用者的詳細資訊。`RoleDetailList` 和 `GroupDetailList` 屬性提供類似的資訊。

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

輸出：

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
```

```
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

輸出：

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetAccountAuthorizationDetails。

Tools for PowerShell V5

範例 1：此範例會取得 AWS 帳戶中身分的授權詳細資訊，並顯示傳回物件的元素清單，包括使用者、群組和角色。例如，**UserDetailList** 屬性會顯示使用者的詳細資訊。**RoleDetailList** 和 **GroupDetailList** 屬性提供類似的資訊。

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

輸出：

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

輸出：

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetAccountAuthorizationDetails。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetAccountAuthorizationDetails。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->getaccountauthdetails( ).
        MESSAGE 'Retrieved account authorization details.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting account authorization details.'
TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetAccountAuthorizationDetails。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetAccountPasswordPolicy` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetAccountPasswordPolicy`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
```

```
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 GetAccountPasswordPolicy。

CLI

AWS CLI

查看目前的帳戶密碼政策

下列 `get-account-password-policy` 命令會顯示有關目前帳戶密碼政策的詳細資訊。

```
aws iam get-account-password-policy
```

輸出：

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

如果沒有為帳戶定義密碼政策，則該命令會傳回 `NoSuchEntity` 錯誤。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的設定 IAM 使用者的帳戶密碼政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetAccountPasswordPolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "log"

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

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
```

```
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 GetAccountPasswordPolicy。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

取得帳戶密碼政策。

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
  const command = new GetAccountPasswordPolicyCommand({});

  const response = await client.send(command);
  console.log(response.PasswordPolicy);
  return response;
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetAccountPasswordPolicy。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 GetAccountPasswordPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回目前帳戶之密碼政策的詳細資訊。如果沒有為帳戶定義密碼政策，命令會傳回 `NoSuchEntity` 錯誤。

```
Get-IAMAccountPasswordPolicy
```

輸出：

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
```

```
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetAccountPasswordPolicy。

Tools for PowerShell V5

範例 1：此範例會傳回目前帳戶之密碼政策的詳細資訊。如果沒有為帳戶定義密碼政策，命令會傳回 **NoSuchEntity** 錯誤。

```
Get-IAMAccountPasswordPolicy
```

輸出：

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetAccountPasswordPolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def print_password_policy():
```

```
    """
    Prints the password policy for the account.
    """
    try:
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
{pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
        print(f"\tpassword_reuse_prevention:
{pw_policy.password_reuse_prevention}")
        print(
            f"\trequire_lowercase_characters:
{pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
{pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetAccountPasswordPolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'IAMPolicyManager'
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    response = @iam_client.get_account_password_policy
    @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.info('The account does not have a password policy.')
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't print the account password policy. Error: #{e.code} -
 #{e.message}")
    raise
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 GetAccountPasswordPolicy。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 GetAccountPasswordPolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->getaccountpasswordpolicy( ).
        MESSAGE 'Retrieved account password policy.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'No password policy exists.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting password policy.' TYPE 'E'.
```

```
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetAccountPasswordPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetAccountSummary 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetAccountSummary。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理您的帳戶

CLI

AWS CLI

取得有關目前帳戶中 IAM 實體用量和 IAM 配額的資訊

下列 get-account-summary 命令會傳回有關帳戶中目前 IAM 實體用量和目前 IAM 實體配額的資訊。

```
aws iam get-account-summary
```

輸出：

```
{
    "SummaryMap": {
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
```

```
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

如需實體限制的詳細資訊，請參閱《IAM 使用者指南》中的 IAM 和 AWS STS 配額。 AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetAccountSummary。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回 AWS 帳戶中目前 IAM 實體用量和目前 IAM 實體配額的相關資訊。

```
Get-IAMAccountSummary
```

輸出：

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
Groups                                  3
UsersQuota                              5000
```

```
RolePolicySizeQuota                     10240
UserPolicySizeQuota                     2048
GroupsPerUserQuota                      10
AssumeRolePolicySizeQuota               2048
AttachedPoliciesPerGroupQuota           2
Roles                                   9
VersionsPerPolicyQuota                  5
GroupsQuota                             100
PolicySizeQuota                         5120
Policies                                5
RolesQuota                              250
ServerCertificates                      0
AttachedPoliciesPerRoleQuota            2
MFADevicesInUse                         2
PoliciesQuota                           1000
AccountMFAEnabled                       1
Providers                               2
InstanceProfilesQuota                   100
MFADevices                              4
AccessKeysPerUserQuota                  2
AttachedPoliciesPerUserQuota            2
SigningCertificatesPerUserQuota         2
PolicyVersionsInUse                     4
InstanceProfiles                        1
...
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetAccountSummary。

Tools for PowerShell V5

範例 1：此範例會傳回 AWS 帳戶中目前 IAM 實體用量和目前 IAM 實體配額的相關資訊。

```
Get-IAMAccountSummary
```

輸出：

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
```

```
AccountSigningCertificatesPresent          0
AccountAccessKeysPresent                   0
Groups                                     3
UsersQuota                                 5000
RolePolicySizeQuota                        10240
UserPolicySizeQuota                        2048
GroupsPerUserQuota                         10
AssumeRolePolicySizeQuota                  2048
AttachedPoliciesPerGroupQuota              2
Roles                                      9
VersionsPerPolicyQuota                     5
GroupsQuota                                100
PolicySizeQuota                            5120
Policies                                   5
RolesQuota                                 250
ServerCertificates                         0
AttachedPoliciesPerRoleQuota               2
MFADevicesInUse                            2
PoliciesQuota                              1000
AccountMFAEnabled                          1
Providers                                  2
InstanceProfilesQuota                      100
MFADevices                                 4
AccessKeysPerUserQuota                     2
AttachedPoliciesPerUserQuota               2
SigningCertificatesPerUserQuota            2
PolicyVersionsInUse                        4
InstanceProfiles                           1
...
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetAccountSummary。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetAccountSummary。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->getaccountsummary( ).
        MESSAGE 'Retrieved account summary.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting account summary.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetAccountSummary。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetContextKeysForCustomPolicy** 與 CLI

下列程式碼範例示範如何使用 GetContextKeysForCustomPolicy。

CLI

AWS CLI

範例 1：列出命令列上以參數形式提供的一個或多個自訂 JSON 政策引用的內容索引鍵

下列 get-context-keys-for-custom-policy 命令會剖析提供的每個政策，並列出這些政策使用的內容索引鍵。使用此命令來識別您必須提供的內容索引鍵值，以成功使用政策模擬器命令 simulate-custom-policy 和 simulate-custom-policy。也可以使用 get-context-keys-for-custom-policy 命令，擷取 IAM 使用者或角色關聯之所有政策所使用的內容索引鍵清單。以 file:// 開頭的參數會告訴命令讀取檔案的內容，並將其用作參數的值 (而不是檔案名稱本身)。

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

輸出：

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

範例 2：列出以檔案輸入形式提供的一個或多個自訂 JSON 政策引用的內容索引鍵

下列 get-context-keys-for-custom-policy 命令與先前的範例相同，但政策是在檔案中提供，而不是以參數形式提供。由於該命令需要的是 JSON 字串清單，而非 JSON 結構清單，因此檔案的結構必須如下所示，但您可以將其摺疊為一個。

```
[
    "Policy1",
    "Policy2"
]
```

例如，包含上一個範例中的政策的檔案必須如下所示。必須在政策字串前面加上反斜線 "，以逸出政策字串中的每個內嵌雙引號。

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow
\", \"Action\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-
west-2:128716708097:table/${aws:username}\", \"Condition\": {\"DateGreaterThan\":
 {\"aws:CurrentTime\": \"2015-08-16T12:00:00Z\"}}}}" ]
```

然後可將此檔案提交至下列命令。

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

輸出：

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM 政策模擬器 (AWS CLI 和 AWS API)。 AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetContextKeysForCustomPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取所提供政策 json 中的所有內容索引鍵。若要提供多個政策，可以採用以逗號分隔的值清單形式提供。

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetContextKeysForCustomPolicy。

Tools for PowerShell V5

範例 1：此範例會擷取所提供政策 json 中的所有內容索引鍵。若要提供多個政策，可以採用以逗號分隔的值清單形式提供。

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetContextKeysForCustomPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`GetContextKeysForPrincipalPolicy`** 與 CLI

下列程式碼範例示範如何使用 `GetContextKeysForPrincipalPolicy`。

CLI

AWS CLI

若要列出與 IAM 主體關聯之所有政策引用的內容索引鍵

下列 get-context-keys-for-principal-policy 命令會擷取連接至使用者 saanvi 及其所屬任何群組的所有政策。然後，它會剖析每個政策，並列出這些政策所使用的內容索引鍵。此命令可用於確定為成功使用 simulate-custom-policy 和 simulate-principal-policy 命令，您必須提供的內容索引鍵值。也可以使用 get-context-keys-for-custom-policy 命令來擷取任意 JSON 政策所使用的內容索引鍵清單。

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

輸出：

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM 政策模擬器 (AWS CLI 和 AWS API)。 AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetContextKeysForPrincipalPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取所提供政策 json 中的所有內容索引鍵，以及連接至 IAM 實體 (使用者/角色等) 的政策。對於 -PolicyInputList，您可以提供包含多個以逗號分隔的值的清單。

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

```
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetContextKeysForPrincipalPolicy。

Tools for PowerShell V5

範例 1：此範例會擷取所提供政策 json 中的所有內容索引鍵，以及連接至 IAM 實體 (使用者/角色等) 的政策。對於 -PolicyInputList，您可以提供包含多個以逗號分隔的值的清單。

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetContextKeysForPrincipalPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetCredentialReport` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetCredentialReport`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理您的帳戶

CLI

AWS CLI

取得憑證報告

此範例會開啟傳回的報告，並以文字行陣列的形式將其輸出至管道。

```
aws iam get-credential-report
```

輸出：

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的取得 AWS 帳戶的登入資料報告。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetCredentialReport。

PowerShell

Tools for PowerShell V4

範例 1：此範例會開啟傳回的報告，並以文字行陣列的形式將其輸出至管道。第一行是帶有以逗號分隔的資料欄名稱的標題。每個連續資料列是一個使用者的詳細資訊列，每個欄位以逗號分隔。必須先使用 **Request-IAMCredentialReport** cmdlet 產生報告，然後才能檢視報告。若要以單一字串擷取報告，請使用 **-Raw**，勿用 **-AsTextArray**。**-AsTextArray** 交換器也接受別名 **-SplitLines**。如需輸出中資料欄的完整清單，請參閱服務 API 參考。請注意，如果不使用 **-AsTextArray** 或 **-SplitLines**，必須使用 .NET **StreamReader** 類從 **.Content** 屬性擷取文字。

```
Request-IAMCredentialReport
```

輸出：

```
Description                                                    State
-----------                                                    -----
```

```
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

輸出：

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetCredentialReport。

Tools for PowerShell V5

範例 1：此範例會開啟傳回的報告，並以文字行陣列的形式將其輸出至管道。第一行是帶有以逗號分隔的資料欄名稱的標題。每個連續資料列是一個使用者的詳細資訊列，每個欄位以逗號分隔。必須先使用 **Request-IAMCredentialReport** cmdlet 產生報告，然後才能檢視報告。若要以單一字串擷取報告，請使用 **-Raw**，勿用 **-AsTextArray**。**-AsTextArray** 交換器也接受別名 **-SplitLines**。如需輸出中資料欄的完整清單，請參閱服務 API 參考。請注意，如果不使用 **-AsTextArray** 或 **-SplitLines**，必須使用 .NET **StreamReader** 類從 **.Content** 屬性擷取文字。

```
Request-IAMCredentialReport
```

輸出：

```
Description                                                          State
-----------                                                          -----
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

輸出：

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetCredentialReport。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetCredentialReport。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
TRY.
    oo_result = lo_iam->getcredentialreport( ).
    MESSAGE 'Retrieved credential report.' TYPE 'I'.
  CATCH /aws1/cx_iamcredrptnotpresen00.
    MESSAGE 'Credential report not present.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptexpiredex.
    MESSAGE 'Credential report expired.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptnotreadyex.
    MESSAGE 'Credential report not ready.' TYPE 'E'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when getting credential report.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetCredentialReport。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`GetGroup`** 與 CLI

下列程式碼範例示範如何使用 `GetGroup`。

CLI

AWS CLI

若要取得 IAM 群組

此範例會傳回 IAM 群組 Admins 的詳細資訊。

```
aws iam get-group \
    --group-name Admins
```

輸出：

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    },
    "Users": []
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 身分 (使用者、使用者群組和角色)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetGroup。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回 IAM 群組 **Testers** 的詳細資訊，包括屬於該群組的所有 IAM 使用者的集合。

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

輸出：

```
Group                                    IsTruncated           Marker
     Users
```

```
-----                                    -----------          ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

輸出：

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetGroup。

Tools for PowerShell V5

範例 1：此範例會傳回 IAM 群組 **Testers** 的詳細資訊，包括屬於該群組的所有 IAM 使用者的集合。

```
$results = Get-IAMGroup -GroupName "Testers"
```

```
$results
```

輸出：

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

輸出：

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetGroupPolicy** 與 CLI

下列程式碼範例示範如何使用 GetGroupPolicy。

CLI

AWS CLI

若要取得連接到 IAM 群組的政策的相關資訊

下列 get-group-policy 命令會取得連接到名為 Test-Group 的群組之指定政策的相關資訊。

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

輸出：

```
{
    "GroupName": "Test-Group",
    "PolicyDocument": {
        "Statement": [
            {
                "Action": [
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetGroupPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回群組 PowerUserAccess-Testers 名為 Testers 的嵌入式內嵌政策的詳細資訊。PolicyDocument 屬性經過 URL 編碼。在此範例中，它會使用 UrlDecode .NET 方法解碼。

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

輸出：

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetGroupPolicy。

Tools for PowerShell V5

範例 1：此範例會傳回群組 PowerUserAccess-Testers 名為 Testers 的嵌入式內嵌政策的詳細資訊。PolicyDocument 屬性經過 URL 編碼。在此範例中，它會使用 UrlDecode .NET 方法解碼。

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

輸出：

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetGroupPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetInstanceProfile** 與 CLI

下列程式碼範例示範如何使用 GetInstanceProfile。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 受管串流入門

CLI

AWS CLI

若要取得有關執行個體設定檔的資訊

下列 get-instance-profile 命令會取得名為 ExampleInstanceProfile 之執行個體設定檔的相關資訊。

```
aws iam get-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

輸出：

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/
ExampleInstanceProfile"
    }
```

```
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用執行個體設定檔。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetInstanceProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回目前 AWS 帳戶中定義的名為 **ec2instancerole** 的執行個體描述檔詳細資訊。

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

輸出：

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetInstanceProfile。

Tools for PowerShell V5

範例 1：此範例會傳回目前 AWS 帳戶中定義的名為 **ec2instancerole** 的執行個體描述檔詳細資訊。

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

輸出：

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
```

```
Roles                : {ec2instancerole}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetInstanceProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetLoginProfile** 與 CLI

下列程式碼範例示範如何使用 GetLoginProfile。

CLI

AWS CLI

若要取得 IAM 使用者的密碼資訊

下列 get-login-profile 命令會取得名為 Bob 之 IAM 使用者的密碼資訊。

```
aws iam get-login-profile \
    --user-name Bob
```

輸出：

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

get-login-profile 命令可用來驗證 IAM 使用者是否有密碼。如果沒有為使用者定義密碼，該命令會傳回錯誤：NoSuchEntity。

不能使用此命令檢視密碼。如果密碼遺失，可以重設使用者的密碼 (update-login-profile)。或者，可以刪除使用者的登入設定檔 (delete-login-profile)，然後建立新檔 (create-login-profile)。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的密碼。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetLoginProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回密碼建立日期，以及 IAM 使用者 David 是否需要重設密碼。

```
Get-IAMLoginProfile -UserName David
```

輸出：

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
12/10/2014 3:39:44 PM      False                              David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetLoginProfile。

Tools for PowerShell V5

範例 1：此範例會傳回密碼建立日期，以及 IAM 使用者 David 是否需要重設密碼。

```
Get-IAMLoginProfile -UserName David
```

輸出：

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
12/10/2014 3:39:44 PM      False                              David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetLoginProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetOpenIdConnectProvider** 與 CLI

下列程式碼範例示範如何使用 GetOpenIdConnectProvider。

CLI

AWS CLI

若要傳回指定 OpenID Connect 提供者的相關資訊

此範例會傳回 ARN 為 arn:aws:iam::123456789012:oidc-provider/server.example.com 的 OpenID Connect 提供者的詳細資訊。

```
aws iam get-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

輸出：

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 Creating OpenID Connect (OIDC) identity providers。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetOpenIdConnectProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回 ARN 為 **arn:aws:iam::123456789012:oidc-provider/accounts.google.com** 的 OpenID Connect 提供者的詳細資訊。**ClientIDList** 屬性是一個集合，包含為此提供者定義的所有用戶端 ID。

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

輸出：

```
ClientIDList         CreateDate                ThumbprintList
           Url
------------         ----------                --------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetOpenIdConnectProvider。

Tools for PowerShell V5

範例 1：此範例會傳回 ARN 為 **arn:aws:iam::123456789012:oidc-provider/accounts.google.com** 的 OpenID Connect 提供者的詳細資訊。**ClientIDList** 屬性是一個集合，包含為此提供者定義的所有用戶端 ID。

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

輸出：

```
ClientIDList         CreateDate                ThumbprintList
           Url
------------         ----------                --------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetOpenIdConnectProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetPolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 受管串流入門
- 使用 IAM 政策產生器 API

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 GetPolicy。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 GetPolicy。

CLI

AWS CLI

擷取指定受管政策的相關資訊

此範例會傳回 ARN 為 `arn:aws:iam::123456789012:policy/MySamplePolicy` 之受管政策的詳細資訊。

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

輸出：

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetPolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
```

```
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 GetPolicy。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

取得政策。

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetPolicy。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
};

iam.getPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
```

```
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetPolicy。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun getIAMPolicy(policyArnVal: String?) {
    val request =
        GetPolicyRequest {
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 GetPolicy。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
    {
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 GetPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回 ARN 為 **arn:aws:iam::123456789012:policy/MySamplePolicy** 之受管政策的詳細資訊。

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

輸出：

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
```

```
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetPolicy。

Tools for PowerShell V5

範例 1：此範例會傳回 ARN 為 **arn:aws:iam::123456789012:policy/MySamplePolicy** 之受管政策的詳細資訊。

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

輸出：

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetPolicy。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def get_default_policy_statement(policy_arn):
```

```
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
policy_arn)
        raise
    else:
        return policy_statement
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetPolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Fetches an IAM policy by its ARN
# @param policy_arn [String] the ARN of the IAM policy to retrieve
# @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
def get_policy(policy_arn)
  response = @iam_client.get_policy(policy_arn: policy_arn)
  policy = response.policy
```

```
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 GetPolicy。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->getpolicy( iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetPolicy。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
        let input = GetPolicyInput(
            policyArn: arn
        )
        do {
            let output = try await client.getPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: getPolicy:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 GetPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetPolicyVersion` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetPolicyVersion`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理政策
- 使用 IAM 政策產生器 API

CLI

AWS CLI

擷取指定受管政策指定版本的相關資訊

此範例會傳回 ARN 為 arn:aws:iam::123456789012:policy/MyManagedPolicy 之 v2 版本政策的政策文件。

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

輸出：

```
{
    "PolicyVersion": {
        "Document": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetPolicyVersion。

PowerShell

Tools for PowerShell V4

此範例會傳回 ARN 為 **arn:aws:iam::123456789012:policy/MyManagedPolicy** 之 **v2** 版本政策的政策文件。**Document** 屬性中的政策文件經過 URL 編碼，在此範例中使用 **UrlDecode** .NET 方法解碼。

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

輸出：

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetPolicyVersion。

Tools for PowerShell V5

此範例會傳回 ARN 為 **arn:aws:iam::123456789012:policy/MyManagedPolicy** 之 **v2** 版本政策的政策文件。**Document** 屬性中的政策文件經過 URL 編碼，在此範例中使用 **UrlDecode** .NET 方法解碼。

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

輸出：

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetPolicyVersion。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetPolicyVersion。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
TRY.
    oo_result = lo_iam->getpolicyversion(
      iv_policyarn = iv_policy_arn
      iv_versionid = iv_version_id ).
    MESSAGE 'Retrieved policy version information.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy or version does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iaminvalidinputex.
    MESSAGE 'Invalid input provided.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetPolicyVersion。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetRole` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetRole`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 設定容器服務連線
- 為無伺服器啟動類型建立容器任務
- 為虛擬機器執行個體建立容器服務
- 開始使用無伺服器資料倉儲

- 開始使用物聯網裝置保護
- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門
- 受管串流入門
- 佈建資料倉儲叢集入門
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
/// <summary>
/// Get information about an IAM role.
/// </summary>
/// <param name="roleName">The name of the IAM role to retrieve information
/// for.</param>
/// <returns>The IAM role that was retrieved.</returns>
public async Task<Role> GetRoleAsync(string roleName)
{
    var response = await _IAMService.GetRoleAsync(new GetRoleRequest
    {
        RoleName = roleName,
    });

    return response.Role;
}
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 GetRole。

CLI

AWS CLI

取得 IAM 角色的相關資訊

下列 get-role 命令會取得名為 Test-Role 之角色的相關資訊。

```
aws iam get-role \
    --role-name Test-Role
```

輸出：

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

該命令會顯示連接至角色的信任政策。若要列出連接至角色的許可政策，請使用 list-role-policies 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetRole。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
```

```
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 GetRole。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

取得角色。

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });

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

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetRole。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 GetRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回 **lamda_exec_role** 的詳細資訊。其中包含指定誰可以擔任此角色的信任政策文件。政策文件經過 URL 編碼，可以使用 .NET **UrlDecode** 方法解碼。在此範例中，原始政策在上傳至政策之前已移除所有空格。若要查看決定擔任該角色之人員可以執行的動作的許可政策文件，請將 **Get-IAMRolePolicy** 用於內嵌政策，並將 **Get-IAMPolicyVersion** 用於連接的受管政策。

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

輸出：

```
Arn                   : arn:aws:iam::123456789012:role/lambda_exec_role
```

```
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate               : 4/2/2015 9:16:11 AM
Path                     : /
RoleId                   : 2YBIKAIBHNKB4EXAMPLE1
RoleName                 : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

輸出：

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetRole。

Tools for PowerShell V5

範例 1：此範例會傳回 **lamda_exec_role** 的詳細資訊。其中包含指定誰可以擔任此角色的信任政策文件。政策文件經過 URL 編碼，可以使用 .NET **UrlDecode** 方法解碼。在此範例中，原始政策在上傳至政策之前已移除所有空格。若要查看決定擔任該角色之人員可以執行的動作的許可政策文件，請將 **Get-IAMRolePolicy** 用於內嵌政策，並將 **Get-IAMPolicyVersion** 用於連接的受管政策。

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

輸出：

```
Arn                      : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
```

```
                            %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                            %22%7D%5D%7D
CreateDate                : 4/2/2015 9:16:11 AM
Path                      : /
RoleId                    : 2YBIKAIBHNKB4EXAMPLE1
RoleName                  : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

輸出：

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetRole。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def get_role(role_name):
    """
    Gets a role by name.

    :param role_name: The name of the role to retrieve.
    :return: The specified role.
    """
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
```

```
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetRole。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Gets data about a role.
#
# @param name [String] The name of the role to look up.
# @return [Aws::IAM::Role] The retrieved role.
def get_role(name)
  role = @iam_client.get_role({
                                role_name: name
                              }).role
  puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
rescue Aws::Errors::ServiceError => e
  puts("Couldn't get data for role '#{name}' Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
else
  role
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 GetRole。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 GetRole。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->getrole( iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved role information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 GetRole。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func getRole(name: String) async throws -> IAMClientTypes.Role {
        let input = GetRoleInput(
            roleName: name
        )
        do {
            let output = try await client.getRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            print("ERROR: getRole:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 GetRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetRolePolicy** 與 CLI

下列程式碼範例示範如何使用 GetRolePolicy。

CLI

AWS CLI

若要取得連接到 IAM 角色的政策的相關資訊

下列 `get-role-policy` 命令會取得連接到名為 `Test-Role` 的角色之指定政策的相關資訊。

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

輸出：

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
      "Statement": [
          {
              "Action": [
                  "s3:ListBucket",
                  "s3:Put*",
                  "s3:Get*",
                  "s3:*MultipartUpload*"
              ],
              "Resource": "*",
              "Effect": "Allow",
              "Sid": "1"
          }
      ]
  }
  "PolicyName": "ExamplePolicy"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetRolePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回內嵌在 IAM 角色 `lamda_exec_role` 中的名為 `oneClick_lambda_exec_role_policy` 之政策的許可政策文件。產生的政策文件經過 URL 編碼。在此範例中，它會使用 `UrlDecode` .NET 方法解碼。

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

輸出：

```
PolicyDocument                                                PolicyName
              UserName
--------------                                                ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

輸出：

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
```

```
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetRolePolicy。

Tools for PowerShell V5

範例 1：此範例會傳回內嵌在 IAM 角色 **lamda_exec_role** 中的名為 **oneClick_lambda_exec_role_policy** 之政策的許可政策文件。產生的政策文件經過 URL 編碼。在此範例中，它會使用 **UrlDecode** .NET 方法解碼。

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

輸出：

```
PolicyDocument                                              PolicyName
              UserName
--------------                                              ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

輸出：

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

```
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetRolePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetSamlProvider** 與 CLI

下列程式碼範例示範如何使用 GetSamlProvider。

CLI

AWS CLI

若要擷取 SAML 提供者中繼資料文件

此範例會擷取 ARM 為 arn:aws:iam::123456789012:saml-provider/SAMLADFS 的 SAML 2.0 提供者的詳細資訊。回應包含您從身分提供者取得以建立 AWS SAML 提供者實體的中繼資料文件，以及建立和過期日期。

```
aws iam get-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

輸出：

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM SAML 身分提供者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetSamlProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取 ARM 為 arn:aws:iam::123456789012:saml-provider/SAMLADFS 的 SAML 2.0 提供者的詳細資訊。回應包含您從身分提供者取得以建立 AWS SAML 提供者實體的中繼資料文件，以及建立和過期日期。

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

輸出：

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetSamlProvider。

Tools for PowerShell V5

範例 1：此範例會擷取 ARM 為 arn:aws:iam::123456789012:saml-provider/SAMLADFS 的 SAML 2.0 提供者的詳細資訊。回應包含您從身分提供者取得以建立 AWS SAML 提供者實體的中繼資料文件，以及建立和過期日期。

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

輸出：

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetSamlProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetServerCertificate 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetServerCertificate`。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
                  std::endl;
    }

    return result;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 GetServerCertificate。

CLI

AWS CLI

取得您 AWS 帳戶中伺服器憑證的詳細資訊

下列get-server-certificate命令會擷取您 AWS 帳戶中指定伺服器憑證的所有詳細資訊。

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

輸出：

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
```

```
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```

若要列出您 AWS 帳戶中可用的伺服器憑證，請使用 `list-server-certificates`命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 IAM 中管理伺服器憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetServerCertificate。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

獲取伺服器憑證。

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
```

```
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v3/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-getting。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetServerCertificate。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
```

```
);
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-getting。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 GetServerCertificate。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取名為 MyServerCertificate 之伺服器憑證的詳細資訊。可以在 CertificateBody 和 ServerCertificateMetadata 屬性中找到憑證詳細資訊。

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

輸出：

```
CertificateBody           : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                              21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
```

```
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

輸出：

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetServerCertificate。

Tools for PowerShell V5

範例 1：此範例會擷取名為 **MyServerCertificate** 之伺服器憑證的詳細資訊。可以在 **CertificateBody** 和 **ServerCertificateMetadata** 屬性中找到憑證詳細資訊。

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

輸出：

```
CertificateBody           : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
```

```
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                            21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

輸出：

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetServerCertificate。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetServiceLastAccessedDetails** 與 CLI

下列程式碼範例示範如何使用 GetServiceLastAccessedDetails。

CLI

AWS CLI

若要擷取服務存取報告

下列 get-service-last-accessed-details 範例將擷取先前產生的報告，其中有列出 IAM 實體存取的服務。若要產生報告，請使用 generate-service-last-accessed-details 命令。

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

輸出：

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的AWS 使用上次存取資訊在 中精簡許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetServiceLastAccessedDetails。

PowerShell

Tools for PowerShell V4

範例 1：此範例提供請求呼叫中關聯之 IAM 實體 (使用者、群組、角色或政策) 上次存取的服務詳細資訊。

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

輸出：

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetServiceLastAccessedDetails。

Tools for PowerShell V5

範例 1：此範例提供請求呼叫中關聯之 IAM 實體 (使用者、群組、角色或政策) 上次存取的服務詳細資訊。

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

輸出：

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetServiceLastAccessedDetails。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetServiceLastAccessedDetailsWithEntities** 與 CLI

下列程式碼範例示範如何使用 GetServiceLastAccessedDetailsWithEntities。

CLI

AWS CLI

若要擷取包含服務詳細資訊的服務存取報告

下列 get-service-last-accessed-details-with-entities 範例會擷取包含 IAM 使用者和其他存取指定服務之實體詳細資訊的報告。若要產生報告，請使用 generate-service-last-accessed-details 命令。若要取得使用命名空間存取之服務的清單，請使用 get-service-last-accessed-details。

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

輸出：

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
```

```
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的AWS 使用上次存取資訊在 中精簡許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetServiceLastAccessedDetailsWithEntities。

PowerShell

Tools for PowerShell V4

範例 1：此範例提供在請求中，該個別 IAM 實體上次存取服務的時間戳記。

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

輸出：

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

輸出：

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

輸出：

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetServiceLastAccessedDetailsWithEntities。

Tools for PowerShell V5

範例 1：此範例提供在請求中，該個別 IAM 實體上次存取服務的時間戳記。

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

輸出：

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

輸出：

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

輸出：

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetServiceLastAccessedDetailsWithEntities。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetServiceLinkedRoleDeletionStatus 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetServiceLinkedRoleDeletionStatus。

CLI

AWS CLI

檢查刪除服務連結角色的請求狀態

下列 get-service-linked-role-deletion-status 範例會顯示刪除服務連結角色之先前請求的狀態。刪除操作會以非同步方式發生。提出請求時，就會取得您提供作為此命令參數的 DeletionTaskId 值。

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

輸出：

```
{
"Status": "SUCCEEDED"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用服務連結角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetServiceLinkedRoleDeletionStatus。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

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

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 GetServiceLinkedRoleDeletionStatus。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetUser` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetUser`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 GetUser。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
```

```
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
########################################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetUser。

CLI

AWS CLI

取得關於 IAM 使用者的資訊

下列 `get-user` 命令會取得名為 `Paulo` 之 IAM 使用者的相關資訊。

```
aws iam get-user \
    --user-name Paulo
```

輸出：

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetUser。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
```

```
 "context"
 "encoding/json"
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}
```

```
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 GetUser。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取名為 David 之使用者的詳細資訊。

```
Get-IAMUser -UserName David
```

輸出：

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

範例 2：此範例會擷取目前登入之 IAM 使用者的詳細資訊。

```
Get-IAMUser
```

輸出：

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetUser。

Tools for PowerShell V5

範例 1：此範例會擷取名為 David 之使用者的詳細資訊。

```
Get-IAMUser -UserName David
```

輸出：

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

範例 2：此範例會擷取目前登入之 IAM 使用者的詳細資訊。

```
Get-IAMUser
```

輸出：

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetUser。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Retrieves a user's details
#
# @param user_name [String] The name of the user to retrieve
```

```
  # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
  def get_user(user_name)
    response = @iam_client.get_user(user_name: user_name)
    response.user
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("User '#{user_name}' not found.")
    nil
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error retrieving user '#{user_name}': #{e.message}")
    nil
  end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 GetUser。

Swift

SDK for Swift

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func getUser(name: String? = nil) async throws -> IAMClientTypes.User
 {
        let input = GetUserInput(
            userName: name
        )
        do {
            let output = try await iamClient.getUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            return user
        } catch {
```

```
            print("ERROR: getUser:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 GetUser。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetUserPolicy** 與 CLI

下列程式碼範例示範如何使用 GetUserPolicy。

CLI

AWS CLI

若要列出 IAM 使用者的政策詳細資訊

下列 get-user-policy 命令會列出連接至名為 Bob 之 IAM 使用者的指定政策的詳細資訊。

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

輸出：

```
{
    "UserName": "Bob",
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    }
```

```
}
```

若要取得 IAM 使用者的政策清單，請使用 `list-user-policies` 命令。

如需詳細資訊，請參閱「IAM 使用者指南」AWS 中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetUserPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取內嵌在名為 **David** 之 IAM 使用者中的內嵌政策 **Davids_IAM_Admin_Policy** 的詳細資訊。政策文件經過 URL 編碼。

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

輸出：

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
    }
```

```
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetUserPolicy。

Tools for PowerShell V5

範例 1：此範例會擷取內嵌在名為 **David** 之 IAM 使用者中的內嵌政策 **Davids_IAM_Admin_Policy** 的詳細資訊。政策文件經過 URL 編碼。

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

輸出：

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetUserPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListAccessKeys` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListAccessKeys`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理存取金鑰

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
# Returns:
#       access_key_ids
#     And:
```

```
#       0 - If successful.
#       1 - If it fails.
################################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
 the specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }

  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
    --output text \
    --query 'AccessKeyMetadata[].AccessKeyId')
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAccessKeys。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
                      << ": " << outcome.GetError().GetMessage() << std::endl;
            return false;
        }
```

```
        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
                      std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                      statusString << std::setw(20) <<
                      key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 ListAccessKeys。

CLI

AWS CLI

列出 IAM 使用者的存取金鑰 ID

下列 `list-access-keys` 命令會列出名為 Bob 之 IAM 使用者的存取金鑰 ID。

```
aws iam list-access-keys \
    --user-name Bob
```

輸出：

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

您無法列出 IAM 使用者的私密存取金鑰。如果私密存取金鑰遺失，您必須使用 `create-access-keys` 命令建立新的存取金鑰。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的存取金鑰。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAccessKeys。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListAccessKeys。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

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

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user for which access keys are
 retrieved.\s
                """;

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

```
        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void listKeys(IamClient iam, String userName) {
        try {
            boolean done = false;
            String newMarker = null;

            while (!done) {
                ListAccessKeysResponse response;

                if (newMarker == null) {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .build();

                    response = iam.listAccessKeys(request);

                } else {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .marker(newMarker)
                            .build();

                    response = iam.listAccessKeys(request);
                }

                for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                    System.out.format("Retrieved access key %s",
metadata.accessKeyId());
                }

                if (!response.isTruncated()) {
```

```
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 ListAccessKeys。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出存取金鑰。

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
```

```
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);

  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListAccessKeys。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
```

```
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListAccessKeys。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun listKeys(userNameVal: String?) {
    val request =
        ListAccessKeysRequest {
            userName = userNameVal
        }
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
```

```
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 ListAccessKeys。

PowerShell

Tools for PowerShell V4

範例 1：此命令會列出名為 **Bob** 之 IAM 使用者的存取金鑰。請注意，IAM 使用者的私密存取金鑰無法列出。如果私密存取金鑰遺失，必須使用 **New-IAMAccessKey** cmdlet 建立新的存取金鑰。

```
Get-IAMAccessKey -UserName "Bob"
```

輸出：

```
AccessKeyId                CreateDate                  Status
 UserName
-----------                ----------                  ------
 --------
AKIAIOSFODNN7EXAMPLE       12/3/2014 10:53:41 AM       Active              Bob
AKIAI44QH8DHBEXAMPLE       6/6/2013 8:42:26 PM         Inactive            Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListAccessKeys。

Tools for PowerShell V5

範例 1：此命令會列出名為 **Bob** 之 IAM 使用者的存取金鑰。請注意，IAM 使用者的私密存取金鑰無法列出。如果私密存取金鑰遺失，必須使用 **New-IAMAccessKey** cmdlet 建立新的存取金鑰。

```
Get-IAMAccessKey -UserName "Bob"
```

輸出：

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListAccessKeys。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListAccessKeys。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、停用和刪除存取金鑰。

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
```

```
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
```

```
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListAccessKeys。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listaccesskeys(
          iv_username = iv_user_name ).
        MESSAGE 'Retrieved access key list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListAccessKeys。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListAccountAliases` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListAccountAliases`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理您的帳戶

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }
```

```
        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 ListAccountAliases。

CLI

AWS CLI

列出帳戶別名

下列 `list-account-aliases` 命令會列出目前帳戶的別名。

```
aws iam list-account-aliases
```

輸出：

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的AWS 您的帳戶 ID 及其別名。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAccountAliases。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

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

        listAliases(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }
```

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 ListAccountAliases。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出帳戶別名。

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
```

```
      yield alias;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
}
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListAccountAliases。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
```

```
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListAccountAliases。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun listAliases() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 ListAccountAliases。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回 AWS 帳戶的帳戶別名。

```
Get-IAMAccountAlias
```

輸出：

```
ExampleCo
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListAccountAliases。

Tools for PowerShell V5

範例 1：此命令會傳回 AWS 帳戶的帳戶別名。

```
Get-IAMAccountAlias
```

輸出：

```
ExampleCo
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListAccountAliases。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
```

```
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListAccountAliases。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出、建立和刪除帳戶別名。

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
```

```
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListAccountAliases。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
TRY.
    oo_result = lo_iam->listaccountaliases( ).
    MESSAGE 'Retrieved account alias list.' TYPE 'I'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when listing account aliases.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListAccountAliases。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`ListAttachedGroupPolicies`** 與 CLI

下列程式碼範例示範如何使用 `ListAttachedGroupPolicies`。

CLI

AWS CLI

若要列出連接至指定群組所有受管政策

此範例會傳回連接到`Admins` AWS 帳戶中名為 之 IAM 群組的受管政策的名稱和 ARNs。

```
aws iam list-attached-group-policies \
    --group-name Admins
```

輸出：

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAttachedGroupPolicies。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回連接到**Admins** AWS 帳戶中名為 之 IAM 群組的受管政策的名稱和 ARNs。若要查看內嵌在群組中的內嵌政策的清單，請使用 **Get-IAMGroupPolicyList** 命令。

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

輸出：

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListAttachedGroupPolicies。

Tools for PowerShell V5

範例 1：此命令會傳回連接到**Admins** AWS 帳戶中名為 之 IAM 群組的受管政策的名稱和 ARNs。若要查看內嵌在群組中的內嵌政策的清單，請使用 **Get-IAMGroupPolicyList** 命令。

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

輸出：

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListAttachedGroupPolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListAttachedRolePolicies 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListAttachedRolePolicies。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
```

```
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListAttachedRolePolicies。

CLI

AWS CLI

列出連接至指定角色的所有受管政策

此命令會傳回連接到SecurityAuditRole AWS 帳戶中名為 之 IAM 角色的受管政策的名稱和 ARNs。

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

輸出：

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
```

```
    "IsTruncated": false
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAttachedRolePolicies。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
```

```
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListAttachedRolePolicies。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出連接至角色的政策。

```
import {
  ListAttachedRolePoliciesCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
```

```
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });

  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListAttachedRolePolicies。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 ListAttachedRolePolicies。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回連接至 AWS 帳戶中名為 **SecurityAuditRole** 之 IAM 角色的受管政策的名稱和 ARN。若要查看內嵌在角色中的內嵌政策清單，請使用 **Get-IAMRolePolicyList** 命令。

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

輸出：

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListAttachedRolePolicies。

Tools for PowerShell V5

範例 1：此命令會傳回連接至 AWS 帳戶中名為 **SecurityAuditRole** 之 IAM 角色的受管政策的名稱和 ARN。若要查看內嵌在角色中的內嵌政策清單，請使用 **Get-IAMRolePolicyList** 命令。

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

輸出：

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListAttachedRolePolicies。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
            logger.info("Got policy %s.", policy.arn)
    except ClientError:
```

```
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListAttachedRolePolicies。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、連接和分離角色政策。

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
```

```
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
```

```
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 ListAttachedRolePolicies。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn list_attached_role_policies(
```

```
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 ListAttachedRolePolicies。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listattachedrolepolicies(
          iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved attached policy list for role.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListAttachedRolePolicies。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3



    /// Returns a list of AWS Identity and Access Management (IAM) policies
    /// that are attached to the role.
    ///
    /// - Parameter role: The IAM role to return the policy list for.
    ///
    /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
    ///   describing each managed policy that's attached to the role.
    public func listAttachedRolePolicies(role: String) async throws ->
 [IAMClientTypes.AttachedPolicy] {
        var policyList: [IAMClientTypes.AttachedPolicy] = []

        // Use "Paginated" to get all the attached role polices.
        // This lets the SDK handle the 'isTruncated' in
 "ListAttachedRolePoliciesOutput".
        let input = ListAttachedRolePoliciesInput(
            roleName: role
        )
        let output = client.listAttachedRolePoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let attachedPolicies = page.attachedPolicies else {
                    print("Error: no attached policies returned.")
```

```
                continue
            }
            for attachedPolicy in attachedPolicies {
                policyList.append(attachedPolicy)
            }
        }
    } catch {
        print("ERROR: listAttachedRolePolicies:", dump(error))
        throw error
    }

    return policyList
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 ListAttachedRolePolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListAttachedUserPolicies** 與 CLI

下列程式碼範例示範如何使用 ListAttachedUserPolicies。

CLI

AWS CLI

若要列出連接至指定使用者的所有受管政策

此命令會傳回 AWS 帳戶中名為 之 IAM 使用者Bob之受管政策的名稱和 ARNs。

```
aws iam list-attached-user-policies \
    --user-name Bob
```

輸出：

```
{
    "AttachedPolicies": [
        {
```

```
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAttachedUserPolicies。

PowerShell

Tools for PowerShell V4

範例 1：此命令會傳回**Bob** AWS 帳戶中名為 的 IAM 使用者之受管政策的名稱和 ARNs。若要查看內嵌在 IAM 使用者中的內嵌政策的清單，請使用 **Get-IAMUserPolicyList** 命令。

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

輸出：

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/TesterPolicy                      TesterPolicy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListAttachedUserPolicies。

Tools for PowerShell V5

範例 1：此命令會傳回**Bob** AWS 帳戶中名為 的 IAM 使用者之受管政策的名稱和 ARNs。若要查看內嵌在 IAM 使用者中的內嵌政策的清單，請使用 **Get-IAMUserPolicyList** 命令。

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

輸出：

```
PolicyArn                                                  PolicyName
---------                                                  ----------
arn:aws:iam::aws:policy/TesterPolicy                       TesterPolicy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListAttachedUserPolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListEntitiesForPolicy** 與 CLI

下列程式碼範例示範如何使用 ListEntitiesForPolicy。

CLI

AWS CLI

若要列出指定受管政策所連接的所有使用者、群組和角色

此範例會傳回已連接政策 arn:aws:iam::123456789012:policy/TestPolicy 的 IAM 群組、角色和使用者的清單。

```
aws iam list-entities-for-policy \
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

輸出：

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
            "UserName": "Alice",
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
```

```
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
    "IsTruncated": false
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListEntitiesForPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回已連接政策 **arn:aws:iam::123456789012:policy/TestPolicy** 的 IAM 群組、角色和使用者的清單。

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

輸出：

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListEntitiesForPolicy。

Tools for PowerShell V5

範例 1：此範例會傳回已連接政策 **arn:aws:iam::123456789012:policy/TestPolicy** 的 IAM 群組、角色和使用者的清單。

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

輸出：

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListEntitiesForPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListGroupPolicies** 與 CLI

下列程式碼範例示範如何使用 ListGroupPolicies。

CLI

AWS CLI

若要列出連接至指定群組的所有內嵌政策

下列 list-group-policies 命令會列出連接至目前帳戶中名為 Admins 之 IAM 群組的內嵌政策名稱。

```
aws iam list-group-policies \
    --group-name Admins
```

輸出：

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListGroupPolicies。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回內嵌在群組 **Testers** 中的內嵌政策清單。若要取得連接到群組的受管政策，請使用 **Get-IAMAttachedGroupPolicyList** 命令。

```
Get-IAMGroupPolicyList -GroupName Testers
```

輸出：

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListGroupPolicies。

Tools for PowerShell V5

範例 1：此範例會傳回內嵌在群組 **Testers** 中的內嵌政策清單。若要取得連接到群組的受管政策，請使用 **Get-IAMAttachedGroupPolicyList** 命令。

```
Get-IAMGroupPolicyList -GroupName Testers
```

輸出：

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListGroupPolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListGroups 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListGroups。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListGroups。

CLI

AWS CLI

列出目前帳戶的 IAM 群組

下列 `list-groups` 命令會列出目前帳戶中的 IAM 群組。

```
aws iam list-groups
```

輸出：

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者群組。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListGroups。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "log"

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

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

// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(ctx context.Context, maxGroups int32)
 ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(ctx, &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
 return groups, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListGroups。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出群組。

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListGroups。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listGroupsArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listGroups($listGroupsArguments);
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 ListGroups。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回目前 中定義之所有 IAM 群組的集合 AWS 帳戶。

```
Get-IAMGroupList
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListGroups。

Tools for PowerShell V5

範例 1：此範例會傳回目前 中定義之所有 IAM 群組的集合 AWS 帳戶。

```
Get-IAMGroupList
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
```

```
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListGroups。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_groups(count):
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListGroups。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListGroups。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 ListGroups。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listgroups( ).
        MESSAGE 'Retrieved group list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing groups.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListGroups。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func listGroups() async throws -> [String] {
        var groupList: [String] = []

        // Use "Paginated" to get all the groups.
        // This lets the SDK handle the 'isTruncated' property in
 "ListGroupsOutput".
        let input = ListGroupsInput()

        let pages = client.listGroupsPaginated(input: input)
        do {
            for try await page in pages {
                guard let groups = page.groups else {
                    print("Error: no groups returned.")
                    continue
                }

                for group in groups {
```

```
                    if let name = group.groupName {
                        groupList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listGroups:", dump(error))
            throw error
        }
        return groupList
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 ListGroups。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListGroupsForUser** 與 CLI

下列程式碼範例示範如何使用 ListGroupsForUser。

CLI

AWS CLI

若要列出 IAM 使用者所屬的群組

下列 list-groups-for-user 命令顯示名為 Bob 之 IAM 使用者所屬的群組。

```
aws iam list-groups-for-user \
    --user-name Bob
```

輸出：

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
```

```
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者群組。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListGroupsForUser。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回 IAM 使用者 **David** 所屬之 IAM 群組的清單。

```
Get-IAMGroupForUser -UserName David
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
```

```
GroupName  : Developers
Path       : /
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListGroupsForUser。

Tools for PowerShell V5

範例 1：此範例會傳回 IAM 使用者 **David** 所屬之 IAM 群組的清單。

```
Get-IAMGroupForUser -UserName David
```

輸出：

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListGroupsForUser。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListInstanceProfiles** 與 CLI

下列程式碼範例示範如何使用 ListInstanceProfiles。

CLI

AWS CLI

若要列出帳戶的執行個體設定檔

下列 `list-instance-profiles` 命令列出與目前帳戶關聯的執行個體設定檔。

```
aws iam list-instance-profiles
```

輸出：

```
{
    "InstanceProfiles": [
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        },
        {
```

```
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用執行個體設定檔。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListInstanceProfiles。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回目前 中定義的執行個體描述檔集合 AWS 帳戶。

```
Get-IAMInstanceProfileList
```

輸出：

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListInstanceProfiles。

Tools for PowerShell V5

範例 1：此範例會傳回目前 中定義的執行個體描述檔集合 AWS 帳戶。

```
Get-IAMInstanceProfileList
```

輸出：

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListInstanceProfiles。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`ListInstanceProfilesForRole`** 與 CLI

下列程式碼範例示範如何使用 `ListInstanceProfilesForRole`。

CLI

AWS CLI

若要列出 IAM 角色的執行個體設定檔

下列 list-instance-profiles-for-role 命令會列出與角色 Test-Role 關聯的執行個體設定檔。

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

輸出：

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用執行個體設定檔。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListInstanceProfilesForRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回與角色 **ec2instancerole** 關聯之執行個體設定檔的詳細資訊。

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

輸出：

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListInstanceProfilesForRole。

Tools for PowerShell V5

範例 1：此範例會傳回與角色 **ec2instancerole** 關聯之執行個體設定檔的詳細資訊。

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

輸出：

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListInstanceProfilesForRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListMfaDevices** 與 CLI

下列程式碼範例示範如何使用 ListMfaDevices。

CLI

AWS CLI

若要列出指定使用者的所有 MFA 裝置

此範例會傳回指派給 IAM 使用者 Bob 之 MFA 裝置的詳細資訊。

```
aws iam list-mfa-devices \
    --user-name Bob
```

輸出：

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 AWS中使用多重要素驗證 (MFA)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListMfaDevices。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回指派給 IAM 使用者 **David** 之 MFA 裝置的詳細資訊。在此範例中，您可以確定這是虛擬裝置，因為 **SerialNumber** 是 ARN，而不是實體裝置的實際序號。

```
Get-IAMMFADevice -UserName David
```

輸出：

```
EnableDate                SerialNumber                            UserName
----------                ------------                            --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David     David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListMfaDevices。

Tools for PowerShell V5

範例 1：此範例會傳回指派給 IAM 使用者 **David** 之 MFA 裝置的詳細資訊。在此範例中，您可以確定這是虛擬裝置，因為 **SerialNumber** 是 ARN，而不是實體裝置的實際序號。

```
Get-IAMMFADevice -UserName David
```

輸出：

```
EnableDate                SerialNumber                            UserName
----------                ------------                            --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David     David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListMfaDevices。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListOpenIdConnectProviders** 與 CLI

下列程式碼範例示範如何使用 `ListOpenIdConnectProviders`。

CLI

AWS CLI

列出 AWS 帳戶中 OpenID Connect 供應商的相關資訊

此範例會傳回目前 AWS 帳戶中定義之所有 OpenID Connect 供應商的 ARNS 清單。

```
aws iam list-open-id-connect-providers
```

輸出：

```
{
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 Creating OpenID Connect (OIDC) identity providers。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListOpenIdConnectProviders。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回目前的 AWS 帳戶中定義之所有 OpenID Connect 提供者的 ARNS 清單。

```
Get-IAMOpenIDConnectProviderList
```

輸出：

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListOpenIdConnectProviders。

Tools for PowerShell V5

範例 1：此範例會傳回目前的 AWS 帳戶中定義之所有 OpenID Connect 提供者的 ARNS 清單。

```
Get-IAMOpenIDConnectProviderList
```

輸出：

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListOpenIdConnectProviders。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListPolicies` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListPolicies`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理政策

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListPolicies。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
```

```
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(64) << "Description" << std::setw(12) <<
                      "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 ListPolicies。

CLI

AWS CLI

列出可供您 AWS 帳戶使用的受管政策

此範例會傳回目前 AWS 帳戶中可用的前兩個受管政策的集合。

```
aws iam list-policies \
    --max-items 3
```

輸出：

```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
```

```
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListPolicies。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
```

```
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListPolicies。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出政策。

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
```

```
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
    Scope: "Local",
  });

  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListPolicies。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 ListPolicies。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回目前 AWS 帳戶中可用的前三個受管政策的集合。由於 **-scope** 未指定，因此預設為 **all**，並同時包含 AWS 受管和客戶受管政策。

```
Get-IAMPolicyList -MaxItem 3
```

輸出：

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

範例 2：此範例會傳回目前 AWS 帳戶中可用的前兩個客戶受管政策的集合。它使用 `-Scope local` 將輸出限制為僅限客戶管理政策。

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

輸出：

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListPolicies。

Tools for PowerShell V5

範例 1：此範例會傳回目前 AWS 帳戶中可用的前三個受管政策的集合。由於 `-scope` 未指定，因此預設為 `all`，並同時包含 AWS 受管和客戶受管政策。

```
Get-IAMPolicyList -MaxItem 3
```

輸出：

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
```

```
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

範例 2：此範例會傳回目前 AWS 帳戶中可用的前兩個客戶受管政策的集合。它使用 `-Scope local` 將輸出限制為僅限客戶管理政策。

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

輸出：

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
```

```
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListPolicies。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListPolicies。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

此範例模組會列出、建立、連接和分離角色政策。

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
```

```
 end

 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end

 # Attaches a policy to a role
 #
 # @param role_name [String] The name of the role
 # @param policy_arn [String] The policy ARN
 # @return [Boolean] true if successful, false otherwise
 def attach_policy_to_role(role_name, policy_arn)
   @iam_client.attach_role_policy(
     role_name: role_name,
     policy_arn: policy_arn
   )
   true
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error attaching policy to role: #{e.message}")
   false
 end

 # Lists policy ARNs attached to a role
 #
 # @param role_name [String] The name of the role
 # @return [Array<String>] List of policy ARNs
 def list_attached_policy_arns(role_name)
   response = @iam_client.list_attached_role_policies(role_name: role_name)
   response.attached_policies.map(&:policy_arn)
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 ListPolicies。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
```

```
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 ListPolicies。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listpolicies( iv_scope = iv_scope ).
        MESSAGE 'Retrieved policy list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policies.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListPolicies。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func listPolicies() async throws -> [MyPolicyRecord] {
        var policyList: [MyPolicyRecord] = []

        // Use "Paginated" to get all the policies.
        // This lets the SDK handle the 'isTruncated' in "ListPoliciesOutput".
        let input = ListPoliciesInput()
        let output = client.listPoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let policies = page.policies else {
                    print("Error: no policies returned.")
                    continue
                }

                for policy in policies {
                    guard let name = policy.policyName,
                          let id = policy.policyId,
                          let arn = policy.arn
                    else {
                        throw ServiceHandlerError.noSuchPolicy
                    }
                    policyList.append(MyPolicyRecord(name: name, id: id, arn:
 arn))
                }
            }
```

```
        } catch {
            print("ERROR: listPolicies:", dump(error))
            throw error
        }

        return policyList
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 ListPolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListPolicyVersions 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListPolicyVersions`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理政策
- 復原政策版本

CLI

AWS CLI

若要列出指定受管政策的版本相關資訊

此範例會傳回 ARN 為 `arn:aws:iam::123456789012:policy/MySamplePolicy` 之政策的可用版本清單。

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

輸出：

```
{
    "IsTruncated": false,
```

```
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱 AWS CLI Command Reference 中的 ListPolicyVersions。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回 ARN 為 `arn:aws:iam::123456789012:policy/MyManagedPolicy` 之政策的可用版本清單。若要取得特定版本的政策文件，請使用 `Get-IAMPolicyVersion` 命令並指定想要文件的 `VersionId`。

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

輸出：

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
2/12/2015 9:39:09 AM                                False
       v1
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListPolicyVersions。

Tools for PowerShell V5

範例 1：此範例會傳回 ARN 為 `arn:aws:iam::123456789012:policy/MyManagedPolicy` 之政策的可用版本清單。若要取得特定版本的政策文件，請使用 `Get-IAMPolicyVersion` 命令並指定想要文件的 `VersionId`。

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

輸出：

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
2/12/2015 9:39:09 AM                                False
       v1
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListPolicyVersions。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listpolicyversions(
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
```

```
            MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
        ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListPolicyVersions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListRolePolicies` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListRolePolicies`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 開始使用物聯網裝置保護

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
```

```
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListRolePolicies。

CLI

AWS CLI

列出連接至 IAM 角色的政策

下列 `list-role-policies` 命令會列出指定 IAM 角色的許可政策名稱。

```
aws iam list-role-policies \
    --role-name Test-Role
```

輸出：

```
{
    "PolicyNames": [
        "ExamplePolicy"
    ]
}
```

若要查看連接至角色的信任政策，請使用 `get-role` 命令。若要查看許可政策的詳細資訊，請使用 `get-role-policy` 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListRolePolicies。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
```

```
  policies = result.PolicyNames
 }
 return policies, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListRolePolicies。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出政策。

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.PolicyNames?.length) {
```

```
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListRolePolicies。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
        if ($marker) {
            $listRolePoliciesArguments['Marker'] = $marker;
        }
```

```
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 ListRolePolicies。

PowerShell

Tools for PowerShell V4

範例 1：此範例會傳回內嵌在 IAM 角色 **lamda_exec_role** 中之內嵌政策的名稱清單。若要查看內嵌政策的詳細資訊，請使用 **Get-IAMRolePolicy** 命令。

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

輸出：

```
oneClick_lambda_exec_role_policy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListRolePolicies。

Tools for PowerShell V5

範例 1：此範例會傳回內嵌在 IAM 角色 **lamda_exec_role** 中之內嵌政策的名稱清單。若要查看內嵌政策的詳細資訊，請使用 **Get-IAMRolePolicy** 命令。

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

輸出：

```
oneClick_lambda_exec_role_policy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListRolePolicies。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_policies(role_name):
    """
    Lists inline policies for a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListRolePolicies。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListRolePolicies。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
```

```
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 ListRolePolicies。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listrolepolicies(
          iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved inline policy list for role.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListRolePolicies。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func listRolePolicies(role: String) async throws -> [String] {
        var policyList: [String] = []

        // Use "Paginated" to get all the role policies.
        // This lets the SDK handle the 'isTruncated' in
 "ListRolePoliciesOutput".
        let input = ListRolePoliciesInput(
            roleName: role
        )
        let pages = client.listRolePoliciesPaginated(input: input)

        do {
            for try await page in pages {
                guard let policies = page.policyNames else {
                    print("Error: no role policies returned.")
                    continue
                }

                for policy in policies {
                    policyList.append(policy)
                }
            }
        } catch {
            print("ERROR: listRolePolicies:", dump(error))
            throw error
        }
        return policyList
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 ListRolePolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListRoleTags** 與 CLI

下列程式碼範例示範如何使用 ListRoleTags。

CLI

AWS CLI

若要列出連接至角色的標籤

下列 list-role-tags 命令會擷取與指定角色關聯的標籤之清單。

```
aws iam list-role-tags \
    --role-name production-role
```

輸出：

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 Tagging IAM resources。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListRoleTags。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取與角色關聯的標籤。

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListRoleTags。

Tools for PowerShell V5

範例 1：此範例會擷取與角色關聯的標籤。

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListRoleTags。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListRoles` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListRoles`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// List IAM roles.
```

```
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListRoles。

CLI

AWS CLI

列出目前帳戶的 IAM 角色

下列 `list-roles` 命令會列出目前帳戶的 IAM 角色。

```
aws iam list-roles
```

輸出：

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
```

```
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListRoles。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "log"

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

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
```

```
 return roles, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListRoles。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出角色。

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
```

```
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListRoles。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
```

```
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listRolesArguments["MaxItems"] = $maxItems;
        }
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 ListRoles。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取 AWS 帳戶中所有 IAM 角色的清單。

```
Get-IAMRoleList
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListRoles。

Tools for PowerShell V5

範例 1：此範例會擷取 AWS 帳戶中所有 IAM 角色的清單。

```
Get-IAMRoleList
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListRoles。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.

    :param count: The number of roles to list.
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListRoles。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
  # Lists IAM roles up to a specified count.
  # @param count [Integer] the maximum number of roles to list.
  # @return [Array<String>] the names of the roles.
  def list_roles(count)
    role_names = []
    roles_counted = 0

    @iam_client.list_roles.each_page do |page|
      page.roles.each do |role|
        break if roles_counted >= count

        @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
        role_names << role.role_name
        roles_counted += 1
      end
      break if roles_counted >= count
    end

    role_names
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't list roles for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListRoles。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
```

```
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 ListRoles。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listroles( ).
        MESSAGE 'Retrieved role list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing roles.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListRoles。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    public func listRoles() async throws -> [String] {
        var roleList: [String] = []

        // Use "Paginated" to get all the roles.
        // This lets the SDK handle the 'isTruncated' in "ListRolesOutput".
        let input = ListRolesInput()
        let pages = client.listRolesPaginated(input: input)

        do {
            for try await page in pages {
                guard let roles = page.roles else {
                    print("Error: no roles returned.")
                    continue
                }

                for role in roles {
                    if let name = role.roleName {
                        roleList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listRoles:", dump(error))
            throw error
        }
        return roleList
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 ListRoles。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListSAMLProviders` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListSAMLProviders`。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListSAMLProviders。

CLI

AWS CLI

列出 AWS 帳戶中的 SAML 供應商

此範例會擷取目前 AWS 帳戶中建立的 SAML 2.0 供應商清單。

```
aws iam list-saml-providers
```

輸出：

```
{
    "SAMLProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM SAML 身分提供者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListSAMLProviders。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "log"

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

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
```

```
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListSAMLProviders。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出 SAML 供應商。

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListSAMLProviders。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 ListSAMLProviders。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取在目前 AWS 帳戶中建立的 SAML 2.0 提供者的清單。它會傳回每個 SAML 提供者的 ARN、建立日期和過期日期。

```
Get-IAMSAMLProviderList
```

輸出：

```
Arn                                                  CreateDate
   ValidUntil
---                                                  ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS     12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListSAMLProviders。

Tools for PowerShell V5

範例 1：此範例會擷取在目前 AWS 帳戶中建立的 SAML 2.0 提供者的清單。它會傳回每個 SAML 提供者的 ARN、建立日期和過期日期。

```
Get-IAMSAMLProviderList
```

輸出：

```
Arn                                                  CreateDate
   ValidUntil
---                                                  ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS     12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListSAMLProviders。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
    try:
        found = 0
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListSAMLProviders。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListSAMLProviders。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 ListSAMLProviders。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        oo_result = lo_iam->listsamlproviders( ).
        MESSAGE 'Retrieved SAML provider list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing SAML providers.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListSAMLProviders。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListServerCertificates` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListServerCertificates`。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::listServerCertificates(
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
```

```
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();

        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
                      certificate.GetArn() << std::setw(14) <<

 certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::setw(14) <<

 certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 ListServerCertificates。

CLI

AWS CLI

列出您 AWS 帳戶中的伺服器憑證

下列`list-server-certificates`命令會列出您 AWS 帳戶中存放和可使用的所有伺服器憑證。

```
aws iam list-server-certificates
```

輸出：

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 IAM 中管理伺服器憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListServerCertificates。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出憑證。

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }

    if (response.IsTruncated) {
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v3/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-listing。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListServerCertificates。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-listing。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListServerCertificates。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取已上傳至目前 AWS 帳戶的伺服器憑證的清單。

```
Get-IAMServerCertificateList
```

輸出：

```
Arn                  : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration           : 1/14/2018 9:52:36 AM
Path                 : /Org1/Org2/
ServerCertificateId  : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate           : 4/21/2015 11:14:16 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListServerCertificates。

Tools for PowerShell V5

範例 1：此範例會擷取已上傳至目前 AWS 帳戶的伺服器憑證的清單。

```
Get-IAMServerCertificateList
```

輸出：

```
Arn                  : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration           : 1/14/2018 9:52:36 AM
Path                 : /Org1/Org2/
ServerCertificateId  : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate           : 4/21/2015 11:14:16 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListServerCertificates。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出、更新和刪除伺服器憑證。

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
```

```
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListServerCertificates。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListSigningCertificates** 與 CLI

下列程式碼範例示範如何使用 ListSigningCertificates。

CLI

AWS CLI

若要列出 IAM 使用者的簽署憑證

下列 list-signing-certificates 命令列出名為 Bob 之 IAM 使用者的簽署憑證。

```
aws iam list-signing-certificates \
```

```
    --user-name Bob
```

輸出：

```
{
    "Certificates": [
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-
body>-----END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的管理簽署憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListSigningCertificates。

PowerShell

Tools for PowerShell V4

範例 1：此範例擷取與名為 **Bob** 之使用者關聯的簽署憑證的詳細資訊。

```
Get-IAMSigningCertificate -UserName Bob
```

輸出：

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
```

```
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListSigningCertificates。

Tools for PowerShell V5

範例 1：此範例擷取與名為 Bob 之使用者關聯的簽署憑證的詳細資訊。

```
Get-IAMSigningCertificate -UserName Bob
```

輸出：

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
```

```
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListSigningCertificates。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListUserPolicies` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListUserPolicies`。

CLI

AWS CLI

列出 IAM 使用者的政策

下列 `list-user-policies` 命令會列出連接至名為 Bob 之 IAM 使用者的政策。

```
aws iam list-user-policies \
    --user-name Bob
```

輸出：

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

如需詳細資訊，請參閱《IAM 使用者指南》中的在 AWS 帳戶中建立 AWS IAM 使用者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListUserPolicies。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
```

```
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListUserPolicies。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取內嵌在名為 **David** 之 IAM 使用者中的內嵌政策的名稱清單。

```
Get-IAMUserPolicyList -UserName David
```

輸出：

```
Davids_IAM_Admin_Policy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListUserPolicies。

Tools for PowerShell V5

範例 1：此範例會擷取內嵌在名為 **David** 之 IAM 使用者中的內嵌政策的名稱清單。

```
Get-IAMUserPolicyList -UserName David
```

輸出：

```
Davids_IAM_Admin_Policy
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListUserPolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListUserTags** 與 CLI

下列程式碼範例示範如何使用 ListUserTags。

CLI

AWS CLI

若要列出連接至使用者的標籤

下列 list-user-tags 命令會擷取與指定 IAM 使用者關聯的標籤之清單。

```
aws iam list-user-tags \
    --user-name alice
```

輸出：

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
```

```
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的 Tagging IAM resources。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListUserTags。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取與使用者關聯的標籤。

```
Get-IAMUserTagList -UserName joe
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListUserTags。

Tools for PowerShell V5

範例 1：此範例會擷取與使用者關聯的標籤。

```
Get-IAMUserTagList -UserName joe
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListUserTags。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListUsers` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListUsers`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建立唯讀和讀寫的使用者

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 ListUsers。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
########################################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListUsers。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
```

```
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 ListUsers。

CLI

AWS CLI

列出 IAM 使用者

下列 `list-users` 命令會列出目前帳戶中的 IAM 使用者。

```
aws iam list-users
```

輸出：

```
{
    "Users": [
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的列出 IAM 使用者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListUsers。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
```

```
 }
 return users, err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 ListUsers。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

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

```
        listAllUsers(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
        try {
            boolean done = false;
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }

                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

permissionsBoundary.permissionsBoundaryTypeAsString());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

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

```
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 ListUsers。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出使用者。

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);

  for (const { UserName, CreateDate } of response.Users) {
    console.log(`${UserName} created on: ${CreateDate}`);
  }
  return response;
};
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListUsers。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListUsers。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun listAllUsers() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null) {
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
            }
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 ListUsers。

PHP

SDK for PHP

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
$uuid = uniqid();
$service = new IAMService();
```

```
    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API Reference 中的 ListUsers。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取目前 中的使用者集合 AWS 帳戶。

```
Get-IAMUserList
```

輸出：

```
Arn              : arn:aws:iam::123456789012:user/Administrator
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE1
UserName         : Administrator

Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/6/2015 12:54:42 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : L3EWNONDOM3YUEXAMPLE2
UserName         : bab
```

```
    Arn              : arn:aws:iam::123456789012:user/David
    CreateDate       : 12/10/2014 3:39:27 PM
    PasswordLastUsed : 3/19/2015 8:44:04 AM
    Path             : /
    UserId           : Y4FKWQCXTA52QEXAMPLE3
    UserName         : David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListUsers。

Tools for PowerShell V5

範例 1：此範例會擷取目前 中的使用者集合 AWS 帳戶。

```
Get-IAMUserList
```

輸出：

```
    Arn              : arn:aws:iam::123456789012:user/Administrator
    CreateDate       : 10/16/2014 9:03:09 AM
    PasswordLastUsed : 3/4/2015 12:12:33 PM
    Path             : /
    UserId           : 7K3GJEANSKZF2EXAMPLE1
    UserName         : Administrator

    Arn              : arn:aws:iam::123456789012:user/Bob
    CreateDate       : 4/6/2015 12:54:42 PM
    PasswordLastUsed : 1/1/0001 12:00:00 AM
    Path             : /
    UserId           : L3EWNONDOM3YUEXAMPLE2
    UserName         : bab

    Arn              : arn:aws:iam::123456789012:user/David
    CreateDate       : 12/10/2014 3:39:27 PM
    PasswordLastUsed : 3/19/2015 8:44:04 AM
    Path             : /
    UserId           : Y4FKWQCXTA52QEXAMPLE3
    UserName         : David
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListUsers。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListUsers。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
  # Lists all users in the AWS account
  #
  # @return [Array<Aws::IAM::Types::User>] An array of user objects
  def list_users
    users = []
    @iam_client.list_users.each_page do |page|
      page.users.each do |user|
        users << user
      end
    end
    users
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing users: #{e.message}")
    []
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 ListUsers。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
pub async fn list_users(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
```

```
        .await?;
    Ok(response)
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Rust API reference 中的 ListUsers。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
  TRY.
      oo_result = lo_iam->listusers( ).
      MESSAGE 'Retrieved user list.' TYPE 'I'.
    CATCH /aws1/cx_iamservicefailureex.
      MESSAGE 'Service failure when listing users.' TYPE 'E'.
  ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 ListUsers。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
```

```
import AWSS3


    public func listUsers() async throws -> [MyUserRecord] {
        var userList: [MyUserRecord] = []

        // Use "Paginated" to get all the users.
        // This lets the SDK handle the 'isTruncated' in "ListUsersOutput".
        let input = ListUsersInput()
        let output = client.listUsersPaginated(input: input)

        do {
            for try await page in output {
                guard let users = page.users else {
                    continue
                }
                for user in users {
                    if let id = user.userId, let name = user.userName {
                        userList.append(MyUserRecord(id: id, name: name))
                    }
                }
            }
        }
        catch {
            print("ERROR: listUsers:", dump(error))
            throw error
        }
       return userList
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 ListUsers。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListVirtualMfaDevices** 與 CLI

下列程式碼範例示範如何使用 ListVirtualMfaDevices。

CLI

AWS CLI

若要列出虛擬 MFA 裝置

下列 `list-virtual-mfa-devices` 命令會列出已為目前帳戶設定的虛擬 MFA 裝置。

```
aws iam list-virtual-mfa-devices
```

輸出：

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的啟用虛擬多重要素驗證 (MFA) 裝置。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListVirtualMfaDevices。

PowerShell

Tools for PowerShell V4

範例 1：此範例會擷取指派給 AWS 帳戶中使用者的虛擬 MFA 裝置集合。每個裝置的 **User** 屬性是一個物件，包含裝置被分派給的 IAM 使用者的詳細資訊。

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

輸出：

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
```

```
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ListVirtualMfaDevices。

Tools for PowerShell V5

範例 1：此範例會擷取指派給 AWS 帳戶中使用者的虛擬 MFA 裝置集合。每個裝置的 **User** 屬性是一個物件，包含裝置被分派給的 IAM 使用者的詳細資訊。

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

輸出：

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ListVirtualMfaDevices。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutGroupPolicy** 與 CLI

下列程式碼範例示範如何使用 PutGroupPolicy。

CLI

AWS CLI

將政策新增至群組

下列 put-group-policy 命令會將政策新增至名為 Admins 的 IAM 群組。

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

此命令不會產生輸出。

會在 AdminPolicy.json 檔案中將此政策定義為 JSON 文件。(檔案名稱和副檔名沒有意義。)

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutGroupPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立名為 **AppTesterPolicy** 的內嵌政策，並將其內嵌在 IAM 群組 **AppTesters** 中。如果已存在同名的內嵌政策，其會被覆寫。JSON 政策內容來自 **apptesterpolicy.json** 檔案。請注意，必須使用 **-Raw** 參數，才能成功處理 JSON 檔案的內容。

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 PutGroupPolicy。

Tools for PowerShell V5

範例 1：此範例會建立名為 **AppTesterPolicy** 的內嵌政策，並將其內嵌在 IAM 群組 **AppTesters** 中。如果已存在同名的內嵌政策，其會被覆寫。JSON 政策內容來自 **apptesterpolicy.json** 檔案。請注意，必須使用 **-Raw** 參數，才能成功處理 JSON 檔案的內容。

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 PutGroupPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutRolePermissionsBoundary** 與 CLI

下列程式碼範例示範如何使用 PutRolePermissionsBoundary。

CLI

AWS CLI

範例 1：根據自訂政策將許可界限套用至 IAM 角色

下列 put-role-permissions-boundary 範例會將名為 intern-boundary 的自訂政策套用為指定 IAM 角色的許可界限。

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --role-name lambda-application-role
```

此命令不會產生輸出。

範例 2：根據 AWS 受管政策將許可界限套用至 IAM 角色

下列put-role-permissions-boundary範例會將 AWS 受管PowerUserAccess政策套用為指定 IAM 角色的許可界限。

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的修改角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutRolePermissionsBoundary。

PowerShell

Tools for PowerShell V4

範例 1：此範例示範如何設定 IAM 角色的許可界限。您可以將 AWS 受管政策或自訂政策設定為許可界限。

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 PutRolePermissionsBoundary。

Tools for PowerShell V5

範例 1：此範例示範如何設定 IAM 角色的許可界限。您可以將 AWS 受管政策或自訂政策設定為許可界限。

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 PutRolePermissionsBoundary。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`PutRolePolicy` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `PutRolePolicy`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 開始使用無伺服器資料倉儲
- 組態管理入門
- 開始使用物聯網裝置保護
- 佈建資料倉儲叢集入門
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 PutRolePolicy。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(policyDocument);

    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 PutRolePolicy。

CLI

AWS CLI

將許可政策連接至 IAM 角色

下列 put-role-policy 命令會將許可政策連接到名為 Test-Role 的角色。

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

此命令不會產生輸出。

會在 AdminPolicy.json 檔案中將此政策定義為 JSON 文件。(檔案名稱和副檔名沒有意義。)

若要將信任政策連接至角色，請使用 update-assume-role-policy 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的修改角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutRolePolicy。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
```

```
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::amzn-s3-demo-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
    },
  ],
});

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 PutRolePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立名為 **FedTesterRolePolicy** 的內嵌政策，並將其內嵌在 IAM 角色 **FedTesterRole** 中。如果已存在同名的內嵌政策，其會被覆寫。JSON 政策內容來自 **FedTesterPolicy.json** 檔案。請注意，必須使用 **-Raw** 參數，才能成功處理 JSON 檔案的內容。

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 PutRolePolicy。

Tools for PowerShell V5

範例 1：此範例會建立名為 **FedTesterRolePolicy** 的內嵌政策，並將其內嵌在 IAM 角色 **FedTesterRole** 中。如果已存在同名的內嵌政策，其會被覆寫。JSON 政策內容來自 **FedTesterPolicy.json** 檔案。請注意，必須使用 **-Raw** 參數，才能成功處理 JSON 檔案的內容。

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 PutRolePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutUserPermissionsBoundary** 與 CLI

下列程式碼範例示範如何使用 PutUserPermissionsBoundary。

CLI

AWS CLI

範例 1：根據自訂政策將許可界限套用至 IAM 使用者

下列 put-user-permissions-boundary 範例會將名為 intern-boundary 的自訂政策套用為指定 IAM 使用者的許可界限。

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

此命令不會產生輸出。

範例 2：根據 AWS 受管政策將許可界限套用至 IAM 使用者

下列put-user-permissions-boundary範例會套用名為 的 AWS 受管污染，PowerUserAccess做為指定 IAM 使用者的許可界限。

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的新增和移除 IAM 身分許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutUserPermissionsBoundary。

PowerShell

Tools for PowerShell V4

範例 1：此範例示範如何設定使用者的許可界限。您可以將 AWS 受管政策或自訂政策設定為許可界限。

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 PutUserPermissionsBoundary。

Tools for PowerShell V5

範例 1：此範例示範如何設定使用者的許可界限。您可以將 AWS 受管政策或自訂政策設定為許可界限。

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 PutUserPermissionsBoundary。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutUserPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutUserPolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

將政策連接至 IAM 使用者

下列 put-user-policy 命令會將政策連接至名為 Bob 的 IAM 使用者。

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

此命令不會產生輸出。

會在 AdminPolicy.json 檔案中將此政策定義為 JSON 文件。(檔案名稱和副檔名沒有意義。)

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的新增和移除 IAM 身分許可。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutUserPolicy。

Go

SDK for Go V2

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

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

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
```

```
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- 如需 API 詳細資訊，請參閱 適用於 Go 的 AWS SDK API Reference 中的 PutUserPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會建立名為 **EC2AccessPolicy** 的內嵌政策，並將其內嵌在 IAM 使用者 **Bob** 中。如果已存在同名的內嵌政策，其會被覆寫。JSON 政策內容來自 **EC2AccessPolicy.json** 檔案。請注意，必須使用 **-Raw** 參數，才能成功處理 JSON 檔案的內容。

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 PutUserPolicy。

Tools for PowerShell V5

範例 1：此範例會建立名為 **EC2AccessPolicy** 的內嵌政策，並將其內嵌在 IAM 使用者 **Bob** 中。如果已存在同名的內嵌政策，其會被覆寫。JSON 政策內容來自 **EC2AccessPolicy.json** 檔案。請注意，必須使用 **-Raw** 參數，才能成功處理 JSON 檔案的內容。

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 PutUserPolicy。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Creates an inline policy for a specified user.
# @param username [String] The name of the IAM user.
# @param policy_name [String] The name of the policy to create.
# @param policy_document [String] The JSON policy document.
# @return [Boolean]
def create_user_policy(username, policy_name, policy_document)
  @iam_client.put_user_policy({
                                user_name: username,
                                policy_name: policy_name,
                                policy_document: policy_document
                              })
  @logger.info("Policy #{policy_name} created for user #{username}.")
  true
rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    false
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 PutUserPolicy。

Swift

SDK for Swift

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSIAM
import AWSS3


    func putUserPolicy(policyDocument: String, policyName: String, user:
IAMClientTypes.User) async throws {
        let input = PutUserPolicyInput(
            policyDocument: policyDocument,
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.putUserPolicy(input: input)
        } catch {
            print("ERROR: putUserPolicy:", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Swift API reference 中的 PutUserPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RemoveClientIdFromOpenIdConnectProvider** 與 CLI

下列程式碼範例示範如何使用 RemoveClientIdFromOpenIdConnectProvider。

CLI

AWS CLI

若要從已向指定 IAM OpenID Connect 提供者註冊的用戶端 ID 清單中移除指定的用戶端 ID

此範例會從與 ARN 為 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com 的 IAM OIDC 提供者關聯的用戶端 ID 清單中移除用戶端 ID My-TestApp-3。

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Creating OpenID Connect (OIDC) identity providers。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RemoveClientIdFromOpenIdConnectProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從與 ARN 為 **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com** 的 IAM OIDC 提供者關聯的用戶端 ID 清單中移除用戶端 ID **My-TestApp-3**。

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 RemoveClientIdFromOpenIdConnectProvider。

Tools for PowerShell V5

範例 1：此範例會從與 ARN 為 **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com** 的 IAM OIDC 提供者關聯的用戶端 ID 清單中移除用戶端 ID **My-TestApp-3**。

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 RemoveClientIdFromOpenIdConnectProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RemoveRoleFromInstanceProfile** 與 CLI

下列程式碼範例示範如何使用 RemoveRoleFromInstanceProfile。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 受管串流入門
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

CLI

AWS CLI

若要從執行個體設定檔中移除角色

下列 remove-role-from-instance-profile 命令會將名為 Test-Role 的角色從名為 ExampleInstanceProfile 的執行個體設定檔中移除。

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
    --role-name Test-Role
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的使用執行個體設定檔。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RemoveRoleFromInstanceProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將名為 **MyNewRole** 的角色從名為 **MyNewRole** 的 EC2 執行個體設定檔中刪除。在 IAM 主控台中建立的執行個體設定檔一律與角色同名，如本範例中所示。如果您在 API 或 CLI 中建立它們，則它們可以有不同的名稱。

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 RemoveRoleFromInstanceProfile。

Tools for PowerShell V5

範例 1：此範例會將名為 **MyNewRole** 的角色從名為 **MyNewRole** 的 EC2 執行個體設定檔中刪除。在 IAM 主控台中建立的執行個體設定檔一律與角色同名，如本範例中所示。如果您在 API 或 CLI 中建立它們，則它們可以有不同的名稱。

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 RemoveRoleFromInstanceProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RemoveUserFromGroup** 與 CLI

下列程式碼範例示範如何使用 RemoveUserFromGroup。

CLI

AWS CLI

從 IAM 群組移除使用者

下列 remove-user-from-group 命令會將名為 Bob 的使用者從名為 Admins 的 IAM 群組中移除。

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 IAM 使用者群組中新增和移除使用者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RemoveUserFromGroup。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從 **Testers** 群組中移除 IAM 使用者 **Bob**。

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

範例 2：此範例會尋找 IAM 使用者 **Theresa** 所屬的任意群組，然後從這些群組中移除 **Theresa**。

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

範例 3：此範例顯示從 **Testers** 群組中移除 IAM 使用者 **Bob** 的替代方式。

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 RemoveUserFromGroup。

Tools for PowerShell V5

範例 1：此範例會從 **Testers** 群組中移除 IAM 使用者 **Bob**。

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

範例 2：此範例會尋找 IAM 使用者 **Theresa** 所屬的任意群組，然後從這些群組中移除 **Theresa**。

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

範例 3：此範例顯示從 **Testers** 群組中移除 IAM 使用者 **Bob** 的替代方式。

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 RemoveUserFromGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ResyncMfaDevice** 與 CLI

下列程式碼範例示範如何使用 ResyncMfaDevice。

CLI

AWS CLI

若要同步 MFA 裝置

下列 resync-mfa-device 範例會將與 IAM 使用者 Bob 關聯且 ARN 為 arn:aws:iam::123456789012:mfa/BobsMFADevice 的 MFA 裝置，與提供兩個驗證碼的身分驗證器程式同步。

```
aws iam resync-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 AWS中使用多重要素驗證 (MFA)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ResyncMfaDevice。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將與 IAM 使用者 Bob 關聯且 ARN 為 arn:aws:iam::123456789012:mfa/bob 的 MFA 裝置，與提供兩個驗證碼的身分驗證器程式同步。

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

範例 2：此範例會將與 IAM 使用者 Theresa 關聯的 IAM MFA 裝置，與序號為 ABCD12345678 並提供兩個驗證碼的實體裝置同步。

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 ResyncMfaDevice。

Tools for PowerShell V5

範例 1：此範例會將與 IAM 使用者 Bob 關聯且 ARN 為 arn:aws:iam::123456789012:mfa/bob 的 MFA 裝置，與提供兩個驗證碼的身分驗證器程式同步。

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

範例 2：此範例會將與 IAM 使用者 Theresa 關聯的 IAM MFA 裝置，與序號為 ABCD12345678 並提供兩個驗證碼的實體裝置同步。

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 ResyncMfaDevice。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

SetDefaultPolicyVersion 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 SetDefaultPolicyVersion。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理政策
- 復原政策版本

CLI

AWS CLI

若要將指定政策的指定版本設定為政策的預設版本。

此範例會將 ARN 為 arn:aws:iam::123456789012:policy/MyPolicy 的政策的 v2 版本設定為預設的作用中版本。

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

如需詳細資訊，請參閱「IAM 使用者指南」AWS 中的 IAM 中的政策和許可。

- 如需 API 詳細資訊，請參閱 AWS CLI Command Reference 中的 SetDefaultPolicyVersion。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將 ARN 為 **arn:aws:iam::123456789012:policy/MyPolicy** 的政策的 **v2** 版本設定為預設的作用中版本。

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 SetDefaultPolicyVersion。

Tools for PowerShell V5

範例 1：此範例會將 ARN 為 **arn:aws:iam::123456789012:policy/MyPolicy** 的政策的 **v2** 版本設定為預設的作用中版本。

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 SetDefaultPolicyVersion。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->setdefaultpolicyversion(
          iv_policyarn = iv_policy_arn
          iv_versionid = iv_version_id ).
        MESSAGE 'Default policy version set successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy or version does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iaminvalidinputex.
        MESSAGE 'Invalid input provided.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 SetDefaultPolicyVersion。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **TagRole** 與 CLI

下列程式碼範例示範如何使用 TagRole。

CLI

AWS CLI

若要將標籤新增至角色

下列 tag-role 命令會將含部門名稱的標籤新增至指定角色。

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Tagging IAM resources。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 TagRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會標籤新增至 Identity Management Service 中的角色

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 TagRole。

Tools for PowerShell V5

範例 1：此範例會標籤新增至 Identity Management Service 中的角色

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 TagRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`TagUser`** 與 CLI

下列程式碼範例示範如何使用 `TagUser`。

CLI

AWS CLI

若要將標籤新增至使用者

下列 `tag-user` 命令將含關聯部門的標籤新增至指定使用者。

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Tagging IAM resources。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 TagUser。

PowerShell

Tools for PowerShell V4

範例 1：此範例會標籤新增至 Identity Management Service 中的使用者

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 TagUser。

Tools for PowerShell V5

範例 1：此範例會標籤新增至 Identity Management Service 中的使用者

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 TagUser。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`UntagRole`** 與 CLI

下列程式碼範例示範如何使用 `UntagRole`。

CLI

AWS CLI

若要從角色中移除標籤

下列 `untag-role` 命令會從指定的角色移除任何金鑰名稱為「Department」的標籤。

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Tagging IAM resources。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UntagRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從名為「MyRoleName」且標籤索引鍵為「abac」的角色中移除標籤。若要移除多個標籤，請提供以逗號分隔的標籤索引鍵清單。

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UntagRole。

Tools for PowerShell V5

範例 1：此範例會從名為「MyRoleName」且標籤索引鍵為「abac」的角色中移除標籤。若要移除多個標籤，請提供以逗號分隔的標籤索引鍵清單。

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UntagRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`UntagUser`** 與 CLI

下列程式碼範例示範如何使用 `UntagUser`。

CLI

AWS CLI

若要從使用者中移除標籤

下列 `untag-user` 命令會從指定的使用者移除任何金鑰名稱為「Department」的標籤。

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Tagging IAM resources。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UntagUser。

PowerShell

Tools for PowerShell V4

範例 1：此範例會從名為「joe」且標籤索引鍵為「abac」和「xyzw」的使用者中移除標籤。若要移除多個標籤，請提供以逗號分隔的標籤索引鍵清單。

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UntagUser。

Tools for PowerShell V5

範例 1：此範例會從名為「joe」且標籤索引鍵為「abac」和「xyzw」的使用者中移除標籤。若要移除多個標籤，請提供以逗號分隔的標籤索引鍵清單。

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UntagUser。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`UpdateAccessKey` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `UpdateAccessKey`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 管理存取金鑰

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
###############################################################################
# function iam_update_access_key
#
# This function can activate or deactivate an IAM access key for the specified
 IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
```

```
#       -k access_key -- The access key to update.
#       -a            -- Activate the selected access key.
#       -d            -- Deactivate the selected access key.
#
# Example:
#       # To deactivate the selected access key for IAM user Bob
#       iam_update_access_key -u Bob -k AKIAIOSFODNN7EXAMPLE -d
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_update_access_key() {
  local user_name access_key status response
  local option OPTARG # Required to use getopts command in a function.
  local activate_flag=false deactivate_flag=false

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_update_access_key"
    echo "Updates the status of an AWS Identity and Access Management (IAM)
 access key for the specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to update."
    echo "  -a              Activate the access key."
    echo "  -d              Deactivate the access key."
    echo ""
  }

  # Retrieve the calling parameters.
    while getopts "u:k:adh" option; do
      case "${option}" in
        u) user_name="${OPTARG}" ;;
        k) access_key="${OPTARG}" ;;
        a) activate_flag=true ;;
        d) deactivate_flag=true ;;
        h)
          usage
          return 0
          ;;
        \?)
          echo "Invalid parameter"
          usage
          return 1
```

```
          ;;
      esac
    done
    export OPTIND=1

  # Validate input parameters
    if [[ -z "$user_name" ]]; then
      errecho "ERROR: You must provide a username with the -u parameter."
      usage
      return 1
    fi

    if [[ -z "$access_key" ]]; then
      errecho "ERROR: You must provide an access key with the -k parameter."
      usage
      return 1
    fi

    # Ensure that only -a or -d is specified
    if [[ "$activate_flag" == true && "$deactivate_flag" == true ]]; then
      errecho "ERROR: You cannot specify both -a (activate) and -d (deactivate)
at the same time."
      usage
      return 1
    fi

    # If neither -a nor -d is provided, return an error
    if [[ "$activate_flag" == false && "$deactivate_flag" == false ]]; then
      errecho "ERROR: You must specify either -a (activate) or -d (deactivate)."
      usage
      return 1
    fi

    # Determine the status based on the flag
    if [[ "$activate_flag" == true ]]; then
      status="Active"
    elif [[ "$deactivate_flag" == true ]]; then
      status="Inactive"
    fi

    iecho "Parameters:\n"
    iecho "    Username:   $user_name"
    iecho "    Access key: $access_key"
    iecho "    New status: $status"
```

```
    iecho ""

    # Update the access key status
    response=$(aws iam update-access-key \
      --user-name "$user_name" \
      --access-key-id "$access_key" \
      --status "$status" 2>&1)

    local error_code=${?}

    if [[ $error_code -ne 0 ]]; then
      aws_cli_error_log $error_code
      errecho "ERROR: AWS reports update-access-key operation failed.\n$response"
      return 1
    fi

    iecho "update-access-key response: $response"
    iecho

    return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateAccessKey。

C++

適用於 C++ 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  Aws::IAM::Model::StatusType status,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
```

```
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 UpdateAccessKey。

CLI

AWS CLI

啟用或停用 IAM 使用者的存取金鑰

下列 update-access-key 命令會為名為 Bob 的 IAM 使用者停用指定的存取金鑰 (存取金鑰 ID 與私密存取金鑰)。

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

此命令不會產生輸出。

停用金鑰表示它無法用於程式設計存取 AWS。但是，金鑰仍然可用，並且可以重新啟用。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的存取金鑰。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateAccessKey。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

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

    private static StatusType statusType;

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

                Usage:
                    <username> <accessId> <status>\s

                Where:
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
```

```
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }

    public static void updateKey(IamClient iam, String username, String accessId,
String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
                    .build();

            iam.updateAccessKey(request);
            System.out.printf("Successfully updated the status of access key %s
to" +
                    "status %s for user %s", accessId, status, username);

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

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 UpdateAccessKey。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

更新存取金鑰。

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
    Status: StatusType.Inactive,
    UserName: userName,
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v3/developer-guide/iam-examples-managing-access-keys.html#iam-examples-managing-access-keys-updating。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateAccessKey。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/iam-examples-managing-access-keys.html#iam-examples-managing-access-keys-updating。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateAccessKey。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將名為 Bob 之 IAM 使用者的存取金鑰 AKIAIOSFODNN7EXAMPLE 的狀態變更為Inactive。

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateAccessKey。

Tools for PowerShell V5

範例 1：此範例會將名為 Bob 之 IAM 使用者的存取金鑰 AKIAIOSFODNN7EXAMPLE 的狀態變更為Inactive。

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateAccessKey。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 UpdateAccessKey。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    TRY.
        lo_iam->updateaccesskey(
          iv_accesskeyid = iv_access_key_id
          iv_status = iv_status
          iv_username = iv_user_name ).
        MESSAGE 'Access key updated successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Access key or user does not exist.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 UpdateAccessKey。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateAccountPasswordPolicy** 與 CLI

下列程式碼範例示範如何使用 UpdateAccountPasswordPolicy。

CLI

AWS CLI

若要設定或變更目前的帳戶密碼政策

下列 update-account-password-policy 命令會設定密碼政策，要求密碼最小長度為 8 個字元，且必須包含一個或多個數字。

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

此命令不會產生輸出。

變更帳戶的密碼政策會影響為帳戶中的 IAM 使用者建立的任何新密碼。密碼政策變更不影響現有密碼。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的設定 IAM 使用者的帳戶密碼政策。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateAccountPasswordPolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會使用指定設定更新帳戶的密碼政策。請注意，命令中未包含的任何參數都不會保持不變。相反，它們會重設為預設值。

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateAccountPasswordPolicy。

Tools for PowerShell V5

範例 1：此範例會使用指定設定更新帳戶的密碼政策。請注意，命令中未包含的任何參數都不會保持不變。相反，它們會重設為預設值。

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateAccountPasswordPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`UpdateAssumeRolePolicy`** 與 CLI

下列程式碼範例示範如何使用 `UpdateAssumeRolePolicy`。

CLI

AWS CLI

若要更新 IAM 角色的信任政策

下列 `update-assume-role-policy` 命令會更新名為 `Test-Role` 之角色的信任政策。

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

此命令不會產生輸出。

在 Test-Role-Trust-Policy.json 檔案中，將信任政策定義為 JSON 文件。(檔案名稱和副檔名沒有意義。) 信任政策必須指定主體。

若要更新角色的許可政策，請使用 put-role-policy 命令。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM 角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateAssumeRolePolicy。

PowerShell

Tools for PowerShell V4

範例 1：此範例會使用新的信任政策更新名為 **ClientRole** 的 IAM 角色，其內容來自 **ClientRolePolicy.json** 檔案。請注意，必須使用 **-Raw** 切換參數，才能成功處理 JSON 檔案的內容。

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateAssumeRolePolicy。

Tools for PowerShell V5

範例 1：此範例會使用新的信任政策更新名為 **ClientRole** 的 IAM 角色，其內容來自 **ClientRolePolicy.json** 檔案。請注意，必須使用 **-Raw** 切換參數，才能成功處理 JSON 檔案的內容。

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateAssumeRolePolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateGroup** 與 CLI

下列程式碼範例示範如何使用 UpdateGroup。

CLI

AWS CLI

若要重新命名 IAM 群組

下列 update-group 命令會將 IAM 群組 Test 的名稱變更為 Test-1。

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的重新命名 IAM 使用者群組。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateGroup。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將 IAM 群組 **Testers** 重新命名為 **AppTesters**。

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

範例 2：此範例會將 IAM 群組 **AppTesters** 的路徑變更為 **/Org1/Org2/**。這會將群組的 ARN 變更為 **arn:aws:iam::123456789012:group/Org1/Org2/AppTesters**。

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateGroup。

Tools for PowerShell V5

範例 1：此範例會將 IAM 群組 **Testers** 重新命名為 **AppTesters**。

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

範例 2：此範例會將 IAM 群組 **AppTesters** 的路徑變更為 **/Org1/Org2/**。這會將群組的 ARN 變更為 **arn:aws:iam::123456789012:group/Org1/Org2/AppTesters**。

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateLoginProfile** 與 CLI

下列程式碼範例示範如何使用 UpdateLoginProfile。

CLI

AWS CLI

若要更新 IAM 使用者的密碼

下列 update-login-profile 命令會為名為 Bob 的 IAM 使用者建立新的密碼。

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

此命令不會產生輸出。

若要設定帳戶的密碼政策，請使用 update-account-password-policy 命令。如果新密碼違反帳戶密碼政策，則命令會傳回 PasswordPolicyViolation 錯誤。

如果帳戶密碼政策允許，IAM 使用者可以使用 change-password 命令變更自己的密碼。

將密碼存放於安全處。密碼一旦遺失，便無法復原，您必須使用 create-login-profile 命令建立新密碼。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的管理 IAM 使用者的密碼。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateLoginProfile。

PowerShell

Tools for PowerShell V4

範例 1：此範例會為 IAM 使用者 **Bob** 設定新的臨時密碼，並在使用者下次登入時要求其變更密碼。

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateLoginProfile。

Tools for PowerShell V5

範例 1：此範例會為 IAM 使用者 **Bob** 設定新的臨時密碼，並在使用者下次登入時要求其變更密碼。

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateLoginProfile。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`UpdateOpenIdConnectProviderThumbprint`** 與 CLI

下列程式碼範例示範如何使用 `UpdateOpenIdConnectProviderThumbprint`。

CLI

AWS CLI

若要以新清單取代現有的伺服器憑證指紋清單

此範例會更新 ARN 為 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com 之 OIDC 提供者的憑證指紋清單，以使用新指紋。

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

如果您 OIDC 提供者的探索和 JWKS 端點使用不同的憑證，請在 --thumbprint-list值中包含兩個端點的指紋。

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Creating OpenID Connect (OIDC) identity providers。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateOpenIdConnectProviderThumbprint。

PowerShell

Tools for PowerShell V4

範例 1：此範例會更新 ARN 為 **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com** 之 OIDC 提供者的憑證指紋清單，以使用新指紋。當與提供者關聯之憑證變更時，OIDC 提供者會共用新值。

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateOpenIdConnectProviderThumbprint。

Tools for PowerShell V5

範例 1：此範例會更新 ARN 為 **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com** 之 OIDC 提供者的憑證指紋清單，以使用新指紋。當與提供者關聯之憑證變更時，OIDC 提供者會共用新值。

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateOpenIdConnectProviderThumbprint。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`UpdateRole`** 與 CLI

下列程式碼範例示範如何使用 `UpdateRole`。

CLI

AWS CLI

若要變更 IAM 角色的描述或工作階段持續時間

下列 `update-role` 命令會將 IAM 角色 `production-role` 的描述變更為 `Main production role`，並將工作階段持續時間上限設定為 12 小時。

```
aws iam update-role \
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的修改角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateRole。

PowerShell

Tools for PowerShell V4

範例 1：此範例會更新角色描述，以及可以請求角色工作階段的工作階段持續時間值上限 (以秒為單位)。

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateRole。

Tools for PowerShell V5

範例 1：此範例會更新角色描述，以及可以請求角色工作階段的工作階段持續時間值上限 (以秒為單位)。

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateRoleDescription** 與 CLI

下列程式碼範例示範如何使用 UpdateRoleDescription。

CLI

AWS CLI

變更 IAM 角色的描述

下列 update-role 命令會將 IAM 角色的描述從 production-role 變更為 Main production role。

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

輸出：

```
{
    "Role": {
```

```
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {}
                }
            ]
        },
        "Description": "Main production role"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的修改角色。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateRoleDescription。

PowerShell

Tools for PowerShell V4

範例 1：此範例會更新您的帳戶中 IAM 角色的描述。

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateRoleDescription。

Tools for PowerShell V5

範例 1：此範例會更新您的帳戶中 IAM 角色的描述。

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateRoleDescription。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateSamlProvider** 與 CLI

下列程式碼範例示範如何使用 UpdateSamlProvider。

CLI

AWS CLI

若要更新現有 SAML 提供者的中繼資料文件

此範例會更新 IAM 中 ARN 為 arn:aws:iam::123456789012:saml-provider/SAMLADFS、使用檔案 SAMLMetaData.xml 中的新 SAML 中繼資料文件的 SAML 提供者。

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

輸出：

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的建立 IAM SAML 身分提供者。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateSamlProvider。

PowerShell

Tools for PowerShell V4

範例 1：此範例會更新 IAM 中 ARN 為 **arn:aws:iam::123456789012:saml-provider/SAMLADFS**、使用檔案 **SAMLMetaData.xml** 中的新 SAML 中繼資料文件的 SAML 提供者。請注意，必須使用 **-Raw** 切換參數，才能成功處理 JSON 檔案的內容。

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateSamlProvider。

Tools for PowerShell V5

範例 1：此範例會更新 IAM 中 ARN 為 **arn:aws:iam::123456789012:saml-provider/SAMLADFS**、使用檔案 **SAMLMetaData.xml** 中的新 SAML 中繼資料文件的 SAML 提供者。請注意，必須使用 **-Raw** 切換參數，才能成功處理 JSON 檔案的內容。

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateSamlProvider。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UpdateServerCertificate 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UpdateServerCertificate。

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
                                          const Aws::String &newCertificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);

    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }

    return result;
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 UpdateServerCertificate。

CLI

AWS CLI

變更 AWS 帳戶中伺服器憑證的路徑或名稱

下列 update-server-certificate 命令會將憑證名稱從 myServerCertificate 變更為 myUpdatedServerCertificate。該命令也會將路徑變更為 /cloudfront/，這樣

Amazon CloudFront 服務就可以存取該路徑。此命令不會產生輸出。您可以透過執行 `list-server-certificates` 命令來查看更新的結果。

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的在 IAM 中管理伺服器憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateServerCertificate。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

更新伺服器憑證。

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });
```

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v3/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-updating。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateServerCertificate。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};

iam.updateServerCertificate(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/iam-examples-server-certificates.html#iam-examples-server-certificates-updating。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateServerCertificate。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將名為 MyServerCertificate 的憑證重新命名為 MyRenamedServerCertificate。

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

範例 2：此範例會將名為 MyServerCertificate 的憑證移至路徑 /Org1/Org2/。這會將資源的 ARN 變更為 arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate。

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateServerCertificate。

Tools for PowerShell V5

範例 1：此範例會將名為 MyServerCertificate 的憑證重新命名為 MyRenamedServerCertificate。

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

範例 2：此範例會將名為 MyServerCertificate 的憑證移至路徑 /Org1/Org2/。這會將資源的 ARN 變更為 arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate。

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateServerCertificate。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

列出、更新和刪除伺服器憑證。

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end
```

```
  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 UpdateServerCertificate。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateSigningCertificate** 與 CLI

下列程式碼範例示範如何使用 UpdateSigningCertificate。

CLI

AWS CLI

若要啟用或停用 IAM 使用者的簽署憑證

下列 update-signing-certificate 命令將停用名為 Bob 之 IAM 使用者的指定簽署憑證。

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

若要取得簽署憑證的 ID，請使用 list-signing-certificates 命令。

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的管理簽署憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateSigningCertificate。

PowerShell

Tools for PowerShell V4

範例 1：此範例會更新與名為 **Bob**、憑證 ID 為 **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** 的 IAM 使用者關聯的憑證，將其標示為非作用中。

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateSigningCertificate。

Tools for PowerShell V5

範例 1：此範例會更新與名為 **Bob**、憑證 ID 為 **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** 的 IAM 使用者關聯的憑證，將其標示為非作用中。

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateSigningCertificate。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`UpdateUser` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `UpdateUser`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 建立唯讀和讀寫的使用者

C++

SDK for C++

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
                             const Aws::String &newUserName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
```

```
    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 UpdateUser。

CLI

AWS CLI

變更 IAM 使用者的名稱

下列 update-user 命令會將 IAM 使用者名稱從 Bob 變更為 Robert。

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的重新命名 IAM 使用者群組。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateUser。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

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

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String curName = args[0];
```

```
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 UpdateUser。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

更新使用者。

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

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

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateUser。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });
```

```
var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- 如需詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateUser。

Kotlin

SDK for Kotlin

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
suspend fun updateIAMUser(
    curName: String?,
    newName: String?,
) {
    val request =
        UpdateUserRequest {
            userName = curName
            newUserName = newName
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
```

```
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Kotlin API reference 中的 UpdateUser。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將 IAM 使用者 **Bob** 重新命名為 **Robert**。

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

範例 2：此範例會將 IAM 使用者 **Bob** 的路徑變更為 **/Org1/Org2/**，這可有效地將使用者的 ARN 變更為 **arn:aws:iam::123456789012:user/Org1/Org2/bob**。

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UpdateUser。

Tools for PowerShell V5

範例 1：此範例會將 IAM 使用者 **Bob** 重新命名為 **Robert**。

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

範例 2：此範例會將 IAM 使用者 **Bob** 的路徑變更為 **/Org1/Org2/**，這可有效地將使用者的 ARN 變更為 **arn:aws:iam::123456789012:user/Org1/Org2/bob**。

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UpdateUser。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 UpdateUser。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
# Updates an IAM user's name
#
# @param current_name [String] The current name of the user
# @param new_name [String] The new name of the user
def update_user_name(current_name, new_name)
  @iam_client.update_user(user_name: current_name, new_user_name: new_name)
  true
rescue StandardError => e
  @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
  false
end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 UpdateUser。

SAP ABAP

適用於 SAP ABAP 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
  TRY.
      lo_iam->updateuser(
        iv_username = iv_user_name
        iv_newusername = iv_new_user_name ).
```

```
        MESSAGE 'User updated successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'New user name already exists.' TYPE 'E'.
    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 UpdateUser。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UploadServerCertificate 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UploadServerCertificate。

CLI

AWS CLI

將伺服器憑證上傳至 AWS 您的帳戶

下列 upload-server-certificate 命令會將伺服器憑證上傳至 AWS 您的帳戶。在此範例中，憑證位於檔案 public_key_cert_file.pem 中，相關聯的私密金鑰位於檔案 my_private_key.pem 中，而憑證授權機構 (CA) 提供的憑證鏈結位於 my_certificate_chain_file.pem 檔案中。檔案上傳完成後，其就會以 myServerCertificate 名稱提供使用。以 file:// 開頭的參數會告訴命令讀取檔案的內容，並將其用作參數值 (而不是檔案名稱本身)。

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

輸出：

```
{
    "ServerCertificateMetadata": {
```

```
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

如需詳細資訊，請參閱《使用 IAM 指南》中的「建立、上傳和刪除伺服器憑證」。

- 如需 API 詳細資訊，請參閱 AWS CLI Command Reference 中的 UploadServrCertificate。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "node:path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
```

```
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

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

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 UploadServerCertificate。

PowerShell

Tools for PowerShell V4

範例 1：此範例會將新伺服器憑證上傳至 IAM 帳戶。包含憑證內文、私有金鑰和 (可選) 憑證鏈的檔案全都必須採用 PEM 編碼。請注意，參數需要檔案的實際內容，而不是檔案名稱。必須使用 **-Raw** 切換參數，才能成功處理檔案內容。

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

輸出：

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UploadServerCertificate。

Tools for PowerShell V5

範例 1：此範例會將新伺服器憑證上傳至 IAM 帳戶。包含憑證內文、私有金鑰和 (可選) 憑證鏈的檔案全都必須採用 PEM 編碼。請注意，參數需要檔案的實際內容，而不是檔案名稱。必須使用 **-Raw** 切換參數，才能成功處理檔案內容。

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

輸出：

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UploadServerCertificate。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UploadSigningCertificate** 與 CLI

下列程式碼範例示範如何使用 UploadSigningCertificate。

CLI

AWS CLI

若要上傳 IAM 使用者的簽署憑證

下列 upload-signing-certificate 命令會上傳名為 Bob 之 IAM 使用者的簽署憑證。

```
aws iam upload-signing-certificate \
    --user-name Bob \
    --certificate-body file://certificate.pem
```

輸出：

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

憑證位於名為 certificate.pem 的 PEM 格式的檔案中。

如需詳細資訊，請參閱《IAM 使用指南》中的「建立和上傳使用者簽署憑證」。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UploadSigningCertificate。

PowerShell

Tools for PowerShell V4

範例 1：此範例會上傳新的 X.509 簽署憑證，並將其與名為 Bob 的 IAM 使用者關聯起來。包含憑證內文的檔案經過 PEM 編碼。CertificateBody 參數需要憑證檔案的實際內容，而不是檔案名稱。必須使用 -Raw 切換參數才能成功處理檔案。

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

輸出：

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                   YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                   21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                   rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                   nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                   NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                   -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
```

```
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 UploadSigningCertificate。

Tools for PowerShell V5

範例 1：此範例會上傳新的 X.509 簽署憑證，並將其與名為 Bob 的 IAM 使用者關聯起來。包含憑證內文的檔案經過 PEM 編碼。CertificateBody 參數需要憑證檔案的實際內容，而不是檔案名稱。必須使用 -Raw 切換參數才能成功處理檔案。

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

輸出：

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
```

```
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 UploadSigningCertificate。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDKs IAM 案例

下列程式碼範例示範如何在 IAM AWS SDKs 中實作常見案例。這些案例會向您展示如何呼叫 IAM 中的多個函數或與其他 AWS 服務組合來完成特定任務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- 使用 AWS SDK 建置和管理彈性服務
- 設定容器服務連線
- 為無伺服器啟動類型建立容器任務
- 使用函數代理整合建立靜態 API
- 使用 AWS SDK 建立唯讀和讀寫 IAM 使用者
- 為虛擬機器執行個體建立容器服務
- 建立受管監控工作區
-
- 建立您的第一個無伺服器函數
- 使用 CLI 開始使用無伺服器資料倉儲
- 組態管理入門
- 開始使用物聯網裝置保護

- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門
- 受管串流入門
- 佈建資料倉儲叢集入門
- 工作流程協同運作入門
- 使用 SDK 管理 IAM 存取金鑰 AWS
- 使用 SDK 管理 IAM 政策 AWS
- 使用 SDK 管理 IAM 角色 AWS
- 使用 SDK 管理您的 IAM 帳戶 AWS
- 將硬式編碼秘密移至安全秘密儲存
- 允許 AWS Compute Optimizer 自動化功能套用建議的動作
- 在整個組織中啟用自動化的政策
- 為您的帳戶啟用自動化的政策
- 授予組織管理帳戶 Compute Optimizer Automation 完整存取權的政策
- 授予獨立 AWS 帳戶 Compute Optimizer Automation 完整存取權的政策
- 為組織的管理帳戶授予 Compute Optimizer Automation 唯讀存取權的政策
- 授予獨立 AWS 帳戶 Compute Optimizer Automation 唯讀存取權的政策
- 授予運算最佳化自動化服務連結角色許可的政策
- 使用 SDK 轉返 IAM 政策版本 AWS
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 使用 AWS Command Line Interface v2 設定 DynamoDB 的屬性型存取控制
- 設定系統管理
- 在監控儀表板中使用屬性變數來監控多個無伺服器函數
- 使用 AWS Command Line Interface v2 處理 DynamoDB 串流和Time-to-Live
- 使用 SDK 搭配 IAM 政策建置器 API AWS

使用 AWS SDK 建置和管理彈性服務

下列程式碼範例示範如何建立負載平衡的 Web 服務，以傳回書籍、影片和歌曲建議。此範例顯示服務如何回應失故障，以及如何在發生故障時重組服務以提高復原能力。

- 使用 Amazon EC2 Auto Scaling 群組根據啟動範本建立 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，並將執行個體數量保持在指定範圍內。
- 使用 Elastic Load Balancing 處理和分發 HTTP 請求。
- 監控 Auto Scaling 群組中執行個體的運作狀態，並且只將請求轉送給運作良好的執行個體。
- 在每個 EC2 執行個體上執行一個 Python Web 伺服器來處理 HTTP 請求。Web 伺服器會回應建議和運作狀態檢查。
- 使用 Amazon DynamoDB 資料表模擬建議服務。
- 透過更新 AWS Systems Manager 參數來控制 Web 伺服器對請求和運作狀態檢查的回應。

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在命令提示中執行互動式案例。

```
    static async Task Main(string[] args)
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
```

```
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
```

```
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
    }

    /// <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
```

```
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
```

```
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();
```

```
        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
 The target group\n"
            + "defines how the load balancer connects to instances. The load
 balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
 instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
 _autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
 _elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
 protocol, port, defaultVpc.VpcId);

        await
 _elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
 subnetIds, targetGroup);
        await
 _autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
 targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
 _elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
 _elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
```

```
            var sshPortIsOpen =
_autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
your default VPC must\n"
                    + "allows access from this computer. You can either add it
automatically from this\n"
                    + "example or add it yourself using the AWS Management
Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound traffic from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound SSH traffic for debugging from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
```

```
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
```

```
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
```

```
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");
```

```
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
```

```
        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
_elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
_elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
_autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
_autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
_autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
```

```
            );
            await
 _recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

建立包裝 Auto Scaling 和 Amazon EC2 動作的類別。

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;
    private readonly ILogger<AutoScalerWrapper> _logger;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";
```

```
    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration,
        ILogger<AutoScalerWrapper> logger)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;
        _logger = logger;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }
```

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {

       var assumeRoleDoc = "{" +
                                   "\"Version\": \"2012-10-17\"," +
                                   "\"Statement\": [{" +
                                        "\"Effect\": \"Allow\"," +
                                        "\"Principal\": {" +
                                        "\"Service\": [" +
                                            "\"ec2.amazonaws.com\"" +
                                        "]" +
                                        "}," +
                                   "\"Action\": \"sts:AssumeRole\"" +
                                   "}]" +
                               "}";

       var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

       var policyArn = "";

       try
       {
           var createPolicyResult = await _amazonIam.CreatePolicyAsync(
               new CreatePolicyRequest
```

```
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
```

```
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
```

```
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }
```

```
   /// <summary>
   /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
   /// The launch template specifies a Bash script in its user data field that
runs after
   /// the instance is started. This script installs the Python packages and
starts a Python
   /// web server on the instance.
   /// </summary>
   /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
   /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
   /// <returns>The template object.</returns>
   public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
   {
       try
       {
           await CreateKeyPair(_keyPairName);
           await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName,
               _instanceProfileName, instancePolicyPath);

           var startServerText = await File.ReadAllTextAsync(startupScriptPath);
           var plainTextBytes =
System.Text.Encoding.UTF8.GetBytes(startServerText);

           var amiLatest = await _amazonSsm.GetParameterAsync(
               new GetParameterRequest() { Name = _amiParam });
           var amiId = amiLatest.Parameter.Value;
           var launchTemplateResponse = await
_amazonEc2.CreateLaunchTemplateAsync(
               new CreateLaunchTemplateRequest()
               {
                   LaunchTemplateName = _launchTemplateName,
                   LaunchTemplateData = new RequestLaunchTemplateData()
                   {
                       InstanceType = _instanceType,
                       ImageId = amiId,
                       IamInstanceProfile =
                           new

LaunchTemplateIamInstanceProfileSpecificationRequest()
```

```
                            {
                                Name = _instanceProfileName
                            },
                        KeyName = _keyPairName,
                        UserData = System.Convert.ToBase64String(plainTextBytes)
                    }
                });
            return launchTemplateResponse.LaunchTemplate;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.AlreadyExistsException")
            {
                _logger.LogError($"Could not create the template, the name
{_launchTemplateName} already exists. " +
                                 $"Please try again with a unique name.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while creating the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        try
        {
            var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
                new DescribeAvailabilityZonesRequest());
            return zoneResponse.AvailabilityZones.Select(z =>
z.ZoneName).ToList();
        }
        catch (AmazonEC2Exception ec2Exception)
```

```
        {
            _logger.LogError($"An Amazon EC2 error occurred while listing
 availability zones.: {ec2Exception.Message}");
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while listing availability
 zones.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
 Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
 size {groupSize}.");
        }
```

```
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        try
        {
            var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
                new DescribeVpcsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("is-default", new List<string>() { "true" })
                    }
                });
            return vpcResponse.Vpcs[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "UnauthorizedOperation")
            {
                _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the vpcs.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
```

```
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        try
        {
            var subnets = new List<Subnet>();
            var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
                new DescribeSubnetsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("vpc-id", new List<string>() { vpcId }),
                        new("availability-zone", availabilityZones),
                        new("default-for-az", new List<string>() { "true" })
                    }
                });

            // Get the entire list using the paginator.
            await foreach (var subnet in subnetPaginator.Subnets)
            {
                subnets.Add(subnet);
            }

            return subnets;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
            {
                _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the
subnets.: {ex.Message}");
```

```
            throw;
        }
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.NotFoundException")
            {
                _logger.LogError(
                    $"Could not delete the template, the name
{_launchTemplateName} was not found.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while deleting the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
```

```
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
```

```
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        try
        {
            var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
                new DescribeIamInstanceProfileAssociationsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("instance-id", new List<string>() { instanceId })
                    },
                });
            return response.IamInstanceProfileAssociations[0];
        }
        catch (AmazonEC2Exception ec2Exception)
```

```
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while creating the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        try
        {
            await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
```

```
            // Allow time before resetting.
            Thread.Sleep(25000);

            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(25000);
            var instanceReady = false;
            var retries = 5;
            while (retries-- > 0 && !instanceReady)
            {
                var instancesPaginator =
                    _amazonSsm.Paginators.DescribeInstanceInformation(
                        new DescribeInstanceInformationRequest());
                // Get the entire list using the paginator.
                await foreach (var instance in
instancesPaginator.InstanceInformationList)
                {
                    instanceReady = instance.InstanceId == instanceId;
                    if (instanceReady)
                    {
                        break;
                    }
                }
            }
            Console.WriteLine("Waiting for instance to be running.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);
            Console.WriteLine("Instance ready.");
            Console.WriteLine($"Sending restart command to instance
{instanceId}");
            await _amazonSsm.SendCommandAsync(
                new SendCommandRequest()
                {
                    InstanceIds = new List<string>() { instanceId },
                    DocumentName = "AWS-RunShellScript",
                    Parameters = new Dictionary<string, List<string>>()
                    {
                        {
                            "commands",
                            new List<string>() { "cd / && sudo python3 server.py
80" }
                        }
                    }
                });
```

```
            Console.WriteLine($"Restarted the web server on instance
{instanceId}");
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
```

```
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
```

```
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
```

```
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }
```

```
                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
```

```
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
 InstanceStateName stateName)
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is in the specified state.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
```

```
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEc2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }
}
```

建立包裝 Elastic Load Balancing 動作的類別。

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
 client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
```

```
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }

        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
```

```
    public async Task<List<TargetHealthDescription>>
 CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
 groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
 how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
```

```
   /// <param name="vpcId">The Id of the Vpc in which the load balancer
exists.</param>
   /// <returns>The new TargetGroup object.</returns>
   public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
ProtocolEnum protocol, int port, string vpcId)
   {
       var createResponse = await
_amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
           new CreateTargetGroupRequest()
           {
               Name = groupName,
               Protocol = protocol,
               Port = port,
               HealthCheckPath = "/healthcheck",
               HealthCheckIntervalSeconds = 10,
               HealthCheckTimeoutSeconds = 5,
               HealthyThresholdCount = 2,
               UnhealthyThresholdCount = 2,
               VpcId = vpcId
           });
       var targetGroup = createResponse.TargetGroups[0];
       return targetGroup;
   }

   /// <summary>
   /// Create an Elastic Load Balancing load balancer that uses the specified
subnets
   /// and forwards requests to the specified target group.
   /// </summary>
   /// <param name="name">The name for the new load balancer.</param>
   /// <param name="subnetIds">Subnets for the load balancer.</param>
   /// <param name="targetGroup">Target group for forwarded requests.</param>
   /// <returns>The new LoadBalancer object.</returns>
   public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
List<string> subnetIds, TargetGroup targetGroup)
   {
       var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
           new CreateLoadBalancerRequest()
           {
               Name = name,
               Subnets = subnetIds
           });
       var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;
```

```
        // Wait for load balancer to be available.
        var loadBalancerReady = false;
        while (!loadBalancerReady)
        {
            try
            {
                var describeResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                        new DescribeLoadBalancersRequest()
                        {
                            Names = new List<string>() { name }
                        });

                var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

                loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
            }
            catch (LoadBalancerNotFoundException)
            {
                loadBalancerReady = false;
            }
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
```

```
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
```

```
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }

    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
        {
            try
            {
                var groupResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
```

```
                    });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
 targetArn });
                Console.WriteLine($"Deleted load balancing target group
 {groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

建立使用 DynamoDB 模擬建議服務的類別。

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
```

```
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
```

```
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
_amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
```

```
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

建立包裝 Systems Manager 動作的類別。

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";

    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
```

```
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的下列主題。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup

- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在命令提示中執行互動式案例。

```
public class Main {

    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
```

```
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException,
 InterruptedException {
        Scanner in = new Scanner(System.in);
        Database database = new Database();
        AutoScaler autoScaler = new AutoScaler();
        LoadBalancer loadBalancer = new LoadBalancer();

        System.out.println(DASHES);
        System.out.println("Welcome to the demonstration of How to Build and
 Manage a Resilient Service!");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("A - SETUP THE RESOURCES");
        System.out.println("Press Enter when you're ready to start deploying
 resources.");
        in.nextLine();
        deploy(loadBalancer);
        System.out.println(DASHES);
        System.out.println(DASHES);
```

```
        System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        demo(loadBalancer);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("C - DELETE THE RESOURCES");
        System.out.println("""
                This concludes the demo of how to build and manage a resilient
service.
                To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
                that were created for this demo.
                """);

        System.out.println("\n Do you want to delete the resources (y/n)? ");
        String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

        if (userInput.equals("y")) {
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
    }

    // Deletes the AWS resources used in this example.
    private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
            throws IOException, InterruptedException {
        loadBalancer.deleteLoadBalancer(lbName);
```

```
        System.out.println("*** Wait 30 secs for resource to be deleted");
        TimeUnit.SECONDS.sleep(30);
        loadBalancer.deleteTargetGroup(targetGroupName);
        autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
        autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
        autoScaler.deleteTemplate(templateName);
        database.deleteTable(tableName);
    }

    private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
        Scanner in = new Scanner(System.in);
        System.out.println(
                """
                        For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                        to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                        against various kinds of failures.

                        Some of the resources create by this demo are:
                        \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                        \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                        \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                        \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                        """);

        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
```

```
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
```

```
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
```

```
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
```

```
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        System.out.println(
                """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                        """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        Instead of failing when the recommendation service fails,
the web service can return a static response.
                        While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                        """);
        paramHelper.put(paramHelper.failureResponse, "static");

        System.out.println("""
                Now, sending a GET request to the load balancer endpoint returns
a static response.
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
```

```
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");
```

```
        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
```

```
   }

   public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
       String[] actions = {
               "Send a GET request to the load balancer endpoint.",
               "Check the health of load balancer targets.",
               "Go to the next part of the demo."
       };
       Scanner scanner = new Scanner(System.in);

       while (true) {
           System.out.println("-".repeat(88));
           System.out.println("See the current state of the service by selecting
one of the following choices:");
           for (int i = 0; i < actions.length; i++) {
               System.out.println(i + ": " + actions[i]);
           }

           try {
               System.out.print("\nWhich action would you like to take? ");
               int choice = scanner.nextInt();
               System.out.println("-".repeat(88));

               switch (choice) {
                   case 0 -> {
                       System.out.println("Request:\n");
                       System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                       CloseableHttpClient httpClient =
HttpClients.createDefault();

                       // Create an HTTP GET request to the ELB.
                       HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                       // Execute the request and get the response.
                       HttpResponse response = httpClient.execute(httpGet);
                       int statusCode =
response.getStatusLine().getStatusCode();
                       System.out.println("HTTP Status Code: " + statusCode);

                       // Display the JSON response
                       BufferedReader reader = new BufferedReader(
```

```
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
target.target().id(),
                                    target.target().port(),
target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
```

```
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

建立包裝 Auto Scaling 和 Amazon EC2 動作的類別。

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;

    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
```

```
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }

    /**
     * Replaces the profile associated with a running instance. After the profile
is
     * replaced, the instance is rebooted to ensure that it uses the new profile.
     * When
     * the instance is ready, Systems Manager is used to restart the Python web
     * server.
     */
    public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
```

```
            throws InterruptedException {
        // Create an IAM instance profile specification.
        software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
                .builder()
                .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                              // name.
                .build();

        // Replace the IAM instance profile association for the EC2 instance.
        ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
```

```
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }
```

```
   /**
    * Detaches a role from an instance profile, detaches policies from the role,
    * and deletes all the resources.
    */
   public void deleteInstanceProfile(String roleName, String profileName) {
       try {
           software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                   .builder()
                   .instanceProfileName(profileName)
                   .build();

           GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
           String name = response.instanceProfile().instanceProfileName();
           System.out.println(name);

           RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                   .instanceProfileName(profileName)
                   .roleName(roleName)
                   .build();

           getIAMClient().removeRoleFromInstanceProfile(profileRequest);
           DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                   .instanceProfileName(profileName)
                   .build();

           getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
           System.out.println("Deleted instance profile " + profileName);

           DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           // List attached role policies.
           ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                   .listAttachedRolePolicies(role -> role.roleName(roleName));
           List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
           for (AttachedPolicy attachedPolicy : attachedPolicies) {
```

```
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }

    /*
     * Verify the default security group of the specified VPC allows ingress from
     * this
     * computer. This can be done by allowing ingress from this computer's IP
     * address. In some situations, such as connecting from a corporate network,
you
```

```
    * must instead specify a prefix list ID. You can also temporarily open the
port
    * to
    * any IP address while running this example. If you do, be sure to remove
    * public
    * access when you're done.
    *
    */
   public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
       boolean portIsOpen = false;
       GroupInfo groupInfo = new GroupInfo();
       try {
           Filter filter = Filter.builder()
                   .name("group-name")
                   .values("default")
                   .build();

           Filter filter1 = Filter.builder()
                   .name("vpc-id")
                   .values(VPC)
                   .build();

           DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                   .filters(filter, filter1)
                   .build();

           DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                   .describeSecurityGroups(securityGroupsRequest);
           String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
           groupInfo.setGroupName(securityGroup);

           for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
               System.out.println("Found security group: " +
secGroup.groupId());

               for (IpPermission ipPermission : secGroup.ipPermissions()) {
                   if (ipPermission.fromPort() == port) {
                       System.out.println("Found inbound rule: " +
ipPermission);
                       for (IpRange ipRange : ipPermission.ipRanges()) {
```

```
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

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

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
     * Scaling group.
     * The target group specifies how the load balancer forward requests to the
     * instances
     * in the group.
     */
    public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
        try {
```

```
            AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                    .autoScalingGroupName(asGroupName)
                    .targetGroupARNs(targetGroupARN)
                    .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
            System.out.println("Attached load balancer to " + asGroupName);

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

    // Creates an EC2 Auto Scaling group with the specified size.
    public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

        // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
                .builder()
                .build();

        DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
        List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
                .collect(Collectors.toList());

        String availabilityZones = String.join(",", availabilityZoneNames);
        LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
                .launchTemplateName(templateName)
                .version("$Default")
                .build();

        String[] zones = availabilityZones.split(",");
```

```
        CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
                .launchTemplate(specification)
                .availabilityZones(zones)
                .maxSize(groupSize)
                .minSize(groupSize)
                .autoScalingGroupName(autoScalingGroupName)
                .build();

        try {
            getAutoScalingClient().createAutoScalingGroup(groupRequest);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
```

```
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }
```

```
   // Gets data about the profile associated with an instance.
   public String getInstanceProfile(String instanceId) {
       Filter filter = Filter.builder()
               .name("instance-id")
               .values(instanceId)
               .build();

       DescribeIamInstanceProfileAssociationsRequest associationsRequest =
 DescribeIamInstanceProfileAssociationsRequest
               .builder()
               .filters(filter)
               .build();

       DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
               .describeIamInstanceProfileAssociations(associationsRequest);
       return response.iamInstanceProfileAssociations().get(0).associationId();
   }

   public void deleteRolesPolicies(String policyName, String roleName, String
 InstanceProfile) {
       ListPoliciesRequest listPoliciesRequest =
 ListPoliciesRequest.builder().build();
       ListPoliciesResponse listPoliciesResponse =
 getIAMClient().listPolicies(listPoliciesRequest);
       for (Policy policy : listPoliciesResponse.policies()) {
           if (policy.policyName().equals(policyName)) {
               // List the entities (users, groups, roles) that are attached to
 the policy.

 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
 listEntitiesRequest =
 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                       .builder()
                       .policyArn(policy.arn())
                       .build();
               ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                       .listEntitiesForPolicy(listEntitiesRequest);
               if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                       || !listEntitiesResponse.policyRoles().isEmpty()) {
                   // Detach the policy from any entities it is attached to.
                   DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
```

```
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
```

```
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

建立包裝 Elastic Load Balancing 動作的類別。

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
 targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
 DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();

        DescribeTargetGroupsResponse tgResponse =
 getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
 DescribeTargetHealthRequest.builder()

 .targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
```

```
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);

        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
```

```
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }
```

```
        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
```

```
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");

            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
CreateListenerRequest.builder()

.loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
```

```
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

建立使用 DynamoDB 模擬建議服務的類別。

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
 DescribeTableRequest.builder()
                    .tableName(tableName)
```

```
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
e.getMessage());
        }
        return false;
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
     * hash key named 'MediaType' that defines the type of media recommended,
such
     * as
     * Book or Movie, and a range key named 'ItemId' that, combined with the
     * MediaType,
     * forms a unique identifier for the recommended item.
     */
    public void createTable(String tableName, String fileName) throws IOException
{
        // First check to see if the table exists.
        boolean doesExist = doesTableExist(tableName);
        if (!doesExist) {
            DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
            CreateTableRequest createTableRequest = CreateTableRequest.builder()
                    .tableName(tableName)
                    .attributeDefinitions(
                            AttributeDefinition.builder()
                                    .attributeName("MediaType")
                                    .attributeType(ScalarAttributeType.S)
                                    .build(),
                            AttributeDefinition.builder()
                                    .attributeName("ItemId")
                                    .attributeType(ScalarAttributeType.N)
                                    .build())
                    .keySchema(
                            KeySchemaElement.builder()
                                    .attributeName("MediaType")
```

```
                                .keyType(KeyType.HASH)
                                .build(),
                        KeySchemaElement.builder()
                                .attributeName("ItemId")
                                .keyType(KeyType.RANGE)
                                .build())
                .provisionedThroughput(
                        ProvisionedThroughput.builder()
                                .readCapacityUnits(5L)
                                .writeCapacityUnits(5L)
                                .build())
                .build();

            getDynamoDbClient().createTable(createTableRequest);
            System.out.println("Creating table " + tableName + "...");

            // Wait until the Amazon DynamoDB table is created.
            DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Table " + tableName + " created.");

            // Add records to the table.
            populateTable(fileName, tableName);
        }
    }

    public void deleteTable(String tableName) {
        getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
        System.out.println("Table " + tableName + " deleted.");
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
```

```
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
 enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

建立包裝 Systems Manager 動作的類別。

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }
```

```
    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers

- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

JavaScript

適用於 JavaScript (v3) 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在命令提示中執行互動式案例。

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
```

```
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "node:url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
    name: "Resilient Workflow",
    synopsis:
      "node index.js --scenario <deploy | demo | destroy> [-h|--help] [-y|--yes]
 [-v|--verbose]",
    description: "Deploy and interact with scalable EC2 instances.",
  });
}
```

建立步驟以部署所有資源。

```
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";
```

```
import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

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

```
import { saveState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
```

```
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
```

```
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
```

```
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
```

```
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
```

```
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
```

```
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
```

```
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
```

```
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
```

```
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
  }),
  new ScenarioOutput(
```

```
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
```

```
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      }
      return MESSAGES.noIpRules;
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      }
      return MESSAGES.noIpRules;
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
```

```
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    }
    return false;
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
  saveState,
];
```

建立步驟以執行示範。

```
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";
```

```
import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

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

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
```

```
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );

  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
```

```
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
);
```

```
const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
```

```
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
```

```
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
```

```
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
```

```
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
```

```
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
```

```
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
```

```
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

建立步驟以銷毀所有資源。

```
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
  RevokeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
```

```
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { loadState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  loadState,
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
```

```
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    }
    return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
    return MESSAGES.deletedKeyPair.replace(
      "${KEY_PAIR_NAME}",
      NAMES.keyPairName,
    );
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
```

```
        }),
      );
    }
  } catch (e) {
    state.detachPolicyFromRoleError = e;
  }
}),
new ScenarioOutput("detachedPolicyFromRole", (state) => {
  if (state.detachPolicyFromRoleError) {
    console.error(state.detachPolicyFromRoleError);
    return MESSAGES.detachPolicyFromRoleError
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }
  return MESSAGES.detachedPolicyFromRole
    .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
}),
new ScenarioAction("deleteInstancePolicy", async (state) => {
  const client = new IAMClient({});
  const policy = await findPolicy(NAMES.instancePolicyName);

  if (!policy) {
    state.deletePolicyError = new Error(
      `Policy ${NAMES.instancePolicyName} not found.`,
    );
  } else {
    return client.send(
      new DeletePolicyCommand({
        PolicyArn: policy.Arn,
      }),
    );
  }
}),
new ScenarioOutput("deletePolicyResult", (state) => {
  if (state.deletePolicyError) {
    console.error(state.deletePolicyError);
    return MESSAGES.deletePolicyError.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }
  return MESSAGES.deletedPolicy.replace(
    "${INSTANCE_POLICY_NAME}",
```

```
      NAMES.instancePolicyName,
    );
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.removedRoleFromInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
```

```
        NAMES.instanceRoleName,
      );
    }
    return MESSAGES.deletedInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    );
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
    return MESSAGES.deletedInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
```

```
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
    return MESSAGES.deletedAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
    return MESSAGES.deletedLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    );
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
```

```
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
```

```
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    }
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
    return MESSAGES.detachedSsmOnlyRoleFromProfile
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
```

```
        RoleName: NAMES.ssmOnlyRoleName,
        PolicyArn: ssmOnlyPolicy.Arn,
      }),
    );
  } catch (e) {
    state.detachSsmOnlyCustomRolePolicyError = e;
  }
}),
new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
  if (state.detachSsmOnlyCustomRolePolicyError) {
    console.error(state.detachSsmOnlyCustomRolePolicyError);
    return MESSAGES.detachSsmOnlyCustomRolePolicyError
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
  }
  return MESSAGES.detachedSsmOnlyCustomRolePolicy
    .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
    .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
}),
new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
  try {
    const iamClient = new IAMClient({});
    await iamClient.send(
      new DetachRolePolicyCommand({
        RoleName: NAMES.ssmOnlyRoleName,
        PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
      }),
    );
  } catch (e) {
    state.detachSsmOnlyAWSRolePolicyError = e;
  }
}),
new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
  if (state.detachSsmOnlyAWSRolePolicyError) {
    console.error(state.detachSsmOnlyAWSRolePolicyError);
    return MESSAGES.detachSsmOnlyAWSRolePolicyError
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
  }
  return MESSAGES.detachedSsmOnlyAWSRolePolicy
    .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
    .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
}),
new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
```

```
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
```

```
    }
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
    return MESSAGES.deletedSsmOnlyRole.replace(
      "${ROLE_NAME}",
      NAMES.ssmOnlyRoleName,
    );
  }),
  new ScenarioAction(
    "revokeSecurityGroupIngress",
    async (
      /** @type {{ myIp: string, defaultSecurityGroup: { GroupId: string } }} */
state,
    ) => {
      const ec2Client = new EC2Client({});

      try {
        await ec2Client.send(
          new RevokeSecurityGroupIngressCommand({
            GroupId: state.defaultSecurityGroup.GroupId,
            CidrIp: `${state.myIp}/32`,
```

```
            FromPort: 80,
            ToPort: 80,
            IpProtocol: "tcp",
          }),
        );
      } catch (e) {
        state.revokeSecurityGroupIngressError = e;
      }
    },
  ),
  new ScenarioOutput("revokeSecurityGroupIngressResult", (state) => {
    if (state.revokeSecurityGroupIngressError) {
      console.error(state.revokeSecurityGroupIngressError);
      return MESSAGES.revokeSecurityGroupIngressError.replace(
        "${IP}",
        state.myIp,
      );
    }
    return MESSAGES.revokedSecurityGroupIngress.replace("${IP}", state.myIp);
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
```

```
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    }
    console.log(err.name);
    throw err;
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
```

```
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在命令提示中執行互動式案例。

```
class Runner:
    """
    Manages the deployment, demonstration, and destruction of resources for the
 resilient service.
    """

    def __init__(
        self,
        resource_path: str,
        recommendation: RecommendationService,
        autoscaler: AutoScalingWrapper,
        loadbalancer: ElasticLoadBalancerWrapper,
        param_helper: ParameterHelper,
    ):
        """
        Initializes the Runner class with the necessary parameters.

        :param resource_path: The path to resource files used by this example,
 such as IAM policies and instance scripts.
        :param recommendation: An instance of the RecommendationService class.
        :param autoscaler: An instance of the AutoScaler class.
        :param loadbalancer: An instance of the LoadBalancer class.
        :param param_helper: An instance of the ParameterHelper class.
        """
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
```

```
        self.ssh_port = 22

        prefix = "doc-example-resilience"
        self.target_group_name = f"{prefix}-tg"
        self.load_balancer_name = f"{prefix}-lb"

    def deploy(self) -> None:
        """
        Deploys the resources required for the resilient service, including the
DynamoDB table,
        EC2 instances, Auto Scaling group, and load balancer.
        """
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        logging.info("Starting deployment of resources for the resilient
service.")

        logging.info(
            "Creating and populating DynamoDB table '%s'.",
            self.recommendation.table_name,
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)

        logging.info(
            "Creating an EC2 launch template with the startup script '%s'.",
            startup_script,
        )
        self.autoscaler.create_template(startup_script, instance_policy)

        logging.info(
            "Creating an EC2 Auto Scaling group across multiple Availability
Zones."
        )
        zones = self.autoscaler.create_autoscaling_group(3)

        logging.info("Creating variables that control the flow of the demo.")
        self.param_helper.reset()

        logging.info("Creating Elastic Load Balancing target group and load
balancer.")
```

```
        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.target_group_name, self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            self.load_balancer_name, [subnet["SubnetId"] for subnet in subnets]
        )
        self.loadbalancer.create_listener(self.load_balancer_name, target_group)

        self.autoscaler.attach_load_balancer_target_group(target_group)

        logging.info("Verifying access to the load balancer endpoint.")
        endpoint = self.loadbalancer.get_endpoint(self.load_balancer_name)
        lb_success = self.loadbalancer.verify_load_balancer_endpoint(endpoint)
        current_ip_address = requests.get("http://
checkip.amazonaws.com").text.strip()

        if not lb_success:
            logging.warning(
                "Couldn't connect to the load balancer. Verifying that the port
 is open..."
            )
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                logging.warning(
                    "The default security group for your VPC must allow access
 from this computer."
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
 address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
```

```
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success =
self.loadbalancer.verify_load_balancer_endpoint(endpoint)

        if lb_success:
            logging.info(
                "Load balancer is ready. Access it at: http://%s",
current_ip_address
            )
        else:
            logging.error(
                "Couldn't get a successful response from the load balancer
endpoint. Please verify your VPC and security group settings."
            )

    def demo_choices(self) -> None:
        """
        Presents choices for interacting with the deployed service, such as
sending requests to
        the load balancer or checking the health of the targets.
        """
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            logging.info("Choose an action to interact with the service.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                logging.info("Sending a GET request to the load balancer
endpoint.")
```

```
                endpoint =
self.loadbalancer.get_endpoint(self.load_balancer_name)
                logging.info("GET http://%s", endpoint)
                response = requests.get(f"http://{endpoint}")
                logging.info("Response: %s", response.status_code)
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                logging.info("Checking the health of load balancer targets.")
                health =
self.loadbalancer.check_target_health(self.target_group_name)
                for target in health:
                    state = target["TargetHealth"]["State"]
                    logging.info(
                        "Target %s on port %d is %s",
                        target["Target"]["Id"],
                        target["Target"]["Port"],
                        state,
                    )
                    if state != "healthy":
                        logging.warning(
                            "%s: %s",
                            target["TargetHealth"]["Reason"],
                            target["TargetHealth"]["Description"],
                        )
                logging.info(
                    "Note that it can take a minute or two for the health check
to update."
                )
            elif choice == 2:
                logging.info("Proceeding to the next part of the demo.")

    def demo(self) -> None:
        """
        Runs the demonstration, showing how the service responds to different
failure scenarios
        and how a resilient architecture can keep the service running.
        """
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        logging.info("Resetting parameters to starting values for the demo.")
        self.param_helper.reset()

        logging.info(
```

```
            "Starting demonstration of the service's resilience under various
failure conditions."
        )
        self.demo_choices()

        logging.info(
            "Simulating failure by changing the Systems Manager parameter to a
non-existent table."
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info("Sending GET requests will now return failure codes.")
        self.demo_choices()

        logging.info("Switching to static response mode to mitigate failure.")
        self.param_helper.put(self.param_helper.failure_response, "static")
        logging.info("Sending GET requests will now return static responses.")
        self.demo_choices()

        logging.info("Restoring normal operation of the recommendation service.")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)

        logging.info(
            "Introducing a failure by assigning bad credentials to one of the
instances."
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        logging.info(
            "Replacing instance profile with bad credentials for instance %s.",
            bad_instance_id,
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
```

```
        )
        logging.info(
            "Sending GET requests may return either a valid recommendation or a
static response."
        )
        self.demo_choices()

        logging.info("Implementing deep health checks to detect unhealthy
instances.")
        self.param_helper.put(self.param_helper.health_check, "deep")
        logging.info("Checking the health of the load balancer targets.")
        self.demo_choices()

        logging.info(
            "Terminating the unhealthy instance to let the auto scaler replace
it."
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        logging.info("The service remains resilient during instance
replacement.")
        self.demo_choices()

        logging.info("Simulating a complete failure of the recommendation
service.")
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info(
            "All instances will report as unhealthy, but the service will still
return static responses."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self, automation=False) -> None:
        """
        Destroys all resources created for the demo, including the load balancer,
Auto Scaling group,
        EC2 instances, and DynamoDB table.
        """
        logging.info(
            "This concludes the demo. Preparing to clean up all AWS resources
created during the demo."
        )
        if automation:
            cleanup = True
```

```
        else:
            cleanup = q.ask(
                "Do you want to clean up all demo resources? (y/n) ", q.is_yesno
            )

        if cleanup:
            logging.info("Deleting load balancer and related resources.")
            self.loadbalancer.delete_load_balancer(self.load_balancer_name)
            self.loadbalancer.delete_target_group(self.target_group_name)
            self.autoscaler.delete_autoscaling_group(self.autoscaler.group_name)
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            logging.info("Deleting DynamoDB table and other resources.")
            self.recommendation.destroy()
        else:
            logging.warning(
                "Resources have not been deleted. Ensure you clean them up
 manually to avoid unexpected charges."
            )


def main() -> None:
    """
    Main function to parse arguments and run the appropriate actions for the
 demo.
    """
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../scenarios/features/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
 policies and\n"
```

```
        "instance scripts.",
    )
    args = parser.parse_args()

    logging.info("Starting the Resilient Service demo.")

    prefix = "doc-example-resilience"

    # Service Clients
    ddb_client = boto3.client("dynamodb")
    elb_client = boto3.client("elbv2")
    autoscaling_client = boto3.client("autoscaling")
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")

    # Wrapper instantiations
    recommendation = RecommendationService(
        "doc-example-recommendation-service", ddb_client
    )
    autoscaling_wrapper = AutoScalingWrapper(
        prefix,
        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    )
    elb_wrapper = ElasticLoadBalancerWrapper(elb_client)
    param_helper = ParameterHelper(recommendation.table_name, ssm_client)

    # Demo invocation
    runner = Runner(
        args.resource_path,
        recommendation,
        autoscaling_wrapper,
        elb_wrapper,
        param_helper,
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
"destroy"]
    for action in actions:
        if action == "deploy":
```

```
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    logging.info("Demo completed successfully.")


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

建立包裝 Auto Scaling 和 Amazon EC2 動作的類別。

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
```

```
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_policy(self, policy_file: str, policy_name: str) -> str:
        """
        Creates a new IAM policy or retrieves the ARN of an existing policy.

        :param policy_file: The path to a JSON file that contains the policy
definition.
        :param policy_name: The name to give the created policy.
        :return: The ARN of the created or existing policy.
        """
        with open(policy_file) as file:
            policy_doc = file.read()

        try:
            response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=policy_doc
            )
            policy_arn = response["Policy"]["Arn"]
            log.info(f"Policy '{policy_name}' created successfully. ARN:
{policy_arn}")
```

```
            return policy_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the policy already exists, get its ARN
                response = self.iam_client.get_policy(
                    PolicyArn=f"arn:aws:iam::{self.account_id}:policy/
{policy_name}"
                )
                policy_arn = response["Policy"]["Arn"]
                log.info(f"Policy '{policy_name}' already exists. ARN:
 {policy_arn}")
                return policy_arn
            log.error(f"Full error:\n\t{err}")

    def create_role(self, role_name: str, assume_role_doc: dict) -> str:
        """
        Creates a new IAM role or retrieves the ARN of an existing role.

        :param role_name: The name to give the created role.
        :param assume_role_doc: The assume role policy document that specifies
 which
                                entities can assume the role.
        :return: The ARN of the created or existing role.
        """
        try:
            response = self.iam_client.create_role(
                RoleName=role_name,
 AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            role_arn = response["Role"]["Arn"]
            log.info(f"Role '{role_name}' created successfully. ARN: {role_arn}")
            return role_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the role already exists, get its ARN
                response = self.iam_client.get_role(RoleName=role_name)
                role_arn = response["Role"]["Arn"]
                log.info(f"Role '{role_name}' already exists. ARN: {role_arn}")
                return role_arn
            log.error(f"Full error:\n\t{err}")

    def attach_policy(
```

```
        self,
        role_name: str,
        policy_arn: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> None:
        """
        Attaches an IAM policy to a role and optionally attaches additional AWS-
managed policies.

        :param role_name: The name of the role to attach the policy to.
        :param policy_arn: The ARN of the policy to attach.
        :param aws_managed_policies: A tuple of AWS-managed policy names to
 attach to the role.
        """
        try:
            self.iam_client.attach_role_policy(RoleName=role_name,
 PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info(f"Attached policy {policy_arn} to role {role_name}.")
        except ClientError as err:
            log.error(f"Failed to attach policy {policy_arn} to role
 {role_name}.")
            log.error(f"Full error:\n\t{err}")

    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
 created by
        this class. An instance's associated profile defines a role that is
 assumed by the
        instance. The role has attached policies that specify the AWS permissions
 granted to
        clients that run on the instance.
```

```
        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
```

```
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn


    def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
            if not response["IamInstanceProfileAssociations"]:
                log.info(f"No instance profile found for instance
{instance_id}.")
            profile_data = response["IamInstanceProfileAssociations"][0]
            log.info(f"Retrieved instance profile for instance {instance_id}.")
            return profile_data
        except ClientError as err:
            log.error(
                f"Failed to retrieve instance profile for instance
{instance_id}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidInstanceID.NotFound":
                log.error(f"The instance ID '{instance_id}' does not exist.")
            log.error(f"Full error:\n\t{err}")


    def replace_instance_profile(
        self,
        instance_id: str,
```

```
        new_instance_profile_name: str,
        profile_association_id: str,
    ) -> None:
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to restart.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)

            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info("Rebooting instance %s.", instance_id)
            waiter = self.ec2_client.get_waiter("instance_running")
            log.info("Waiting for instance %s to be running.", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info("Instance %s is now running.", instance_id)

            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
```

```
            log.info(f"Restarted the Python web server on instance
'{instance_id}'.")
        except ClientError as err:
            log.error("Failed to replace instance profile.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidAssociationID.NotFound":
                log.error(
                    f"Association ID '{profile_association_id}' does not exist."
                    "Please check the association ID and try again."
                )
            if error_code == "InvalidInstanceId":
                log.error(
                    f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
                    f"Please verify the instance ID and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
```

```
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )


    def create_key_pair(self, key_pair_name: str) -> None:
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to create key pair {key_pair_name}.")
            if error_code == "InvalidKeyPair.Duplicate":
                log.error(f"A key pair with the name '{key_pair_name}' already
exists.")
            log.error(f"Full error:\n\t{err}")


    def delete_key_pair(self) -> None:
        """
        Deletes a key pair.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
```

```
        except ClientError as err:
            log.error(f"Couldn't delete key pair '{self.key_pair_name}'.")
            log.error(f"Full error:\n\t{err}")
        except FileNotFoundError as err:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
            log.error(f"Full error:\n\t{err}")


    def create_template(
        self, server_startup_script_file: str, instance_policy_file: str
    ) -> Dict[str, Any]:
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            # Create key pair and instance profile
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )

            # Read the startup script
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
```

```
            # Get the latest AMI ID
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]

            # Create the launch template
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
            )
        except ClientError as err:
            log.error(f"Failed to create launch template
{self.launch_template_name}.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
                log.info(
                    f"Launch template {self.launch_template_name} already exists,
nothing to do."
                )
            log.error(f"Full error:\n\t{err}")
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
```

```
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            log.error(f"Full error:\n\t{err}")


    def get_availability_zones(self) -> List[str]:
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
            log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
        except ClientError as err:
            log.error("Failed to retrieve availability zones.")
            log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def create_autoscaling_group(self, group_size: int) -> List[str]:
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        try:
```

```
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
                f"Created EC2 Auto Scaling group {self.group_name} with
availability zones {zones}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            if error_code == "AlreadyExists":
                log.info(
                    f"EC2 Auto Scaling group {self.group_name} already exists,
nothing to do."
                )
            else:
                log.error(f"Failed to create EC2 Auto Scaling group
{self.group_name}.")
                log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def get_instances(self) -> List[str]:
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: A list of instance IDs in the Auto Scaling group.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
```

```
            log.info(
                f"Retrieved {len(instance_ids)} instances for Auto Scaling group
{self.group_name}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to retrieve instances for Auto Scaling group
{self.group_name}."
            )
            if error_code == "ResourceNotFound":
                log.error(f"The Auto Scaling group '{self.group_name}' does not
exist.")
            log.error(f"Full error:\n\t{err}")
        else:
            return instance_ids


    def terminate_instance(self, instance_id: str, decrementsetting=False) ->
None:
        """
        Terminates an instance in an EC2 Auto Scaling group. After an instance is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        :param decrementsetting: If True, do not replace terminated instances.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id,
                ShouldDecrementDesiredCapacity=decrementsetting,
            )
            log.info("Terminated instance %s.", instance_id)

            # Adding a waiter to ensure the instance is terminated
            waiter = self.ec2_client.get_waiter("instance_terminated")
            log.info("Waiting for instance %s to be terminated...", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info(
                f"Instance '{instance_id}' has been terminated and will be
replaced."
            )

        except ClientError as err:
```

```
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to terminate instance '{instance_id}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to terminate the instance again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            log.error(f"Full error:\n\t{err}")

    def attach_load_balancer_target_group(
        self, lb_target_group: Dict[str, Any]
    ) -> None:
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forwards requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
```

```
            )
            if error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
 the resource."
                )
            elif error_code == "ServiceLinkedRoleFailure":
                log.error(
                    "The operation failed because the service-linked role is not
 ready or does not exist. "
                    "Check that the service-linked role exists and is correctly
 configured."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_autoscaling_group(self, group_name: str) -> None:
        """
        Terminates all instances in the group, then deletes the EC2 Auto Scaling
 group.

        :param group_name: The name of the group to delete.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self.terminate_instance(inst_id)

                # Wait for all instances to be terminated
                if instance_ids:
                    waiter = self.ec2_client.get_waiter("instance_terminated")
                    log.info("Waiting for all instances to be terminated...")
                    waiter.wait(InstanceIds=instance_ids)
                    log.info("All instances have been terminated.")
```

```
            else:
                log.info(f"No groups found named '{group_name}'! Nothing to do.")
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to delete the group again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_default_vpc(self) -> Dict[str, Any]:
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error("Failed to retrieve the default VPC.")
            if error_code == "UnauthorizedOperation":
                log.error(
                    "You do not have the necessary permissions to describe VPCs.
"
                    "Ensure that your AWS IAM user or role has the correct
permissions."
                )
            elif error_code == "InvalidParameterValue":
                log.error(
                    "One or more parameters are invalid. Check the request
parameters."
```

```
                )

            log.error(f"Full error:\n\t{err}")
        else:
            if "Vpcs" in response and response["Vpcs"]:
                log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
                return response["Vpcs"][0]
            else:
                pass


    def verify_inbound_port(
        self, vpc: Dict[str, Any], port: int, ip_address: str
    ) -> Tuple[Dict[str, Any], bool]:
        """
        Verify the default security group of the specified VPC allows ingress
 from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
 you
        must instead specify a prefix list ID. You can also temporarily open the
 port to
        any IP address while running this example. If you do, be sure to remove
 public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
        :return: The default security group of the specified VPC, and a value
 that indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info(f"Found default security group {sec_group['GroupId']}.")
```

```
            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info(f"Found inbound rule: {ip_perm}")
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
                    if not port_is_open:
                        log.info(
                            f"The inbound rule does not appear to be open to
either this computer's IP "
                            f"address of {ip_address}, to all IP addresses
(0.0.0.0/0), or to a prefix list ID."
                        )
                    else:
                        break
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to verify inbound rule for port {port} for VPC
{vpc['VpcId']}."
            )
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    f"The specified VPC ID '{vpc['VpcId']}' does not exist.
Please check the VPC ID."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id: str, port: int, ip_address: str) ->
None:
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
```

```
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to authorize ingress to security group '{sec_group_id}'
on port {port} from {ip_address}."
            )
            if error_code == "InvalidGroupId.Malformed":
                log.error(
                    "The security group ID is malformed. "
                    "Please verify that the security group ID is correct."
                )
            elif error_code == "InvalidPermission.Duplicate":
                log.error(
                    "The specified rule already exists in the security group. "
                    "Check the existing rules for this security group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
```

```
        """
        # Ensure that 'zones' is a list, even if None is passed
        if zones is None:
            zones = []
        try:
            paginator = self.ec2_client.get_paginator("describe_subnets")
            page_iterator = paginator.paginate(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )

            subnets = []
            for page in page_iterator:
                subnets.extend(page["Subnets"])

            log.info("Found %s subnets for the specified zones.", len(subnets))
            return subnets
        except ClientError as err:
            log.error(
                f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
{zones}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    "The specified VPC ID does not exist. "
                    "Please check the VPC ID and try again."
                )
            # Add more error-specific handling as needed
            log.error(f"Full error:\n\t{err}")
```

建立包裝 Elastic Load Balancing 動作的類別。

```
class ElasticLoadBalancerWrapper:
    """Encapsulates Elastic Load Balancing (ELB) actions."""
```

```
    def __init__(self, elb_client: boto3.client):
        """
        Initializes the LoadBalancer class with the necessary parameters.
        """
        self.elb_client = elb_client


    def create_target_group(
        self, target_group_name: str, protocol: str, port: int, vpc_id: str
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing target group. The target group
specifies how
        the load balancer forwards requests to instances in the group and how
instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
times and
        lower thresholds. In production, you might want to decrease the
sensitivity of
        your health checks to avoid unwanted failures.

        :param target_group_name: The name of the target group to create.
        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info(f"Created load balancing target group
'{target_group_name}'.")
```

```
            return target_group
        except ClientError as err:
            log.error(
                f"Couldn't create load balancing target group
'{target_group_name}'."
            )
            error_code = err.response["Error"]["Code"]

            if error_code == "DuplicateTargetGroupName":
                log.error(
                    f"Target group name {target_group_name} already exists. "
                    "Check if the target group already exists."
                    "Consider using a different name or deleting the existing
target group if appropriate."
                )
            elif error_code == "TooManyTargetGroups":
                log.error(
                    "Too many target groups exist in the account. "
                    "Consider deleting unused target groups to create space for
new ones."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_target_group(self, target_group_name) -> None:
        """
        Deletes the target group.
        """
        try:
            # Describe the target group to get its ARN
            response =
self.elb_client.describe_target_groups(Names=[target_group_name])
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]

            # Delete the target group
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
            log.info("Deleted load balancing target group %s.",
target_group_name)

            # Use a custom waiter to wait until the target group is no longer
available
            self.wait_for_target_group_deletion(self.elb_client, tg_arn)
            log.info("Target group %s successfully deleted.", target_group_name)
```

```
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete target group '{target_group_name}'.")
            if error_code == "TargetGroupNotFound":
                log.error(
                    "Load balancer target group either already deleted or never
existed. "
                    "Verify the name and check that the resource exists in the
AWS Console."
                )
            elif error_code == "ResourceInUseException":
                log.error(
                    "Target group still in use by another resource. "
                    "Ensure that the target group is no longer associated with
any load balancers or resources.",
                )
            log.error(f"Full error:\n\t{err}")

    def wait_for_target_group_deletion(
        self, elb_client, target_group_arn, max_attempts=10, delay=30
    ):
        for attempt in range(max_attempts):
            try:

elb_client.describe_target_groups(TargetGroupArns=[target_group_arn])
                print(
                    f"Attempt {attempt + 1}: Target group {target_group_arn}
still exists."
                )
            except ClientError as e:
                if e.response["Error"]["Code"] == "TargetGroupNotFound":
                    print(
                        f"Target group {target_group_arn} has been successfully
deleted."
                    )
                    return
                else:
                    raise
            time.sleep(delay)
        raise TimeoutError(
            f"Target group {target_group_arn} was not deleted after {max_attempts
* delay} seconds."
        )
```

```
   def create_load_balancer(
       self,
       load_balancer_name: str,
       subnet_ids: List[str],
   ) -> Dict[str, Any]:
       """
       Creates an Elastic Load Balancing load balancer that uses the specified
subnets
       and forwards requests to the specified target group.

       :param load_balancer_name: The name of the load balancer to create.
       :param subnet_ids: A list of subnets to associate with the load balancer.
       :return: Data about the newly created load balancer.
       """
       try:
           response = self.elb_client.create_load_balancer(
               Name=load_balancer_name, Subnets=subnet_ids
           )
           load_balancer = response["LoadBalancers"][0]
           log.info(f"Created load balancer '{load_balancer_name}'.")

           waiter = self.elb_client.get_waiter("load_balancer_available")
           log.info(
               f"Waiting for load balancer '{load_balancer_name}' to be
available..."
           )
           waiter.wait(Names=[load_balancer_name])
           log.info(f"Load balancer '{load_balancer_name}' is now available!")

       except ClientError as err:
           error_code = err.response["Error"]["Code"]
           log.error(
               f"Failed to create load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
           )

           if error_code == "DuplicateLoadBalancerNameException":
               log.error(
                   f"A load balancer with the name '{load_balancer_name}'
already exists. "
                   "Load balancer names must be unique within the AWS region. "
                   "Please choose a different name and try again."
               )
```

```
            if error_code == "TooManyLoadBalancersException":
                log.error(
                    "The maximum number of load balancers has been reached in
this account and region. "
                    "You can delete unused load balancers or request an increase
in the service quota from AWS Support."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return load_balancer


    def create_listener(
        self,
        load_balancer_name: str,
        target_group: Dict[str, Any],
    ) -> Dict[str, Any]:
        """
        Creates a listener for the specified load balancer that forwards requests
to the
        specified target group.

        :param load_balancer_name: The name of the load balancer to create a
listener for.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created listener.
        """
        try:
            # Retrieve the load balancer ARN
            load_balancer_response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            load_balancer_arn = load_balancer_response["LoadBalancers"][0][
                "LoadBalancerArn"
            ]

            # Create the listener
            response = self.elb_client.create_listener(
                LoadBalancerArn=load_balancer_arn,
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
```

```
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                f"Created listener to forward traffic from load balancer
'{load_balancer_name}' to target group '{target_group['TargetGroupName']}'."
            )
            return response["Listeners"][0]
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to add a listener on '{load_balancer_name}' for target
group '{target_group['TargetGroupName']}'."
            )

            if error_code == "ListenerNotFoundException":
                log.error(
                    f"The listener could not be found for the load balancer
'{load_balancer_name}'. "
                    "Please check the load balancer name and target group
configuration."
                )
            if error_code == "InvalidConfigurationRequestException":
                log.error(
                    f"The configuration provided for the listener on load
balancer '{load_balancer_name}' is invalid. "
                    "Please review the provided protocol, port, and target group
settings."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_load_balancer(self, load_balancer_name) -> None:
        """
        Deletes a load balancer.

        :param load_balancer_name: The name of the load balancer to delete.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
```

```
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[load_balancer_name])
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Couldn't delete load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    f"The load balancer '{load_balancer_name}' does not exist. "
                    "Please check the name and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def get_endpoint(self, load_balancer_name) -> str:
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            return response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            log.error(
                f"Couldn't get the endpoint for load balancer
{load_balancer_name}"
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Verify load balancer name and ensure it exists in the AWS
console."
                )
```

```
            log.error(f"Full error:\n\t{err}")

    @staticmethod
    def verify_load_balancer_endpoint(endpoint) -> bool:
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.

        :param endpoint: The endpoint to verify.
        :return: True if the GET request is successful, False otherwise.
        """
        retries = 3
        verified = False
        while not verified and retries > 0:
            try:
                lb_response = requests.get(f"http://{endpoint}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    verified = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
                )
                retries -= 1
                time.sleep(10)
        return verified

    def check_target_health(self, target_group_name: str) -> List[Dict[str,
Any]]:
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[target_group_name]
            )
```

```
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            log.error(f"Couldn't check health of {target_group_name} target(s).")
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Load balancer associated with the target group was not
found. "
                    "Ensure the load balancer exists, is in the correct AWS
region, and "
                    "that you have the necessary permissions to access it.",
                )
            elif error_code == "TargetGroupNotFoundException":
                log.error(
                    "Target group was not found. "
                    "Verify the target group name, check that it exists in the
correct region, "
                    "and ensure it has not been deleted or created in a different
account.",
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return health_response["TargetHealthDescriptions"]
```

建立使用 DynamoDB 模擬建議服務的類別。

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name: str, dynamodb_client: boto3.client):
        """
        Initializes the RecommendationService class with the necessary
 parameters.
```

```
        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    def create(self) -> Dict[str, Any]:
        """
        Creates a DynamoDB table to use as a recommendation service. The table
has a
        hash key named 'MediaType' that defines the type of media recommended,
such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        :raises RecommendationServiceError: If the table creation fails.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be done.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
```

```
                )
        else:
            return response

    def populate(self, data_file: str) -> None:
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        :raises RecommendationServiceError: If the table population fails.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self) -> None:
        """
        Deletes the recommendations table.

        :raises RecommendationServiceError: If the table deletion fails.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
self.table_name)
            else:
```

```
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

建立包裝 Systems Manager 動作的類別。

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table: str = "doc-example-resilient-architecture-table"
    failure_response: str = "doc-example-resilient-architecture-failure-response"
    health_check: str = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name: str, ssm_client: boto3.client):
        """
        Initializes the ParameterHelper class with the necessary parameters.

        :param table_name: The name of the DynamoDB table that is used as a
 recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    def reset(self) -> None:
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
 a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
```

```
        self.put(self.health_check, "shallow")

    def put(self, name: str, value: str) -> None:
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        :raises ParameterHelperError: If the parameter value cannot be set.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting parameter %s to '%s'.", name, value)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to set parameter {name}.")
            if error_code == "ParameterLimitExceeded":
                log.error(
                    "The parameter limit has been exceeded. "
                    "Consider deleting unused parameters or request a limit
increase."
                )
            elif error_code == "ParameterAlreadyExists":
                log.error(
                    "The parameter already exists and overwrite is set to False.
"
                    "Use Overwrite=True to update the parameter."
                )
            log.error(f"Full error:\n\t{err}")
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的下列主題。
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate

- CreateListener
- CreateLoadBalancer
- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

設定容器服務連線

以下程式碼範例顯示做法：

- 建立 VPC 基礎設施
- 設定記錄
- 建立 ECS 叢集

- 設定 IAM 角色
- 使用 Service Connect 建立服務
- 驗證部署
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)
```

```
# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
```

```
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
```

```
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"

    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true
```

```
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
```

```
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
```

```
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
```

```
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."

    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF
```

```
    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"

    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
```

```
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
```

```
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"

    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
```

```
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi

    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "============================================="
    echo "CREATED RESOURCES"
    echo "============================================="
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "============================================="
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
```

```
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
```

```
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"
```

```
    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups

    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources

    # Ask user if they want to clean up
    echo ""
    echo "============================================"
    echo "CLEANUP CONFIRMATION"
    echo "============================================"
    echo "Do you want to clean up all created resources? (y/n): "
    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
cleanup function."
        echo ""
```

```
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
main "$@"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteSecurityGroup
  - DeleteService (AWS Cloud Map)
  - DeleteService (Amazon ECS)
  - DeregisterTaskDefinition
  - DescribeClusters
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - DetachRolePolicy

- GetCallerIdentity
- GetRole
- ListNamespaces
- ListServices
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- 等待 (Amazon EC2)
- 等待 (Amazon ECS)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

為無伺服器啟動類型建立容器任務

以下程式碼範例顯示做法：

- 建立叢集
- 建立任務定義
- 建立 服務
- 清除

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash
```

```
# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error

# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
execute_command() {
    local cmd="$1"
    local description="$2"

    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "================================================"
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "================================================"

    local output
    local exit_code
```

```
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
        echo "OUTPUT: $output"
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
        echo "Output: $output"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
```

```
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
            echo "No network interfaces found using security group
 $security_group_id"
            return 0
        else
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}
```

```
# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "============================================="
```

```
   echo "CLEANUP PROCESS"
   echo "========================================="
   echo "The following resources were created:"
   for resource in "${CREATED_RESOURCES[@]}"; do
       echo "  - $resource"
   done
   echo ""
   echo "Auto-confirming cleanup of all created resources..."

   CLEANUP_CHOICE="y"

   if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
       echo "Starting cleanup process..."

       # Step 1: Scale service to 0 tasks first, then delete service
       if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
then
           echo ""
           echo "Step 1: Scaling service to 0 tasks..."
           if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
               echo "Waiting for service to stabilize after scaling to 0..."
               execute_command "aws ecs wait services-stable --cluster
'$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"

               echo "Deleting service..."
               execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME'" "Delete ECS service"
           else
               echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
               execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
           fi
       fi

       # Step 2: Wait a bit for tasks to fully terminate
       echo ""
       echo "Step 2: Waiting for tasks to fully terminate..."
       sleep 15

       # Step 3: Delete cluster
       if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
```

```
            echo ""
            echo "Step 3: Deleting cluster..."
            execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
 "Delete ECS cluster"
        fi

        # Step 4: Wait for network interfaces to be cleaned up, then delete
 security group
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."

            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."

            # Get all revisions of the task definition
            local revisions
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "========================================="
```

```
        echo "CLEANUP COMPLETED"
        echo "========================================="
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
        echo ""
        echo "To clean up manually later, use the following commands in order:"
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
}

# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "========================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "========================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
```

```
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"
```

```
    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi

# Step 2: Create ECS cluster
echo ""
echo "============================================="
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================="

CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "============================================="
echo "STEP 3: CREATE TASK DEFINITION"
echo "============================================="

# Create task definition JSON with validated inputs
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
```

```
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

# Validate JSON before using
if ! jq empty task-definition.json 2>/dev/null; then
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "============================================="
```

```
echo "STEP 4: SET UP NETWORKING"
echo "========================================="

# Get default VPC ID
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi

# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi
```

```
echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "=========================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "=========================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"
```

```
echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"

CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "============================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "============================================="

echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi

echo "Task ARN: $TASK_ARN"

# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"
```

```
# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
    else
        echo ""
        echo "=========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "=========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
echo "=========================================="
echo "SERVICE INFORMATION"
echo "=========================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "=========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "=========================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi
```

```
echo ""
echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeNetworkInterfaces
  - DescribeSecurityGroups
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - GetCallerIdentity
  - GetRole
  - ListTaskDefinitions
  - ListTasks
  - RegisterTaskDefinition
  - UpdateService
  - 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用函數代理整合建立靜態 API

以下程式碼範例顯示做法：

- 建立用於 Lambda 執行的 IAM 角色
- 建立和部署 Lambda 函數
- 建立 REST API
- 設定 Lambda 代理整合
- 部署和測試 API
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
```

```
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."

# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
```

```
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")

        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
```

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration

# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
```

```
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."

# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
```

```
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}

# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
```

```
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}

echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"

# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."
```

```
# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2

# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2

aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddPermission
  - AttachRolePolicy
  - CreateDeployment
  - CreateFunction
  - CreateResource
  - CreateRestApi
  - CreateRole
  - DeleteFunction
  - DeleteRestApi
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetResources
  - GetRestApis

- PutIntegration
- PutMethod

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 建立唯讀和讀寫 IAM 使用者

下列程式碼範例示範如何建立使用者並為它們連接政策。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

- 建立兩個 IAM 使用者。
- 對一位使用者連接政策，並將物件放在 Amazon S3 儲存貯體中。
- 為第二位使用者連接政策，以便從儲存貯體取得物件。
- 依據使用者憑證取得儲存貯體的不同許可。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 使用者動作的函數。

```
import logging
import time

import boto3
from botocore.exceptions import ClientError
```

```
import access_key_wrapper
import policy_wrapper

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user



def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
```

```
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise



def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

建立可包裝 IAM 政策動作的函數。

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.
```

```
    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

建立可包裝 IAM 存取金鑰動作的函數。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
```

```
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

使用包裝函數建立具有不同政策的使用者，並使用其憑證存取 Amazon S3 儲存貯體。

```
def usage_demo():
    """
    Shows how to manage users, keys, and policies.
    This demonstration creates two users: one user who can put and get objects in
 an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
 permitted
    to perform.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
```

```
    print(f"Created two IAM users: {user_read_writer.name} and
{user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
    )
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
```

```
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )

    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
```

```
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
    detach_policy(user_reader.name, read_policy.arn)
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的下列主題。
  - AttachUserPolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteUser
  - DetachUserPolicy
  - ListUsers
  - UpdateUser

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

為虛擬機器執行個體建立容器服務

以下程式碼範例顯示做法：

- 建立 ECS 叢集
- 建立和監控服務
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")

# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"
```

```
    echo ""
    echo "============================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "============================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}

# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi

        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
```

```
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Deregister task definition
    if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
        log "Deregistering task definition: $TASK_DEFINITION_ARN"
        if ! aws ecs deregister-task-definition --task-definition
 "$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to deregister task definition
 $TASK_DEFINITION_ARN"
            ((cleanup_errors++))
        fi
    fi

    # Terminate EC2 instance
    if [[ -n "${INSTANCE_ID:-}" ]]; then
        log "Terminating EC2 instance: $INSTANCE_ID"
        if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
 2>>"$LOG_FILE"; then
            log "WARNING: Failed to terminate instance $INSTANCE_ID"
            ((cleanup_errors++))
        else
            log "Waiting for instance to terminate..."
            if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
 2>>"$LOG_FILE"; then
                log "WARNING: Instance termination wait failed for $INSTANCE_ID"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Delete security group with retry logic
```

```
    if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
        log "Deleting security group: $SECURITY_GROUP_ID"
        local retry_count=0
        local max_retries=3

        while [[ $retry_count -lt $max_retries ]]; do
            if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
                log "Successfully deleted security group"
                break
            else
                ((retry_count++))
                if [[ $retry_count -lt $max_retries ]]; then
                    log "Retry $retry_count/$max_retries: Waiting 10 seconds
before retrying security group deletion..."
                    sleep 10
                else
                    log "ERROR: Failed to delete security group after
$max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
```

```
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"

    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
```

```
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
    log "Creating ECS cluster: $CLUSTER_NAME"

    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}

# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")
```

```
    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)

    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
```

```
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."

        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
```

```
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole

        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."

    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
```

```
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
```

```
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
        exit 1
    fi

    rm -f ecs-user-data.sh
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
```

```
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi

    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi
```

```
    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"

    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}
```

```
# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun

    # List tasks
    log "Listing tasks in service:"
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table

    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
```

```
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "=========================================="
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "=========================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
```

```
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "=========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "=========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "=========================================="
    echo "CLEANUP CONFIRMATION"
    echo "=========================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
}

# Run main function
main "$@"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile

- CreateKeyPair
- CreateRole
- CreateSecurityGroup
- CreateService
- DeleteCluster
- DeleteKeyPair
- DeleteSecurityGroup
- DeleteService
- DeregisterTaskDefinition
- DescribeContainerInstances
- DescribeInstances
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetParameters
- GetRole
- ListContainerInstances
- ListServices
- ListTasks
- RegisterTaskDefinition
- RunInstances
- StopTask
- TerminateInstances
- UpdateService
- 等待 (Amazon EC2)
- 等待 (Amazon ECS)
- 等待 (IAM)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立受管監控工作區

以下程式碼範例顯示做法：

- 為您的工作區建立 IAM 角色
- 建立 Grafana 工作區
- 設定身分驗證
- 設定選用設定
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it

# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2
```

```
    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."

    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
    fi

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"
```

```
echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."

# Create trust policy document
cat > trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://trust-policy.json \
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
```

```
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"

ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
```

```
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)

if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
```

```
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi

# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "==========================================="
echo "WORKSPACE INFORMATION"
echo "==========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "==========================================="

# Step 6: Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
```

```
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
 grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done

    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
 still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
```

```
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace
  - DescribeWorkspace
  - DetachRolePolicy
  - GetCallerIdentity

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

以下程式碼範例顯示做法：

- 建立 CloudWatch 儀表板

-
-
- 清除資源

Bash

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "$(date): Starting CloudWatch dashboard creation script"

# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "============================================="
    echo "CLEANUP CONFIRMATION"
    echo "============================================="
    echo "An error occurred. Proceeding with automatic cleanup..."
```

```
    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true

    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi

# Get the current region securely
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")
```

```
if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."

    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi

    if ! zip -q function.zip index.js; then
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
```

```
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null
```

```
    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
```

```
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
```

```
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi
```

```
# Verify the dashboard was created
echo "Verifying dashboard creation..."
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
```

```
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
```

```
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi

    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards

- DeleteFunction
- DeleteRole
- DetachRolePolicy
- GetCallerIdentity
- GetDashboard
- Invoke
- ListDashboards
- ListFunctions
- PutDashboard

此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立您的第一個無伺服器函數

以下程式碼範例顯示做法：

- 建立 Lambda 的 IAM 角色
- 建立函數程式碼
- 建立 Lambda 函式
- 測試您的 Lambda 函數
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash
# AWS Lambda - Create Your First Function
# This script creates a Lambda function, invokes it with a test event,
```

```
# views CloudWatch logs, and cleans up all resources.
#
# Source: https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html
#
# Resources created:
#   - IAM role (Lambda execution role with basic logging permissions)
#   - Lambda function (Python 3.13 or Node.js 22.x runtime)
#   - CloudWatch log group (created automatically by Lambda on invocation)

set -eE -o pipefail

################################################################################
# Setup
################################################################################

UNIQUE_ID=$(head -c 8 /dev/urandom | od -An -tx1 | tr -d ' ')
FUNCTION_NAME="my-lambda-function-${UNIQUE_ID}"
ROLE_NAME="lambda-execution-role-${UNIQUE_ID}"
LOG_GROUP_NAME="/aws/lambda/${FUNCTION_NAME}"

TEMP_DIR=$(mktemp -d)
readonly TEMP_DIR
LOG_FILE="${TEMP_DIR}/lambda-gettingstarted.log"

exec > >(tee -a "$LOG_FILE") 2>&1

declare -a CREATED_RESOURCES

################################################################################
# Helper functions
################################################################################

cleanup_resources() {
    # Disable error trap to prevent recursion during cleanup
    trap - ERR
    set +eE

    echo ""
    echo "Cleaning up resources..."
    echo ""

    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        local RESOURCE="${CREATED_RESOURCES[$i]}"
        local TYPE="${RESOURCE%%:*}"
```

```
        local NAME="${RESOURCE#*:}"

        case "$TYPE" in
            log-group)
                echo "Deleting CloudWatch log group: ${NAME}"
                aws logs delete-log-group \
                    --log-group-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete log group ${NAME}."
                ;;
            lambda-function)
                echo "Deleting Lambda function: ${NAME}"
                aws lambda delete-function \
                    --function-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete Lambda function ${NAME}."
                echo "  Waiting for function deletion to complete..."
                local DELETE_WAIT=0
                while aws lambda get-function --function-name "$NAME" > /dev/null
2>&1; do
                    sleep 2
                    DELETE_WAIT=$((DELETE_WAIT + 2))
                    if [ "$DELETE_WAIT" -ge 60 ]; then
                        echo "  WARNING: Timed out waiting for function
deletion."
                        break
                    fi
                done
                ;;
            iam-role-policy)
                local ROLE_PART="${NAME%%|*}"
                local POLICY_PART="${NAME#*|}"
                echo "Detaching policy from role: ${ROLE_PART}"
                aws iam detach-role-policy \
                    --role-name "$ROLE_PART" \
                    --policy-arn "$POLICY_PART" 2>&1 || echo "  WARNING: Could
not detach policy from role ${ROLE_PART}."
                ;;
            iam-role)
                echo "Deleting IAM role: ${NAME}"
                aws iam delete-role \
                    --role-name "$NAME" 2>&1 || echo "  WARNING: Could not delete
IAM role ${NAME}."
                ;;
        esac
    done
```

```
    if [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi

    echo ""
    echo "Cleanup complete."
}

handle_error() {
    echo ""
    echo "========================================="
    echo "ERROR: Script failed at $1"
    echo "========================================="
    echo ""
    if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
        echo "Attempting to clean up ${#CREATED_RESOURCES[@]} resource(s)..."
        cleanup_resources
    fi
    exit 1
}

trap 'handle_error "line $LINENO"' ERR

wait_for_resource() {
    local DESCRIPTION="$1"
    local COMMAND="$2"
    local TARGET_VALUE="$3"
    local TIMEOUT=300
    local ELAPSED=0
    local INTERVAL=5

    echo "Waiting for ${DESCRIPTION}..."
    while true; do
        local RESULT
        RESULT=$(eval "$COMMAND" 2>&1) || true
        if echo "$RESULT" | grep -q "$TARGET_VALUE"; then
            echo "  ${DESCRIPTION} is ready."
            return 0
        fi
        if [ "$ELAPSED" -ge "$TIMEOUT" ]; then
            echo "ERROR: Timed out waiting for ${DESCRIPTION} after ${TIMEOUT}
 seconds."
            return 1
```

```
        fi
        sleep "$INTERVAL"
        ELAPSED=$((ELAPSED + INTERVAL))
    done
}

validate_input() {
    local input="$1"
    local pattern="$2"
    if ! [[ "$input" =~ $pattern ]]; then
        echo "ERROR: Invalid input: $input"
        return 1
    fi
    return 0
}

################################################################################
# Region pre-check
################################################################################

CONFIGURED_REGION=$(aws configure get region 2>/dev/null || true)
if [ -z "$CONFIGURED_REGION" ] && [ -z "$AWS_DEFAULT_REGION" ] && [ -z
 "$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Run 'aws configure set region <region>' or export AWS_DEFAULT_REGION."
    exit 1
fi

################################################################################
# Runtime selection
################################################################################

echo ""
echo "=========================================="
echo "AWS Lambda - Create Your First Function"
echo "=========================================="
echo ""
echo "Select a runtime for your Lambda function:"
echo "  1) Python 3.13"
echo "  2) Node.js 22.x"
echo ""
echo "Using default: Python 3.13"
RUNTIME_CHOICE="1"
```

```
case "$RUNTIME_CHOICE" in
    1)
        RUNTIME="python3.13"
        HANDLER="lambda_function.lambda_handler"
        CODE_FILE="lambda_function.py"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'PYTHON_EOF'
import json
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
def lambda_handler(event, context):
    if not isinstance(event, dict) or 'length' not in event or 'width' not in
 event:
        raise ValueError('Event must contain length and width')
    try:
        length = float(event['length'])
        width = float(event['width'])
        if length < 0 or width < 0:
            raise ValueError('Length and width must be non-negative')
        area = calculate_area(length, width)
        print(f'The area is {area}')
        logger.info(f'CloudWatch logs group: {context.log_group_name}')
        return json.dumps({'area': area})
    except (TypeError, ValueError) as e:
        logger.error(f'Error processing input: {str(e)}')
        raise
def calculate_area(length, width):
    return length * width
PYTHON_EOF
        echo "Selected runtime: Python 3.13"
        ;;
    2)
        RUNTIME="nodejs22.x"
        HANDLER="index.handler"
        CODE_FILE="index.mjs"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'NODEJS_EOF'
export const handler = async (event, context) => {
  if (!event || typeof event.length !== 'number' || typeof event.width !==
 'number') {
    throw new Error('Event must contain numeric length and width');
  }
  if (event.length < 0 || event.width < 0) {
    throw new Error('Length and width must be non-negative');
  }
```

```
  const area = event.length * event.width;
  console.log(`The area is ${area}`);
  console.log('CloudWatch log group: ', context.logGroupName);
  return JSON.stringify({area});
};
NODEJS_EOF
        echo "Selected runtime: Node.js 22.x"
        ;;
    *)
        echo "ERROR: Invalid choice. Please enter 1 or 2."
        exit 1
        ;;
esac

# ============================================================================
# Step 1: Create IAM execution role
# ============================================================================

echo ""
echo "=========================================="
echo "Step 1: Create IAM execution role"
echo "=========================================="
echo ""

TRUST_POLICY='{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "lambda.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}'

echo "Creating IAM role: ${ROLE_NAME}"
ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" \
    --query 'Role.Arn' \
    --output text 2>&1)
```

```
if ! validate_input "$ROLE_OUTPUT" "^arn:aws:iam::[0-9]+:role/"; then
    echo "ERROR: Failed to create IAM role"
    exit 1
fi

echo "$ROLE_OUTPUT"
ROLE_ARN="$ROLE_OUTPUT"
CREATED_RESOURCES+=("iam-role:${ROLE_NAME}")
echo "Role ARN: ${ROLE_ARN}"

aws iam tag-role \
    --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=lambda-gettingstarted

echo ""
echo "Attaching AWSLambdaBasicExecutionRole policy..."
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>&1
CREATED_RESOURCES+=("iam-role-policy:${ROLE_NAME}|arn:aws:iam::aws:policy/
service-role/AWSLambdaBasicExecutionRole")
echo "Policy attached."

# IAM roles can take a few seconds to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

################################################################################
# Step 2: Create Lambda function
################################################################################

echo ""
echo "========================================="
echo "Step 2: Create Lambda function"
echo "========================================="
echo ""

echo "Creating deployment package..."
ORIGINAL_DIR=$(pwd)
cd "$TEMP_DIR" || exit 1
zip -j function.zip "$CODE_FILE" > /dev/null 2>&1 || {
    echo "ERROR: Failed to create deployment package"
    exit 1
```

```
}
cd "$ORIGINAL_DIR" || exit 1

if [ ! -f "${TEMP_DIR}/function.zip" ]; then
    echo "ERROR: Deployment package creation failed"
    exit 1
fi

echo "Creating Lambda function: ${FUNCTION_NAME}"
echo "  Runtime: ${RUNTIME}"
echo "  Handler: ${HANDLER}"
echo ""

CREATE_OUTPUT=$(aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime "$RUNTIME" \
    --role "$ROLE_ARN" \
    --handler "$HANDLER" \
    --architectures x86_64 \
    --zip-file "fileb://${TEMP_DIR}/function.zip" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted \
    --query '[FunctionName, FunctionArn, Runtime, State]' \
    --output text 2>&1)

if [ -z "$CREATE_OUTPUT" ]; then
    echo "ERROR: Failed to create Lambda function"
    exit 1
fi

echo "$CREATE_OUTPUT"
CREATED_RESOURCES+=("lambda-function:${FUNCTION_NAME}")

wait_for_resource "Lambda function to become Active" \
    "aws lambda get-function-configuration --function-name ${FUNCTION_NAME} --
query State --output text" \
    "Active"

##############################################################################
# Step 3: Invoke the function
##############################################################################

echo ""
echo "============================================"
echo "Step 3: Invoke the function"
```

```
echo "============================================="
echo ""

TEST_EVENT='{"length": 6, "width": 7}'
echo "Invoking function with test event: ${TEST_EVENT}"
echo ""

echo "$TEST_EVENT" > "${TEMP_DIR}/test-event.json"

if ! validate_input "$TEST_EVENT" '"length": [0-9]+, "width": [0-9]+'; then
    echo "ERROR: Invalid test event format"
    exit 1
fi

INVOKE_OUTPUT=$(aws lambda invoke \
    --function-name "$FUNCTION_NAME" \
    --payload "fileb://${TEMP_DIR}/test-event.json" \
    --cli-read-timeout 30 \
    "${TEMP_DIR}/response.json" 2>&1)
echo "$INVOKE_OUTPUT"

if [ ! -f "${TEMP_DIR}/response.json" ]; then
    echo "ERROR: No response file generated"
    exit 1
fi

RESPONSE=$(cat "${TEMP_DIR}/response.json")
echo ""
echo "Function response: ${RESPONSE}"
echo ""

if echo "$INVOKE_OUTPUT" | grep -qi "functionerror"; then
    echo "WARNING: Function returned an error."
fi

################################################################################
# Step 4: View CloudWatch logs
################################################################################

echo ""
echo "============================================="
echo "Step 4: View CloudWatch Logs"
echo "============================================="
echo ""
```

```
echo "Log group: ${LOG_GROUP_NAME}"
echo ""

echo "Waiting for CloudWatch logs to be available..."

LOG_STREAMS=""
for i in $(seq 1 6); do
    LOG_STREAMS=$(aws logs describe-log-streams \
        --log-group-name "$LOG_GROUP_NAME" \
        --order-by LastEventTime \
        --descending \
        --query 'logStreams[0].logStreamName' \
        --output text 2>/dev/null) || true
    if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
        break
    fi
    LOG_STREAMS=""
    sleep 5
done

if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
    echo "Latest log stream: ${LOG_STREAMS}"
    echo ""
    echo "--- Log events ---"
    LOG_EVENTS=$(aws logs get-log-events \
        --log-group-name "$LOG_GROUP_NAME" \
        --log-stream-name "$LOG_STREAMS" \
        --query 'events[].message' \
        --output text 2>&1) || true
    echo "$LOG_EVENTS"
    echo "--- End of log events ---"
else
    echo "No log streams found yet. Logs may take a moment to appear."
    echo "You can view them in the CloudWatch console:"
    echo "  Log group: ${LOG_GROUP_NAME}"
fi

CREATED_RESOURCES+=("log-group:${LOG_GROUP_NAME}")

aws logs tag-log-group \
    --log-group-name "$LOG_GROUP_NAME" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted
```

```
################################################################################
# Summary and cleanup
################################################################################

echo ""
echo "==========================================="
echo "SUMMARY"
echo "==========================================="
echo ""
echo "Resources created:"
echo "  IAM role:          ${ROLE_NAME}"
echo "  Lambda function:   ${FUNCTION_NAME}"
echo "  CloudWatch logs:   ${LOG_GROUP_NAME}"
echo ""
echo "==========================================="
echo "CLEANUP"
echo "==========================================="
echo ""
echo "Cleaning up all created resources..."
cleanup_resources

echo ""
echo "Done."
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteFunction
  - DeleteLogGroup
  - DeleteRole
  - DescribeLogStreams
  - DetachRolePolicy
  - GetFunction
  - GetFunctionConfiguration
  - GetLogEvents
  - Invoke

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 CLI 開始使用無伺服器資料倉儲

以下程式碼範例顯示做法：

- 使用 secrets-manager CreateSecret
- 使用 secrets-manager DeleteSecret
- 使用 secrets-manager GetSecretValue
- 使用 Redshift CreateNamespace
- 使用 Redshift CreateWorkgroup
- 使用 redshift DeleteNamespace
- 使用 iam CreateRole

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon Redshift Serverless Tutorial Script with Secrets Manager (No jq
 dependency)
# This script creates a Redshift Serverless environment, loads sample data, and
 runs queries
# Uses AWS Secrets Manager for secure password management without requiring jq

# Set up logging
LOG_FILE="redshift-serverless-tutorial-v4.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift Serverless tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
```

```
# Function to check for errors in command output
check_error() {
  local output=$1
  local cmd=$2

  if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Command failed: $cmd"
    echo "Output: $output"
    cleanup_resources
    exit 1
  fi
}

# Function to generate a secure password that meets Redshift requirements
generate_secure_password() {
  # Redshift password requirements:
  # - 8-64 characters
  # - At least one uppercase letter
  # - At least one lowercase letter
  # - At least one decimal digit
  # - Can contain printable ASCII characters except /, ", ', \, @, space

  local password=""
  local valid=false
  local attempts=0
  local max_attempts=10

  while [[ "$valid" == false && $attempts -lt $max_attempts ]]; do
    # Generate base password with safe characters
    local base=$(openssl rand -base64 12 | tr -d '/+=' | head -c 12)

    # Ensure we have at least one of each required character type
    local upper=$(echo "ABCDEFGHIJKLMNOPQRSTUVWXYZ" | fold -w1 | shuf -n1)
    local lower=$(echo "abcdefghijklmnopqrstuvwxyz" | fold -w1 | shuf -n1)
    local digit=$(echo "0123456789" | fold -w1 | shuf -n1)
    local special=$(echo "!#$%&*()_+-=[]{}|;:,.<>?" | fold -w1 | shuf -n1)

    # Combine and shuffle
    password="${base}${upper}${lower}${digit}${special}"
    password=$(echo "$password" | fold -w1 | shuf | tr -d '\n')

    # Validate password meets requirements
    if [[ ${#password} -ge 8 && ${#password} -le 64 ]] && \
```

```
        [[ "$password" =~ [A-Z] ]] && \
        [[ "$password" =~ [a-z] ]] && \
        [[ "$password" =~ [0-9] ]] && \
        [[ ! "$password" =~ [/\"\'\\@[:space:]] ]]; then
      valid=true
    fi

    ((attempts++))
  done

  if [[ "$valid" == false ]]; then
    echo "ERROR: Failed to generate valid password after $max_attempts attempts"
    exit 1
  fi

  echo "$password"
}

# Function to create secret in AWS Secrets Manager
create_secret() {
  local secret_name=$1
  local username=$2
  local password=$3
  local description=$4

  echo "Creating secret in AWS Secrets Manager: $secret_name"

  # Create the secret using AWS CLI without jq
  local secret_output=$(aws secretsmanager create-secret \
    --name "$secret_name" \
    --description "$description" \
    --secret-string "{\"username\":\"$username\",\"password\":\"$password\"}" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
 2>&1)

  if echo "$secret_output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Failed to create secret: $secret_output"
    return 1
  fi

  echo "Secret created successfully: $secret_name"
  return 0
}
```

```
# Function to retrieve password from AWS Secrets Manager
get_password_from_secret() {
  local secret_name=$1

  # Get the secret value and extract password using sed/grep instead of jq
  local secret_value=$(aws secretsmanager get-secret-value \
    --secret-id "$secret_name" \
    --query 'SecretString' \
    --output text 2>/dev/null)

  if [[ $? -eq 0 ]]; then
    # Extract password from JSON using sed
    echo "$secret_value" | sed -n 's/.*"password":"\([^"]*\)".*/\1/p'
  else
    echo ""
  fi
}

# Function to wait for a resource to be available
wait_for_resource() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be available..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>/dev/null)
    local status=$(echo "$output" | grep -o '"Status": "[^"]*' | cut -d'"' -f4 ||
 echo "")

    if [[ "$status" == "AVAILABLE" ]]; then
      echo "$resource_type $resource_name is now available"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name status:
 $status. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done
```

```
  echo "ERROR: Timed out waiting for $resource_type $resource_name to be
 available"
  return 1
}

# Function to wait for a resource to be deleted
wait_for_resource_deletion() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be deleted..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>&1)

    if echo "$output" | grep -i "not found\|does not exist" > /dev/null; then
      echo "$resource_type $resource_name has been deleted"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name is still being
 deleted. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done

  echo "ERROR: Timed out waiting for $resource_type $resource_name to be deleted"
  return 1
}

# Function to clean up resources
cleanup_resources() {
  echo ""
  echo "========================================="
  echo "CLEANUP CONFIRMATION"
  echo "========================================="
  echo "The following resources were created:"
  echo "- Redshift Serverless Workgroup: $WORKGROUP_NAME"
  echo "- Redshift Serverless Namespace: $NAMESPACE_NAME"
  echo "- IAM Role: $ROLE_NAME"
  echo "- Secrets Manager Secret: $SECRET_NAME"
  echo ""
```

```
  echo "Do you want to clean up all created resources? (y/n): "
  read -r CLEANUP_CHOICE

  if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the workgroup
    echo "Deleting Redshift Serverless workgroup $WORKGROUP_NAME..."
    WORKGROUP_DELETE_OUTPUT=$(aws redshift-serverless delete-workgroup --
workgroup-name "$WORKGROUP_NAME" 2>&1)
    echo "$WORKGROUP_DELETE_OUTPUT"

    # Wait for workgroup to be deleted before deleting namespace
    wait_for_resource_deletion "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-
serverless get-workgroup --workgroup-name $WORKGROUP_NAME"

    # Delete the namespace
    echo "Deleting Redshift Serverless namespace $NAMESPACE_NAME..."
    NAMESPACE_DELETE_OUTPUT=$(aws redshift-serverless delete-namespace --
namespace-name "$NAMESPACE_NAME" 2>&1)
    echo "$NAMESPACE_DELETE_OUTPUT"

    # Wait for namespace to be deleted
    wait_for_resource_deletion "namespace" "$NAMESPACE_NAME" 20 30 "aws redshift-
serverless get-namespace --namespace-name $NAMESPACE_NAME"

    # Delete the IAM role policy
    echo "Deleting IAM role policy..."
    POLICY_DELETE_OUTPUT=$(aws iam delete-role-policy --role-name "$ROLE_NAME" --
policy-name S3Access 2>&1)
    echo "$POLICY_DELETE_OUTPUT"

    # Delete the IAM role
    echo "Deleting IAM role $ROLE_NAME..."
    ROLE_DELETE_OUTPUT=$(aws iam delete-role --role-name "$ROLE_NAME" 2>&1)
    echo "$ROLE_DELETE_OUTPUT"

    # Delete the secret
    echo "Deleting Secrets Manager secret $SECRET_NAME..."
    SECRET_DELETE_OUTPUT=$(aws secretsmanager delete-secret --secret-id
 "$SECRET_NAME" --force-delete-without-recovery 2>&1)
    echo "$SECRET_DELETE_OUTPUT"

    echo "Cleanup completed."
```

```
  else
    echo "Cleanup skipped. Resources will remain in your AWS account."
  fi
}

# Check if required tools are available
if ! command -v openssl &> /dev/null; then
    echo "ERROR: openssl is required but not installed. Please install openssl to
 continue."
    exit 1
fi

# Generate unique names for resources
RANDOM_SUFFIX=$(cat /dev/urandom | tr -dc 'a-z0-9' | head -c 6)
NAMESPACE_NAME="rs-namespace-${RANDOM_SUFFIX}"
WORKGROUP_NAME="rs-workgroup-${RANDOM_SUFFIX}"
ROLE_NAME="RedshiftServerlessS3Role-${RANDOM_SUFFIX}"
SECRET_NAME="redshift-serverless-admin-${RANDOM_SUFFIX}"
DB_NAME="dev"
ADMIN_USERNAME="admin"

# Generate secure password
echo "Generating secure password..."
ADMIN_PASSWORD=$(generate_secure_password)

# Create secret in AWS Secrets Manager
create_secret "$SECRET_NAME" "$ADMIN_USERNAME" "$ADMIN_PASSWORD" "Admin
 credentials for Redshift Serverless namespace $NAMESPACE_NAME"
if [[ $? -ne 0 ]]; then
  echo "ERROR: Failed to create secret in AWS Secrets Manager"
  exit 1
fi

# Track created resources
CREATED_RESOURCES=()

echo "Using the following resource names:"
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- IAM Role: $ROLE_NAME"
echo "- Secret: $SECRET_NAME"
echo "- Database: $DB_NAME"
echo "- Admin Username: $ADMIN_USERNAME"
echo "- Admin Password: [STORED IN SECRETS MANAGER]"
```

```
# Step 1: Create IAM role for S3 access
echo "Creating IAM role for Redshift Serverless S3 access..."

# Create trust policy document
cat > redshift-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift-serverless.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create S3 access policy document
cat > redshift-s3-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role $ROLE_NAME..."
ROLE_OUTPUT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://redshift-trust-policy.json 2>&1)
```

```
echo "$ROLE_OUTPUT"
check_error "$ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
CREATED_RESOURCES+=("IAM Role: $ROLE_NAME")

# Attach S3 policy to the role
echo "Attaching S3 access policy to role $ROLE_NAME..."
POLICY_OUTPUT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 S3Access --policy-document file://redshift-s3-policy.json 2>&1)
echo "$POLICY_OUTPUT"
check_error "$POLICY_OUTPUT" "aws iam put-role-policy"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Step 2: Create a namespace
echo "Creating Redshift Serverless namespace $NAMESPACE_NAME..."
NAMESPACE_OUTPUT=$(aws redshift-serverless create-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --admin-username "$ADMIN_USERNAME" \
  --admin-user-password "$ADMIN_PASSWORD" \
  --db-name "$DB_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$NAMESPACE_OUTPUT"
check_error "$NAMESPACE_OUTPUT" "aws redshift-serverless create-namespace"
CREATED_RESOURCES+=("Redshift Serverless Namespace: $NAMESPACE_NAME")

# Wait for namespace to be available
wait_for_resource "namespace" "$NAMESPACE_NAME" 10 30 "aws redshift-serverless
 get-namespace --namespace-name $NAMESPACE_NAME"

# Associate IAM role with namespace
echo "Associating IAM role with namespace..."
UPDATE_NAMESPACE_OUTPUT=$(aws redshift-serverless update-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --iam-roles "$ROLE_ARN" 2>&1)
echo "$UPDATE_NAMESPACE_OUTPUT"
check_error "$UPDATE_NAMESPACE_OUTPUT" "aws redshift-serverless update-namespace"

# Step 3: Create a workgroup
echo "Creating Redshift Serverless workgroup $WORKGROUP_NAME..."
```

```
WORKGROUP_OUTPUT=$(aws redshift-serverless create-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --namespace-name "$NAMESPACE_NAME" \
  --base-capacity 8 \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$WORKGROUP_OUTPUT"
check_error "$WORKGROUP_OUTPUT" "aws redshift-serverless create-workgroup"
CREATED_RESOURCES+=("Redshift Serverless Workgroup: $WORKGROUP_NAME")

# Wait for workgroup to be available
wait_for_resource "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-serverless
 get-workgroup --workgroup-name $WORKGROUP_NAME"

# Get workgroup endpoint
WORKGROUP_ENDPOINT=$(aws redshift-serverless get-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --query 'workgroup.endpoint.address' \
  --output text)
echo "Workgroup endpoint: $WORKGROUP_ENDPOINT"

# Wait additional time for the endpoint to be fully operational
echo "Waiting for endpoint to be fully operational..."
sleep 60

# Step 4: Create tables for sample data
echo "Creating tables for sample data..."

# Create users table
echo "Creating users table..."
USERS_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE users(
    userid INTEGER NOT NULL DISTKEY SORTKEY,
    username CHAR(8),
    firstname VARCHAR(30),
    lastname VARCHAR(30),
    city VARCHAR(30),
    state CHAR(2),
    email VARCHAR(100),
    phone CHAR(14),
    likesports BOOLEAN,
    liketheatre BOOLEAN,
    likeconcerts BOOLEAN,
```

```
    likejazz BOOLEAN,
    likeclassical BOOLEAN,
    likeopera BOOLEAN,
    likerock BOOLEAN,
    likevegas BOOLEAN,
    likebroadway BOOLEAN,
    likemusicals BOOLEAN
  );" 2>&1)
echo "$USERS_TABLE_OUTPUT"
check_error "$USERS_TABLE_OUTPUT" "aws redshift-data execute-statement (users
 table)"
USERS_QUERY_ID=$(echo "$USERS_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for users table creation to complete..."
sleep 5

# Create event table
echo "Creating event table..."
EVENT_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE event(
    eventid INTEGER NOT NULL DISTKEY,
    venueid SMALLINT NOT NULL,
    catid SMALLINT NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    eventname VARCHAR(200),
    starttime TIMESTAMP
  );" 2>&1)
echo "$EVENT_TABLE_OUTPUT"
check_error "$EVENT_TABLE_OUTPUT" "aws redshift-data execute-statement (event
 table)"
EVENT_QUERY_ID=$(echo "$EVENT_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for event table creation to complete..."
sleep 5

# Create sales table
echo "Creating sales table..."
SALES_TABLE_OUTPUT=$(aws redshift-data execute-statement \
```

```
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE sales(
    salesid INTEGER NOT NULL,
    listid INTEGER NOT NULL DISTKEY,
    sellerid INTEGER NOT NULL,
    buyerid INTEGER NOT NULL,
    eventid INTEGER NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    qtysold SMALLINT NOT NULL,
    pricepaid DECIMAL(8,2),
    commission DECIMAL(8,2),
    saletime TIMESTAMP
  );" 2>&1)
echo "$SALES_TABLE_OUTPUT"
check_error "$SALES_TABLE_OUTPUT" "aws redshift-data execute-statement (sales
 table)"
SALES_QUERY_ID=$(echo "$SALES_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for tables to be created
echo "Waiting for tables to be created..."
sleep 10

# Step 5: Load sample data from Amazon S3
echo "Loading sample data from Amazon S3..."

# Load data into users table
echo "Loading data into users table..."
USERS_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY users
    FROM 's3://redshift-downloads/tickit/allusers_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$USERS_LOAD_OUTPUT"
check_error "$USERS_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 users)"
USERS_LOAD_QUERY_ID=$(echo "$USERS_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)
```

```
# Wait for data loading to complete
echo "Waiting for users data loading to complete..."
sleep 10

# Load data into event table
echo "Loading data into event table..."
EVENT_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY event
    FROM 's3://redshift-downloads/tickit/allevents_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$EVENT_LOAD_OUTPUT"
check_error "$EVENT_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 event)"
EVENT_LOAD_QUERY_ID=$(echo "$EVENT_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for event data loading to complete..."
sleep 10

# Load data into sales table
echo "Loading data into sales table..."
SALES_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY sales
    FROM 's3://redshift-downloads/tickit/sales_tab.txt'
    DELIMITER '\t'
    TIMEFORMAT 'MM/DD/YYYY HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$SALES_LOAD_OUTPUT"
check_error "$SALES_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 sales)"
SALES_LOAD_QUERY_ID=$(echo "$SALES_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for sales data loading to complete..."
```

```
sleep 30

# Step 6: Run sample queries
echo "Running sample queries..."

# Query 1: Find top 10 buyers by quantity
echo "Running query: Find top 10 buyers by quantity..."
QUERY1_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT firstname, lastname, total_quantity
    FROM (SELECT buyerid, sum(qtysold) total_quantity
          FROM sales
          GROUP BY buyerid
          ORDER BY total_quantity desc limit 10) Q, users
    WHERE Q.buyerid = userid
    ORDER BY Q.total_quantity desc;" 2>&1)
echo "$QUERY1_OUTPUT"
check_error "$QUERY1_OUTPUT" "aws redshift-data execute-statement (query 1)"
QUERY1_ID=$(echo "$QUERY1_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 1 to complete..."
sleep 10

# Get query 1 results
echo "Getting results for query 1..."
QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
echo "$QUERY1_STATUS_OUTPUT"
check_error "$QUERY1_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 1)"

QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY1_STATUS" == "FINISHED" ]; then
  QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID" 2>&1)
  echo "Query 1 Results:"
  echo "$QUERY1_RESULTS"
else
  echo "Query 1 is not yet complete. Status: $QUERY1_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20
```

```
  # Check again
  QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
  QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY1_STATUS" == "FINISHED" ]; then
    QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID"
 2>&1)
    echo "Query 1 Results:"
    echo "$QUERY1_RESULTS"
  else
    echo "Query 1 is still not complete. Status: $QUERY1_STATUS"
  fi
fi

# Query 2: Find events in the 99.9 percentile in terms of all time total sales
echo "Running query: Find events in the 99.9 percentile in terms of all time
 total sales..."
QUERY2_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT eventname, total_price
    FROM (SELECT eventid, total_price, ntile(1000) over(order by total_price
 desc) as percentile
          FROM (SELECT eventid, sum(pricepaid) total_price
                FROM sales
                GROUP BY eventid)) Q, event E
    WHERE Q.eventid = E.eventid
    AND percentile = 1
    ORDER BY total_price desc;" 2>&1)
echo "$QUERY2_OUTPUT"
check_error "$QUERY2_OUTPUT" "aws redshift-data execute-statement (query 2)"
QUERY2_ID=$(echo "$QUERY2_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 2 to complete..."
sleep 10

# Get query 2 results
echo "Getting results for query 2..."
QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
echo "$QUERY2_STATUS_OUTPUT"
```

```
check_error "$QUERY2_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 2)"

QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY2_STATUS" == "FINISHED" ]; then
  QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID" 2>&1)
  echo "Query 2 Results:"
  echo "$QUERY2_RESULTS"
else
  echo "Query 2 is not yet complete. Status: $QUERY2_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20

  # Check again
  QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
  QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY2_STATUS" == "FINISHED" ]; then
    QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID"
 2>&1)
    echo "Query 2 Results:"
    echo "$QUERY2_RESULTS"
  else
    echo "Query 2 is still not complete. Status: $QUERY2_STATUS"
  fi
fi

# Summary
echo ""
echo "=========================================="
echo "TUTORIAL SUMMARY"
echo "=========================================="
echo "You have successfully:"
echo "1. Created a Redshift Serverless namespace and workgroup"
echo "2. Created an IAM role with S3 access permissions"
echo "3. Stored admin credentials securely in AWS Secrets Manager"
echo "4. Created tables for sample data"
echo "5. Loaded sample data from Amazon S3"
echo "6. Run sample queries on the data"
echo ""
echo "Redshift Serverless Resources:"
```

```
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- Database: $DB_NAME"
echo "- Endpoint: $WORKGROUP_ENDPOINT"
echo "- Credentials Secret: $SECRET_NAME"
echo ""
echo "To connect to your Redshift Serverless database using SQL tools:"
echo "- Host: $WORKGROUP_ENDPOINT"
echo "- Database: $DB_NAME"
echo "- Username: $ADMIN_USERNAME"
echo "- Password: Retrieve from AWS Secrets Manager secret '$SECRET_NAME'"
echo ""
echo "To retrieve the password from Secrets Manager (without jq):"
echo "aws secretsmanager get-secret-value --secret-id $SECRET_NAME --query
 'SecretString' --output text | sed -n 's/.*\"password\":\"\([^\"]*\)\".*/\1/p'"
echo ""

# Clean up temporary files
rm -f redshift-trust-policy.json redshift-s3-policy.json

# Clean up resources
cleanup_resources

echo "Tutorial completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - CreateNamespace
  - CreateRole
  - CreateSecret
  - CreateWorkgroup
  - DeleteNamespace
  - DeleteRole
  - DeleteRolePolicy
  - DeleteSecret
  - DeleteWorkgroup
  - GetNamespace
  - GetRole

- GetSecretValue
- GetWorkgroup
- PutRolePolicy
- UpdateNamespace

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

組態管理入門

以下程式碼範例顯示做法：

- 建立 Amazon S3 儲存貯體
- 建立 Amazon SNS 主題
- 為 Config 建立 IAM 角色
- 設定 Config 組態記錄器
- 設定 Config 交付管道
- 啟動組態記錄器
- 驗證 Config 設定

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI

# Error handling
set -e
```

```
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
```

```
    fi

    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}

# Function to display created resources
```

```
display_resources() {
    echo ""
    echo "=========================================="
    echo "CREATED RESOURCES"
    echo "=========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "=========================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
```

```
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
```

```
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"

# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"

echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15

# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
```

```
CREATED_NEW_CONFIG_RECORDER="false"

echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"

if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
```

```
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"

    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
```

```
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"

    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"

echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
```

```
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"

echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder
  - DeleteDeliveryChannel
  - DeleteRole
  - DeleteRolePolicy

- DeleteTopic
- DescribeConfigurationRecorderStatus
- DescribeConfigurationRecorders
- DescribeDeliveryChannels
- DetachRolePolicy
- GetCallerIdentity
- PutConfigurationRecorder
- PutDeliveryChannel
- PutPublicAccessBlock
- PutRolePolicy
- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

開始使用物聯網裝置保護

以下程式碼範例顯示做法：

- 建立必要的 IAM 角色
- 啟用 IoT Device Defender 稽核檢查
- 執行隨需稽核
- 建立緩解動作
- 將緩解動作套用至調查結果
- 設定 SNS 通知（選用）
- 啟用 IoT 記錄

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS IoT Device Defender Getting Started Script
# This script demonstrates how to use AWS IoT Device Defender to enable audit
 checks,
# view audit results, create mitigation actions, and apply them to findings.

set -euo pipefail

# Set up logging
LOG_FILE="iot-device-defender-script-$(date +%Y%m%d%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "=================================================="
echo "AWS IoT Device Defender Getting Started Script"
echo "=================================================="
echo "Starting script execution at $(date)"
echo ""

# Function to check for errors in command output
check_error() {
    if echo "$1" | grep -iE "An error occurred|Exception|Failed|usage: aws" > /
dev/null; then
        echo "ERROR: Command failed with the following output:"
        echo "$1"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values using jq
extract_json_value() {
```

```
    local json="$1"
    local key="$2"
    echo "$json" | jq -r ".${key} // empty" 2>/dev/null || echo ""
}

# Function to validate JSON
validate_json() {
    local json="$1"
    echo "$json" | jq empty 2>/dev/null
}

# Function to check AWS CLI availability
check_aws_cli() {
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed or not in PATH"
        return 1
    fi
    if ! command -v jq &> /dev/null; then
        echo "ERROR: jq is not installed or not in PATH"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    local account_id
    account_id=$(aws sts get-caller-identity --query 'Account' --output text 2>/
dev/null) || true
    if [ -z "$account_id" ]; then
        echo "ERROR: Could not retrieve AWS account ID"
        return 1
    fi
    echo "$account_id"
    return 0
}

# Function to create IAM roles with retry logic
create_iam_role() {
    local ROLE_NAME=$1
    local TRUST_POLICY=$2
    local MANAGED_POLICY=$3
    local RETRY_COUNT=0
    local MAX_RETRIES=3
```

```
    echo "Creating IAM role: $ROLE_NAME"

    # Validate trust policy JSON
    if ! validate_json "$TRUST_POLICY"; then
        echo "ERROR: Invalid trust policy JSON for role $ROLE_NAME"
        return 1
    fi

    # Check if role already exists
    if aws iam get-role --role-name "$ROLE_NAME" >/dev/null 2>&1; then
        echo "Role $ROLE_NAME already exists, skipping creation"
        ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn'
 --output text 2>/dev/null) || true
        if [ -z "$ROLE_ARN" ]; then
            echo "ERROR: Could not retrieve ARN for existing role $ROLE_NAME"
            return 1
        fi
        echo "Role ARN: $ROLE_ARN"
        return 0
    fi

    # Create the role with trust policy and retry logic
    while [ $RETRY_COUNT -lt $MAX_RETRIES ]; do
        ROLE_RESULT=$(aws iam create-role \
            --role-name "$ROLE_NAME" \
            --assume-role-policy-document "$TRUST_POLICY" 2>&1) || true

        if check_error "$ROLE_RESULT"; then
            break
        fi

        RETRY_COUNT=$((RETRY_COUNT + 1))
        if [ $RETRY_COUNT -lt $MAX_RETRIES ]; then
            echo "Retrying role creation (attempt $((RETRY_COUNT + 1))/
$MAX_RETRIES)..."
            sleep $((RETRY_COUNT * 2))
        fi
    done

    if ! check_error "$ROLE_RESULT"; then
        echo "Failed to create role $ROLE_NAME after $MAX_RETRIES attempts"
        return 1
    fi
```

```
    aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

    # For IoT logging role, create an inline policy instead of using a managed
 policy
    if [[ "$ROLE_NAME" == "AWSIoTLoggingRole" ]]; then
        local LOGGING_POLICY
        LOGGING_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:PutMetricFilter",
                "logs:PutRetentionPolicy",
                "logs:GetLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        }
    ]
}
EOF
)

        if ! validate_json "$LOGGING_POLICY"; then
            echo "ERROR: Invalid logging policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$LOGGING_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
            return 1
        fi
```

```
    elif [[ "$ROLE_NAME" == "IoTMitigationActionErrorLoggingRole" ]]; then
        local MITIGATION_POLICY
        MITIGATION_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:UpdateCACertificate",
                "iot:UpdateCertificate",
                "iot:SetV2LoggingOptions",
                "iot:SetLoggingOptions",
                "iot:AddThingToThingGroup"
            ],
            "Resource": "arn:aws:iot:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "iot.amazonaws.com"
                }
            }
        }
    ]
}
EOF
)

        if ! validate_json "$MITIGATION_POLICY"; then
            echo "ERROR: Invalid mitigation policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$MITIGATION_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
```

```
            return 1
        fi
    else
        # Attach managed policy to role if provided
        if [ -n "$MANAGED_POLICY" ]; then
            ATTACH_RESULT=$(aws iam attach-role-policy \
                --role-name "$ROLE_NAME" \
                --policy-arn "$MANAGED_POLICY" 2>&1) || true

            if ! check_error "$ATTACH_RESULT"; then
                echo "Failed to attach policy to role $ROLE_NAME"
                return 1
            fi
        fi
    fi

    echo "Role $ROLE_NAME created successfully"

    # Get the role ARN with error handling
    ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --
output text 2>/dev/null) || true
    if [ -z "$ROLE_ARN" ]; then
        echo "ERROR: Could not retrieve ARN for newly created role $ROLE_NAME"
        return 1
    fi
    echo "Role ARN: $ROLE_ARN"
    return 0
}

# Array to store created resources for cleanup
declare -a CREATED_RESOURCES

# Validate prerequisites
echo "Validating prerequisites..."
if ! check_aws_cli; then
    echo "ERROR: Prerequisites not met"
    exit 1
fi

ACCOUNT_ID=$(get_account_id) || exit 1
echo "AWS Account ID: $ACCOUNT_ID"
echo ""

# Step 1: Create IAM roles needed for the tutorial
```

```
echo "================================================="
echo "Step 1: Creating required IAM roles"
echo "================================================="

# Create IoT Device Defender Audit role
IOT_DEFENDER_AUDIT_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "AWSIoTDeviceDefenderAuditRole"
 "$IOT_DEFENDER_AUDIT_TRUST_POLICY" "arn:aws:iam::aws:policy/service-role/
AWSIoTDeviceDefenderAudit"; then
    echo "ERROR: Failed to create audit role"
    exit 1
fi
AUDIT_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTDeviceDefenderAuditRole")

# Create IoT Logging role
IOT_LOGGING_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
EOF
)

if ! create_iam_role "AWSIoTLoggingRole" "$IOT_LOGGING_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create logging role"
    exit 1
fi
LOGGING_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTLoggingRole")

# Create IoT Mitigation Action role
IOT_MITIGATION_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "IoTMitigationActionErrorLoggingRole"
 "$IOT_MITIGATION_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create mitigation role"
    exit 1
fi
MITIGATION_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: IoTMitigationActionErrorLoggingRole")

# Wait for IAM role propagation
echo "Waiting for IAM role propagation..."
sleep 5

# Step 2: Enable audit checks
echo ""
echo "================================================="
echo "Step 2: Enabling AWS IoT Device Defender audit checks"
echo "================================================="
```

```
# Get current audit configuration
echo "Getting current audit configuration..."
CURRENT_CONFIG=$(aws iot describe-account-audit-configuration --output json 2>&1)
 || true
if validate_json "$CURRENT_CONFIG"; then
    echo "$CURRENT_CONFIG" | jq '.' 2>/dev/null || echo "Could not parse current
 configuration"
fi

# Enable specific audit checks with proper JSON escaping
echo "Enabling audit checks..."
AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":true}}'

if ! validate_json "$AUDIT_CONFIG"; then
    echo "ERROR: Invalid audit configuration JSON"
    exit 1
fi

UPDATE_RESULT=$(aws iot update-account-audit-configuration \
  --role-arn "$AUDIT_ROLE_ARN" \
  --audit-check-configurations "$AUDIT_CONFIG" 2>&1) || true

if ! check_error "$UPDATE_RESULT"; then
    echo "Failed to update audit configuration"
    exit 1
fi

echo "Audit checks enabled successfully"

# Step 3: Run an on-demand audit
echo ""
echo "=================================================="
echo "Step 3: Running an on-demand audit"
echo "=================================================="

echo "Starting on-demand audit task..."
AUDIT_TASK_RESULT=$(aws iot start-on-demand-audit-task \
  --target-check-names LOGGING_DISABLED_CHECK --output json 2>&1) || true

if ! check_error "$AUDIT_TASK_RESULT"; then
    echo "Failed to start on-demand audit task"
    exit 1
fi
```

```
TASK_ID=$(extract_json_value "$AUDIT_TASK_RESULT" "taskId")
if [ -z "$TASK_ID" ]; then
    echo "ERROR: Could not extract task ID from response"
    exit 1
fi

echo "Audit task started with ID: $TASK_ID"
CREATED_RESOURCES+=("Audit Task: $TASK_ID")

# Tag the audit task via IoT service
aws iot tag-resource --resource-arn "arn:aws:iot:$(aws configure get region):
${ACCOUNT_ID}:audittask/${TASK_ID}" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

# Wait for the audit task to complete
echo "Waiting for audit task to complete (this may take a few minutes)..."
TASK_STATUS="IN_PROGRESS"
TIMEOUT=0
MAX_TIMEOUT=600
POLL_INTERVAL=15

while [ "$TASK_STATUS" != "COMPLETED" ]; do
    if [ $TIMEOUT -ge $MAX_TIMEOUT ]; then
        echo "WARNING: Audit task did not complete within ${MAX_TIMEOUT} seconds,
 continuing..."
        break
    fi

    sleep "$POLL_INTERVAL"
    TIMEOUT=$((TIMEOUT + POLL_INTERVAL))

    TASK_DETAILS=$(aws iot describe-audit-task --task-id "$TASK_ID" --output json
 2>&1) || true
    if validate_json "$TASK_DETAILS"; then
        TASK_STATUS=$(extract_json_value "$TASK_DETAILS" "taskStatus")
        echo "Current task status: $TASK_STATUS (elapsed: ${TIMEOUT}s)"

        if [ "$TASK_STATUS" = "FAILED" ]; then
            echo "WARNING: Audit task failed, continuing with script..."
            FAILURE_REASON=$(extract_json_value "$TASK_DETAILS"
 "taskStatistics.failedChecksNotApplicable")
            if [ -n "$FAILURE_REASON" ]; then
                echo "Reason: $FAILURE_REASON"
```

```
            fi
            break
        fi
    else
        echo "WARNING: Could not parse task details, retrying..."
    fi
done

echo "Audit task processing completed"

# Get audit findings (non-blocking)
echo "Getting audit findings..."
FINDINGS=$(aws iot list-audit-findings \
  --task-id "$TASK_ID" --output json 2>&1) || true

if validate_json "$FINDINGS"; then
    FINDING_COUNT=$(echo "$FINDINGS" | jq '.findings | length' 2>/dev/null ||
 echo "0")
    echo "Audit findings count: $FINDING_COUNT"
    if [ "$FINDING_COUNT" -gt 0 ]; then
        echo "Sample finding:"
        echo "$FINDINGS" | jq '.findings[0]' 2>/dev/null || echo "Could not parse
 finding"
    fi
else
    echo "WARNING: Could not parse audit findings response"
    FINDINGS='{"findings":[]}'
fi

# Check if we have any non-compliant findings
FINDING_ID=$(extract_json_value "$FINDINGS" "findings[0].findingId")
if [ -n "$FINDING_ID" ]; then
    echo "Found non-compliant finding with ID: $FINDING_ID"
    HAS_FINDINGS=true
else
    echo "No non-compliant findings detected"
    HAS_FINDINGS=false
fi

# Step 4: Create a mitigation action
echo ""
echo "================================================="
echo "Step 4: Creating a mitigation action"
echo "================================================="
```

```
# Check if mitigation action already exists and delete it
if aws iot describe-mitigation-action --action-name "EnableErrorLoggingAction" >/
dev/null 2>&1; then
    echo "Mitigation action 'EnableErrorLoggingAction' already exists, deleting
 it first..."
    aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction"
 2>&1 || true
    sleep 2
fi

echo "Creating mitigation action to enable AWS IoT logging..."

# Build mitigation action parameters JSON
MITIGATION_PARAMS=$(cat <<EOF
{
  "enableIoTLoggingParams": {
    "roleArnForLogging": "$LOGGING_ROLE_ARN",
    "logLevel": "ERROR"
  }
}
EOF
)

if ! validate_json "$MITIGATION_PARAMS"; then
    echo "ERROR: Invalid mitigation parameters JSON"
    exit 1
fi

MITIGATION_RESULT=$(aws iot create-mitigation-action \
  --action-name "EnableErrorLoggingAction" \
  --role-arn "$MITIGATION_ROLE_ARN" \
  --action-params "$MITIGATION_PARAMS" --output json 2>&1) || true

if ! check_error "$MITIGATION_RESULT"; then
    echo "Failed to create mitigation action"
    exit 1
fi

if validate_json "$MITIGATION_RESULT"; then
    echo "Mitigation action created successfully"
    MITIGATION_ACTION_ARN=$(extract_json_value "$MITIGATION_RESULT" "actionArn")
    if [ -n "$MITIGATION_ACTION_ARN" ]; then
        echo "Mitigation Action ARN: $MITIGATION_ACTION_ARN"
```

```
        aws iot tag-resource --resource-arn "$MITIGATION_ACTION_ARN" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs 2>&1
 || true
    fi
else
    echo "WARNING: Could not validate mitigation action response, but action may
 have been created"
fi

CREATED_RESOURCES+=("Mitigation Action: EnableErrorLoggingAction")

# Step 5: Apply mitigation action to findings (if any)
if [ "$HAS_FINDINGS" = true ]; then
    echo ""
    echo "=================================================="
    echo "Step 5: Applying mitigation action to findings"
    echo "=================================================="

    MITIGATION_TASK_ID="MitigationTask-$(date +%s)"
    echo "Starting mitigation actions task with ID: $MITIGATION_TASK_ID"

    # Build target JSON
    TARGET_JSON=$(cat <<EOF
{
  "findingIds": ["$FINDING_ID"]
}
EOF
)

    if ! validate_json "$TARGET_JSON"; then
        echo "ERROR: Invalid target JSON"
        exit 1
    fi

    # Build audit check to actions mapping JSON
    AUDIT_CHECK_MAPPING=$(cat <<EOF
{
  "LOGGING_DISABLED_CHECK": ["EnableErrorLoggingAction"]
}
EOF
)

    if ! validate_json "$AUDIT_CHECK_MAPPING"; then
        echo "ERROR: Invalid audit check mapping JSON"
```

```
        exit 1
    fi

    MITIGATION_TASK_RESULT=$(aws iot start-audit-mitigation-actions-task \
      --task-id "$MITIGATION_TASK_ID" \
      --target "$TARGET_JSON" \
      --audit-check-to-actions-mapping "$AUDIT_CHECK_MAPPING" --output json 2>&1)
 || true

    if ! check_error "$MITIGATION_TASK_RESULT"; then
        echo "WARNING: Failed to start mitigation actions task, continuing..."
    else
        echo "Mitigation actions task started successfully"
        CREATED_RESOURCES+=("Mitigation Task: $MITIGATION_TASK_ID")
    fi
else
    echo ""
    echo "=================================================="
    echo "Step 5: Skipping mitigation action application (no findings)"
    echo "=================================================="
fi

# Step 6: Set up SNS notifications (optional)
echo ""
echo "=================================================="
echo "Step 6: Setting up SNS notifications"
echo "=================================================="

# Check if SNS topic already exists
SNS_TOPICS=$(aws sns list-topics --output json 2>&1) || true
TOPIC_ARN=""
if validate_json "$SNS_TOPICS"; then
    TOPIC_ARN=$(echo "$SNS_TOPICS" | jq -r '.Topics[] | select(.TopicArn |
 contains("IoTDDNotifications")) | .TopicArn' 2>/dev/null | head -1 || echo "")
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "SNS topic 'IoTDDNotifications' already exists, using existing topic..."
    echo "Topic ARN: $TOPIC_ARN"
else
    echo "Creating SNS topic for notifications..."
    SNS_RESULT=$(aws sns create-topic --name "IoTDDNotifications" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs --
output json 2>&1) || true
```

```
    if ! check_error "$SNS_RESULT"; then
        echo "WARNING: Failed to create SNS topic, continuing..."
        SNS_RESULT=""
    else
        TOPIC_ARN=$(extract_json_value "$SNS_RESULT" "TopicArn")
        if [ -n "$TOPIC_ARN" ]; then
            echo "SNS topic created with ARN: $TOPIC_ARN"
            CREATED_RESOURCES+=("SNS Topic: IoTDDNotifications")
        fi
    fi
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "Updating audit configuration to enable SNS notifications..."

    # Build SNS notification configuration JSON
    SNS_CONFIG=$(cat <<EOF
{
  "SNS": {
    "targetArn": "$TOPIC_ARN",
    "roleArn": "$AUDIT_ROLE_ARN",
    "enabled": true
  }
}
EOF
)

    if ! validate_json "$SNS_CONFIG"; then
        echo "ERROR: Invalid SNS configuration JSON"
        exit 1
    fi

    SNS_UPDATE_RESULT=$(aws iot update-account-audit-configuration \
      --audit-notification-target-configurations "$SNS_CONFIG" 2>&1) || true

    if ! check_error "$SNS_UPDATE_RESULT"; then
        echo "WARNING: Failed to update audit configuration for SNS
 notifications"
    else
        echo "SNS notifications enabled successfully"
    fi
else
    echo "Skipping SNS configuration due to topic creation failure"
```

```
fi

# Step 7: Enable AWS IoT logging
echo ""
echo "=================================================="
echo "Step 7: Enabling AWS IoT logging"
echo "=================================================="

echo "Setting up AWS IoT logging options..."

LOGGING_RESULT=$(aws iot set-v2-logging-options \
  --role-arn "$LOGGING_ROLE_ARN" \
  --default-log-level "ERROR" 2>&1) || true

if ! check_error "$LOGGING_RESULT"; then
    echo "V2 logging setup failed, trying v1 logging..."

    V1_LOGGING_CONFIG=$(cat <<EOF
{
  "roleArn": "$LOGGING_ROLE_ARN",
  "logLevel": "ERROR"
}
EOF
)

    if ! validate_json "$V1_LOGGING_CONFIG"; then
        echo "ERROR: Invalid v1 logging configuration JSON"
        exit 1
    fi

    LOGGING_RESULT_V1=$(aws iot set-logging-options \
      --logging-options-payload "$V1_LOGGING_CONFIG" 2>&1) || true

    if ! check_error "$LOGGING_RESULT_V1"; then
        echo "WARNING: Failed to set up AWS IoT logging with both v1 and v2
 methods, continuing..."
    else
        echo "AWS IoT v1 logging enabled successfully"
    fi
else
    echo "AWS IoT v2 logging enabled successfully"
fi

# Verify logging is enabled
```

```
echo "Verifying logging configuration..."
LOGGING_CONFIG=$(aws iot get-v2-logging-options --output json 2>&1) || true
if [ -n "$LOGGING_CONFIG" ] && ! check_error "$LOGGING_CONFIG" && validate_json
 "$LOGGING_CONFIG"; then
    echo "Logging configuration verified"
    echo "$LOGGING_CONFIG" | jq '.' 2>/dev/null || echo "Configuration retrieved
 but could not display details"
else
    echo "Could not verify logging configuration, but setup completed"
fi

# Script completed successfully
echo ""
echo "================================================="
echo "AWS IoT Device Defender setup completed successfully!"
echo "================================================="
echo "The following resources were created:"
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "- $resource"
done
echo ""

# Cleanup phase
echo "=========================================="
echo "CLEANUP"
echo "=========================================="
echo "Starting automatic cleanup of resources..."
echo "Waiting 10 seconds before cleanup to allow resource stabilization..."
sleep 10

# Disable AWS IoT logging
echo "Disabling AWS IoT logging..."

DISABLE_V2_RESULT=$(aws iot set-v2-logging-options \
  --default-log-level "DISABLED" 2>&1) || true

if check_error "$DISABLE_V2_RESULT"; then
    echo "V2 logging disabled successfully"
else
    echo "Attempting v1 logging disable..."

    V1_DISABLE_CONFIG=$(cat <<'EOF'
{
  "logLevel": "DISABLED"
```

```
}
EOF
)

    DISABLE_V1_RESULT=$(aws iot set-logging-options \
      --logging-options-payload "$V1_DISABLE_CONFIG" 2>&1) || true

    if check_error "$DISABLE_V1_RESULT"; then
        echo "V1 logging disabled successfully"
    else
        echo "WARNING: Could not disable logging"
    fi
fi

# Delete mitigation action
echo "Deleting mitigation action..."
aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction" 2>&1 ||
 true

# Reset audit configuration
echo "Resetting IoT Device Defender audit configuration..."
RESET_AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":false}}'
aws iot update-account-audit-configuration \
  --audit-check-configurations "$RESET_AUDIT_CONFIG" 2>&1 || true

# Delete SNS topic
echo "Deleting SNS topic..."
if [ -n "${TOPIC_ARN:-}" ] && [ "$TOPIC_ARN" != "null" ]; then
    aws sns delete-topic --topic-arn "$TOPIC_ARN" 2>&1 || true
fi

# Clean up IAM roles with improved error handling
echo "Cleaning up IAM roles..."

cleanup_role() {
    local role_name=$1
    echo "Cleaning up role: $role_name"

    if aws iam get-role --role-name "$role_name" >/dev/null 2>&1; then
        ROLE_POLICIES=$(aws iam list-role-policies --role-name "$role_name" --
output json 2>&1 || echo '{"PolicyNames":[]}')
        if validate_json "$ROLE_POLICIES"; then
            while IFS= read -r policy_name; do
                if [ -n "$policy_name" ] && [ "$policy_name" != "null" ]; then
```

```
                    echo "  Deleting inline policy: $policy_name"
                    aws iam delete-role-policy \
                        --role-name "$role_name" \
                        --policy-name "$policy_name" 2>&1 || true
                fi
            done < <(echo "$ROLE_POLICIES" | jq -r '.PolicyNames[]' 2>/dev/null
 || echo "")
        fi

        ATTACHED_POLICIES=$(aws iam list-attached-role-policies --role-name
 "$role_name" --output json 2>&1 || echo '{"AttachedPolicies":[]}')
        if validate_json "$ATTACHED_POLICIES"; then
            while IFS= read -r policy_arn; do
                if [ -n "$policy_arn" ] && [ "$policy_arn" != "null" ]; then
                    echo "  Detaching managed policy: $policy_arn"
                    aws iam detach-role-policy \
                        --role-name "$role_name" \
                        --policy-arn "$policy_arn" 2>&1 || true
                fi
            done < <(echo "$ATTACHED_POLICIES" | jq -r
 '.AttachedPolicies[].PolicyArn' 2>/dev/null || echo "")
        fi

        echo "  Deleting role: $role_name"
        aws iam delete-role --role-name "$role_name" 2>&1 || true
    else
        echo "  Role $role_name does not exist or already deleted"
    fi
}

cleanup_role "AWSIoTDeviceDefenderAuditRole"
cleanup_role "AWSIoTLoggingRole"
cleanup_role "IoTMitigationActionErrorLoggingRole"

echo "Cleanup completed successfully"

echo ""
echo "Script execution completed at $(date)"
echo "Log file: $LOG_FILE"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateMitigationAction

- CreateRole
- CreateTopic
- DeleteAccountAuditConfiguration
- DeleteMitigationAction
- DeleteRole
- DeleteRolePolicy
- DeleteTopic
- DescribeAccountAuditConfiguration
- DescribeAuditTask
- DetachRolePolicy
- GetLoggingOptions
- GetRole
- GetV2LoggingOptions
- ListAuditFindings
- ListAuditMitigationActionsTasks
- ListMitigationActions
- ListRolePolicies
- ListTopics
- PutRolePolicy
- SetLoggingOptions
- SetV2LoggingOptions
- StartAuditMitigationActionsTask
- StartOnDemandAuditTask
- UpdateAccountAuditConfiguration

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

機器學習功能存放區的入門

以下程式碼範例顯示做法：

- 設定 IAM 許可
- 建立一個 SageMaker 執行角色
- 建立特徵群組
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command status
```

```
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"

        case "$resource_type" in
            "FeatureGroup")
```

```
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'
```

```
    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"
```

```
    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
```

```
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
    exit 1
fi
```

```
echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
```

```
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
```

```
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER1_RESPONSE"

CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
```

```
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"

ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
```

```
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)
```

```
if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "============================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "============================================="
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "============================================="
echo ""

# Handle cleanup
```

```
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateBucket
  - CreateFeatureGroup
  - CreateRole
  - DeleteBucket

- DeleteFeatureGroup
- DeleteRole
- DescribeFeatureGroup
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListFeatureGroups
- PutPublicAccessBlock
- Rm

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

受管 kubernetes 叢集入門

以下程式碼範例顯示做法：

- 為您的 EKS 叢集建立 VPC
- 為您的 EKS 叢集建立 IAM 角色
- 建立 EKS 叢集
- 設定 kubectl 以與您的叢集通訊
- 建立受管節點群組
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash
```

```
# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
        echo ""
        echo "  3. Move the binary to your PATH:"
```

```
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"

# Array to track created resources for cleanup
```

```
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
```

```
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi

    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."

# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")
```

```
echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."

# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"
```

```
# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"
```

```
# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"

# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
```

```
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"
```

```
NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
```

```
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi

# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
```

```
done
echo "============================================"

# Prompt for cleanup
echo ""
echo "============================================"
echo "CLEANUP CONFIRMATION"
echo "============================================"
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster
  - DeleteNodegroup

- DeleteRole
- DeleteStack
- DescribeCluster
- DescribeNodegroup
- DescribeStacks
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNodegroups
- UpdateKubeconfig
- 等待 (CloudFormation)
- 等待 (Amazon EKS)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

受管串流入門

以下程式碼範例顯示做法：

- 建立 MSK 叢集
- 建立 MSK 存取的 IAM 許可
- 建立用戶端機器
- 取得引導代理程式
- 設定用戶端機器
- 建立主題並產生/使用資料
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
```

```
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
}

# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi
```

```
    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"

                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
```

```
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."

                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete EC2 instance if it exists
    if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
        echo "Terminating EC2 instance: $INSTANCE_ID"
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
 to terminate instance"
        echo "Waiting for instance to terminate..."
        aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
 "Failed to wait for instance termination"
    fi
```

```
   # Delete MSK cluster first (to remove dependencies on security group)
   if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
       echo "Deleting MSK cluster: $CLUSTER_ARN"
       aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
delete cluster"

       # Wait a bit for the cluster deletion to start
       echo "Waiting 30 seconds for cluster deletion to begin..."
       sleep 30
   fi

   # Remove security group references before attempting deletion
   if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
"$CLIENT_SG_ID"; then
       remove_security_group_references "$CLIENT_SG_ID"

       echo "Deleting security group: $CLIENT_SG_ID"
       # Try multiple times with longer delays to ensure dependencies are
removed
       for i in {1..10}; do
           if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
               echo "Security group deleted successfully"
               break
           fi
           echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
           sleep 30
       done
   fi

   # Delete key pair if it exists
   if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
       echo "Deleting key pair: $KEY_NAME"
       aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
   fi

   # Remove role from instance profile
   if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
       echo "Removing role from instance profile"
       aws iam remove-role-from-instance-profile \
           --instance-profile-name "$INSTANCE_PROFILE_NAME" \
```

```
            --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
 profile"
    fi

    # Delete instance profile
    if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
 "$INSTANCE_PROFILE_NAME"; then
        echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
        aws iam delete-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
 delete instance profile"
    fi

    # Detach policy from role
    if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
 "$ROLE_NAME"; then
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
 role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
 delete policy"
    fi

    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")
```

```
    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"

            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)

            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
        done
    done

    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
```

```
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"

# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
```

```
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
```

```
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi

    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi
```

```
if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
```

```
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"

# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs
```

```
# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi

echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi

echo "Role added to instance profile"

# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "============================================="
```

```
# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)

if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"

# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi

echo "MSK security group ID: $MSK_SG_ID"

# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
```

```
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi

echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)

if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
```

```
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
```

```
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"

INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)

if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi
```

```
echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
echo "Getting bootstrap brokers"
MAX_RETRIES=10
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
```

```
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi

    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
done

echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
```

```
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"

# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
cd $KAFKA_ROOT/libs
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
```

```
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
```

```
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
echo "============================================="

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCE SUMMARY"
echo "============================================="
```

```
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "================================================"
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
```

```
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
```

```
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreatePolicy
  - CreateRole
  - CreateSecurityGroup
  - DeleteCluster
  - DeleteInstanceProfile
  - DeleteKeyPair
  - DeletePolicy
  - DeleteRole
  - DeleteSecurityGroup
  - DescribeAvailabilityZones

- DescribeCluster
- DescribeImages
- DescribeInstanceTypeOfferings
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DescribeSubnets
- DescribeVpcs
- DetachRolePolicy
- GetBootstrapBrokers
- GetCallerIdentity
- GetInstanceProfile
- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- TerminateInstances
- 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

佈建資料倉儲叢集入門

以下程式碼範例顯示做法：

- 建立 Redshift 叢集
- 建立 S3 存取的 IAM 角色
- 建立資料表並載入資料
- 執行範例查詢

- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon Redshift Provisioned Cluster Tutorial Script
# This script creates a Redshift cluster, loads sample data, runs queries, and
 cleans up resources
# Version 4: Security improvements and best practices

set -euo pipefail

# Set up logging
LOG_FILE="redshift_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1" >&2
    echo "Resources created so far:"
    if [ -n "${CLUSTER_ID:-}" ]; then echo "- Redshift Cluster: $CLUSTER_ID"; fi
    if [ -n "${ROLE_NAME:-}" ]; then echo "- IAM Role: $ROLE_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
```

```
    echo "Cleaning up resources..."

    # Delete the cluster if it exists
    if [ -n "${CLUSTER_ID:-}" ]; then
        echo "Deleting Redshift cluster: $CLUSTER_ID"
        aws redshift delete-cluster --cluster-identifier "$CLUSTER_ID" --skip-
final-cluster-snapshot 2>/dev/null || true
        echo "Waiting for cluster deletion to complete..."
        aws redshift wait cluster-deleted --cluster-identifier "$CLUSTER_ID" 2>/
dev/null || true
        echo "Cluster deleted successfully."
    fi

    # Delete the IAM role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Removing IAM role policy..."
        aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 RedshiftS3Access 2>/dev/null || true

        echo "Deleting IAM role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    # Clean up temporary files
    rm -f redshift-trust-policy.json redshift-s3-policy.json

    echo "Cleanup completed."
}

# Trap errors and cleanup
trap 'handle_error "Script interrupted"' INT TERM

# Function to wait for SQL statement to complete
wait_for_statement() {
    local statement_id=$1
    local max_attempts=30
    local attempt=1
    local status=""

    echo "Waiting for statement $statement_id to complete..."

    while [ $attempt -le $max_attempts ]; do
        status=$(aws redshift-data describe-statement --id "$statement_id" --
query 'Status' --output text 2>/dev/null || echo "")
```

```
        if [ "$status" == "FINISHED" ]; then
            echo "Statement completed successfully."
            return 0
        elif [ "$status" == "FAILED" ]; then
            local error=$(aws redshift-data describe-statement --id
 "$statement_id" --query 'Error' --output text 2>/dev/null || echo "Unknown
 error")
            echo "Statement failed with error: $error" >&2
            return 1
        elif [ "$status" == "ABORTED" ]; then
            echo "Statement was aborted." >&2
            return 1
        fi

        echo "Statement status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 10
        ((attempt++))
    done

    echo "Timed out waiting for statement to complete." >&2
    return 1
}

# Function to check if IAM role is attached to cluster
check_role_attached() {
    local role_arn=$1
    local max_attempts=10
    local attempt=1

    echo "Checking if IAM role is attached to the cluster..."

    while [ $attempt -le $max_attempts ]; do
        local status=$(aws redshift describe-clusters \
            --cluster-identifier "$CLUSTER_ID" \
            --query "Clusters[0].IamRoles[?IamRoleArn=='$role_arn'].ApplyStatus"
 \
            --output text 2>/dev/null || echo "")

        if [ "$status" == "in-sync" ]; then
            echo "IAM role is successfully attached to the cluster."
            return 0
        fi
```

```
        echo "IAM role status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 30
        ((attempt++))
    done

    echo "Timed out waiting for IAM role to be attached." >&2
    return 1
}

# Validate required commands
for cmd in aws jq; do
    if ! command -v "$cmd" &> /dev/null; then
        handle_error "Required command '$cmd' not found. Please install it and
 try again."
    fi
done

# Validate AWS credentials
if ! aws sts get-caller-identity &>/dev/null; then
    handle_error "AWS credentials not configured or invalid"
fi

# Variables to track created resources
CLUSTER_ID="examplecluster"
ROLE_NAME="RedshiftS3Role-$(date +%s)"
DB_NAME="dev"
DB_USER="awsuser"

# Generate secure password using AWS Secrets Manager or random string
if command -v openssl &> /dev/null; then
    DB_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-20)
else
    DB_PASSWORD="TempPass$(date +%s | md5sum | cut -c1-20)"
fi

# Validate password meets requirements
if [ ${#DB_PASSWORD} -lt 8 ]; then
    handle_error "Generated password does not meet minimum length requirement"
fi

# Store password securely (optional: use AWS Secrets Manager in production)
echo "Generated database password (store securely): $DB_PASSWORD"
```

```
echo "=== Step 1: Creating Amazon Redshift Cluster ==="

# Create the Redshift cluster with encryption and audit logging enabled
echo "Creating Redshift cluster: $CLUSTER_ID"
CLUSTER_RESULT=$(aws redshift create-cluster \
  --cluster-identifier "$CLUSTER_ID" \
  --node-type ra3.xlplus \
  --number-of-nodes 2 \
  --master-username "$DB_USER" \
  --master-user-password "$DB_PASSWORD" \
  --db-name "$DB_NAME" \
  --port 5439 \
  --encrypted \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-provisioned \
  2>&1) || handle_error "Failed to create Redshift cluster"

echo "$CLUSTER_RESULT"
echo "Waiting for cluster to become available..."

# Wait for the cluster to be available
aws redshift wait cluster-available --cluster-identifier "$CLUSTER_ID" ||
 handle_error "Timeout waiting for cluster to become available"

# Get cluster status to confirm
CLUSTER_STATUS=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].ClusterStatus' \
  --output text)

echo "Cluster status: $CLUSTER_STATUS"

echo "=== Step 2: Creating IAM Role for S3 Access ==="

# Create trust policy file with restricted permissions
echo "Creating trust policy for Redshift"
cat > redshift-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

chmod 600 redshift-trust-policy.json

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://redshift-trust-policy.json 2>&1) ||
 handle_error "Failed to create IAM role"

echo "$ROLE_RESULT"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Tag the IAM role
echo "Tagging IAM role: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=redshift-provisioned

# Create policy document for S3 access with principle of least privilege
echo "Creating S3 access policy"
cat > redshift-s3-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
```

```
    }
  ]
}
EOF

chmod 600 redshift-s3-policy.json

# Attach policy to role
echo "Attaching S3 access policy to role"
POLICY_RESULT=$(aws iam put-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-name RedshiftS3Access \
  --policy-document file://redshift-s3-policy.json 2>&1) || handle_error "Failed
 to attach policy to role"

echo "$POLICY_RESULT"

# Attach role to cluster
echo "Attaching IAM role to Redshift cluster"
ATTACH_ROLE_RESULT=$(aws redshift modify-cluster-iam-roles \
  --cluster-identifier "$CLUSTER_ID" \
  --add-iam-roles "$ROLE_ARN" 2>&1) || handle_error "Failed to attach role to
 cluster"

echo "$ATTACH_ROLE_RESULT"

# Wait for the role to be attached
echo "Waiting for IAM role to be attached to the cluster..."
if ! check_role_attached "$ROLE_ARN"; then
    handle_error "Failed to attach IAM role to cluster"
fi

echo "=== Step 3: Getting Cluster Connection Information ==="

# Get cluster endpoint
CLUSTER_INFO=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].Endpoint.{Address:Address,Port:Port}' \
  --output json)

echo "Cluster endpoint information:"
echo "$CLUSTER_INFO"

echo "=== Step 4: Creating Tables and Loading Data ==="
```

```
echo "Creating sales table"
SALES_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS sales; CREATE TABLE sales(salesid integer not null,
 listid integer not null distkey, sellerid integer not null, buyerid integer
 not null, eventid integer not null, dateid smallint not null sortkey, qtysold
 smallint not null, pricepaid decimal(8,2), commission decimal(8,2), saletime
 timestamp);" \
  --query 'Id' --output text)

echo "Sales table creation statement ID: $SALES_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_TABLE_ID"; then
    handle_error "Failed to create sales table"
fi

echo "Creating date table"
DATE_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS date; CREATE TABLE date(dateid smallint not null
 distkey sortkey, caldate date not null, day character(3) not null, week smallint
 not null, month character(5) not null, qtr character(5) not null, year smallint
 not null, holiday boolean default('N'));" \
  --query 'Id' --output text)

echo "Date table creation statement ID: $DATE_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_TABLE_ID"; then
    handle_error "Failed to create date table"
fi

echo "Loading data into sales table"
SALES_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
```

```
  --sql "COPY sales FROM 's3://redshift-downloads/tickit/sales_tab.txt' DELIMITER
 '\t' TIMEFORMAT 'MM/DD/YYYY HH:MI:SS' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';"
 \
  --query 'Id' --output text)

echo "Sales data load statement ID: $SALES_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_LOAD_ID"; then
    handle_error "Failed to load data into sales table"
fi

echo "Loading data into date table"
DATE_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY date FROM 's3://redshift-downloads/tickit/date2008_pipe.txt'
 DELIMITER '|' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';" \
  --query 'Id' --output text)

echo "Date data load statement ID: $DATE_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_LOAD_ID"; then
    handle_error "Failed to load data into date table"
fi

echo "=== Step 5: Running Example Queries ==="

echo "Running query: Get definition for the sales table"
QUERY1_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT * FROM pg_table_def WHERE tablename = 'sales';" \
  --query 'Id' --output text)

echo "Query 1 statement ID: $QUERY1_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY1_ID"; then
    handle_error "Query 1 failed"
fi
```

```
# Get and display results
echo "Query 1 results (first 10 rows):"
aws redshift-data get-statement-result --id "$QUERY1_ID" --max-items 10

echo "Running query: Find total sales on a given calendar date"
QUERY2_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT sum(qtysold) FROM sales, date WHERE sales.dateid = date.dateid
 AND caldate = '2008-01-05';" \
  --query 'Id' --output text)

echo "Query 2 statement ID: $QUERY2_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY2_ID"; then
    handle_error "Query 2 failed"
fi

# Get and display results
echo "Query 2 results:"
aws redshift-data get-statement-result --id "$QUERY2_ID"

echo "=== Tutorial Complete ==="
echo "The following resources were created:"
echo "- Redshift Cluster: $CLUSTER_ID"
echo "- IAM Role: $ROLE_NAME"

echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Cleaning up all created resources..."
cleanup_resources
echo "All resources have been cleaned up."

# Securely clear password from memory
DB_PASSWORD=""

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。

- CreateCluster
- CreateRole
- DeleteCluster
- DeleteRole
- DeleteRolePolicy
- DescribeClusters
- GetRole
- ModifyClusterIamRoles
- PutRolePolicy
- 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

工作流程協同運作入門

以下程式碼範例顯示做法：

- 為 Step Functions 建立 IAM 角色
- 建立您的第一個狀態機器
- 啟動您的狀態機器執行
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash
```

```
# AWS Step Functions Getting Started Tutorial Script
# This script creates and runs a Step Functions state machine based on the AWS
 Step Functions Getting Started tutorial

set -euo pipefail

# Security: Restrict umask to prevent unintended file permissions
umask 077

# Parse command line arguments
AUTO_CLEANUP=true
while [[ $# -gt 0 ]]; do
    case $1 in
        --auto-cleanup)
            AUTO_CLEANUP=true
            shift
            ;;
        -h|--help)
            echo "Usage: $0 [--auto-cleanup] [--help]"
            echo "  --auto-cleanup: Automatically clean up resources without
 prompting"
            echo "  --help: Show this help message"
            exit 0
            ;;
        *)
            echo "Unknown option: $1"
            echo "Use --help for usage information"
            exit 1
            ;;
    esac
done

# Set up logging with secure permissions
LOG_FILE="step-functions-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"

# Security: Use process substitution with explicit FD cleanup
exec 3>&1 4>&2
exec > >(tee -a "$LOG_FILE") 2>&1
trap 'exec 1>&3 2>&4 3>&- 4>&-' EXIT

echo "Starting AWS Step Functions Getting Started Tutorial..."
echo "Logging to $LOG_FILE"
```

```
# Verify AWS CLI is installed and configured
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed"
    exit 1
fi

# Verify AWS credentials are configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS credentials are not configured or invalid"
    exit 1
fi

# Check if jq is available for better JSON parsing
if ! command -v jq &> /dev/null; then
    echo "WARNING: jq is not installed. Using basic JSON parsing which may be
 less reliable."
    echo "Consider installing jq for better error handling: brew install jq
 (macOS) or apt-get install jq (Ubuntu)"
    USE_JQ=false
else
    USE_JQ=true
fi

# Use fixed region that supports Amazon Comprehend
CURRENT_REGION="us-west-2"
echo "Using fixed AWS region: $CURRENT_REGION (supports Amazon Comprehend)"

# Set AWS CLI to use the fixed region for all commands
export AWS_DEFAULT_REGION="$CURRENT_REGION"
export AWS_REGION="$CURRENT_REGION"

# Amazon Comprehend is available in us-west-2, so we can always enable it
echo "Amazon Comprehend is available in region $CURRENT_REGION"
SKIP_COMPREHEND=false

# Security: Initialize all resource variables
STATE_MACHINE_ARN=""
ROLE_NAME=""
ROLE_ARN=""
POLICY_ARN=""
STEPFUNCTIONS_POLICY_ARN=""
EXECUTION_ARN=""
EXECUTION2_ARN=""
```

```
EXECUTION3_ARN=""

# Performance: Cache for AWS API calls to reduce redundant requests
declare -A API_CACHE

# Function to make cached AWS CLI calls
aws_call_cached() {
    local cache_key="$1"
    shift

    if [[ -v API_CACHE["$cache_key"] ]]; then
        echo "${API_CACHE[$cache_key]}"
        return 0
    fi

    local result
    result=$(aws "$@" 2>&1) || return $?
    API_CACHE["$cache_key"]="$result"
    echo "$result"
}

# Function to check for API errors in JSON response with optimized jq usage
check_api_error() {
    local response="$1"
    local operation="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        # Use jq for more reliable JSON parsing with efficient error detection
        if echo "$response" | jq -e '.Error // .error // empty' > /dev/null 2>&1;
 then
            local error_message=$(echo "$response" | jq -r
 '.Error.Message // .Error.Code // .error // "Unknown error"' 2>/dev/null)
            handle_error "$operation failed: $error_message"
        fi
    else
        # Fallback to grep-based detection with optimized pattern
        if echo "$response" | grep -qE '"[Ee]rror":|"error":'; then
            handle_error "$operation failed: $response"
        fi
    fi
}

# Function to extract JSON field efficiently
extract_json_field() {
```

```
    local json="$1"
    local field="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        echo "$json" | jq -r "$field" 2>/dev/null
    else
        echo "$json" | grep -oP "\"${field}\":\s*\"\K[^\"]+|\"${field}\":\s*
\K[^,}]+" | head -1
    fi
}

# Function to securely wait for resource propagation with exponential backoff
wait_for_propagation() {
    local resource_type="$1"
    local wait_time="${2:-10}"

    # Validate wait_time is a positive integer
    if ! [[ "$wait_time" =~ ^[0-9]+$ ]] || [ "$wait_time" -lt 1 ] ||
 [ "$wait_time" -gt 300 ]; then
        echo "WARNING: Invalid wait time $wait_time, using default 10 seconds"
        wait_time=10
    fi

    echo "Waiting for $resource_type to propagate ($wait_time seconds)..."
    sleep "$wait_time"
}

# Function to validate JSON file efficiently
validate_json_file() {
    local file="$1"

    if [[ "$USE_JQ" == "true" ]]; then
        if ! jq empty "$file" 2>/dev/null; then
            handle_error "Invalid JSON in $file"
        fi
    fi
}

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created:"
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
        echo "- State Machine: $STATE_MACHINE_ARN"
```

```
    fi
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "- IAM Role: $ROLE_NAME"
    fi
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "- IAM Policy: $POLICY_ARN"
    fi
    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "- Step Functions Policy: $STEPFUNCTIONS_POLICY_ARN"
    fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to securely clean up resources with parallel deletion
cleanup() {
    echo "Cleaning up resources..."

    # Delete state machine if it exists
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
        echo "Deleting state machine: $STATE_MACHINE_ARN"
        aws stepfunctions delete-state-machine --state-machine-arn
 "$STATE_MACHINE_ARN" 2>/dev/null &
    fi

    # Detach and delete policies if they exist
    if [ -n "${POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Comprehend policy $POLICY_ARN from role $ROLE_NAME"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Step Functions policy $STEPFUNCTIONS_POLICY_ARN from role
 $ROLE_NAME"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/null &
    fi

    # Wait for detach operations to complete
    wait 2>/dev/null || true
```

```
    # Delete custom policies if they exist
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "Deleting Comprehend policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "Deleting Step Functions policy: $STEPFUNCTIONS_POLICY_ARN"
        aws iam delete-policy --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/
null &
    fi

    # Wait for policy deletion to complete
    wait 2>/dev/null || true

    # Delete role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || echo "Failed
 to delete role"
    fi

    # Remove temporary files securely
    echo "Removing temporary files"
    local temp_files=(
        "hello-world.json"
        "updated-hello-world.json"
        "sentiment-hello-world.json"
        "step-functions-trust-policy.json"
        "comprehend-policy.json"
        "stepfunctions-policy.json"
        "input.json"
        "sentiment-input.json"
    )

    for file in "${temp_files[@]}"; do
        if [ -f "$file" ]; then
            if command -v shred &> /dev/null; then
                shred -vfz -n 3 "$file" 2>/dev/null || rm -f "$file"
            else
                rm -f "$file"
            fi
        fi
    done
```

```
}

# Security: Set trap to cleanup on script exit
trap cleanup EXIT

# Generate a secure random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
ROLE_NAME="StepFunctionsHelloWorldRole-${RANDOM_ID}"
POLICY_NAME="DetectSentimentPolicy-${RANDOM_ID}"
STATE_MACHINE_NAME="MyFirstStateMachine-${RANDOM_ID}"

echo "Using random identifier: $RANDOM_ID"
echo "Role name: $ROLE_NAME"
echo "Policy name: $POLICY_NAME"
echo "State machine name: $STATE_MACHINE_NAME"

# Step 1: Create the state machine definition
echo "Creating state machine definition..."
cat > hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Result": {
        "IsHelloWorldExample": true,
        "ExecutionWaitTimeInSeconds": 10
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
    "WaitState": {
```

```
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
```

```
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
      "Cause": "The IsHelloWorldExample value was false"
    }
  }
}
EOF

# Create IAM role trust policy
echo "Creating IAM role trust policy..."
cat > step-functions-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "states.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://step-functions-trust-policy.json 2>&1)

check_api_error "$ROLE_RESULT" "Create IAM role"
echo "Role created successfully"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-step-functions-gs

# Get the role ARN
ROLE_ARN=$(extract_json_field "$ROLE_RESULT" ".Role.Arn")

if [ -z "$ROLE_ARN" ]; then
    handle_error "Failed to extract role ARN"
fi
echo "Role ARN: $ROLE_ARN"
```

```
# Create a custom policy for Step Functions with least privilege
echo "Creating custom policy for Step Functions..."
cat > stepfunctions-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution",
        "states:DescribeExecution",
        "states:StopExecution"
      ],
      "Resource": "arn:aws:states:*:*:stateMachine:*"
    }
  ]
}
EOF

# Create the policy
echo "Creating Step Functions policy..."
STEPFUNCTIONS_POLICY_RESULT=$(aws iam create-policy \
  --policy-name "StepFunctionsPolicy-${RANDOM_ID}" \
  --policy-document file://stepfunctions-policy.json 2>&1)

check_api_error "$STEPFUNCTIONS_POLICY_RESULT" "Create Step Functions policy"
echo "Step Functions policy created successfully"

# Get the policy ARN
STEPFUNCTIONS_POLICY_ARN=$(extract_json_field "$STEPFUNCTIONS_POLICY_RESULT"
 ".Policy.Arn")

if [ -z "$STEPFUNCTIONS_POLICY_ARN" ]; then
    handle_error "Failed to extract Step Functions policy ARN"
fi
echo "Step Functions policy ARN: $STEPFUNCTIONS_POLICY_ARN"

# Attach policy to the role
echo "Attaching Step Functions policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>&1)
```

```
if [ $? -ne 0 ]; then
    handle_error "Failed to attach Step Functions policy to role: $ATTACH_RESULT"
fi

# Wait for role to propagate (IAM changes can take time to propagate)
wait_for_propagation "IAM role" 8

# Create state machine
echo "Creating state machine: $STATE_MACHINE_NAME"
SM_RESULT=$(aws stepfunctions create-state-machine \
  --name "$STATE_MACHINE_NAME" \
  --definition file://hello-world.json \
  --role-arn "$ROLE_ARN" \
  --type STANDARD \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-step-functions-gs
 2>&1)

check_api_error "$SM_RESULT" "Create state machine"
echo "State machine created successfully"

# Get the state machine ARN
STATE_MACHINE_ARN=$(extract_json_field "$SM_RESULT" ".stateMachineArn")

if [ -z "$STATE_MACHINE_ARN" ]; then
    handle_error "Failed to extract state machine ARN"
fi
echo "State machine ARN: $STATE_MACHINE_ARN"

# Step 2: Start the state machine execution
echo "Starting state machine execution..."
EXEC_RESULT=$(aws stepfunctions start-execution \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello001-${RANDOM_ID}" 2>&1)

check_api_error "$EXEC_RESULT" "Start execution"
echo "Execution started successfully"

# Get the execution ARN
EXECUTION_ARN=$(extract_json_field "$EXEC_RESULT" ".executionArn")

if [ -z "$EXECUTION_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
echo "Execution ARN: $EXECUTION_ARN"
```

```
# Wait for execution to complete (the workflow has a 10-second wait state)
echo "Waiting for execution to complete (12 seconds)..."
sleep 12

# Check execution status
echo "Checking execution status..."
EXEC_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION_ARN" 2>&1)

echo "Execution status: $EXEC_STATUS"

# Step 3: Update state machine to process external input
echo "Updating state machine to process external input..."
cat > updated-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
```

```
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
      "Cause": "The IsHelloWorldExample value was false"
```

```
    }
  }
}
EOF

# Update state machine
echo "Updating state machine..."
UPDATE_RESULT=$(aws stepfunctions update-state-machine \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --definition file://updated-hello-world.json \
  --role-arn "$ROLE_ARN" 2>&1)

check_api_error "$UPDATE_RESULT" "Update state machine"
echo "State machine updated successfully"

# Create input file with strict validation
echo "Creating input file..."
cat > input.json << 'EOF'
{
  "wait": 5,
  "hello_world": true
}
EOF

# Validate input JSON
validate_json_file "input.json"

# Start execution with input
echo "Starting execution with input..."
EXEC2_RESULT=$(aws stepfunctions start-execution \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello002-${RANDOM_ID}" \
  --input file://input.json 2>&1)

check_api_error "$EXEC2_RESULT" "Start execution with input"
echo "Execution with input started successfully"

# Get the execution ARN
EXECUTION2_ARN=$(extract_json_field "$EXEC2_RESULT" ".executionArn")

if [ -z "$EXECUTION2_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
echo "Execution ARN: $EXECUTION2_ARN"
```

```
# Wait for execution to complete (the workflow has a 5-second wait state)
echo "Waiting for execution to complete (8 seconds)..."
sleep 8

# Check execution status
echo "Checking execution status..."
EXEC2_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION2_ARN" 2>&1)

echo "Execution status: $EXEC2_STATUS"

# Step 4: Integrate Amazon Comprehend for sentiment analysis (if available)
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Creating policy for Amazon Comprehend access with least privilege..."
    cat > comprehend-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "comprehend:DetectSentiment"
      ],
      "Resource": "*"
    }
  ]
}
EOF

    # Create policy
    echo "Creating IAM policy: $POLICY_NAME"
    POLICY_RESULT=$(aws iam create-policy \
      --policy-name "$POLICY_NAME" \
      --policy-document file://comprehend-policy.json 2>&1)

    check_api_error "$POLICY_RESULT" "Create Comprehend policy"
    echo "Comprehend policy created successfully"

    # Get policy ARN
    POLICY_ARN=$(extract_json_field "$POLICY_RESULT" ".Policy.Arn")

    if [ -z "$POLICY_ARN" ]; then
        handle_error "Failed to extract policy ARN"
```

```
    fi
    echo "Policy ARN: $POLICY_ARN"

    # Attach policy to role
    echo "Attaching policy to role..."
    ATTACH2_RESULT=$(aws iam attach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN" 2>&1)

    if [ $? -ne 0 ]; then
        handle_error "Failed to attach policy to role: $ATTACH2_RESULT"
    fi

    # Create updated state machine definition with sentiment analysis
    echo "Creating updated state machine definition with sentiment analysis..."
    cat > sentiment-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example with sentiment analysis",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait",
        "FeedbackComment.$": "$.feedback_comment"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "DetectSentiment"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
```

```
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "DetectSentiment": {
      "Type": "Task",
      "Resource": "arn:aws:states:::aws-sdk:comprehend:detectSentiment",
      "Parameters": {
        "LanguageCode": "en",
        "Text.$": "$.FeedbackComment"
```

```
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    }
  }
}
EOF

    # Validate sentiment state machine JSON
    validate_json_file "sentiment-hello-world.json"

    # Wait for IAM changes to propagate
    wait_for_propagation "IAM changes" 8

    # Update state machine
    echo "Updating state machine with sentiment analysis..."
    UPDATE2_RESULT=$(aws stepfunctions update-state-machine \
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --definition file://sentiment-hello-world.json \
      --role-arn "$ROLE_ARN" 2>&1)

    check_api_error "$UPDATE2_RESULT" "Update state machine with sentiment
 analysis"
    echo "State machine updated with sentiment analysis successfully"

    # Create input file with feedback comment
    echo "Creating input file with feedback comment..."
    cat > sentiment-input.json << 'EOF'
{
  "hello_world": false,
  "wait": 5,
  "feedback_comment": "This getting started with Step Functions workshop is a
 challenge!"
}
EOF

    # Validate sentiment input JSON
    validate_json_file "sentiment-input.json"

    # Start execution with sentiment analysis input
    echo "Starting execution with sentiment analysis input..."
    EXEC3_RESULT=$(aws stepfunctions start-execution \
```

```
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --name "hello003-${RANDOM_ID}" \
      --input file://sentiment-input.json 2>&1)

    check_api_error "$EXEC3_RESULT" "Start execution with sentiment analysis"
    echo "Execution with sentiment analysis started successfully"

    # Get the execution ARN
    EXECUTION3_ARN=$(extract_json_field "$EXEC3_RESULT" ".executionArn")

    if [ -z "$EXECUTION3_ARN" ]; then
        handle_error "Failed to extract execution ARN"
    fi
    echo "Execution ARN: $EXECUTION3_ARN"

    # Wait for execution to complete
    echo "Waiting for execution to complete (3 seconds)..."
    sleep 3

    # Check execution status
    echo "Checking execution status..."
    EXEC3_STATUS=$(aws stepfunctions describe-execution \
      --execution-arn "$EXECUTION3_ARN" 2>&1)

    echo "Execution status: $EXEC3_STATUS"
else
    echo "Skipping Amazon Comprehend integration (not available in
 $CURRENT_REGION)"
    EXECUTION3_ARN=""
fi

# Display summary of resources created
echo ""
echo "================================================"
echo "RESOURCES CREATED"
echo "================================================"
echo "State Machine: $STATE_MACHINE_ARN"
echo "IAM Role: $ROLE_NAME"
echo "Step Functions Policy: StepFunctionsPolicy-${RANDOM_ID}
 ($STEPFUNCTIONS_POLICY_ARN)"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Comprehend Policy: $POLICY_NAME ($POLICY_ARN)"
fi
echo "Executions:"
```

```
echo "  - hello001-${RANDOM_ID}: $EXECUTION_ARN"
echo "  - hello002-${RANDOM_ID}: $EXECUTION2_ARN"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "  - hello003-${RANDOM_ID}: $EXECUTION3_ARN"
fi
echo "========================================="

# Cleanup
echo ""
echo "========================================="
echo "CLEANUP"
echo "========================================="
echo "Auto-cleanup enabled. Cleaning up resources..."

echo "All resources have been cleaned up."

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateStateMachine
  - DeletePolicy
  - DeleteRole
  - DeleteStateMachine
  - DescribeExecution
  - DetachRolePolicy
  - StartExecution
  - UpdateStateMachine

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 管理 IAM 存取金鑰 AWS

下列程式碼範例示範如何管理存取金鑰。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

- 建立並列出存取金鑰。
- 找出上次使用存取金鑰的時間和方式。
- 更新和刪除存取金鑰。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 存取金鑰動作的函數。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
```

```
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
```

```
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response



def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.
```

```
    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

使用包裝函數執行目前使用者的存取金鑰動作。

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
    if len(all_keys) == 2:
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
```

```
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
{new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
{last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的下列主題。
  - CreateAccessKey
  - DeleteAccessKey
  - GetAccessKeyLastUsed
  - ListAccessKeys
  - UpdateAccessKey

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 管理 IAM 政策 AWS

以下程式碼範例顯示做法：

- 建立並列出政策。
- 建立並取得政策版本。

- 將政策復原至先前的版本。
- 刪除政策。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 政策動作的函數。

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
```

```
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
```

```
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version



def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
```

```
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
```

```
        logger.info("Set %s as the default version.",
 rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
 default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
 so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

使用包裝函數來建立政策、更新版本，以及取得有關它們的資訊。

```
def usage_demo():
```

```
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::amzn-s3-demo-bucket"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的下列主題。
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersion
  - GetPolicyVersion
  - ListPolicies
  - ListPolicyVersions
  - SetDefaultPolicyVersion

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 管理 IAM 角色 AWS

以下程式碼範例顯示做法：

- 建立 IAM 角色。
- 連接和分離角色的政策。
- 刪除角色。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 角色動作的函數。

```
import json
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise



def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
```

```
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

使用包裝函數建立角色，然後連接和分離政策。

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
 "batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
    pprint.pprint(role.assume_role_policy_document)
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- 如需 API 的詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的下列主題。
  - AttachRolePolicy
  - CreateRole
  - DeleteRole
  - DetachRolePolicy

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 管理您的 IAM 帳戶 AWS

以下程式碼範例顯示做法：

- 取得並更新帳戶別名。
- 產生使用者及憑證的報告。
- 取得帳戶用量摘要。
- 取得帳戶中所有使用者、群組、角色和政策的詳細資訊，包含這些項目彼此之間的關係。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可包裝 IAM 帳戶動作的函數。

```
import logging
import pprint
import sys
import time
import boto3
from botocore.exceptions import ClientError
```

```
logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]



def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise



def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.
```

```
    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
```

```
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

呼叫包裝函數以變更帳戶別名並取得有關帳戶的報告。

```
def usage_demo():
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
```

```
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
    cred_lines = [
        line.split(",")[:col_count] for line in
 cred_report.decode("utf-8").split("\n")
    ]
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
 ")
    if see_pw_policy.lower() == "y":
```

```
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
            print("Password policy deleted.")

    print("The SAML providers for your account are:")
    list_saml_providers(10)

    print("-" * 88)
    print("Thanks for watching.")
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的下列主題。
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialReport
  - GetAccountAuthorizationDetails
  - GetAccountSummary
  - GetCredentialReport
  - ListAccountAliases

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

將硬式編碼秘密移至安全秘密儲存

以下程式碼範例顯示做法：

- 建立 IAM 角色
- 在 Secrets Manager 中建立秘密
- 更新您的應用程式程式碼
- 更新秘密
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Script to move hardcoded secrets to AWS Secrets Manager
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail

# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "============================================================"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2
```

```
    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED"
    echo "========================================="

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi

    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi

    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
```

```
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done

        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
```

```
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)

# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"
```

```
# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
```

```
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
                }
            }
        }
    ]
}
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)

check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
```

```
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)

check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "========================================"
```

```
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi

echo ""
echo "============================================================"
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""

echo "Script completed at $(date)"
exit 0
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy
  - GetCallerIdentity
  - GetSecretValue
  - PutResourcePolicy
  - UpdateSecret

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

允許 AWS Compute Optimizer 自動化功能套用建議的動作

下列程式碼範例示範如何此以許可為基礎的政策允許 AWS Compute Optimizer 自動化功能套用建議的動作

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aco-automation.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

在整個組織中啟用自動化的政策

下列程式碼範例示範如何將此以許可為基礎的政策啟用整個組織的自動化

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:AssociateAccounts",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:DisassociateAccounts",
            "Resource": "*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:ListAccounts",
            "Resource": "*"
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

為您的帳戶啟用自動化的政策

下列程式碼範例示範如何為您的帳戶啟用以許可為基礎的政策 enablesAutomation

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
```

```
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

授予組織管理帳戶 Compute Optimizer Automation 完整存取權的政策

下列程式碼範例示範如何將此以許可為基礎的政策授予組織管理帳戶對 Compute Optimizer Automation 的完整存取權

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:*",
               "ec2:DescribeVolumes",
               "organizations:ListAccounts",
               "organizations:DescribeOrganization",
               "organizations:DescribeAccount",
               "organizations:EnableAWSServiceAccess",
               "organizations:ListDelegatedAdministrators",
               "organizations:RegisterDelegatedAdministrator",
               "organizations:DeregisterDelegatedAdministrator"
            ],
            "Resource": "*"
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

授予獨立 AWS 帳戶 Compute Optimizer Automation 完整存取權的政策

下列程式碼範例示範如何將此以許可為基礎的政策授予獨立 AWS 帳戶對 Compute Optimizer Automation 的完整存取權

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:*",
            "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

為組織的管理帳戶授予 Compute Optimizer Automation 唯讀存取權的政策

下列程式碼範例示範如何針對組織的管理帳戶，此以許可為基礎的政策授予 Compute Optimizer Automation 的唯讀存取權

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:GetEnrollmentConfiguration",
               "aco-automation:GetAutomationEvent",
               "aco-automation:GetAutomationRule",
```

```
                "aco-automation:ListAccounts",
                "aco-automation:ListAutomationEvents",
                "aco-automation:ListAutomationEventSteps",
                "aco-automation:ListAutomationEventSummaries",
                "aco-automation:ListAutomationRules",
                "aco-automation:ListAutomationRulePreview",
                "aco-automation:ListAutomationRulePreviewSummaries",
                "aco-automation:ListRecommendedActions",
                "aco-automation:ListRecommendedActionSummaries",
                "aco-automation:ListTagsForResource",
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

授予獨立 AWS 帳戶 Compute Optimizer Automation 唯讀存取權的政策

下列程式碼範例示範如何將此以許可為基礎的政策授予獨立 AWS 帳戶的 Compute Optimizer Automation 唯讀存取權

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aco-automation:GetEnrollmentConfiguration",
                "aco-automation:GetAutomationEvent",
                "aco-automation:GetAutomationRule",
                "aco-automation:ListAutomationEvents",
                "aco-automation:ListAutomationEventSteps",
                "aco-automation:ListAutomationEventSummaries",
                "aco-automation:ListAutomationRules",
                "aco-automation:ListAutomationRulePreview",
```

```
                "aco-automation:ListAutomationRulePreviewSummaries",
                "aco-automation:ListRecommendedActions",
                "aco-automation:ListRecommendedActionSummaries",
                "aco-automation:ListTagsForResource",
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

授予運算最佳化自動化服務連結角色許可的政策

下列程式碼範例示範如何將此以許可為基礎的政策授予運算最佳化自動化的服務連結角色許可

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": "iam:PutRolePolicy",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        }
    ]
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 轉返 IAM 政策版本 AWS

以下程式碼範例顯示做法：

- 依日期順序取得政策版本清單。
- 查找預設政策版本。
- 將先前的政策版本設為預設值。
- 刪除舊的預設版本。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
```

```
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的下列主題。
  - DeletePolicyVersion
  - ListPolicyVersions
  - SetDefaultPolicyVersion

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

以下程式碼範例顯示做法：

- 建立 IAM 角色
- 建立 CloudWatch 警示
- 建立實驗範本
- 執行實驗
- 驗證結果
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial

#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "================================================"
```

```
# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
            return 0
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
```

```
    fi

    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true

        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" || true
    fi

    if [ -n "$FIS_ROLE_NAME" ]; then
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
 "$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()
```

```
echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
      ],
      "Resource": "*"
    }
  ]
}
EOF
```

```
# Attach policy to the role
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
cat > ec2-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
```

```
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")

# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"
```

```
# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")

# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
```

```
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"

# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60

# Check alarm state
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"

    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi

echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
```

```
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
```

```
  "stopConditions": [
    {
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
EOF

# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")

echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
```

```
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")

echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail

    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi

    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done

if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi

echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
```

```
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"

# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")

# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
```

```
echo "$METRICS_OUTPUT"

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCES CREATED"
echo "============================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "============================================="

# Prompt for cleanup
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
 "$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"

    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
 "$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"
```

```
# Terminate EC2 instance
echo "Terminating EC2 instance $INSTANCE_ID..."
TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
echo "Waiting for instance to terminate..."
aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"

# Clean up IAM resources
echo "Removing role from instance profile..."
REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"

echo "Deleting instance profile..."
DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

echo "Deleting FIS role policy..."
DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME")
check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

echo "Detaching policy from EC2 role..."
DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

echo "Deleting FIS role..."
DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

echo "Deleting EC2 role..."
DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
  --role-name "$EC2_ROLE_NAME")
check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

# Clean up temporary files
echo "Cleaning up temporary files..."
```

```
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
 experiment-template.json metric-query.json

    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi

echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile
  - DeleteRole
  - DeleteRolePolicy
  - DescribeAlarms
  - DescribeImages
  - DetachRolePolicy
  - GetCallerIdentity
  - GetExperiment
  - GetMetricData
  - GetRole
  - MonitorInstances
  - PutMetricAlarm
  - PutRolePolicy

- RemoveRoleFromInstanceProfile
- RunInstances
- StartExperiment
- StopExperiment
- TerminateInstances
- 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS Command Line Interface v2 設定 DynamoDB 的屬性型存取控制

下列程式碼範例示範如何實作 DynamoDB 的屬性型存取控制 (ABAC)。

- 建立 ABAC 的 IAM 政策。
- 為不同部門建立具有標籤的資料表。
- 根據標籤列出和篩選資料表。

Bash

AWS CLI 使用 Bash 指令碼

建立 ABAC 的 IAM 政策。

```
# Step 1: Create a policy document for ABAC
cat > abac-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
```

```
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}",
          "aws:ResourceTag/Environment": "Development"
        }
      }
    }
  ]
}
EOF

# Step 2: Create the IAM policy
aws iam create-policy \
    --policy-name DynamoDBDepartmentBasedAccess \
    --policy-document file://abac-policy.json
```

為不同部門建立具有標籤的資料表。

```
# Create a DynamoDB table with tags for ABAC
aws dynamodb create-table \
    --table-name FinanceData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Finance \
```

```
        Key=Environment,Value=Development

# Create another table with different tags
aws dynamodb create-table \
    --table-name MarketingData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Marketing \
        Key=Environment,Value=Production
```

根據標籤列出和篩選資料表。

```
# List all DynamoDB tables
echo "Listing all tables:"
aws dynamodb list-tables

# Get ARNs for all tables
echo -e "\nGetting ARNs for all tables:"
TABLE_ARNS=$(aws dynamodb list-tables --query "TableNames[*]" --output text |
 xargs -I {} aws dynamodb describe-table --table-name {} --query "Table.TableArn"
 --output text)

# For each table ARN, list its tags
echo -e "\nListing tags for each table:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    echo -e "\nTags for table: $TABLE_NAME"
    aws dynamodb list-tags-of-resource --resource-arn $ARN
done

# Example: Find tables with a specific tag
echo -e "\nFinding tables with Environment=Production tag:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    TAGS=$(aws dynamodb list-tags-of-resource --resource-arn $ARN --query "Tags[?
Key=='Environment' && Value=='Production']" --output text)
    if [ ! -z "$TAGS" ]; then
        echo "Table with Production tag: $TABLE_NAME"
```

```
    fi
done
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - CreatePolicy
  - CreateTable
  - ListTables

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

設定系統管理

以下程式碼範例顯示做法：

- 建立 Systems Manager 的 IAM 許可
- 建立 Systems Manager 的 IAM 角色
- 設定 Systems Manager
- 驗證設定
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
# Version 17 fixes:
```

```
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}
```

```
# Function to clean up resources on error
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"

        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
                ;;
```

```
            "SSM_CONFIG_MANAGER")
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done

    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================"
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""

# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"

# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."

# Create policy document
```

```
cat > ssm-onboarding-policy.json << 'EOF'
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmReadOnly",
       "Effect": "Allow",
       "Action": [
         "ssm:DescribeAutomationExecutions",
         "ssm:GetAutomationExecution",
         "ssm:ListAssociations",
         "ssm:DescribeAssociation",
         "ssm:ListDocuments",
         "ssm:ListResourceDataSync",
         "ssm:DescribePatchBaselines",
         "ssm:GetPatchBaseline",
         "ssm:DescribeMaintenanceWindows",
         "ssm:DescribeMaintenanceWindowTasks"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmDocument",
       "Effect": "Allow",
       "Action": [
         "ssm:GetDocument",
         "ssm:DescribeDocument"
       ],
       "Resource": [
         "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
         "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
       ]
     },
     {
       "Sid": "SsmEnableExplorer",
       "Effect": "Allow",
```

```
      "Action": "ssm:StartAutomationExecution",
      "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
    },
    {
      "Sid": "SsmExplorerRds",
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsSummary",
        "ssm:CreateResourceDataSync",
        "ssm:UpdateResourceDataSync"
      ],
      "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
    },
    {
      "Sid": "OrgsReadOnly",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListRoots",
        "organizations:ListParents",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAWSServiceAccessForOrganization"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgsAdministration",
      "Effect": "Allow",
      "Action": [
        "organizations:EnableAWSServiceAccess",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:DeregisterDelegatedAdministrator"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "organizations:ServicePrincipal": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "member.org.stacksets.cloudformation.amazonaws.com",
            "resource-explorer-2.amazonaws.com"
```

```
            ]
          }
        }
      },
      {
        "Sid": "CfnReadOnly",
        "Effect": "Allow",
        "Action": [
          "cloudformation:ListStacks",
          "cloudformation:DescribeStacks",
          "cloudformation:ListStackSets",
          "cloudformation:DescribeOrganizationsAccess"
        ],
        "Resource": "*"
      },
      {
        "Sid": "OrgCfnAccess",
        "Effect": "Allow",
        "Action": [
          "cloudformation:ActivateOrganizationsAccess"
        ],
        "Resource": "*"
      },
      {
        "Sid": "CfnStackActions",
        "Effect": "Allow",
        "Action": [
          "cloudformation:CreateStack",
          "cloudformation:DeleteStack",
          "cloudformation:DescribeStackResources",
          "cloudformation:DescribeStackEvents",
          "cloudformation:GetTemplate",
          "cloudformation:RollbackStack",
          "cloudformation:TagResource",
          "cloudformation:UntagResource",
          "cloudformation:UpdateStack"
        ],
        "Resource": [
          "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
          "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
          "arn:aws:cloudformation:*:*:type/resource/*"
        ]
      },
      {
```

```
        "Sid": "CfnStackSetActions",
        "Effect": "Allow",
        "Action": [
          "cloudformation:CreateStackInstances",
          "cloudformation:CreateStackSet",
          "cloudformation:DeleteStackInstances",
          "cloudformation:DeleteStackSet",
          "cloudformation:DescribeStackInstance",
          "cloudformation:DetectStackSetDrift",
          "cloudformation:ListStackInstanceResourceDrifts",
          "cloudformation:DescribeStackSet",
          "cloudformation:DescribeStackSetOperation",
          "cloudformation:ListStackInstances",
          "cloudformation:ListStackSetOperations",
          "cloudformation:ListStackSetOperationResults",
          "cloudformation:TagResource",
          "cloudformation:UntagResource",
          "cloudformation:UpdateStackSet"
        ],
        "Resource": [
          "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
          "arn:aws:cloudformation:*:*:type/resource/*",
          "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
        ]
      },
      {
        "Sid": "ValidationReadonlyActions",
        "Effect": "Allow",
        "Action": [
          "iam:ListRoles",
          "iam:GetRole"
        ],
        "Resource": "*"
      },
      {
        "Sid": "IamRolesMgmt",
        "Effect": "Allow",
        "Action": [
          "iam:CreateRole",
          "iam:DeleteRole",
          "iam:GetRole",
          "iam:AttachRolePolicy",
          "iam:DetachRolePolicy",
          "iam:GetRolePolicy",
```

```
          "iam:ListRolePolicies"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ]
      },
      {
        "Sid": "IamPassRole",
        "Effect": "Allow",
        "Action": [
          "iam:PassRole"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ],
        "Condition": {
          "StringEquals": {
            "iam:PassedToService": [
              "ssm.amazonaws.com",
              "ssm-quicksetup.amazonaws.com",
              "cloudformation.amazonaws.com"
            ]
          }
        }
      },
      {
        "Sid": "IamRolesPoliciesMgmt",
        "Effect": "Allow",
        "Action": [
          "iam:AttachRolePolicy",
          "iam:DetachRolePolicy"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ],
        "Condition": {
          "ArnEquals": {
            "iam:PolicyARN": [
              "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
              "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
```

```
            ]
          }
        }
      },
      {
        "Sid": "CfnStackSetsSLR",
        "Effect": "Allow",
        "Action": [
          "iam:CreateServiceLinkedRole"
        ],
        "Resource": [
          "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
          "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
          "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
          "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
          "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
        ]
      }
    ]
}
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created policy
track_resource "IAM_POLICY" "$POLICY_ARN"
```

```
aws iam tag-policy --policy-arn "$POLICY_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"

# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
```

```
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json

# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"
```

```
echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"

# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF

echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi
```

```
# Track the created configuration manager
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30

# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"

echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "==========================================="
echo "CREATED RESOURCES"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
    echo "Cleanup completed." | tee -a "$LOG_FILE"
```

```
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateConfigurationManager
  - CreatePolicy
  - CreateRole
  - DeleteConfigurationManager
  - DeletePolicy
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetConfigurationManager

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

在監控儀表板中使用屬性變數來監控多個無伺服器函數

以下程式碼範例顯示做法：

- 建立 Lambda 函數以進行監控
- 建立 CloudWatch 儀表板
- 將屬性變數新增至儀表板
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# CloudWatch Dashboard with Lambda Function Variable Script
# This script creates a CloudWatch dashboard with a property variable for Lambda
 function names

set -euo pipefail

# Security: Set restrictive umask
umask 0077

# Set up logging with secure permissions
LOG_FILE="cloudwatch-dashboard-script-v4.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
echo "Starting script execution at $(date)" >> "$LOG_FILE"

# Function to log commands and their output (with sensitive data sanitization)
log_cmd() {
    local cmd="$1"
    local sanitized_cmd="${cmd//--password*/--password [REDACTED]}"
    sanitized_cmd="${sanitized_cmd//--secret*/--secret [REDACTED]}"
    echo "$(date): Running command: $sanitized_cmd" >> "$LOG_FILE"
    eval "$cmd" 2>&1 | tee -a "$LOG_FILE"
    return ${PIPESTATUS[0]}
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"
```

```
    if [ $cmd_status -ne 0 ] || echo "$cmd_output" | grep -qi "error"; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        # Sanitize output before logging
        local sanitized_output="${cmd_output//arn:aws:iam::[0-9]*/
arn:aws:iam::ACCOUNT_ID}"
        echo "Command output: $sanitized_output" | tee -a "$LOG_FILE"
        cleanup_resources
        exit 1
    fi
}

# Trap errors and cleanup
trap 'cleanup_resources' EXIT ERR INT TERM

# Function to clean up resources
cleanup_resources() {
    local exit_code=$?

    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "CLEANUP PROCESS" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"

    if [ -n "${DASHBOARD_NAME:-}" ]; then
        echo "Deleting CloudWatch dashboard: $DASHBOARD_NAME" | tee -a
 "$LOG_FILE"
        aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" 2>&1
 >> "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION1:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION1" 2>&1 >>
 "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION2:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION2" 2>&1 >>
 "$LOG_FILE" || true
    fi

    if [ -n "${ROLE_NAME:-}" ]; then
```

```
        echo "Detaching policy from role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole 2>&1 >>
 "$LOG_FILE" || true

        echo "Deleting IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam delete-role --role-name "$ROLE_NAME" 2>&1 >> "$LOG_FILE" || true
    fi

    # Clean up temporary files securely
    shred -vfz -n 3 trust-policy.json lambda_function.py lambda_function.zip 2>/
dev/null || rm -f trust-policy.json lambda_function.py lambda_function.zip

    echo "Cleanup completed." | tee -a "$LOG_FILE"

    return $exit_code
}

# Validate AWS CLI is installed and authenticated
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed" | tee -a "$LOG_FILE"
    exit 1
fi

if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not properly authenticated" | tee -a "$LOG_FILE"
    exit 1
fi

# Get AWS region with validation
AWS_REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION" | tee -a
 "$LOG_FILE"
else
    echo "Using AWS region: $AWS_REGION" | tee -a "$LOG_FILE"
fi

# Validate region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    echo "ERROR: Invalid AWS region format: $AWS_REGION" | tee -a "$LOG_FILE"
    exit 1
fi
```

```
# Generate unique identifiers using secure random with validation
RANDOM_ID=$(openssl rand -hex 6)
if [ -z "$RANDOM_ID" ] || [ ${#RANDOM_ID} -ne 12 ]; then
    echo "ERROR: Failed to generate valid random identifier" | tee -a "$LOG_FILE"
    exit 1
fi

DASHBOARD_NAME="LambdaMetricsDashboard-${RANDOM_ID}"
LAMBDA_FUNCTION1="TestFunction1-${RANDOM_ID}"
LAMBDA_FUNCTION2="TestFunction2-${RANDOM_ID}"
ROLE_NAME="LambdaExecutionRole-${RANDOM_ID}"

# Validate resource names don't exceed AWS limits
if [ ${#DASHBOARD_NAME} -gt 128 ] || [ ${#LAMBDA_FUNCTION1} -gt 64 ] ||
 [ ${#ROLE_NAME} -gt 64 ]; then
    echo "ERROR: Generated resource names exceed AWS limits" | tee -a "$LOG_FILE"
    exit 1
fi

echo "Using random identifier: $RANDOM_ID" | tee -a "$LOG_FILE"
echo "Dashboard name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "Lambda function names: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "IAM role name: $ROLE_NAME" | tee -a "$LOG_FILE"

# Create IAM role for Lambda functions
echo "Creating IAM role for Lambda..." | tee -a "$LOG_FILE"
TRUST_POLICY='{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}'

echo "$TRUST_POLICY" > trust-policy.json
chmod 600 trust-policy.json
```

```
# Validate JSON before use
if ! python3 -m json.tool trust-policy.json > /dev/null 2>&1; then
    echo "ERROR: Invalid trust policy JSON" | tee -a "$LOG_FILE"
    exit 1
fi

ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name '$ROLE_NAME' --assume-
role-policy-document file://trust-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=cloudwatch-streams --output json")
check_error "$ROLE_OUTPUT" $? "Failed to create IAM role"

ROLE_ARN=$(echo "$ROLE_OUTPUT" | python3 -c "import sys, json;
 print(json.load(sys.stdin)['Role']['Arn'])" 2>/dev/null)
if [ -z "$ROLE_ARN" ]; then
    echo "ERROR: Failed to extract Role ARN" | tee -a "$LOG_FILE"
    exit 1
fi
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Attach Lambda basic execution policy to the role
echo "Attaching Lambda execution policy to role..." | tee -a "$LOG_FILE"
POLICY_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name '$ROLE_NAME' --
policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole")
check_error "$POLICY_OUTPUT" $? "Failed to attach policy to role"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..." | tee -a "$LOG_FILE"
sleep 10

# Create simple Python Lambda function code with security validation
echo "Creating Lambda function code..." | tee -a "$LOG_FILE"
cat > lambda_function.py << 'LAMBDA_EOF'
def handler(event, context):
    print("Lambda function executed successfully")
    return {
        'statusCode': 200,
        'body': 'Success'
    }
LAMBDA_EOF

chmod 600 lambda_function.py

# Validate Python syntax
if ! python3 -m py_compile lambda_function.py 2>/dev/null; then
```

```
    echo "ERROR: Invalid Python syntax in Lambda function" | tee -a "$LOG_FILE"
    exit 1
fi

# Zip the Lambda function code
zip -j -q lambda_function.zip lambda_function.py
if [ ! -f lambda_function.zip ]; then
    echo "ERROR: Failed to create lambda_function.zip" | tee -a "$LOG_FILE"
    exit 1
fi
chmod 600 lambda_function.zip

# Validate zip file integrity
if ! unzip -t lambda_function.zip > /dev/null 2>&1; then
    echo "ERROR: Created zip file is corrupted" | tee -a "$LOG_FILE"
    exit 1
fi

# Create first Lambda function
echo "Creating first Lambda function: $LAMBDA_FUNCTION1..." | tee -a "$LOG_FILE"
LAMBDA1_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION1' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA1_OUTPUT" $? "Failed to create first Lambda function"

# Create second Lambda function
echo "Creating second Lambda function: $LAMBDA_FUNCTION2..." | tee -a "$LOG_FILE"
LAMBDA2_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION2' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA2_OUTPUT" $? "Failed to create second Lambda function"

# Invoke Lambda functions to generate some metrics
echo "Invoking Lambda functions to generate metrics..." | tee -a "$LOG_FILE"
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION1' --payload '{}' /
dev/null" || true
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION2' --payload '{}' /
dev/null" || true

# Create CloudWatch dashboard with property variable
echo "Creating CloudWatch dashboard with property variable..." | tee -a
 "$LOG_FILE"
```

```
# Create dashboard body with proper escaping and validation
DASHBOARD_BODY=$(cat <<EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "$LAMBDA_FUNCTION1" ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$AWS_REGION",
        "title": "Lambda Invocations",
        "period": 300,
        "stat": "Sum"
      }
    }
  ]
}
EOF
)

# Validate JSON before sending
if ! echo "$DASHBOARD_BODY" | python3 -m json.tool > /dev/null 2>&1; then
    echo "ERROR: Dashboard body is not valid JSON" | tee -a "$LOG_FILE"
    exit 1
fi

# First create a basic dashboard without variables
echo "Creating initial dashboard without variables..." | tee -a "$LOG_FILE"
DASHBOARD_OUTPUT=$(aws cloudwatch put-dashboard --dashboard-name
 "$DASHBOARD_NAME" --dashboard-body "$DASHBOARD_BODY" --output json 2>&1)
check_error "$DASHBOARD_OUTPUT" $? "Failed to create initial CloudWatch
 dashboard"

# Now let's try to add a property variable using the console instructions
echo "To complete the tutorial, please follow these steps in the CloudWatch
 console:" | tee -a "$LOG_FILE"
```

```
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/" | tee -a "$LOG_FILE"
echo "2. Navigate to Dashboards and select your dashboard: $DASHBOARD_NAME" | tee
 -a "$LOG_FILE"
echo "3. Choose Actions > Variables > Create a variable" | tee -a "$LOG_FILE"
echo "4. Choose Property variable" | tee -a "$LOG_FILE"
echo "5. For Property that the variable changes, choose FunctionName" | tee -a
 "$LOG_FILE"
echo "6. For Input type, choose Select menu (dropdown)" | tee -a "$LOG_FILE"
echo "7. Choose Use the results of a metric search" | tee -a "$LOG_FILE"
echo "8. Choose Pre-built queries > Lambda > Errors" | tee -a "$LOG_FILE"
echo "9. Choose By Function Name and then choose Search" | tee -a "$LOG_FILE"
echo "10. (Optional) Configure any secondary settings as desired" | tee -a
 "$LOG_FILE"
echo "11. Choose Add variable" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "The dashboard has been created and can be accessed at:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"

# Verify dashboard creation
echo "Verifying dashboard creation..." | tee -a "$LOG_FILE"
VERIFY_OUTPUT=$(aws cloudwatch get-dashboard --dashboard-name "$DASHBOARD_NAME"
 --output json 2>&1)
check_error "$VERIFY_OUTPUT" $? "Failed to verify dashboard creation"

echo "" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "DASHBOARD CREATED SUCCESSFULLY" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "Dashboard Name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "Lambda Functions: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "You can view your dashboard in the CloudWatch console:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"

# Auto-confirm cleanup
echo "" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
```

```
echo "CLEANUP CONFIRMATION" | tee -a "$LOG_FILE"
echo "========================================" | tee -a "$LOG_FILE"
echo "The following resources were created:" | tee -a "$LOG_FILE"
echo "- CloudWatch Dashboard: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
echo "- IAM Role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "Auto-confirming cleanup of all created resources..." | tee -a "$LOG_FILE"

echo "Script completed successfully." | tee -a "$LOG_FILE"
exit 0
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole
  - DetachRolePolicy
  - GetDashboard
  - Invoke
  - PutDashboard

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS Command Line Interface v2 處理 DynamoDB 串流和Time-to-Live

下列程式碼範例示範如何管理 DynamoDB 串流和存留時間功能。

- 建立啟用串流的資料表。
- 描述串流。
- 建立 Lambda 函式以處理串流。
- 啟用資料表上的 TTL。
- 新增具有 TTL 屬性的項目。

- 描述 TTL 設定。

Bash

AWS CLI 使用 Bash 指令碼

建立啟用串流的資料表。

```
# Create a table with DynamoDB Streams enabled
aws dynamodb create-table \
    --table-name StreamsDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES
```

描述串流。

```
# Get information about the stream
aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.StreamSpecification"

# Get the stream ARN
STREAM_ARN=$(aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.LatestStreamArn" \
    --output text)

echo "Stream ARN: $STREAM_ARN"

# Describe the stream
aws dynamodbstreams describe-stream \
    --stream-arn $STREAM_ARN
```

為串流建立 Lambda 函式。

```
# Step 1: Create an IAM role for the Lambda function
```

```
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

aws iam create-role \
    --role-name DynamoDBStreamsLambdaRole \
    --assume-role-policy-document file://trust-policy.json

# Step 2: Attach permissions to the role
aws iam attach-role-policy \
    --role-name DynamoDBStreamsLambdaRole \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AWSLambdaDynamoDBExecutionRole

# Step 3: Create a Lambda function (code would be in a separate file)
echo "Lambda function creation would be done separately with appropriate code"

# Step 4: Create an event source mapping
echo "Example command to create event source mapping:"
echo "aws lambda create-event-source-mapping \\"
echo "    --function-name ProcessDynamoDBRecords \\"
echo "    --event-source $STREAM_ARN \\"
echo "    --batch-size 100 \\"
echo "    --starting-position LATEST"
```

啟用資料表上的 TTL。

```
# Create a table for TTL demonstration
aws dynamodb create-table \
    --table-name TTLDemo \
    --attribute-definitions \
```

```
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST

# Wait for table to become active
aws dynamodb wait table-exists --table-name TTLDemo

# Enable TTL on the table
aws dynamodb update-time-to-live \
    --table-name TTLDemo \
    --time-to-live-specification "Enabled=true, AttributeName=ExpirationTime"
```

新增具有 TTL 屬性的項目。

```
# Calculate expiration time (current time + 1 day in seconds)
EXPIRATION_TIME=$(date -d "+1 day" +%s)

# Add an item with TTL attribute
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item1"},
        "Data": {"S": "This item will expire in 1 day"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME'"}
    }'

# Add an item that expires in 1 hour
EXPIRATION_TIME_HOUR=$(date -d "+1 hour" +%s)
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item2"},
        "Data": {"S": "This item will expire in 1 hour"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME_HOUR'"}
    }'
```

描述 TTL 設定。

```
# Describe TTL settings for a table
aws dynamodb describe-time-to-live \
```

```
    --table-name TTLDemo
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateRole
  - CreateTable
  - DescribeTable
  - DescribeTimeToLive
  - PutItem
  - UpdateTimeToLive

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 搭配 IAM 政策建置器 API AWS

以下程式碼範例顯示做法：

- 使用物件導向 API 建立 IAM 政策。
- 將 IAM 政策產生器 API 與 IAM 服務搭配使用。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

這些範例使用下列匯入。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
```

```
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

建立以時間為基礎的政策。

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_GREATER_THAN)

 .key("aws:CurrentTime")

 .value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_LESS_THAN)

 .key("aws:CurrentTime")

 .value("2020-06-30T23:59:59Z")))
                                .build();
```

```
                // Use an IamPolicyWriter to write out the JSON string to a more
 readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
                                .build());
        }
```

建立具有多個條件的政策。

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

 .addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

 .addAction("dynamodb:BatchWriteItem")

 .addResource("arn:aws:dynamodb:*:*:table/table-name")

 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                        List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))

 .key("dynamodb:Select")
```

```
 .value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

在政策中使用主體。

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
                                                .addPrincipal(IamPrincipal.ALL)
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.ARN_NOT_EQUALS)

 .key("aws:PrincipalArn")

 .value("arn:aws:iam::444455556666:user/user-name")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

允許跨帳戶 存取權。

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)

 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
```

```
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                    .key("s3:x-amz-
acl")
                                                                    .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

建置並上傳 IamPolicy。

```
        public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
                // Build the policy.
                IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                                .addStatement(IamStatement.builder()
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
                                                                    + ":table/
exampleTableName")
                                                .build())
                                .build();
                // Upload the policy.
                iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
                return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

下載並使用 IamPolicy。

```
        public String createNewBasedOnExistingPolicyExample(IamClient iam, String
 accountID, String policyName,
```

```
                        String newPolicyName) {

                String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
policyName;
                GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
r.policyArn(policyArn));

                String policyVersion =
getPolicyResponse.policy().defaultVersionId();
                GetPolicyVersionResponse getPolicyVersionResponse = iam
                                .getPolicyVersion(r ->
r.policyArn(policyArn).versionId(policyVersion));

                // Create an IamPolicy instance from the JSON string returned
from IAM.
                String decodedPolicy =
URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
                                StandardCharsets.UTF_8);
                IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

                /*
                 * All IamPolicy components are immutable, so use the copy method
that creates a
                 * new instance that
                 * can be altered in the same method call.
                 *
                 * Add the ability to get an item from DynamoDB as an additional
action.
                 */
                IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

                // Create a new statement that replaces the original statement.
                IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

                // Upload the new policy. IAM now has both policies.
                iam.createPolicy(r -> r.policyName(newPolicyName)
                                .policyDocument(newPolicy.toJson()));

                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- 如需詳細資訊，請參閱《AWS SDK for Java 2.x 開發人員指南》。
- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
  - CreatePolicy
  - GetPolicy
  - GetPolicyVersion

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS STS 使用 AWS SDKs程式碼範例

下列程式碼範例示範如何使用 AWS STS 搭配 AWS 軟體開發套件 (SDK)。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS STS

- AWS STS 使用 AWS SDKs的基本範例
  - AWS STS 使用 AWS SDKs的動作
    - AssumeRole 搭配 AWS SDK 或 CLI 使用
    - 搭配使用 AssumeRoleWithWebIdentity 與 CLI
    - 搭配使用 DecodeAuthorizationMessage 與 CLI
    - 搭配使用 GetFederationToken 與 CLI
    - GetSessionToken 搭配 AWS SDK 或 CLI 使用
- AWS STS 使用 AWS SDKs案例
  - 使用 SDK 搭配 擔任需要 MFA 字符 AWS STS 的 IAM 角色 AWS
  - 設定容器服務連線

- 使用 AWS SDK AWS STS 為聯合身分使用者使用 建構 URL
- 為無伺服器啟動類型建立容器任務
- 使用函數代理整合建立靜態 API
- 為虛擬機器執行個體建立容器服務
- 建立受管監控工作區
- 建立以函數名稱做為變數的監控儀表板
- AWS STS 使用 AWS SDK 透過 取得需要 MFA 字符的工作階段字符
- 組態管理入門
- 容器登錄檔入門
- 開始使用物聯網簡訊
- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門
- 受管串流入門
- 推送通知入門
- 搜尋和分析引擎入門
- Web 應用程式防火牆入門
- 將硬式編碼秘密移至安全秘密儲存
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 設定系統管理

AWS STS 使用 AWS SDKs的基本範例

下列程式碼範例示範如何 AWS Security Token Service 搭配 AWS SDKs 使用 的基本概念。

範例

- AWS STS 使用 AWS SDKs的動作
  - AssumeRole 搭配 AWS SDK 或 CLI 使用
  - 搭配使用 AssumeRoleWithWebIdentity 與 CLI
  - 搭配使用 DecodeAuthorizationMessage 與 CLI
  - 搭配使用 GetFederationToken 與 CLI
  - GetSessionToken 搭配 AWS SDK 或 CLI 使用

AWS STS 使用 AWS SDKs的動作

下列程式碼範例示範如何使用 AWS SDKs執行個別 AWS STS 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 AWS STS API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 AWS STS 使用 AWS SDKs案例 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《AWS Security Token Service API 參考》。

範例

- AssumeRole 搭配 AWS SDK 或 CLI 使用
- 搭配使用 AssumeRoleWithWebIdentity 與 CLI
- 搭配使用 DecodeAuthorizationMessage 與 CLI
- 搭配使用 GetFederationToken 與 CLI
- GetSessionToken 搭配 AWS SDK 或 CLI 使用

AssumeRole 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `AssumeRole`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 擔任需要 MFA 字符的 IAM 角色
- 為聯合身分使用者建構 URL

.NET

適用於 .NET 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
using System;
```

```
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
```

```
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 適用於 .NET 的 AWS SDK API Reference 中的 AssumeRole。

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
```

```
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
##############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
```

```
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AssumeRole。

C++

適用於 C++ 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }
```

```
    return outcome.IsSuccess();
}
```

- 如需 API 詳細資訊，請參閱 適用於 C++ 的 AWS SDK API Reference 中的 AssumeRole。

CLI

AWS CLI

擔任角色

下列 assume-role 命令會為 IAM 角色 s3-access-example 擷取一組短期憑證。

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

輸出：

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

該命令的輸出包含存取金鑰、私密金鑰以及可用來向 AWS進行驗證的工作階段字符。

對於 AWS CLI 使用，您可以設定與角色相關聯的具名設定檔。當您使用設定檔時，CLI AWS 會呼叫 assume-role 並為您管理登入資料。如需詳細資訊，請參閱《CLI 使用者指南》中的在 AWS CLI 中使用 IAM 角色。 AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AssumeRole。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
```

```
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, arn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }
```

```
    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API Reference 中的 AssumeRole。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立用戶端。

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

擔任 IAM 角色。

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
```

```
  } catch (err) {
    console.error(err);
  }
};
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 AssumeRole。

SDK for JavaScript (v2)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
```

```
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- 如需 API 詳細資訊，請參閱 適用於 JavaScript 的 AWS SDK API Reference 中的 AssumeRole。

PowerShell

Tools for PowerShell V4

範例 1：傳回一組臨時登入資料（存取金鑰、私密金鑰和工作階段字符），可用於存取請求使用者通常無法存取 AWS 的資源一小時。傳回的憑證擁有所擔任角色之存取政策和提供的政策所允許的許可 (您無法使用提供的政策來授予超過所擔任角色之存取政策所定義的許可)。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

範例 2：傳回一組臨時憑證，有效期為一小時，這些憑證擁有所擔任角色之存取政策中定義的相同許可。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

範例 3：傳回一組臨時憑證，提供序號，以及與用於執行 cmdlet 之使用者憑證相關聯的 MFA 產生的權杖。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

範例 4：傳回一組臨時憑證，這些臨時憑證擔任了客戶帳戶中定義的角色。對於第三方可以擔任的每個角色，客戶帳戶必須使用識別符來建立角色，必須在每次擔任角色時在 -ExternalId 參數中傳遞該識別符。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AssumeRole。

Tools for PowerShell V5

範例 1：傳回一組臨時登入資料（存取金鑰、私密金鑰和工作階段字符），可用於存取請求使用者通常無法存取 AWS 的資源一小時。傳回的憑證擁有所擔任角色之存取政策和提供的政策所允許的許可 (您無法使用提供的政策來授予超過所擔任角色之存取政策所定義的許可)。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

範例 2：傳回一組臨時憑證，有效期為一小時，這些憑證擁有所擔任角色之存取政策中定義的相同許可。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

範例 3：傳回一組臨時憑證，提供序號，以及與用於執行 cmdlet 之使用者憑證相關聯的 MFA 產生的權杖。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

範例 4：傳回一組臨時憑證，這些臨時憑證擔任了客戶帳戶中定義的角色。對於第三方可以擔任的每個角色，客戶帳戶必須使用識別符來建立角色，必須在每次擔任角色時在 -ExternalId 參數中傳遞該識別符。

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AssumeRole。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

擔任需要 MFA 字符的 IAM 角色，並使用暫時性憑證列出該帳戶的 Amazon S3 儲存貯體。

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
```

```
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 AssumeRole。

Ruby

SDK for Ruby

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
```

```
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

- 如需 API 詳細資訊，請參閱 適用於 Ruby 的 AWS SDK API Reference 中的 AssumeRole。

Rust

適用於 Rust 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
```

```
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的AWS SDK API 參考》中的 AssumeRole。

Swift

適用於 Swift 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
```

```
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- 如需 API 詳細資訊，請參閱 《適用於 Swift 的AWS SDK API 參考》中的 AssumeRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **AssumeRoleWithWebIdentity** 與 CLI

下列程式碼範例示範如何使用 AssumeRoleWithWebIdentity。

CLI

AWS CLI

取得使用 Web Identity (OAuth 2."0) 驗證之角色的短期憑證

下列 assume-role-with-web-identity 命令會為 IAM 角色 app1 擷取一組短期憑證。使用指定 Web 身分提供程式提供的 Web 身分權杖來驗證身分。兩個額外的政策會套用至工作階段，進一步限制使用者可以執行的動作。傳回的憑證會在產生後一小時過期。

```
aws sts assume-role-with-web-identity \
    --duration-seconds 3600 \
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/
webidentitydemopolicy2" \
```

```
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

輸出：

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A",
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    },
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的請求臨時安全憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AssumeRoleWithWebIdentity。

PowerShell

Tools for PowerShell V4

範例 1：針對已使用 Login with Amazon 身分提供程式進行身分驗證的使用者，傳回一組臨時憑證，有效期為一小時。憑證會擔任與角色 ARN 所識別角色相關聯的存取政策。或者，您可以將 JSON 政策傳遞至 -Policy 參數，以進一步改進存取許可 (您授予的許可不得超過與角色相關聯的許可中可用的許可)。提供給 -WebIdentityToken 的值是身分提供程式傳回的唯一使用者識別符。

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 AssumeRoleWithWebIdentity。

Tools for PowerShell V5

範例 1：針對已使用 Login with Amazon 身分提供程式進行身分驗證的使用者，傳回一組臨時憑證，有效期為一小時。憑證會擔任與角色 ARN 所識別角色相關聯的存取政策。或者，您可以將 JSON 政策傳遞至 -Policy 參數，以進一步改進存取許可 (您授予的許可不得超過與角色相關聯的許可中可用的許可)。提供給 -WebIdentityToken 的值是身分提供程式傳回的唯一使用者識別符。

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 AssumeRoleWithWebIdentity。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DecodeAuthorizationMessage** 與 CLI

下列程式碼範例示範如何使用 DecodeAuthorizationMessage。

CLI

AWS CLI

對回應請求時傳回的編碼授權訊息進行解碼

以下 decode-authorization-message 範例將編碼訊息中與請求授權狀態相關的更多資訊進行解碼，以回應 Amazon Web Services 請求。

```
aws sts decode-authorization-message \
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3
```

```
0jau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKE
```

輸出：

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
\"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
\":\"DENY\",\"principals\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE
\"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
\":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
\"}]},\"conditions\":{\"items\":[]}}]},\"failures\":{\"items\":[]},
\"context\":{\"principal\":{\"id\":\"AROA123456789EXAMPLE:Ana\",\"arn
\":\"arn:aws:sts::111122223333:assumed-role/Developer/Ana\"},\"action\":
\"RunInstances\",\"resource\":\"arn:aws:ec2:us-east-1:111122223333:instance/*
\",\"conditions\":{\"items\":[{\"key\":\"ec2:MetadataHttpPutResponseHopLimit\",
\"values\":{\"items\":[{\"value\":\"2\"}]}},{\"key\":\"ec2:InstanceMarketType
\",\"values\":{\"items\":[{\"value\":\"on-demand\"}]}},{\"key\":\"aws:Resource
\",\"values\":{\"items\":[{\"value\":\"instance/*\"}]}},{\"key\":\"aws:Account
\",\"values\":{\"items\":[{\"value\":\"111122223333\"}]}},{\"key\":
\"ec2:AvailabilityZone\",\"values\":{\"items\":[{\"value\":\"us-east-1f\"}]}},
{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items\":[{\"value\":\"false\"}]}},
{\"key\":\"ec2:IsLaunchTemplateResource\",\"values\":{\"items\":[{\"value\":
\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":{\"items\":[{\"value
\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values\":{\"items\":
[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values\":{\"items
\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":{\"items
\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items\":
[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items
\":[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items
\":[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":
[{\"value\":\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}}"
}
```

如需詳細資訊，請參閱 AWS IAM User Guide 中的 Policy evaluation logic。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DecodeAuthorizationMessage。

PowerShell

Tools for PowerShell V4

範例 1：解碼回應請求時傳回的所提供編碼訊息內容中包含的額外資訊。會對額外資訊進行編碼，因為授權狀態的詳細資訊可能構成請求該動作的使用者不應看到的特權資訊。

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 DecodeAuthorizationMessage。

Tools for PowerShell V5

範例 1：解碼回應請求時傳回的所提供編碼訊息內容中包含的額外資訊。會對額外資訊進行編碼，因為授權狀態的詳細資訊可能構成請求該動作的使用者不應看到的特權資訊。

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 DecodeAuthorizationMessage。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **`GetFederationToken`** 與 CLI

下列程式碼範例示範如何使用 `GetFederationToken`。

CLI

AWS CLI

使用 IAM 使用者存取金鑰憑證傳回一組臨時安全憑證

以下 `get-federation-token` 範例會為使用者傳回一組臨時安全憑證 (包括存取金鑰 ID、私密存取金鑰和安全權杖)。必須使用 IAM 使用者的長期安全憑證來呼叫 `GetFederationToken` 操作。

```
aws sts get-federation-token \
```

```
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

myfile.json 的內容：

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

輸出：

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
```

```
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
    },
    "PackedPolicySize": 36
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的請求臨時安全憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetFederationToken。

PowerShell

Tools for PowerShell V4

範例 1：使用 "Bob" 作為聯合身分使用者的名稱，請求有效期為一小時的聯合權杖。此名稱可用於參考資源型政策 (例如 Amazon S3 儲存貯體政策) 中的聯合身分使用者名稱。提供的 JSON 格式的 IAM 政策用於縮小 IAM 使用者可用的許可範圍。提供的政策授予的許可不得超過授予給請求使用者的許可，根據傳遞的政策和 IAM 使用者政策的交集，聯合身分使用者的最終許可是最具限制性的集合。

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetFederationToken。

Tools for PowerShell V5

範例 1：使用 "Bob" 作為聯合身分使用者的名稱，請求有效期為一小時的聯合權杖。此名稱可用於參考資源型政策 (例如 Amazon S3 儲存貯體政策) 中的聯合身分使用者名稱。提供的 JSON 格式的 IAM 政策用於縮小 IAM 使用者可用的許可範圍。提供的政策授予的許可不得超過授予給請求使用者的許可，根據傳遞的政策和 IAM 使用者政策的交集，聯合身分使用者的最終許可是最具限制性的集合。

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetFederationToken。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetSessionToken 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetSessionToken。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 獲取需要 MFA 字符的工作階段字符

CLI

AWS CLI

為 IAM 身分取得一組短期憑證

下列 get-session-token 命令會為進行呼叫的 IAM 身分擷取一組短期憑證。產生的憑證可用於政策要求多重要素驗證 (MFA) 的請求。憑證會在產生後的 15 分鐘過期。

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

輸出：

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
```

```
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

如需詳細資訊，請參閱《AWS IAM 使用者指南》中的請求臨時安全憑證。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetSessionToken。

PowerShell

Tools for PowerShell V4

範例 1：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為設定時間段的臨時憑證。從目前 Shell 預設值中推斷出用於請求臨時憑證的憑證。若要指定其他憑證，請使用 -ProfileName 或 -AccessKey/-SecretKey 參數。

```
Get-STSSessionToken
```

輸出：

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

範例 2：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。從指定的設定檔中取得用於提出請求的憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

輸出：

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

```
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

範例 3：使用在設定檔 'myprofilename' 中指定其憑證的帳戶關聯的 MFA 裝置識別碼和裝置提供的值，傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

輸出：

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 GetSessionToken。

Tools for PowerShell V5

範例 1：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為設定時間段的臨時憑證。從目前 Shell 預設值中推斷出用於請求臨時憑證的憑證。若要指定其他憑證，請使用 -ProfileName 或 -AccessKey/-SecretKey 參數。

```
Get-STSSessionToken
```

輸出：

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

範例 2：傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。從指定的設定檔中取得用於提出請求的憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

輸出：

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

範例 3：使用在設定檔 'myprofilename' 中指定其憑證的帳戶關聯的 MFA 裝置識別碼和裝置提供的值，傳回 **Amazon.RuntimeAWSCredentials** 執行個體，其中包含有效期為一小時的臨時憑證。

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

輸出：

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 GetSessionToken。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

透過傳遞 MFA 字符取得工作階段字符，並使用它列出該帳戶的 Amazon S3 儲存貯體。

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetSessionToken。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS STS 使用 AWS SDKs案例

下列程式碼範例示範如何在 AWS SDKs中 AWS STS 實作常見案例。這些案例說明如何透過在 中呼叫多個函數 AWS STS 或與其他函數結合，來完成特定任務 AWS 服務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- 使用 SDK 搭配 擔任需要 MFA 字符 AWS STS 的 IAM 角色 AWS
- 設定容器服務連線
- 使用 AWS SDK AWS STS 為聯合身分使用者使用 建構 URL
- 為無伺服器啟動類型建立容器任務
- 使用函數代理整合建立靜態 API
- 為虛擬機器執行個體建立容器服務
- 建立受管監控工作區
- 建立以函數名稱做為變數的監控儀表板
- AWS STS 使用 AWS SDK 透過 取得需要 MFA 字符的工作階段字符
- 組態管理入門
- 容器登錄檔入門
- 開始使用物聯網簡訊
- 機器學習功能存放區的入門
- 受管 kubernetes 叢集入門
- 受管串流入門
- 推送通知入門
- 搜尋和分析引擎入門
- Web 應用程式防火牆入門
- 將硬式編碼秘密移至安全秘密儲存
- 使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試
- 設定系統管理

使用 SDK 搭配 擔任需要 MFA 字符 AWS STS 的 IAM 角色 AWS

下列程式碼範例示範如何擔任需要 MFA 權杖的角色。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

- 建立一個可授予許可的 IAM 角色，以列出 Amazon S3 儲存貯體。
- 建立 IAM 使用者，該使用者只有在提供 MFA 憑證時才具有擔任該角色的許可。
- 為使用者註冊 MFA 裝置。
- 擔任角色並使用暫時性憑證列出 S3 儲存貯體。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立 IAM 使用者，註冊 MFA 裝置，並建立一個可授予許可以列出 S3 儲存貯體的角色。使用者只有擔任該角色的權利。

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role and requires
 MFA.
    Creates a policy that allows listing Amazon S3 buckets.
```

```
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    For demonstration purposes, the user is created in the same account as the
role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")
```

```
    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")
```

```
    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

說明如果沒有 MFA 字符，不允許擔任角色。

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
 results
    in an AccessDenied error.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
```

```
    try:
        sts_client.assume_role(RoleArn=assume_role_arn,
 RoleSessionName=session_name)
        raise RuntimeError("Expected AccessDenied error.")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Got AccessDenied.")
        else:
            raise
```

擔任可授予許可以列出 S3 儲存貯體的角色，傳遞所需 MFA 字符，並顯示可列出的儲存貯體。

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
```

```
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

銷毀為示範所建立的資源。

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

使用先前定義的函數執行此案例。

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 AssumeRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

設定容器服務連線

以下程式碼範例顯示做法：

- 建立 VPC 基礎設施
- 設定記錄
- 建立 ECS 叢集
- 設定 IAM 角色
- 使用 Service Connect 建立服務
- 驗證部署
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
```

```
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
```

```
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
```

```
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"

    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
```

```
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
```

```
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
```

```
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
```

```
        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."

    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
```

```
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"

    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
```

```
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
```

```
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"

    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
```

```
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi

    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "============================================="
    echo "CREATED RESOURCES"
    echo "============================================="
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "============================================="
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
```

```
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
```

```
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json
```

```
    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups

    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources

    # Ask user if they want to clean up
    echo ""
    echo "============================================="
    echo "CLEANUP CONFIRMATION"
    echo "============================================="
    echo "Do you want to clean up all created resources? (y/n): "
```

```
    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
main "$@"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteSecurityGroup
  - DeleteService (AWS Cloud Map)
  - DeleteService (Amazon ECS)
  - DeregisterTaskDefinition

- DescribeClusters
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNamespaces
- ListServices
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- 等待 (Amazon EC2)
- 等待 (Amazon ECS)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK AWS STS 為聯合身分使用者使用 建構 URL

以下程式碼範例顯示做法：

- 建立可對目前帳戶的 Amazon S3 資源授予唯讀存取權的 IAM 角色。
- 從 AWS 聯合端點取得安全字符。
- 建構可用來使用聯合憑證存取主控台的 URL。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立可對目前帳戶的 S3 資源授予唯讀存取權的角色。

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
                         that has the permission to create a role.
    :return: The newly created role.
    """
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")

    print("Give AWS time to propagate these new resources and connections.",
 end="")
```

```
    progress_bar(10)

    return role
```

從 AWS 聯合端點取得安全字符，並建構可用於使用聯合憑證存取主控台的 URL。

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
 Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
 that
       can be used to assume a role with limited permissions.
    2. Uses the temporary credentials to request a sign-in token from the
       AWS federation endpoint.
    3. Builds a URL that can be used in a browser to navigate to the AWS
 federation
       endpoint, includes the sign-in token for authentication, and redirects to
       the AWS Management Console with permissions defined by the role that was
       specified in step 1.

    :param assume_role_arn: The role that specifies the permissions that are
 granted.
                            The current user must have permission to assume the
 role.
    :param session_name: The name for the STS session.
    :param issuer: The organization that issues the URL.
    :param sts_client: A Boto3 STS instance that can assume the role.
    :return: The federated URL.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn, RoleSessionName=session_name
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    session_data = {
        "sessionId": temp_credentials["AccessKeyId"],
```

```
        "sessionKey": temp_credentials["SecretAccessKey"],
        "sessionToken": temp_credentials["SessionToken"],
    }
    aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

    # Make a request to the AWS federation endpoint to get a sign-in token.
    # The requests.get function URL-encodes the parameters and builds the query
 string
    # before making the request.
    response = requests.get(
        aws_federated_signin_endpoint,
        params={
            "Action": "getSigninToken",
            "SessionDuration": str(datetime.timedelta(hours=12).seconds),
            "Session": json.dumps(session_data),
        },
    )
    signin_token = json.loads(response.text)
    print(f"Got a sign-in token from the AWS sign-in federation endpoint.")

    # Make a federated URL that can be used to sign into the AWS Management
 Console.
    query_string = urllib.parse.urlencode(
        {
            "Action": "login",
            "Issuer": issuer,
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

銷毀為示範所建立的資源。

```
def teardown(role):
    """
    Removes all resources created during setup.

    :param role: The demo role.
```

```
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

使用先前定義的函數執行此案例。

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 AssumeRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

為無伺服器啟動類型建立容器任務

以下程式碼範例顯示做法：

- 建立叢集
- 建立任務定義
- 建立 服務
- 清除

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error

# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1
```

```
echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
execute_command() {
    local cmd="$1"
    local description="$2"

    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "==========================================="
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "==========================================="

    local output
    local exit_code
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
        echo "OUTPUT: $output"
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
```

```
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
        echo "Output: $output"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."
```

```
    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
            echo "No network interfaces found using security group
 $security_group_id"
            return 0
        else
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}

# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi
```

```
    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "============================================"
    echo "CLEANUP PROCESS"
    echo "============================================"
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "Auto-confirming cleanup of all created resources..."

    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
```

```
        echo "Starting cleanup process..."

        # Step 1: Scale service to 0 tasks first, then delete service
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
then
            echo ""
            echo "Step 1: Scaling service to 0 tasks..."
            if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
                echo "Waiting for service to stabilize after scaling to 0..."
                execute_command "aws ecs wait services-stable --cluster
'$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"

                echo "Deleting service..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME'" "Delete ECS service"
            else
                echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
            fi
        fi

        # Step 2: Wait a bit for tasks to fully terminate
        echo ""
        echo "Step 2: Waiting for tasks to fully terminate..."
        sleep 15

        # Step 3: Delete cluster
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
            echo ""
            echo "Step 3: Deleting cluster..."
            execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
"Delete ECS cluster"
        fi

        # Step 4: Wait for network interfaces to be cleaned up, then delete
security group
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."
```

```
            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."

            # Get all revisions of the task definition
            local revisions
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "============================================"
        echo "CLEANUP COMPLETED"
        echo "============================================"
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
        echo ""
        echo "To clean up manually later, use the following commands in order:"
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
```

```
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
}

# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "==========================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "==========================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
```

```
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"

    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi

# Step 2: Create ECS cluster
echo ""
echo "============================================="
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================="
```

```
CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "=========================================="
echo "STEP 3: CREATE TASK DEFINITION"
echo "=========================================="

# Create task definition JSON with validated inputs
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
```

```
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

# Validate JSON before using
if ! jq empty task-definition.json 2>/dev/null; then
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "============================================="
echo "STEP 4: SET UP NETWORKING"
echo "============================================="

# Get default VPC ID
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi
```

```
# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi

echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
```

```
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "==========================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "==========================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"

echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"

CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "==========================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "==========================================="
```

```
echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi

echo "Task ARN: $TASK_ARN"

# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"

# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
```

```
    else
        echo ""
        echo "==========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "==========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
echo "==========================================="
echo "SERVICE INFORMATION"
echo "==========================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "==========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "==========================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi

echo ""
echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateRole

- CreateSecurityGroup
- CreateService
- DeleteCluster
- DeleteSecurityGroup
- DeleteService
- DeregisterTaskDefinition
- DescribeNetworkInterfaces
- DescribeSecurityGroups
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetRole
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用函數代理整合建立靜態 API

以下程式碼範例顯示做法：

- 建立用於 Lambda 執行的 IAM 角色
- 建立和部署 Lambda 函數
- 建立 REST API
- 設定 Lambda 代理整合
- 部署和測試 API

- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."

# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
```

```
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")
```

```
        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration

# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."
```

```
# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}
```

```
# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}

echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"
```

```
# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."

# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2

# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2
```

```
aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddPermission
  - AttachRolePolicy
  - CreateDeployment
  - CreateFunction
  - CreateResource
  - CreateRestApi
  - CreateRole
  - DeleteFunction
  - DeleteRestApi
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetResources
  - GetRestApis
  - PutIntegration
  - PutMethod

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

為虛擬機器執行個體建立容器服務

以下程式碼範例顯示做法：

- 建立和監控服務
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")
```

```
# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"

    echo ""
    echo "========================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "========================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}

# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
```

```
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi

        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Deregister task definition
    if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
        log "Deregistering task definition: $TASK_DEFINITION_ARN"
        if ! aws ecs deregister-task-definition --task-definition
 "$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to deregister task definition
 $TASK_DEFINITION_ARN"
            ((cleanup_errors++))
        fi
    fi

    # Terminate EC2 instance
    if [[ -n "${INSTANCE_ID:-}" ]]; then
        log "Terminating EC2 instance: $INSTANCE_ID"
```

```
        if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
            log "WARNING: Failed to terminate instance $INSTANCE_ID"
            ((cleanup_errors++))
        else
            log "Waiting for instance to terminate..."
            if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
                log "WARNING: Instance termination wait failed for $INSTANCE_ID"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Delete security group with retry logic
    if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
        log "Deleting security group: $SECURITY_GROUP_ID"
        local retry_count=0
        local max_retries=3

        while [[ $retry_count -lt $max_retries ]]; do
            if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
                log "Successfully deleted security group"
                break
            else
                ((retry_count++))
                if [[ $retry_count -lt $max_retries ]]; then
                    log "Retry $retry_count/$max_retries: Waiting 10 seconds
before retrying security group deletion..."
                    sleep 10
                else
                    log "ERROR: Failed to delete security group after
$max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
```

```
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
 remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
```

```
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"

    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
    log "Creating ECS cluster: $CLUSTER_NAME"

    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}
```

```
# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")

    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)

    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
```

```
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."

        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
```

```
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole

        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
```

```
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."

    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")
```

```
    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
        exit 1
    fi

    rm -f ecs-user-data.sh
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
```

```
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
```

```
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi

    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi

    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
```

```
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"

    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}

# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun

    # List tasks
    log "Listing tasks in service:"
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table
```

```
    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "============================================="
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "============================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
```

```
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "==========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "==========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
```

```
}

# Run main function
main "$@"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeContainerInstances
  - DescribeInstances
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - GetCallerIdentity
  - GetParameters
  - GetRole
  - ListContainerInstances
  - ListServices
  - ListTasks

- RegisterTaskDefinition
- RunInstances
- StopTask
- TerminateInstances
- UpdateService
- 等待 (Amazon EC2)
- 等待 (Amazon ECS)
- 等待 (IAM)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立受管監控工作區

以下程式碼範例顯示做法：

- 為您的工作區建立 IAM 角色
- 建立 Grafana 工作區
- 設定身分驗證
- 設定選用設定
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it
```

```
# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."

    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
```

```
    fi

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"

echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."

# Create trust policy document
cat > trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
```

```
  --assume-role-policy-document file://trust-policy.json \
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"
```

```
ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)

if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
```

```
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi

# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "==========================================="
echo "WORKSPACE INFORMATION"
echo "==========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
```

```
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "========================================="

# Step 6: Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
 grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done
```

```
    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
 still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace

- DescribeWorkspace
- DetachRolePolicy
- GetCallerIdentity

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立以函數名稱做為變數的監控儀表板

以下程式碼範例顯示做法：

- 建立 CloudWatch 儀表板
- 使用函數名稱變數新增 Lambda 指標小工具
- 驗證儀表板
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "$(date): Starting CloudWatch dashboard creation script"
```

```
# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "An error occurred. Proceeding with automatic cleanup..."

    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true

    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi

# Get the current region securely
```

```
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")

if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."

    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi

    if ! zip -q function.zip index.js; then
```

```
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
```

```
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null

    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
```

```
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
```

```
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
```

```
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi

# Verify the dashboard was created
echo "Verifying dashboard creation..."
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
```

```
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"
```

```
if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi
```

```
    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetDashboard
  - Invoke
  - ListDashboards
  - ListFunctions
  - PutDashboard

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS STS 使用 AWS SDK 透過 取得需要 MFA 字符的工作階段字符

下列程式碼範例示範如何獲取需要 MFA 權杖的工作階段字符。

> **Warning**
>
> 為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 AWS IAM Identity Center。

- 建立一個可授予許可的 IAM 角色，以列出 Amazon S3 儲存貯體。
- 建立 IAM 使用者，該使用者只有在提供 MFA 憑證時才具有擔任該角色的許可。
- 為使用者註冊 MFA 裝置。
- 提供 MFA 憑證以取得工作階段權杖，並使用暫時性憑證列出 S3 儲存貯體。

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

建立 IAM 使用者，註冊 MFA 裝置，並建立角色，授予許可以便在使用 MFA 憑證時讓使用者僅列出 S3 儲存貯體。

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
```

```
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
```

```
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
 True}},
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

透過傳遞 MFA 字符獲取暫時性工作階段憑證，並使用憑證列出該帳戶的 S3 儲存貯體。

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
```

```
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

銷毀為示範所建立的資源。

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

使用先前定義的函數執行此案例。

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 GetSessionToken。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

組態管理入門

以下程式碼範例顯示做法：

- 建立 Amazon S3 儲存貯體
- 建立 Amazon SNS 主題
- 為 Config 建立 IAM 角色
- 設定 Config 組態記錄器
- 設定 Config 交付管道
- 啟動組態記錄器
- 驗證 Config 設定

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI

# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
```

```
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
    fi

    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
```

```
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}

# Function to display created resources
display_resources() {
    echo ""
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
```

```
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "============================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
```

```
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"

# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"

echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15

# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"

echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
```

```
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"

if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
```

```
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"

    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"
```

```
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"

echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"

echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
```

```
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder
  - DeleteDeliveryChannel
  - DeleteRole
  - DeleteRolePolicy
  - DeleteTopic
  - DescribeConfigurationRecorderStatus
  - DescribeConfigurationRecorders
  - DescribeDeliveryChannels
  - DetachRolePolicy
  - GetCallerIdentity
  - PutConfigurationRecorder
  - PutDeliveryChannel
  - PutPublicAccessBlock
  - PutRolePolicy

- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

容器登錄檔入門

以下程式碼範例顯示做法：

- 建立 Docker 映像
- 建立 Amazon ECR 儲存庫
- 刪除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon ECR Getting Started Script
# This script demonstrates the lifecycle of a Docker image in Amazon ECR

# Set up logging
LOG_FILE="ecr-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "Amazon ECR Getting Started Tutorial"
echo "================================================="
echo "This script will:"
echo "1. Create a Docker image"
```

```
echo "2. Create an Amazon ECR repository"
echo "3. Authenticate to Amazon ECR"
echo "4. Push the image to Amazon ECR"
echo "5. Pull the image from Amazon ECR"
echo "6. Clean up resources (optional)"
echo "================================================="

# Check prerequisites
echo "Checking prerequisites..."

# Check if AWS CLI is installed
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed. Please install it before running this
 script."
    echo "Visit https://docs.aws.amazon.com/cli/latest/userguide/install-
cliv2.html for installation instructions."
    exit 1
fi

# Check if AWS CLI is configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not configured properly. Please run 'aws configure'
 to set up your credentials."
    exit 1
fi

# Check if Docker is installed
if ! command -v docker &> /dev/null; then
    echo "ERROR: Docker is not installed. Please install Docker before running
 this script."
    echo "Visit https://docs.docker.com/get-docker/ for installation
 instructions."
    exit 1
fi

# Check if Docker daemon is running
if ! docker info &> /dev/null; then
    echo "ERROR: Docker daemon is not running. Please start Docker and try
 again."
    exit 1
fi

echo "All prerequisites met."
```

```
# Initialize variables
REPO_URI=""
TIMEOUT_CMD="timeout 300"  # 5-minute timeout for long-running commands

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Check the log file for details: $LOG_FILE"

    echo "=================================================="
    echo "Resources created:"
    echo "- Docker image: hello-world (local)"
    if [ -n "$REPO_URI" ]; then
        echo "- ECR Repository: hello-repository"
        echo "- ECR Image: $REPO_URI:latest"
    fi
    echo "=================================================="

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "=================================================="
    echo "Cleaning up resources..."

    # Delete the image from ECR if it exists
    if [ -n "$REPO_URI" ]; then
        echo "Deleting image from ECR repository..."
        aws ecr batch-delete-image --repository-name hello-repository --image-ids
 imageTag=latest || echo "Failed to delete image, it may not exist or may have
 already been deleted."
    fi

    # Delete the ECR repository if it exists
    if [ -n "$REPO_URI" ]; then
        echo "Deleting ECR repository..."
        aws ecr delete-repository --repository-name hello-repository --force ||
 echo "Failed to delete repository, it may not exist or may have already been
 deleted."
    fi
```

```
    # Remove local Docker image
    echo "Removing local Docker image..."
    docker rmi hello-world:latest 2>/dev/null || echo "Failed to remove local
 image, it may not exist or may have already been deleted."
    if [ -n "$REPO_URI" ]; then
        docker rmi "$REPO_URI:latest" 2>/dev/null || echo "Failed to remove
 tagged image, it may not exist or may have already been deleted."
    fi

    echo "Cleanup completed."
    echo "================================================="
}

# Step 1: Create a Docker image
echo "Step 1: Creating a Docker image"

# Create Dockerfile
echo "Creating Dockerfile..."
cat > Dockerfile << 'EOF'
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Install dependencies
RUN yum update -y && \
 yum install -y httpd

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

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

EXPOSE 80

CMD /root/run_apache.sh
EOF

# Build Docker image
echo "Building Docker image..."
$TIMEOUT_CMD docker build -t hello-world . || handle_error "Failed to build
 Docker image or operation timed out after 5 minutes"
```

```
# Verify image was created
echo "Verifying Docker image..."
docker images --filter reference=hello-world || handle_error "Failed to list
 Docker images"

echo "Docker image created successfully."

# Step 2: Create an Amazon ECR repository
echo "Step 2: Creating an Amazon ECR repository"

# Get AWS account ID
echo "Getting AWS account ID..."
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ -z "$AWS_ACCOUNT_ID" || "$AWS_ACCOUNT_ID" == *"error"* ]]; then
    handle_error "Failed to get AWS account ID. Make sure your AWS credentials
 are configured correctly."
fi
echo "AWS Account ID: $AWS_ACCOUNT_ID"

# Get current region
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
    echo "No AWS region configured, defaulting to $AWS_REGION"
else
    echo "Using AWS region: $AWS_REGION"
fi

# Create ECR repository
echo "Creating ECR repository..."
REPO_RESULT=$(aws ecr create-repository --repository-name hello-repository --
tags key=project,value=doc-smith key=tutorial,value=amazon-elastic-container-
registry-gs)
if [[ -z "$REPO_RESULT" || "$REPO_RESULT" == *"error"* ]]; then
    handle_error "Failed to create ECR repository"
fi

# Extract repository URI
REPO_URI="$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/hello-repository"
echo "Repository URI: $REPO_URI"

# Step 3: Authenticate to Amazon ECR
echo "Step 3: Authenticating to Amazon ECR"
```

```
echo "Getting ECR login password..."
aws ecr get-login-password --region "$AWS_REGION" | docker login --username
 AWS --password-stdin "$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com" ||
 handle_error "Failed to authenticate to ECR"

echo "Successfully authenticated to ECR."

# Step 4: Push the image to Amazon ECR
echo "Step 4: Pushing the image to Amazon ECR"

# Tag the image
echo "Tagging Docker image..."
docker tag hello-world:latest "$REPO_URI:latest" || handle_error "Failed to tag
 Docker image"

# Push the image with timeout
echo "Pushing image to ECR..."
$TIMEOUT_CMD docker push "$REPO_URI:latest" || handle_error "Failed to push image
 to ECR or operation timed out after 5 minutes"

echo "Successfully pushed image to ECR."

# Step 5: Pull the image from Amazon ECR
echo "Step 5: Pulling the image from Amazon ECR"

# Remove local image to ensure we're pulling from ECR
echo "Removing local tagged image..."
docker rmi "$REPO_URI:latest" || echo "Warning: Failed to remove local tagged
 image"

# Pull the image with timeout
echo "Pulling image from ECR..."
$TIMEOUT_CMD docker pull "$REPO_URI:latest" || handle_error "Failed to pull image
 from ECR or operation timed out after 5 minutes"

echo "Successfully pulled image from ECR."

# List resources created
echo "================================================="
echo "Resources created:"
echo "- Docker image: hello-world (local)"
echo "- ECR Repository: hello-repository"
echo "- ECR Image: $REPO_URI:latest"
echo "================================================="
```

```
# Ask user if they want to clean up resources
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    # Step 6: Delete the image from ECR
    echo "Step 6: Deleting the image from ECR"

    DELETE_IMAGE_RESULT=$(aws ecr batch-delete-image --repository-name hello-
repository --image-ids imageTag=latest)
    if [[ -z "$DELETE_IMAGE_RESULT" || "$DELETE_IMAGE_RESULT" == *"error"* ]];
 then
        echo "Warning: Failed to delete image from ECR"
    else
        echo "Successfully deleted image from ECR."
    fi

    # Step 7: Delete the ECR repository
    echo "Step 7: Deleting the ECR repository"

    DELETE_REPO_RESULT=$(aws ecr delete-repository --repository-name hello-
repository --force)
    if [[ -z "$DELETE_REPO_RESULT" || "$DELETE_REPO_RESULT" == *"error"* ]]; then
        echo "Warning: Failed to delete ECR repository"
    else
        echo "Successfully deleted ECR repository."
    fi

    # Remove local Docker images
    echo "Removing local Docker images..."
    docker rmi hello-world:latest 2>/dev/null || echo "Warning: Failed to remove
 local image"

    echo "All resources have been cleaned up."
else
    echo "Resources were not cleaned up. You can manually clean up later with:"
    echo "aws ecr batch-delete-image --repository-name hello-repository --image-
ids imageTag=latest"
    echo "aws ecr delete-repository --repository-name hello-repository --force"
```

```
    echo "docker rmi hello-world:latest"
    echo "docker rmi $REPO_URI:latest"
fi

echo "============================================================="
echo "Tutorial completed!"
echo "Log file: $LOG_FILE"
echo "============================================================="
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - BatchDeleteImage
  - CreateRepository
  - DeleteRepository
  - GetCallerIdentity
  - GetLoginPassword

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

開始使用物聯網簡訊

以下程式碼範例顯示做法：

- 建立 IoT 資源
- 設定您的裝置
- 執行範例應用程式
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS IoT Core Getting Started Script
# This script creates AWS IoT resources, configures a device, and runs a sample
 application

# Set up logging
LOG_FILE="iot-core-setup.log"
echo "Starting AWS IoT Core setup at $(date)" > $LOG_FILE

# Function to log commands and their outputs
log_cmd() {
    echo "$(date): Running command: $1" >> $LOG_FILE
    eval "$1" 2>&1 | tee -a $LOG_FILE
    return ${PIPESTATUS[0]}
}

# Function to check for errors
check_error() {
    if [ $1 -ne 0 ]; then
        echo "ERROR: Command failed with exit code $1" | tee -a $LOG_FILE
        echo "Please check the log file $LOG_FILE for details" | tee -a $LOG_FILE
        cleanup_on_error
        exit $1
    fi
}

# Function to cleanup resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..." | tee -a
 $LOG_FILE
    echo "Resources created:" | tee -a $LOG_FILE
    if [ ! -z "$CERTIFICATE_ARN" ]; then
        echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
        if [ ! -z "$POLICY_NAME" ]; then
            log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
        fi
        if [ ! -z "$THING_NAME" ]; then
            log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --
principal $CERTIFICATE_ARN"
        fi
        if [ ! -z "$CERTIFICATE_ID" ]; then
```

```
            log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID
 --new-status INACTIVE"
            log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"
        fi
    fi
    if [ ! -z "$THING_NAME" ]; then
        echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-thing --thing-name $THING_NAME"
    fi
    if [ ! -z "$POLICY_NAME" ]; then
        echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    fi
    if [ ! -z "$SHARED_POLICY_NAME" ]; then
        echo "Shared Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"
    fi
}

# Generate unique identifiers
RANDOM_SUFFIX=$(openssl rand -hex 4)
THING_NAME="MyIoTThing-${RANDOM_SUFFIX}"
POLICY_NAME="MyIoTPolicy-${RANDOM_SUFFIX}"
SHARED_POLICY_NAME="SharedSubPolicy-${RANDOM_SUFFIX}"
CERTS_DIR="$HOME/certs"

echo "================================================" | tee -a $LOG_FILE
echo "AWS IoT Core Getting Started" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "This script will:" | tee -a $LOG_FILE
echo "1. Create AWS IoT resources (policy, thing, certificate)" | tee -a
 $LOG_FILE
echo "2. Configure your device" | tee -a $LOG_FILE
echo "3. Set up for running the sample application" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Certificates Directory: $CERTS_DIR" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE

# Get AWS account ID
echo "Getting AWS account ID..." | tee -a $LOG_FILE
ACCOUNT_ID=$(log_cmd "aws sts get-caller-identity --query Account --output text")
```

```
check_error $?

# Get AWS region
echo "Getting AWS region..." | tee -a $LOG_FILE
REGION=$(log_cmd "aws configure get region")
check_error $?
if [ -z "$REGION" ]; then
    echo "AWS region not configured. Please run 'aws configure' to set your
 region." | tee -a $LOG_FILE
    exit 1
fi

echo "Using AWS Account ID: $ACCOUNT_ID and Region: $REGION" | tee -a $LOG_FILE

# Step 1: Create AWS IoT Resources
echo "" | tee -a $LOG_FILE
echo "Step 1: Creating AWS IoT Resources..." | tee -a $LOG_FILE

# Create IoT policy
echo "Creating IoT policy document..." | tee -a $LOG_FILE
cat > iot-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/test-*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic"
      ]
    }
  ]
}
EOF

echo "Creating IoT policy: $POLICY_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-policy --policy-name $POLICY_NAME --policy-
document file://iot-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?

# Create IoT thing
echo "Creating IoT thing: $THING_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-thing --thing-name $THING_NAME"
check_error $?

THING_ARN=$(aws iot describe-thing --thing-name "$THING_NAME" --query 'thingArn'
 --output text)
aws iot tag-resource --resource-arn "$THING_ARN" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-iot-core-gs

# Create directory for certificates
echo "Creating certificates directory..." | tee -a $LOG_FILE
log_cmd "mkdir -p $CERTS_DIR"
check_error $?

# Create keys and certificate
echo "Creating keys and certificate..." | tee -a $LOG_FILE
CERT_OUTPUT=$(log_cmd "aws iot create-keys-and-certificate --set-as-active
 --certificate-pem-outfile $CERTS_DIR/device.pem.crt --public-key-outfile
 $CERTS_DIR/public.pem.key --private-key-outfile $CERTS_DIR/private.pem.key")
check_error $?

# Extract certificate ARN and ID
CERTIFICATE_ARN=$(echo "$CERT_OUTPUT" | grep "certificateArn" | cut -d'"' -f4)
CERTIFICATE_ID=$(echo "$CERTIFICATE_ARN" | cut -d/ -f2)
```

```
if [ -z "$CERTIFICATE_ARN" ] || [ -z "$CERTIFICATE_ID" ]; then
    echo "Failed to extract certificate ARN or ID" | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE

# Attach policy to certificate
echo "Attaching policy to certificate..." | tee -a $LOG_FILE
log_cmd "aws iot attach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Attach certificate to thing
echo "Attaching certificate to thing..." | tee -a $LOG_FILE
log_cmd "aws iot attach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"
check_error $?

# Download Amazon Root CA certificate
echo "Downloading Amazon Root CA certificate..." | tee -a $LOG_FILE
log_cmd "curl -s -o $CERTS_DIR/Amazon-root-CA-1.pem https://www.amazontrust.com/
repository/AmazonRootCA1.pem"
check_error $?

# Step 2: Configure Your Device
echo "" | tee -a $LOG_FILE
echo "Step 2: Configuring Your Device..." | tee -a $LOG_FILE

# Check if Git is installed
echo "Checking if Git is installed..." | tee -a $LOG_FILE
if ! command -v git &> /dev/null; then
    echo "Git is not installed. Please install Git and run this script again." |
 tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Check if Python is installed
echo "Checking if Python is installed..." | tee -a $LOG_FILE
if ! command -v python3 &> /dev/null; then
```

```
    echo "Python 3 is not installed. Please install Python 3 and run this script
 again." | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Install AWS IoT Device SDK for Python
echo "Installing AWS IoT Device SDK for Python..." | tee -a $LOG_FILE
log_cmd "python3 -m pip install awsiotsdk"
check_error $?

# Clone the AWS IoT Device SDK for Python repository
echo "Cloning AWS IoT Device SDK for Python repository..." | tee -a $LOG_FILE
if [ ! -d "$HOME/aws-iot-device-sdk-python-v2" ]; then
    log_cmd "cd $HOME && git clone https://github.com/aws/aws-iot-device-sdk-
python-v2.git"
    check_error $?
else
    echo "AWS IoT Device SDK for Python repository already exists." | tee -a
 $LOG_FILE
fi

# Step 3: Get AWS IoT Endpoint
echo "" | tee -a $LOG_FILE
echo "Step 3: Getting AWS IoT Endpoint..." | tee -a $LOG_FILE

IOT_ENDPOINT=$(log_cmd "aws iot describe-endpoint --endpoint-type iot:Data-ATS --
query endpointAddress --output text")
check_error $?

echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE

# Create a shared subscription policy (optional)
echo "" | tee -a $LOG_FILE
echo "Creating shared subscription policy (optional)..." | tee -a $LOG_FILE

cat > shared-sub-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
```

```
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic",
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/\$share/*/test/topic"
      ]
    }
  ]
}
EOF

log_cmd "aws iot create-policy --policy-name $SHARED_POLICY_NAME --policy-
document file://shared-sub-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?

log_cmd "aws iot attach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Summary of created resources
echo "" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Setup Complete! Resources Created:" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
```

```
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Shared Subscription Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE
echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate Files Location: $CERTS_DIR" | tee -a $LOG_FILE
echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE

# Instructions for running the sample application
echo "" | tee -a $LOG_FILE
echo "To run the sample application, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 pubsub.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "To run the shared subscription example, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 mqtt5_shared_subscription.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key \\" | tee -a $LOG_FILE
echo "  --group_identifier consumer" | tee -a $LOG_FILE

# Ask if user wants to clean up resources
echo "" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "CLEANUP CONFIRMATION" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Do you want to clean up all created resources? (y/n): " | tee -a $LOG_FILE
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a $LOG_FILE

    # Detach policies from certificate
    echo "Detaching policies from certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
    log_cmd "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"
```

```
    # Detach certificate from thing
    echo "Detaching certificate from thing..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"

    # Update certificate status to INACTIVE
    echo "Setting certificate to inactive..." | tee -a $LOG_FILE
    log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE"

    # Delete certificate
    echo "Deleting certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"

    # Delete thing
    echo "Deleting thing..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-thing --thing-name $THING_NAME"

    # Delete policies
    echo "Deleting policies..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"

    echo "Cleanup complete!" | tee -a $LOG_FILE
else
    echo "Resources were not cleaned up. You can manually clean them up later." |
 tee -a $LOG_FILE
    echo "To clean up resources, run the following commands:" | tee -a $LOG_FILE
    echo "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE" | tee -a $LOG_FILE
    echo "aws iot delete-certificate --certificate-id $CERTIFICATE_ID" | tee -a
 $LOG_FILE
    echo "aws iot delete-thing --thing-name $THING_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $POLICY_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $SHARED_POLICY_NAME" | tee -a
 $LOG_FILE
fi
```

```
echo "" | tee -a $LOG_FILE
echo "Script execution completed. See $LOG_FILE for details." | tee -a $LOG_FILE
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachPolicy
  - AttachThingPrincipal
  - CreateKeysAndCertificate
  - CreatePolicy
  - CreateThing
  - DeleteCertificate
  - DeletePolicy
  - DeleteThing
  - DescribeEndpoint
  - DetachPolicy
  - DetachThingPrincipal
  - GetCallerIdentity
  - UpdateCertificate

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

機器學習功能存放區的入門

以下程式碼範例顯示做法：

- 設定 IAM 許可
- 建立一個 SageMaker 執行角色
- 建立特徵群組
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command status
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
```

```
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"

        case "$resource_type" in
            "FeatureGroup")
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
```

```
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'

    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
```

```
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"

    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}
```

```
# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
```

```
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
    exit 1
fi

echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1
```

```
# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
```

```
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
```

```
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER1_RESPONSE"

CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)
```

```
if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"

ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
```

```
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)

if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi
```

```
echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "============================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "============================================="
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "============================================="
echo ""

# Handle cleanup
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
```

```
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateBucket
  - CreateFeatureGroup
  - CreateRole
  - DeleteBucket
  - DeleteFeatureGroup
  - DeleteRole
  - DescribeFeatureGroup
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole

- ListFeatureGroups
- PutPublicAccessBlock
- Rm

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

受管 kubernetes 叢集入門

以下程式碼範例顯示做法：

- 為您的 EKS 叢集建立 VPC
- 為您的 EKS 叢集建立 IAM 角色
- 建立 EKS 叢集
- 設定 kubectl 以與您的叢集通訊
- 建立受管節點群組
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1
```

```
echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
        echo ""
        echo "  3. Move the binary to your PATH:"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
```

```
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
```

```
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
```

```
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi

    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."

# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")

echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."
```

```
# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"

# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"

# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
```

```
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"

# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
```

```
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"

NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
```

```
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi
```

```
# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "========================================="
echo "RESOURCES CREATED"
echo "========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "========================================="

# Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
```

```
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster
  - DeleteNodegroup
  - DeleteRole
  - DeleteStack
  - DescribeCluster
  - DescribeNodegroup
  - DescribeStacks
  - DetachRolePolicy

- GetCallerIdentity
- GetRole
- ListNodegroups
- UpdateKubeconfig
- 等待 (CloudFormation)
- 等待 (Amazon EKS)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

受管串流入門

以下程式碼範例顯示做法：

- 建立 MSK 叢集
- 建立 MSK 存取的 IAM 許可
- 建立用戶端機器
- 取得引導代理程式
- 設定用戶端機器
- 建立主題並產生/使用資料
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
```

```
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
```

```
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
}

# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi

    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
```

```
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"

                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."
```

```
                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete EC2 instance if it exists
    if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
        echo "Terminating EC2 instance: $INSTANCE_ID"
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
 to terminate instance"
        echo "Waiting for instance to terminate..."
        aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
 "Failed to wait for instance termination"
    fi

    # Delete MSK cluster first (to remove dependencies on security group)
    if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
        echo "Deleting MSK cluster: $CLUSTER_ARN"
        aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
 delete cluster"

        # Wait a bit for the cluster deletion to start
        echo "Waiting 30 seconds for cluster deletion to begin..."
        sleep 30
    fi

    # Remove security group references before attempting deletion
    if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
 "$CLIENT_SG_ID"; then
        remove_security_group_references "$CLIENT_SG_ID"
```

```
       echo "Deleting security group: $CLIENT_SG_ID"
       # Try multiple times with longer delays to ensure dependencies are
removed
       for i in {1..10}; do
           if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
               echo "Security group deleted successfully"
               break
           fi
           echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
           sleep 30
       done
   fi

   # Delete key pair if it exists
   if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
       echo "Deleting key pair: $KEY_NAME"
       aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
   fi

   # Remove role from instance profile
   if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
       echo "Removing role from instance profile"
       aws iam remove-role-from-instance-profile \
           --instance-profile-name "$INSTANCE_PROFILE_NAME" \
           --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
profile"
   fi

   # Delete instance profile
   if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
"$INSTANCE_PROFILE_NAME"; then
       echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
       aws iam delete-instance-profile \
           --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
delete instance profile"
   fi

   # Detach policy from role
   if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
"$ROLE_NAME"; then
```

```
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
 role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
 delete policy"
    fi

    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")

    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"
```

```
            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)

            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
        done
    done

    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"
```

```
# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
```

```
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi
```

```
    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi

if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
```

```
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
```

```
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"

# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
```

```
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi

echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi

echo "Role added to instance profile"

# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "============================================="

# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)
```

```
if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"

# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi

echo "MSK security group ID: $MSK_SG_ID"

# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi
```

```
echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)

if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
```

```
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"
```

```
INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)

if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi

echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
```

```
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
echo "Getting bootstrap brokers"
MAX_RETRIES=10
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi
```

```
    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
done

echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"
```

```
# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
cd $KAFKA_ROOT/libs
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"
```

```
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
```

```
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
echo "============================================="

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCE SUMMARY"
echo "============================================="
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "============================================="
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
```

```
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
```

```
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
echo ""
echo "=========================================="
echo "CLEANUP CONFIRMATION"
echo "=========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreatePolicy
  - CreateRole
  - CreateSecurityGroup
  - DeleteCluster
  - DeleteInstanceProfile
  - DeleteKeyPair
  - DeletePolicy
  - DeleteRole
  - DeleteSecurityGroup
  - DescribeAvailabilityZones
  - DescribeCluster
  - DescribeImages
  - DescribeInstanceTypeOfferings
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DescribeSubnets
  - DescribeVpcs
  - DetachRolePolicy
  - GetBootstrapBrokers
  - GetCallerIdentity
  - GetInstanceProfile

- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- TerminateInstances
- 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

推送通知入門

以下程式碼範例顯示做法：

- 建立應用程式
- 啟用推送通知管道
- 傳送推播通知
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS End User Messaging Push Getting Started Script
# This script creates an AWS End User Messaging Push application and demonstrates
# how to enable push notification channels and send a test message.
```

```
#
# Prerequisites:
# - AWS CLI installed and configured
# - Appropriate IAM permissions for Pinpoint operations
#
# Usage: ./aws-end-user-messaging-gs.sh [--auto-cleanup]

set -uo pipefail

# Security: Set secure umask for created files
umask 0077

# Set up logging
LOG_DIR="./aws-eump-logs"
mkdir -p "$LOG_DIR"
LOG_FILE="$LOG_DIR/aws-end-user-messaging-push-script-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS End User Messaging Push setup script..."
echo "Logging to $LOG_FILE"
echo "Timestamp: $(date)"

# Security: Track created resources for cleanup
declare -a TEMP_FILES=()
declare -a AWS_RESOURCES=()

# Cleanup function with improved security
cleanup() {
    local exit_code=$?
    echo "Cleaning up temporary resources..."

    # Remove temporary files securely
    for temp_file in "${TEMP_FILES[@]+"${TEMP_FILES[@]}"}"; do
        if [ -f "$temp_file" ]; then
            shred -vfz -n 3 "$temp_file" 2>/dev/null || rm -f "$temp_file"
        fi
    done

    # Optionally delete AWS resources
    if [ "${DELETE_AWS_RESOURCES:-false}" = "true" ]; then
        for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
            echo "Deleting AWS resource: $resource"
            aws pinpoint delete-app --application-id "$resource" 2>/dev/null || \
                echo "Warning: Failed to delete application $resource"
```

```
        done
    fi

    exit "$exit_code"
}

trap cleanup EXIT INT TERM

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2
    local ignore_error=${3:-false}

    if echo "$output" | grep -qi "error\|exception\|fail"; then
        echo "ERROR: Command failed: $cmd" >&2
        echo "Error details: $output" >&2

        if [ "$ignore_error" = "true" ]; then
            echo "Ignoring error and continuing..." >&2
            return 1
        else
            return 2
        fi
    fi

    return 0
}

# Function to validate AWS CLI is configured
validate_aws_cli() {
    echo "Validating AWS CLI configuration..."

    # Check if AWS CLI is installed
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed. Please install it first." >&2
        echo "Visit: https://docs.aws.amazon.com/cli/latest/userguide/getting-
started-install.html" >&2
        return 1
    fi

    # Check AWS CLI version
    AWS_VERSION=$(aws --version 2>&1 | head -n1)
    echo "AWS CLI version: $AWS_VERSION"
```

```
    # Verify credentials are set (check for credential env vars or config file)
    if ! aws sts get-caller-identity &> /dev/null; then
        echo "ERROR: AWS CLI credentials are not configured or invalid." >&2
        echo "Please configure credentials via environment variables, credential
 file, or 'aws configure'" >&2
        return 1
    fi

    # Get current AWS identity and region
    CALLER_IDENTITY=$(aws sts get-caller-identity)
    CURRENT_REGION=$(aws configure get region 2>/dev/null || echo "us-east-1")
    echo "AWS CLI configured for:"
    echo "$CALLER_IDENTITY"
    echo "Current region: $CURRENT_REGION"
    echo ""

    return 0
}

# Function to check if jq is available for JSON parsing
check_json_tools() {
    if command -v jq &> /dev/null; then
        USE_JQ=true
        echo "jq is available for JSON parsing"
    else
        USE_JQ=false
        echo "jq is not available, using grep for JSON parsing"
        echo "Consider installing jq for better JSON handling: https://
stedolan.github.io/jq/"
    fi
}

# Function to extract JSON values safely
extract_json_value() {
    local json=$1
    local key=$2

    if [ "$USE_JQ" = "true" ]; then
        echo "$json" | jq -r ".ApplicationResponse.$key // empty" 2>/dev/null ||
 echo ""
    else
        # Fallback to grep method with better validation
```

```
        echo "$json" | grep -o "\"$key\": \"[^\"]*" | cut -d'"' -f4 | head -n1 ||
 echo ""
    fi
}

# Function to validate required IAM permissions
validate_permissions() {
    echo "Validating IAM permissions..."

    # Test basic Pinpoint permissions
    if ! aws pinpoint get-apps > /dev/null 2>&1; then
        echo "WARNING: Unable to list Pinpoint applications." >&2
        echo "Please ensure you have appropriate IAM permissions for Pinpoint
 operations." >&2
        echo "Required permissions:" >&2
        echo "  - mobiletargeting:GetApps" >&2
        echo "  - mobiletargeting:CreateApp" >&2
        echo "  - mobiletargeting:DeleteApp" >&2
        echo "  - mobiletargeting:UpdateGcmChannel" >&2
        echo "  - mobiletargeting:UpdateApnsChannel" >&2
        echo "  - mobiletargeting:SendMessages" >&2
    else
        echo "Basic Pinpoint permissions validated."
    fi
}

# Function to validate input parameters
validate_input() {
    local app_name=$1

    # Validate app name length and characters
    if [ ${#app_name} -gt 64 ]; then
        echo "ERROR: Application name exceeds maximum length of 64 characters"
 >&2
        return 1
    fi

    if ! [[ "$app_name" =~ ^[a-zA-Z0-9_-]+$ ]]; then
        echo "ERROR: Application name contains invalid characters" >&2
        return 1
    fi

    return 0
}
```

```
# Function to create secure temporary files
create_temp_file() {
    local temp_file
    temp_file=$(mktemp) || {
        echo "ERROR: Failed to create temporary file" >&2
        return 1
    }
    chmod 600 "$temp_file"
    TEMP_FILES+=("$temp_file")
    echo "$temp_file"
}

# Validate prerequisites
if ! validate_aws_cli; then
    exit 1
fi

check_json_tools
validate_permissions

# Generate a random suffix for resource names to avoid conflicts
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n1)
APP_NAME="PushNotificationApp-${RANDOM_SUFFIX}"

# Validate input
if ! validate_input "$APP_NAME"; then
    exit 1
fi

echo "Creating application with name: $APP_NAME"

# Step 1: Create an application
echo "Executing: aws pinpoint create-app --create-application-request Name=
${APP_NAME}"
CREATE_APP_OUTPUT=$(aws pinpoint create-app --create-application-request "Name=
${APP_NAME},tags={project=doc-smith,tutorial=aws-end-user-messaging-gs}" 2>&1)

if ! check_error "$CREATE_APP_OUTPUT" "create-app"; then
    exit 1
fi

echo "Application created successfully:"
echo "$CREATE_APP_OUTPUT"
```

```
# Extract the application ID from the output
APP_ID=$(extract_json_value "$CREATE_APP_OUTPUT" "Id")

if [ -z "$APP_ID" ] || [ "$APP_ID" = "null" ]; then
    echo "ERROR: Failed to extract application ID from output" >&2
    echo "Output was: $CREATE_APP_OUTPUT" >&2
    exit 1
fi

echo "Application ID: $APP_ID"
AWS_RESOURCES+=("$APP_ID")

# Step 2: Enable FCM (GCM) channel with a sample API key
echo ""
echo "========================================="
echo "ENABLING FCM (GCM) CHANNEL"
echo "========================================="
echo "Note: This is using a placeholder API key for demonstration purposes only."
echo "In a production environment, you should use your actual FCM API key from
 Firebase Console."
echo ""
echo "IMPORTANT: The following command will likely fail because we're using a
 placeholder API key."
echo "This is expected behavior for this demonstration script."

echo "Executing: aws pinpoint update-gcm-channel --application-id $APP_ID --gcm-
channel-request ..."
UPDATE_GCM_OUTPUT=$(aws pinpoint update-gcm-channel \
    --application-id "$APP_ID" \
    --gcm-channel-request '{"Enabled": true, "ApiKey": "sample-fcm-api-key-for-
demo-only"}' 2>&1)

# We'll ignore this specific error since we're using a placeholder API key
if check_error "$UPDATE_GCM_OUTPUT" "update-gcm-channel" "true"; then
    echo "FCM channel enabled successfully:"
    echo "$UPDATE_GCM_OUTPUT"
else
    echo "As expected, FCM channel update failed with the placeholder API key."
    echo "Error details: $UPDATE_GCM_OUTPUT"
    echo ""
    echo "To enable FCM in production:"
    echo "1. Go to Firebase Console (https://console.firebase.google.com/)"
    echo "2. Create or select your project"
```

```
    echo "3. Go to Project Settings > Cloud Messaging"
    echo "4. Copy the Server Key"
    echo "5. Replace 'sample-fcm-api-key-for-demo-only' with your actual Server
 Key"
fi

# Step 3: Try to enable APNS channel (this will also fail without real
 certificates)
echo ""
echo "=========================================="
echo "ENABLING APNS CHANNEL (OPTIONAL)"
echo "=========================================="
echo "Attempting to enable APNS channel with placeholder certificate..."
echo "This will also fail without real APNS certificates, which is expected."

# Create a placeholder APNS configuration
echo "Executing: aws pinpoint update-apns-channel --application-id $APP_ID --
apns-channel-request ..."
UPDATE_APNS_OUTPUT=$(aws pinpoint update-apns-channel \
    --application-id "$APP_ID" \
    --apns-channel-request '{"Enabled": true, "Certificate": "placeholder-
certificate", "PrivateKey": "placeholder-private-key"}' 2>&1)

if check_error "$UPDATE_APNS_OUTPUT" "update-apns-channel" "true"; then
    echo "APNS channel enabled successfully:"
    echo "$UPDATE_APNS_OUTPUT"
else
    echo "As expected, APNS channel update failed with placeholder certificates."
    echo "Error details: $UPDATE_APNS_OUTPUT"
    echo ""
    echo "To enable APNS in production:"
    echo "1. Generate APNS certificates from Apple Developer Console"
    echo "2. Convert certificates to PEM format"
    echo "3. Use the actual certificate and private key in the update-apns-
channel command"
fi

# Step 4: Create message files for different platforms
echo ""
echo "=========================================="
echo "CREATING MESSAGE FILES"
echo "=========================================="

# Create FCM message file securely
```

```
GCM_MESSAGE_FILE=$(create_temp_file)
echo "Creating FCM message file..."
cat > "$GCM_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-FCM": {
      "ChannelType": "GCM"
    }
  },
  "MessageConfiguration": {
    "GCMMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an FCM
 notification.",
      "Priority": "normal",
      "SilentPush": false,
      "Title": "My First FCM Push Notification",
      "TimeToLive": 30,
      "Data": {
        "key1": "value1",
        "key2": "value2"
      }
    }
  }
}
EOF

# Create APNS message file securely
APNS_MESSAGE_FILE=$(create_temp_file)
echo "Creating APNS message file..."
cat > "$APNS_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-APNS": {
      "ChannelType": "APNS"
    }
  },
  "MessageConfiguration": {
    "APNSMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an APNS
 notification.",
      "Priority": "normal",
      "SilentPush": false,
```

```
      "Title": "My First APNS Push Notification",
      "TimeToLive": 30,
      "Badge": 1,
      "Sound": "default"
    }
  }
}
EOF

echo "Message files created:"
echo "- FCM message file (for FCM/Android)"
echo "- APNS message file (for APNS/iOS)"
echo ""
echo "Note: These messages use placeholder device tokens and will not actually be
 delivered."
echo "To send real messages, you would need to replace the sample device tokens
 with actual ones."

# Step 5: Demonstrate how to send messages (this will fail with placeholder
 tokens)
echo ""
echo "============================================="
echo "DEMONSTRATING MESSAGE SENDING"
echo "============================================="
echo "Attempting to send FCM message (will fail with placeholder token)..."

echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<gcm-message>"
SEND_FCM_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$GCM_MESSAGE_FILE" 2>&1)

if check_error "$SEND_FCM_OUTPUT" "send-messages (FCM)" "true"; then
    echo "FCM message sent successfully:"
    echo "$SEND_FCM_OUTPUT"
else
    echo "As expected, FCM message sending failed with placeholder token."
    echo "Error details: $SEND_FCM_OUTPUT"
fi

echo ""
echo "Attempting to send APNS message (will fail with placeholder token)..."
```

```
echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<apns-message>"
SEND_APNS_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$APNS_MESSAGE_FILE" 2>&1)

if check_error "$SEND_APNS_OUTPUT" "send-messages (APNS)" "true"; then
    echo "APNS message sent successfully:"
    echo "$SEND_APNS_OUTPUT"
else
    echo "As expected, APNS message sending failed with placeholder token."
    echo "Error details: $SEND_APNS_OUTPUT"
fi

# Step 6: Show application details
echo ""
echo "==========================================="
echo "APPLICATION DETAILS"
echo "==========================================="
echo "Retrieving application details..."

echo "Executing: aws pinpoint get-app --application-id $APP_ID"
GET_APP_OUTPUT=$(aws pinpoint get-app --application-id "$APP_ID" 2>&1)
if check_error "$GET_APP_OUTPUT" "get-app"; then
    echo "Application details:"
    echo "$GET_APP_OUTPUT"
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
echo "AWS Resources:"
for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
    echo "- Application: $resource"
done

echo ""
echo "Files created:"
echo "- $LOG_FILE (script log)"

# Auto-cleanup information
echo ""
```

```
echo "============================================="
echo "CLEANUP INFORMATION"
echo "============================================="
echo "This script created AWS resources that may incur charges."
echo "AWS resources will be automatically cleaned up on script exit."
echo ""
echo "To manually delete resources later, use:"
echo "  aws pinpoint delete-app --application-id $APP_ID"

# Set flag to delete AWS resources on cleanup
DELETE_AWS_RESOURCES=true

echo ""
echo "============================================="
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================="
echo "This script demonstrated:"
echo "1. Creating an AWS End User Messaging Push application"
echo "2. Attempting to enable FCM and APNS channels (with placeholder
 credentials)"
echo "3. Creating message templates for different platforms"
echo "4. Demonstrating message sending commands (with placeholder tokens)"
echo "5. Retrieving application details"
echo "6. Proper cleanup of resources"
echo ""
echo "Security best practices implemented:"
echo "- Secure temporary file handling with restricted permissions"
echo "- Input validation for application names"
echo "- Error handling with proper exit codes"
echo "- Credential validation before AWS operations"
echo "- Automatic cleanup on script exit"
echo "- Secure file destruction for sensitive data"
echo ""
echo "For production use:"
echo "- Replace placeholder API keys with real FCM server keys"
echo "- Replace placeholder certificates with real APNS certificates"
echo "- Replace placeholder device tokens with real device tokens"
echo "- Use AWS IAM roles instead of long-term credentials"
echo "- Implement comprehensive error handling"
echo "- Store sensitive credentials in AWS Secrets Manager or Parameter Store"
echo "- Enable CloudTrail for audit logging"
echo "- Use VPC endpoints for private AWS API access"
echo ""
echo "Log file: $LOG_FILE"
```

```
echo "Script completed at: $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - CreateApp
  - DeleteApp
  - GetApp
  - GetApps
  - GetCallerIdentity
  - SendMessages
  - UpdateApnsChannel
  - UpdateGcmChannel

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搜尋和分析引擎入門

以下程式碼範例顯示做法：

- 建立 OpenSearch Service 網域
- 將資料上傳至您的網域
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Amazon OpenSearch Service Getting Started Script - Version 8 Fixed
```

```
# This script creates an OpenSearch domain, uploads data, searches documents, and
 cleans up resources
# Based on the tested and working 4-tutorial-final.md

# SECURITY IMPROVEMENTS IN THIS VERSION:
# 1. Removed hardcoded passwords - now generated securely
# 2. Improved access policy to use principle of least privilege
# 3. Added credential masking in logs
# 4. Removed credentials from command output
# 5. Added input validation for AWS region
# 6. Improved error handling and validation
# 7. Added secure temporary file handling
# 8. Removed unnecessary curl insecurity flags
# 9. Added better secret management practices
# 10. Improved resource tagging for better governance

set -e  # Exit on any error

# Set up logging with restricted permissions
LOG_FILE="opensearch_tutorial_v8_fixed.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon OpenSearch Service tutorial script v8-fixed at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Track if domain was successfully created
DOMAIN_CREATED=false
DOMAIN_ACTIVE=false

# Secure temporary directory
TEMP_DIR=$(mktemp -d)
trap 'rm -rf "$TEMP_DIR"' EXIT
chmod 700 "$TEMP_DIR"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}
```

```
# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    if [[ "$DOMAIN_CREATED" == "true" ]]; then
        echo "Checking if domain $DOMAIN_NAME exists before attempting to
 delete..."

        # Check if domain exists before trying to delete
        if aws opensearch describe-domain --domain-name "$DOMAIN_NAME" > /dev/
null 2>&1; then
            echo "Domain $DOMAIN_NAME exists. Proceeding with deletion."
            aws opensearch delete-domain --domain-name "$DOMAIN_NAME"
            echo "Domain deletion initiated. This may take several minutes to
 complete."
        else
            echo "Domain $DOMAIN_NAME does not exist or is not accessible. No
 deletion needed."
        fi
    else
        echo "No domain was successfully created. Nothing to clean up."
    fi

    # Securely clean up sensitive files
    if [[ -d "$TEMP_DIR" ]]; then
        shred -vfz -n 3 "$TEMP_DIR"/* 2>/dev/null || true
        rm -rf "$TEMP_DIR"
    fi
}

# Generate a random identifier for resource names to avoid conflicts
RANDOM_ID=$(openssl rand -hex 4)
DOMAIN_NAME="movies-${RANDOM_ID}"
MASTER_USER="master-user"

# Generate secure password using openssl instead of hardcoding
MASTER_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-25)

# Mask password in logs
echo "Using domain name: $DOMAIN_NAME"
echo "Using master username: $MASTER_USER"
echo "Using master password: [REDACTED]"

# Get AWS account ID (matches tutorial)
```

```
echo "Retrieving AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ $? -ne 0 ]] || [[ -z "$ACCOUNT_ID" ]]; then
    handle_error "Failed to retrieve AWS account ID. Please check your AWS
 credentials."
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Get current region (matches tutorial)
echo "Retrieving current AWS region..."
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION"
else
    echo "Using AWS region: $AWS_REGION"
fi

# Validate AWS region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    handle_error "Invalid AWS region format: $AWS_REGION"
fi

# Step 1: Create an OpenSearch Service Domain
echo "Creating OpenSearch Service domain..."
echo "This may take 15-30 minutes to complete."

# SECURITY IMPROVED: Use least privilege access policy
# This policy restricts access to specific actions and should be further
 restricted to specific principals in production
# Create the domain using CloudFormation (handles the 15-20 min creation wait)
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
STACK_NAME="tutorial-opensearch-${RANDOM_ID}"

echo "Creating domain $DOMAIN_NAME via CloudFormation..."
echo "This typically takes 15-20 minutes. CloudFormation will wait for
 completion."
aws cloudformation deploy \
  --template-file "$SCRIPT_DIR/cfn-opensearch-domain.yaml" \
  --stack-name "$STACK_NAME" \
  --parameter-overrides \
    DomainName="$DOMAIN_NAME" \
    MasterUserName="$MASTER_USER" \
    MasterUserPassword="$MASTER_PASSWORD" \
```

```
  --tags project=doc-smith tutorial=opensearch-service-gs \
  --no-fail-on-empty-changeset 2>&1 || handle_error "CloudFormation stack
 creation failed"

DOMAIN_CREATED=true
echo "Domain created successfully via CloudFormation."

# Get domain endpoint from stack outputs
DOMAIN_ENDPOINT=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query 'Stacks[0].Outputs[?OutputKey==`DomainEndpoint`].OutputValue' \
  --output text)

if [[ -z "$DOMAIN_ENDPOINT" ]] || [[ "$DOMAIN_ENDPOINT" == "None" ]]; then
    handle_error "Failed to get domain endpoint from CloudFormation outputs"
fi

echo "Domain endpoint: $DOMAIN_ENDPOINT"

# Wait for fine-grained access control to be fully ready
echo "Waiting for fine-grained access control to initialize..."
sleep 120

# Verify variables are set correctly (matches tutorial)
echo "Verifying configuration..."
echo "Domain endpoint: $DOMAIN_ENDPOINT"
echo "Master user: $MASTER_USER"
echo "Password set: $(if [ -n "$MASTER_PASSWORD" ]; then echo "Yes"; else echo
 "No"; fi)"

# Step 2: Upload Data to the Domain
echo "Preparing to upload data to the domain..."

# Create a file for the single document (matches tutorial exactly)
echo "Creating single document JSON file..."
SINGLE_MOVIE_FILE="$TEMP_DIR/single_movie.json"
cat > "$SINGLE_MOVIE_FILE" << 'EOF'
{
  "director": "Burton, Tim",
  "genre": ["Comedy","Sci-Fi"],
  "year": 1996,
  "actor": ["Jack Nicholson","Pierce Brosnan","Sarah Jessica Parker"],
  "title": "Mars Attacks!"
}
```

```
EOF
chmod 600 "$SINGLE_MOVIE_FILE"

# Create a file for bulk documents (matches tutorial exactly)
echo "Creating bulk documents JSON file..."
BULK_MOVIES_FILE="$TEMP_DIR/bulk_movies.json"
cat > "$BULK_MOVIES_FILE" << 'EOF'
{ "index" : { "_index": "movies", "_id" : "2" } }
{"director": "Frankenheimer, John", "genre": ["Drama", "Mystery", "Thriller",
 "Crime"], "year": 1962, "actor": ["Lansbury, Angela", "Sinatra, Frank", "Leigh,
 Janet", "Harvey, Laurence", "Silva, Henry", "Frees, Paul", "Gregory, James",
 "Bissell, Whit", "McGiver, John", "Parrish, Leslie", "Edwards, James", "Flowers,
 Bess", "Dhiegh, Khigh", "Payne, Julie", "Kleeb, Helen", "Gray, Joe", "Nalder,
 Reggie", "Stevens, Bert", "Masters, Michael", "Lowell, Tom"], "title": "The
 Manchurian Candidate"}
{ "index" : { "_index": "movies", "_id" : "3" } }
{"director": "Baird, Stuart", "genre": ["Action", "Crime", "Thriller"], "year":
 1998, "actor": ["Downey Jr., Robert", "Jones, Tommy Lee", "Snipes, Wesley",
 "Pantoliano, Joe", "Jacob, Irène", "Nelligan, Kate", "Roebuck, Daniel",
 "Malahide, Patrick", "Richardson, LaTanya", "Wood, Tom", "Kosik, Thomas",
 "Stellate, Nick", "Minkoff, Robert", "Brown, Spitfire", "Foster, Reese",
 "Spielbauer, Bruce", "Mukherji, Kevin", "Cray, Ed", "Fordham, David", "Jett,
 Charlie"], "title": "U.S. Marshals"}
{ "index" : { "_index": "movies", "_id" : "4" } }
{"director": "Ray, Nicholas", "genre": ["Drama", "Romance"], "year": 1955,
 "actor": ["Hopper, Dennis", "Wood, Natalie", "Dean, James", "Mineo, Sal",
 "Backus, Jim", "Platt, Edward", "Ray, Nicholas", "Hopper, William", "Allen,
 Corey", "Birch, Paul", "Hudson, Rochelle", "Doran, Ann", "Hicks, Chuck", "Leigh,
 Nelson", "Williams, Robert", "Wessel, Dick", "Bryar, Paul", "Sessions, Almira",
 "McMahon, David", "Peters Jr., House"], "title": "Rebel Without a Cause"}
EOF
chmod 600 "$BULK_MOVIES_FILE"

# Check if curl is installed
if ! command -v curl &> /dev/null; then
    echo "Warning: curl is not installed. Skipping data upload and search steps."
    echo "You can manually upload the data later using the commands in the
 tutorial."
else
    # IMPROVED: Test authentication with multiple approaches
    echo "Testing authentication with the OpenSearch domain..."
    echo "This test checks if fine-grained access control is ready for data
 operations."
```

```
    # Create credentials file for secure handling
    CREDENTIALS_FILE="$TEMP_DIR/.credentials"
    echo "$MASTER_USER:$MASTER_PASSWORD" > "$CREDENTIALS_FILE"
    chmod 600 "$CREDENTIALS_FILE"

    # Test 1: Basic authentication with root endpoint
    echo "Testing basic authentication with root endpoint..."
    AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
        --netrc-file "$CREDENTIALS_FILE" \
        --request GET \
        "https://${DOMAIN_ENDPOINT}/" 2>&1)

    echo "Basic auth test result:"
    echo "$AUTH_TEST_RESULT"

    # Extract HTTP status code
    HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)

    # Function to check if HTTP code is 2xx
    is_success_code() {
        local code=$1
        if [[ "$code" =~ ^2[0-9][0-9]$ ]]; then
            return 0
        else
            return 1
        fi
    }

    # Check if basic authentication test was successful (200 or 2xx status codes)
    if is_success_code "$HTTP_CODE"; then
        echo "# Basic authentication test successful! (HTTP $HTTP_CODE)"
        AUTH_SUCCESS=true
        AUTH_METHOD="basic"
    else
        echo "Basic authentication failed with HTTP code: $HTTP_CODE"

        # Test 2: Try cluster health endpoint
        echo "Testing with cluster health endpoint..."
        HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

        echo "Cluster health test result:"
```

```
        echo "$HEALTH_TEST_RESULT"

        HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

        if is_success_code "$HEALTH_HTTP_CODE"; then
            echo "# Cluster health authentication test successful! (HTTP
 $HEALTH_HTTP_CODE)"
            AUTH_SUCCESS=true
            AUTH_METHOD="basic"
        else
            echo "Cluster health authentication also failed with HTTP code:
 $HEALTH_HTTP_CODE"

            # Check for specific error patterns
            if echo "$AUTH_TEST_RESULT" | grep -q "anonymous is not authorized";
 then
                echo "Error: Request is being treated as anonymous
 (authentication not working)"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Unauthorized"; then
                echo "Error: Authentication credentials rejected"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Forbidden"; then
                echo "Error: Authentication succeeded but access is forbidden"
            fi

            echo "Waiting additional time and retrying with exponential
 backoff..."

            # Retry authentication test with exponential backoff
            AUTH_RETRY_COUNT=0
            MAX_AUTH_RETRIES=5
            WAIT_TIME=60
            AUTH_SUCCESS=false

            while [[ $AUTH_RETRY_COUNT -lt $MAX_AUTH_RETRIES ]]; do
                echo "Retrying authentication test (attempt $((AUTH_RETRY_COUNT
+1))/$MAX_AUTH_RETRIES) after ${WAIT_TIME} seconds..."
                sleep $WAIT_TIME

                # Try both endpoints
                AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/" 2>&1)
```

```
                HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

                echo "Retry result (HTTP $HTTP_CODE):"
                echo "$AUTH_TEST_RESULT"

                if is_success_code "$HTTP_CODE"; then
                    echo "# Authentication test successful after retry! (HTTP
 $HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi

                # Also try cluster health
                HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

                HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:"
 | cut -d: -f2)

                if is_success_code "$HEALTH_HTTP_CODE"; then
                    echo "# Cluster health authentication successful after retry!
 (HTTP $HEALTH_HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi

                AUTH_RETRY_COUNT=$((AUTH_RETRY_COUNT+1))
                # Exponential backoff: double the wait time each retry (max 10
 minutes)
                WAIT_TIME=$((WAIT_TIME * 2))
                if [[ $WAIT_TIME -gt 600 ]]; then
                    WAIT_TIME=600
                fi
            done
        fi
    fi

    # Proceed with data operations if authentication is working
```

```
    if [[ "$AUTH_SUCCESS" == "true" ]]; then
        echo "Authentication successful using $AUTH_METHOD method. Proceeding
 with data operations."

        # Upload single document (matches tutorial exactly)
        echo "Uploading single document..."
        UPLOAD_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request PUT \
            --header 'Content-Type: application/json' \
            --data @"$SINGLE_MOVIE_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_doc/1" 2>&1)

        echo "Upload response:"
        echo "$UPLOAD_RESULT"

        UPLOAD_HTTP_CODE=$(echo "$UPLOAD_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        if is_success_code "$UPLOAD_HTTP_CODE" && echo "$UPLOAD_RESULT" | grep -q
 '"result"'; then
            echo "# Single document uploaded successfully! (HTTP
 $UPLOAD_HTTP_CODE)"
        else
            echo "# Warning: Single document upload may have failed (HTTP
 $UPLOAD_HTTP_CODE)"
        fi

        # Upload bulk documents (matches tutorial exactly)
        echo "Uploading bulk documents..."
        BULK_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request POST \
            --header 'Content-Type: application/x-ndjson' \
            --data-binary @"$BULK_MOVIES_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_bulk" 2>&1)

        echo "Bulk upload response:"
        echo "$BULK_RESULT"

        BULK_HTTP_CODE=$(echo "$BULK_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)
        if is_success_code "$BULK_HTTP_CODE" && echo "$BULK_RESULT" | grep -q
 '"errors": false'; then
            echo "# Bulk documents uploaded successfully! (HTTP $BULK_HTTP_CODE)"
        else
```

```
            echo "# Warning: Bulk document upload may have failed (HTTP
 $BULK_HTTP_CODE)"
        fi

        # Wait a moment for indexing
        echo "Waiting for documents to be indexed..."
        sleep 5

        # Step 3: Search Documents (matches tutorial exactly)
        echo "Searching for documents containing 'mars'..."
        SEARCH_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true" 2>&1)

        SEARCH_HTTP_CODE=$(echo "$SEARCH_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        echo "Search results for 'mars' (HTTP $SEARCH_HTTP_CODE):"
        echo "$SEARCH_RESULT"

        echo "Searching for documents containing 'rebel'..."
        REBEL_SEARCH=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=rebel&pretty=true" 2>&1)

        REBEL_HTTP_CODE=$(echo "$REBEL_SEARCH" | grep "HTTP_CODE:" | cut -d: -f2)
        echo "Search results for 'rebel' (HTTP $REBEL_HTTP_CODE):"
        echo "$REBEL_SEARCH"

        # Verify search results
        if is_success_code "$SEARCH_HTTP_CODE" && echo "$SEARCH_RESULT" | grep -q
 '"hits"'; then
            echo "# Search functionality is working!"
        else
            echo "# Warning: Search may not be working properly."
        fi

    else
        echo ""
        echo "========================================="
        echo "AUTHENTICATION TROUBLESHOOTING"
        echo "========================================="
        echo "Authentication failed after all retries. This may be due to:"
```

```
        echo "1. Fine-grained access control not fully initialized (most common)"
        echo "2. Domain configuration issues"
        echo "3. Network connectivity issues"
        echo "4. AWS credentials or permissions issues"
        echo ""
        echo "DOMAIN CONFIGURATION DEBUG:"
        echo "Let's check the domain configuration..."

        # Debug domain configuration
        DOMAIN_CONFIG=$(aws opensearch describe-domain --domain-
name "$DOMAIN_NAME" --query 'DomainStatus.{AdvancedSecurityOptions:
 AdvancedSecurityOptions, AccessPolicies: AccessPolicies}' --output json 2>&1)
        echo "Domain configuration:"
        echo "$DOMAIN_CONFIG"

        echo ""
        echo "MANUAL TESTING COMMANDS:"
        echo "You can try these commands manually in 10-15 minutes:"
        echo ""
        echo "# Test basic authentication:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/\""
        echo ""
        echo "# Test cluster health:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/_cluster/health\""
        echo ""
        echo "# Upload single document:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" --request PUT --
header 'Content-Type: application/json' --data @single_movie.json \"https://
${DOMAIN_ENDPOINT}/movies/_doc/1\""
        echo ""
        echo "# Search for documents:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true\""
        echo ""
        echo "TROUBLESHOOTING TIPS:"
        echo "- Wait 10-15 more minutes and try the manual commands"
        echo "- Check AWS CloudTrail logs for authentication errors"
        echo "- Verify your AWS region is correct: $AWS_REGION"
        echo "- Ensure your AWS credentials have OpenSearch permissions"
        echo "- Try accessing OpenSearch Dashboards to verify the master user
 works"
        echo ""
```

```
        echo "Skipping data upload and search operations for now."
        echo "The domain is created and accessible via OpenSearch Dashboards."
    fi

    # Securely remove credentials file
    shred -vfz -n 3 "$CREDENTIALS_FILE" 2>/dev/null || true
fi

# Display OpenSearch Dashboards URL (matches tutorial)
echo ""
echo "============================================"
echo "OPENSEARCH DASHBOARDS ACCESS"
echo "============================================"
echo "OpenSearch Dashboards URL: https://${DOMAIN_ENDPOINT}/_dashboards/"
echo "Username: $MASTER_USER"
echo "Password: [REDACTED - stored securely]"
echo ""
echo "You can access OpenSearch Dashboards using the credentials provided during
 domain creation."
echo "If you uploaded data successfully, you can create an index pattern for
 'movies'."
echo ""

# Summary of created resources
echo ""
echo "============================================"
echo "RESOURCES CREATED"
echo "============================================"
echo "OpenSearch Domain Name: $DOMAIN_NAME"
echo "OpenSearch Domain Endpoint: $DOMAIN_ENDPOINT"
echo "AWS Region: $AWS_REGION"
echo "Master Username: $MASTER_USER"
echo ""
echo "ESTIMATED COST: ~$0.038/hour (~$0.91/day) until deleted"
echo ""
echo "Make sure to save the domain name and endpoint for future reference."
echo ""

# Auto-confirm cleanup: Yes, clean up all resources
echo ""
echo "============================================"
echo "CLEANUP CONFIRMATION"
echo "============================================"
echo "Automatically cleaning up all created resources..."
```

```
CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."
    if [ -n "${STACK_NAME:-}" ]; then
        echo "Deleting CloudFormation stack $STACK_NAME..."
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "# Stack deletion initiated. This may take several minutes."
        echo "  Monitor: aws cloudformation describe-stacks --stack-name
 $STACK_NAME"
    else
        aws opensearch delete-domain --domain-name "$DOMAIN_NAME" 2>/dev/null
        echo "# Domain deletion initiated."
    fi
else
    echo "Resources will NOT be deleted automatically."
    echo ""
    echo "To delete later:"
    echo "  aws cloudformation delete-stack --stack-name ${STACK_NAME:-tutorial-
opensearch-XXXX}"
    echo ""
    echo "# IMPORTANT: Keeping these resources will incur ongoing AWS charges!"
fi

# Clean up temporary files (handled by trap)
echo "Cleaning up temporary files..."

# Disable the trap since we're handling cleanup manually
trap - EXIT

# Final cleanup
rm -rf "$TEMP_DIR" 2>/dev/null || true

echo ""
echo "============================================="
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================="
echo "Script completed at $(date)"
echo "All output has been logged to: $LOG_FILE"
echo ""
echo "Next steps:"
echo "1. Access OpenSearch Dashboards at: https://${DOMAIN_ENDPOINT}/
_dashboards/"
echo "2. Create visualizations and dashboards"
```

```
echo "3. Explore the OpenSearch API"
echo "4. Remember to delete resources when done to avoid charges"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - CreateDomain
  - DeleteDomain
  - DescribeDomain
  - GetCallerIdentity

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Web 應用程式防火牆入門

以下程式碼範例顯示做法：

- 建立 Web ACL
- 新增字串比對規則
- 新增受管規則
- 設定 記錄
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS WAF Getting Started Script
# This script creates a Web ACL with a string match rule and AWS Managed Rules,
# associates it with a CloudFront distribution, and then cleans up all resources.
```

```
set -euo pipefail

# Security: Restrict file permissions
umask 077

# Set up logging with secure file handling
LOG_FILE="waf-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "AWS WAF Getting Started Tutorial"
echo "================================================="
echo "This script will create AWS WAF resources and associate"
echo "them with a CloudFront distribution."
echo ""

# Maximum number of retries for operations
MAX_RETRIES=3

# Function to handle errors securely
handle_error() {
    local error_msg="$1"
    echo "ERROR: $error_msg" >&2
    echo "Check the log file for details: $LOG_FILE" >&2
    cleanup_resources
    exit 1
}

# Function to validate AWS CLI JSON output
validate_json() {
    local json_string="$1"
    if ! echo "$json_string" | jq empty 2>/dev/null; then
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
extract_json_value() {
    local json_string="$1"
    local key_path="$2"
```

```
    if ! validate_json "$json_string"; then
        return 1
    fi

    echo "$json_string" | jq -r "$key_path" 2>/dev/null || return 1
}

# Function to clean up resources securely
cleanup_resources() {
    echo ""
    echo "=================================================="
    echo "CLEANING UP RESOURCES"
    echo "=================================================="

    if [ -n "${DISTRIBUTION_ID:-}" ] && [ -n "${WEB_ACL_ARN:-}" ]; then
        echo "Disassociating Web ACL from CloudFront distribution..."
        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
            echo "Warning: Could not retrieve AWS account ID"
            return
        fi

        local disassociate_result
        disassociate_result=$(aws wafv2 disassociate-web-acl \
            --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
            --region us-east-1 2>&1) || true

        if echo "$disassociate_result" | grep -qi "error"; then
            echo "Warning: Failed to disassociate Web ACL: $disassociate_result"
        else
            echo "Web ACL disassociated successfully."
        fi
    fi

    if [ -n "${WEB_ACL_ID:-}" ] && [ -n "${WEB_ACL_NAME:-}" ]; then
        echo "Deleting Web ACL..."

        local get_result
        get_result=$(aws wafv2 get-web-acl \
```

```
            --name "$WEB_ACL_NAME" \
            --scope CLOUDFRONT \
            --id "$WEB_ACL_ID" \
            --region us-east-1 2>&1) || true

        if echo "$get_result" | grep -qi "error"; then
            echo "Warning: Failed to get Web ACL for deletion: $get_result"
            echo "You may need to manually delete the Web ACL using the AWS
 Console."
        else
            local latest_token
            latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken'
 2>/dev/null) || latest_token=""

            if [ -n "$latest_token" ]; then
                local delete_result
                delete_result=$(aws wafv2 delete-web-acl \
                    --name "$WEB_ACL_NAME" \
                    --scope CLOUDFRONT \
                    --id "$WEB_ACL_ID" \
                    --lock-token "$latest_token" \
                    --region us-east-1 2>&1) || true

                if echo "$delete_result" | grep -qi "error"; then
                    echo "Warning: Failed to delete Web ACL: $delete_result"
                    echo "You may need to manually delete the Web ACL using the
 AWS Console."
                else
                    echo "Web ACL deleted successfully."
                fi
            else
                echo "Warning: Could not extract lock token for deletion. You may
 need to manually delete the Web ACL."
            fi
        fi
    fi

    echo "Cleanup process completed."
}

# Security: Trap EXIT to ensure cleanup on any exit
trap cleanup_resources EXIT

# Generate a random identifier for resource names using secure method
```

```
RANDOM_ID=$(openssl rand -hex 4) || handle_error "Failed to generate random ID"
WEB_ACL_NAME="MyWebACL-${RANDOM_ID}"
METRIC_NAME="MyWebACLMetrics-${RANDOM_ID}"

echo "Using Web ACL name: $WEB_ACL_NAME"

# Step 1: Create a Web ACL
echo ""
echo "================================================="
echo "STEP 1: Creating Web ACL"
echo "================================================="

local create_result
create_result=$(aws wafv2 create-web-acl \
    --name "$WEB_ACL_NAME" \
    --scope "CLOUDFRONT" \
    --default-action Allow={} \
    --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-waf-gs \
    --region us-east-1 2>&1) || handle_error "Failed to create Web ACL"

if ! validate_json "$create_result"; then
    handle_error "Invalid JSON response from create-web-acl"
fi

# Extract Web ACL ID, ARN, and Lock Token from the response
WEB_ACL_ID=$(extract_json_value "$create_result" '.Summary.Id') || handle_error
 "Failed to extract Web ACL ID"
WEB_ACL_ARN=$(extract_json_value "$create_result" '.Summary.ARN') || handle_error
 "Failed to extract Web ACL ARN"
LOCK_TOKEN=$(extract_json_value "$create_result" '.Summary.LockToken') ||
 handle_error "Failed to extract Lock Token"

echo "Web ACL created successfully with ID: $WEB_ACL_ID"
echo "Lock Token: [REDACTED]"

# Step 2: Add a String Match Rule
echo ""
echo "================================================="
echo "STEP 2: Adding String Match Rule"
echo "================================================="
```

```
for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add string match rule..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true

    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to extract lock token after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    fi

    local update_result
```

```
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
```

```
            handle_error "Failed to add string match rule after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
        handle_error "Failed to add string match rule: $update_result"
    else
        echo "String match rule added successfully."
        break
    fi
done

# Step 3: Add AWS Managed Rules
echo ""
echo "================================================="
echo "STEP 3: Adding AWS Managed Rules"
echo "================================================="

for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add AWS Managed Rules..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true

    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
```

```
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to extract lock token after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    fi

    local update_result
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
```

```
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        },
        {
            "Name": "AWS-AWSManagedRulesCommonRuleSet",
            "Priority": 1,
            "Statement": {
                "ManagedRuleGroupStatement": {
                    "VendorName": "AWS",
                    "Name": "AWSManagedRulesCommonRuleSet",
                    "ExcludedRules": []
                }
            },
            "OverrideAction": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "AWS-AWSManagedRulesCommonRuleSet"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to add AWS Managed Rules after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
```

```
        handle_error "Failed to add AWS Managed Rules: $update_result"
    else
        echo "AWS Managed Rules added successfully."
        break
    fi
done

# Step 4: List CloudFront distributions
echo ""
echo "=================================================="
echo "STEP 4: Listing CloudFront Distributions"
echo "=================================================="

local cf_result
cf_result=$(aws cloudfront list-distributions --query "DistributionList.Items[*].
{Id:Id,DomainName:DomainName}" --output table 2>&1) || cf_result=""

if echo "$cf_result" | grep -qi "error"; then
    echo "Warning: Failed to list CloudFront distributions: $cf_result"
    echo "Continuing without CloudFront association."
    DISTRIBUTION_ID=""
else
    echo "$cf_result"

    # Auto-select first CloudFront distribution if available
    echo ""
    echo "=================================================="
    echo "STEP 5: Associate Web ACL with CloudFront Distribution"
    echo "=================================================="

    local first_dist
    first_dist=$(aws cloudfront list-distributions --query
 "DistributionList.Items[0].Id" --output text 2>&1) || first_dist=""

    if [ -n "$first_dist" ] && [ "$first_dist" != "None" ] && ! echo
 "$first_dist" | grep -qi "error"; then
        DISTRIBUTION_ID="$first_dist"
        echo "Using CloudFront distribution: $DISTRIBUTION_ID"

        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
```

```
            echo "Warning: Could not retrieve AWS account ID for association"
            DISTRIBUTION_ID=""
        else
            local associate_result
            associate_result=$(aws wafv2 associate-web-acl \
                --web-acl-arn "$WEB_ACL_ARN" \
                --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
                --region us-east-1 2>&1) || true

            if echo "$associate_result" | grep -qi "error"; then
                echo "Warning: Failed to associate Web ACL with CloudFront
 distribution: $associate_result"
                echo "Continuing without CloudFront association."
                DISTRIBUTION_ID=""
            else
                echo "Web ACL associated with CloudFront distribution
 successfully."
            fi
        fi
    else
        echo "No CloudFront distributions available. Skipping association."
        DISTRIBUTION_ID=""
    fi
fi

# Display summary of created resources
echo ""
echo "=================================================="
echo "RESOURCE SUMMARY"
echo "=================================================="
echo "Web ACL Name: $WEB_ACL_NAME"
echo "Web ACL ID: $WEB_ACL_ID"
echo "Web ACL ARN: $WEB_ACL_ARN"
if [ -n "${DISTRIBUTION_ID:-}" ]; then
    echo "Associated CloudFront Distribution: $DISTRIBUTION_ID"
fi
echo ""

# Auto-confirm cleanup
echo "=================================================="
echo "CLEANUP CONFIRMATION"
echo "=================================================="
echo "Proceeding with automatic cleanup of all created resources..."
```

```
echo ""
echo "============================================================"
echo "Tutorial completed!"
echo "============================================================"
echo "Log file: $LOG_FILE"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AssociateWebAcl
  - CreateWebAcl
  - DeleteWebAcl
  - DisassociateWebAcl
  - GetCallerIdentity
  - GetWebAcl
  - ListDistributions
  - UpdateWebAcl

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

將硬式編碼秘密移至安全秘密儲存

以下程式碼範例顯示做法：

- 建立 IAM 角色
- 在 Secrets Manager 中建立秘密
- 更新您的應用程式程式碼
- 更新秘密
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# Script to move hardcoded secrets to AWS Secrets Manager
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail

# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "============================================================"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
```

```
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED"
    echo "========================================="

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi

    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi

    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
```

```
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done

        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)

# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"

# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs
```

```
# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
```

```
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
                }
            }
        }
    ]
}
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)

check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
```

```
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)

check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "======================================="
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi
```

```
echo ""
echo "============================================================"
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""

echo "Script completed at $(date)"
exit 0
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy
  - GetCallerIdentity

- GetSecretValue
- PutResourcePolicy
- UpdateSecret

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用錯誤注入在虛擬機器執行個體上執行 CPU 壓力測試

以下程式碼範例顯示做法：

- 建立 IAM 角色
- 建立 CloudWatch 警示
- 建立實驗範本
- 執行實驗
- 驗證結果
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial

#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
```

```
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "============================================="

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
            return 0
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
```

```
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
    fi

    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true

        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" || true
    fi

    if [ -n "$FIS_ROLE_NAME" ]; then
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
 "$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
```

```
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()

echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
```

```
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Attach policy to the role
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
cat > ec2-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
```

```
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")

# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
```

```
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"

# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")
```

```
# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"

# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60

# Check alarm state
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"

    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi
```

```
echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
```

```
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
  "stopConditions": [
    {
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
EOF

# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")

echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
```

```
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")

echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail

    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi

    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done

if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi
```

```
echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"

# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")

# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
```

```
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
echo "$METRICS_OUTPUT"

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCES CREATED"
echo "============================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "============================================="

# Prompt for cleanup
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
 "$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"
```

```
    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
"$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"

    # Terminate EC2 instance
    echo "Terminating EC2 instance $INSTANCE_ID..."
    TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
    check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
    echo "Waiting for instance to terminate..."
    aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"

    # Clean up IAM resources
    echo "Removing role from instance profile..."
    REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME" \
      --role-name "$EC2_ROLE_NAME")
    check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"

    echo "Deleting instance profile..."
    DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME")
    check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

    echo "Deleting FIS role policy..."
    DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
      --role-name "$FIS_ROLE_NAME" \
      --policy-name "$FIS_POLICY_NAME")
    check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

    echo "Detaching policy from EC2 role..."
    DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
      --role-name "$EC2_ROLE_NAME" \
      --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
    check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

    echo "Deleting FIS role..."
    DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$FIS_ROLE_NAME")
    check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

    echo "Deleting EC2 role..."
    DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
```

```
      --role-name "$EC2_ROLE_NAME")
    check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

    # Clean up temporary files
    echo "Cleaning up temporary files..."
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
 experiment-template.json metric-query.json

    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi

echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile
  - DeleteRole
  - DeleteRolePolicy
  - DescribeAlarms
  - DescribeImages
  - DetachRolePolicy
  - GetCallerIdentity
  - GetExperiment
  - GetMetricData

- GetRole
- MonitorInstances
- PutMetricAlarm
- PutRolePolicy
- RemoveRoleFromInstanceProfile
- RunInstances
- StartExperiment
- StopExperiment
- TerminateInstances
- 等候

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

設定系統管理

以下程式碼範例顯示做法：

- 建立 Systems Manager 的 IAM 許可
- 建立 Systems Manager 的 IAM 角色
- 設定 Systems Manager
- 驗證設定
- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在範例開發人員教學課程儲存庫中設定和執行。

```
#!/bin/bash
```

```
# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
# Version 17 fixes:
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
```

```
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"

        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
```

```
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
                ;;
            "SSM_CONFIG_MANAGER")
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done

    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================"
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""

# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"
```

```
# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."

# Create policy document
cat > ssm-onboarding-policy.json << 'EOF'
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmReadOnly",
       "Effect": "Allow",
       "Action": [
         "ssm:DescribeAutomationExecutions",
         "ssm:GetAutomationExecution",
         "ssm:ListAssociations",
         "ssm:DescribeAssociation",
         "ssm:ListDocuments",
         "ssm:ListResourceDataSync",
         "ssm:DescribePatchBaselines",
         "ssm:GetPatchBaseline",
         "ssm:DescribeMaintenanceWindows",
         "ssm:DescribeMaintenanceWindowTasks"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmDocument",
       "Effect": "Allow",
       "Action": [
         "ssm:GetDocument",
         "ssm:DescribeDocument"
       ],
       "Resource": [
         "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
         "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
```

```
        ]
      },
      {
        "Sid": "SsmEnableExplorer",
        "Effect": "Allow",
        "Action": "ssm:StartAutomationExecution",
        "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
      },
      {
        "Sid": "SsmExplorerRds",
        "Effect": "Allow",
        "Action": [
          "ssm:GetOpsSummary",
          "ssm:CreateResourceDataSync",
          "ssm:UpdateResourceDataSync"
        ],
        "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
      },
      {
        "Sid": "OrgsReadOnly",
        "Effect": "Allow",
        "Action": [
          "organizations:DescribeAccount",
          "organizations:DescribeOrganization",
          "organizations:ListDelegatedAdministrators",
          "organizations:ListRoots",
          "organizations:ListParents",
          "organizations:ListOrganizationalUnitsForParent",
          "organizations:DescribeOrganizationalUnit",
          "organizations:ListAWSServiceAccessForOrganization"
        ],
        "Resource": "*"
      },
      {
        "Sid": "OrgsAdministration",
        "Effect": "Allow",
        "Action": [
          "organizations:EnableAWSServiceAccess",
          "organizations:RegisterDelegatedAdministrator",
          "organizations:DeregisterDelegatedAdministrator"
        ],
        "Resource": "*",
        "Condition": {
          "StringEquals": {
```

```
            "organizations:ServicePrincipal": [
              "ssm.amazonaws.com",
              "ssm-quicksetup.amazonaws.com",
              "member.org.stacksets.cloudformation.amazonaws.com",
              "resource-explorer-2.amazonaws.com"
            ]
          }
        }
      },
      {
        "Sid": "CfnReadOnly",
        "Effect": "Allow",
        "Action": [
          "cloudformation:ListStacks",
          "cloudformation:DescribeStacks",
          "cloudformation:ListStackSets",
          "cloudformation:DescribeOrganizationsAccess"
        ],
        "Resource": "*"
      },
      {
        "Sid": "OrgCfnAccess",
        "Effect": "Allow",
        "Action": [
          "cloudformation:ActivateOrganizationsAccess"
        ],
        "Resource": "*"
      },
      {
        "Sid": "CfnStackActions",
        "Effect": "Allow",
        "Action": [
          "cloudformation:CreateStack",
          "cloudformation:DeleteStack",
          "cloudformation:DescribeStackResources",
          "cloudformation:DescribeStackEvents",
          "cloudformation:GetTemplate",
          "cloudformation:RollbackStack",
          "cloudformation:TagResource",
          "cloudformation:UntagResource",
          "cloudformation:UpdateStack"
        ],
        "Resource": [
          "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
```

```
        "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*"
      ]
    },
    {
      "Sid": "CfnStackSetActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStackInstances",
        "cloudformation:CreateStackSet",
        "cloudformation:DeleteStackInstances",
        "cloudformation:DeleteStackSet",
        "cloudformation:DescribeStackInstance",
        "cloudformation:DetectStackSetDrift",
        "cloudformation:ListStackInstanceResourceDrifts",
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStackSet"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*",
        "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
      ]
    },
    {
      "Sid": "ValidationReadonlyActions",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:GetRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamRolesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
```

```
        "iam:DeleteRole",
        "iam:GetRole",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ]
    },
    {
      "Sid": "IamPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "cloudformation.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "IamRolesPoliciesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
```

```
        "ArnEquals": {
          "iam:PolicyARN": [
            "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
            "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
          ]
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
        "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
        "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
        "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
        "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
```

```
    exit 1
fi

# Track the created policy
track_resource "IAM_POLICY" "$POLICY_ARN"
aws iam tag-policy --policy-arn "$POLICY_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"

# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
```

```
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json

# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
```

```
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"

echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"

# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF

echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
```

```
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created configuration manager
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30

# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"

echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "============================================="
echo "CREATED RESOURCES"
echo "============================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
```

```
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
    echo "Cleanup completed." | tee -a "$LOG_FILE"
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
  - AttachRolePolicy
  - CreateConfigurationManager
  - CreatePolicy
  - CreateRole
  - DeleteConfigurationManager
  - DeletePolicy
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetConfigurationManager

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

IAM 和 中的安全性 AWS STS

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構專為滿足最安全敏感組織的需求而建置。

安全性是 AWS 與您之間的共同責任。共同責任模式將其描述為雲端的安全性，和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 Cloud AWS 中執行 AWS 服務的基礎設施。 AWS 也為您提供可安全使用的服務。在AWS 合規計畫中，第三方稽核人員會定期測試和驗證我們安全的有效性。若要了解適用於 AWS Identity and Access Management (IAM) 的合規計劃，請參閱AWS 合規計劃的 服務範圍。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您了解如何在使用 AWS Identity and Access Management (IAM) 和 AWS Security Token Service () 時套用共同責任模型AWS STS。下列主題說明如何設定 IAM 和 AWS STS 以符合您的安全與合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 IAM 資源。

目錄

- AWS 安全登入資料
- AWS 安全稽核準則
- 中的資料保護 AWS Identity and Access Management
- 在 中記錄和監控 AWS Identity and Access Management
- 的合規驗證 AWS Identity and Access Management
- 中的彈性 AWS Identity and Access Management
- 中的基礎設施安全 AWS Identity and Access Management
- 中的組態和漏洞分析 AWS Identity and Access Management
- AWS 的 受管政策 AWS Identity and Access Management、IAM Access Analyzer 和帳戶存取管理員
- IAM 外部的安全功能

AWS 安全登入資料

當您與 互動時 AWS，您可以指定 AWS 安全登入資料來驗證您的身分，以及您是否有權存取您請求的資源。 AWS 會使用安全登入資料來驗證和授權您的請求。

例如，如果要從 Amazon Simple Storage Service (Amazon S3) 儲存貯體下載受保護的檔案，您的憑證必須允許此存取動作。如果您的登入資料未顯示您獲得下載檔案的授權， 會 AWS 拒絕您的請求。不過，您不需要 AWS 安全登入資料，即可在公開共用的 Amazon S3 儲存貯體中下載檔案。

中有不同類型的使用者 AWS，每個使用者都有自己的安全登入資料：

- 帳戶擁有者 （根使用者） — 建立 AWS 帳戶 且具有完整存取權的使用者。
- AWS IAM Identity Center 使用者 — 在 中管理的使用者 AWS IAM Identity Center。
- 聯合主體 — 來自外部身分提供者的使用者，他們 AWS 透過聯合獲得對 的臨時存取權。如需聯合身分的詳細資訊，請參閱 身分提供者和 的聯合身分 AWS。
- IAM 使用者 — 在 AWS Identity and Access Management (IAM) 服務中建立的個別使用者。

使用者擁有長期或暫時安全憑證。根使用者、IAM 使用者和存取金鑰具有不會過期的長期安全憑證。若要保護長期憑證，您應有現成的程序以管理存取金鑰、變更密碼和啟用 MFA。

若要簡化 中跨成員帳戶的根使用者憑證管理 AWS Organizations，您可以使用 集中保護 AWS 帳戶 受管 的根使用者憑證 AWS Organizations。 集中管理成員帳戶的根存取權可讓您集中移除和防止長期根使用者憑證復原，防止大規模的意外根存取。

IAM AWS STS 角色 AWS IAM Identity Center 的使用者和聯合身分使用者主體具有臨時安全登入資料。暫時安全憑證會在定義的一段時間或使用者結束工作階段後過期。暫時憑證的作用幾乎與長期憑證完全相同，而其差異如下：

- 暫時安全憑證是「短期」的，如其名稱所暗示。其可設定為在任何地方持續幾分鐘到幾小時不等。登入資料過期後， AWS 就無法再辨識這些登入資料，也無法允許從使用它們提出的 API 請求進行任何類型的存取。
- 暫時性安全憑證不會與使用者一起儲存，但會動態產生並在請求時提供給使用者。當暫時性安全憑證到期時 (或者即使在此之前)，使用者可以請求新的憑證，只要使用者的請求仍具有可這麼做的許可。

因此，相較於長期憑證，暫時性憑證具有下列優點：

- 您不需要向應用程式分發或嵌入長期 AWS 安全登入資料。

- 您可以為使用者提供 AWS 資源的存取權，而不必為其定義 AWS 身分。暫時憑證是角色和聯合身分的基礎。
- 臨時安全憑證的存留期有限，因此當不再需要時，您不需要更新它們或明確予以撤銷。暫時性安全憑證到期之後，就無法重複使用。您可以指定憑證的有效期，達到最長限制。

安全考量

建議您在決定適用於 AWS 帳戶的安全規定時，考量下列資訊：

- 當您建立 時 AWS 帳戶，我們會建立 帳戶根使用者。根使用者 (帳戶擁有者) 的憑證可以完整存取帳戶中的所有資源。您使用根使用者執行的第一個任務是將另一個使用者管理許可授予 ， AWS 帳戶以便將根使用者的用量降至最低。
- 多重要素驗證 (MFA) 為可以存取 AWS 帳戶的使用者提供多一層級的安全防護。為了提高安全性，我們建議您對 AWS 帳戶根使用者 登入資料和所有 IAM 使用者要求 MFA。如需詳細資訊，請參閱AWS IAM 中的多重要素驗證。
- AWS 需要不同類型的安全登入資料，取決於您的存取方式 AWS 和 AWS 使用者類型。例如，您在使用存取金鑰進行程式設計呼叫 AWS 管理主控台 時使用 的登入憑證 AWS。如需判斷使用者類型和登入頁面的說明，請參閱AWS 登入 《 使用者指南》中的什麼是 AWS 登入。
- 您無法使用 IAM 政策來明確拒絕根使用者存取權。您只能使用 AWS Organizations 服務控制政策 (SCP) 來限制根使用者的許可。
- 如果您忘記或遺失根使用者密碼，則必須擁有與您帳戶相關聯之電子郵件地址的存取權，才能重設密碼。
- 如果遺失根使用者存取金鑰，則您必須能夠以根使用者身分登入您的帳戶，才能建立新的金鑰。
- 請勿將您的根使用者用於日常任務。請將其用來執行只能由根使用者執行的任務。如需檢視需要您以根使用者身分登入的任務完整清單，請參閱 需要根使用者憑證的任務。
- 安全登入資料是帳戶專屬的資料。如果您可以存取多個 AWS 帳戶，則每個帳戶都有單獨憑證。
- 政策會決定使用者、角色或使用者群組成員可以對哪些 AWS 資源執行的動作，以及在哪些條件下執行的動作。使用 政策，您可以安全地控制對 中 AWS 服務 和資源的存取 AWS 帳戶。如果您必須修改或撤銷許可以回應安全性事件，則您可以刪除或修改政策，而不是直接變更身分。
- 請務必將緊急存取 IAM 使用者的登入憑證，以及為程式設計存取而建立的任何存取金鑰儲存在安全的位置。如果您遺失存取金鑰，則必須登入帳戶建立新金鑰。
- 強烈建議您使用 IAM 角色和聯合身分主體提供的臨時憑證，而不要使用 IAM 使用者和存取金鑰提供的長期憑證。

使用 AWS 安全登入資料進行程式設計存取

我們建議您盡可能使用短期存取金鑰對 AWS 或 進行程式設計呼叫，以使用 AWS Command Line Interface 或 AWS Tools for PowerShell。不過，您也可以針對這些目的使用長期 AWS 存取金鑰。

當您建立長期存取金鑰時，會將存取金鑰 ID (例如 AKIAIOSFODNN7EXAMPLE) 和私密存取金鑰 (例如 wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY) 建立為一個集合。只有在您建立的時候才可使用私密存取金鑰進行下載。如果您沒有下載私密存取金鑰，或者遺失，則必須重新建立。

在許多情況下，您不需要永遠不會過期的長期存取金鑰 (就像您為 IAM 使用者建立存取金鑰時一樣)。反之，您可以建立 IAM 角色並產生暫時安全憑證。暫時安全憑證包含存取金鑰 ID 和私密存取金鑰，但其中也包含指出憑證何時到期的安全權杖。臨時安全資料過期之後即不再有效。如需詳細資訊，請參閱長期存取金鑰的替代方案

以 開頭的存取金鑰 IDs AKIA是 IAM 使用者或 AWS 帳戶 根使用者的長期存取金鑰。以 開頭的存取金鑰 IDs ASIA是您使用 AWS STS 操作建立的臨時憑證存取金鑰。

如果使用者想要與 AWS 外部互動，則需要程式設計存取 AWS 管理主控台。授予程式設計存取權的方式取決於正在存取的使用者類型 AWS。

若要授予使用者程式設計存取權，請選擇下列其中一個選項。

哪個使用者需要程式設計存取權？	到	根據
IAM	（建議）使用主控台登入資料做為臨時登入資料，以簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	請依照您要使用的介面所提供的指示操作。 • 如需 AWS CLI，請參閱AWS Command Line Interface 《使用者指南》中的登入以進行 AWS 本機開發。 • AWS SDKs請參閱 AWS SDKs和工具參考指南中的登入以進行 AWS 本機開發。
人力資源身分	使用暫時登入資料簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	請依照您要使用的介面所提供的指示操作。

哪個使用者需要程式設計存取權？	到	根據
(IAM Identity Center 中管理的使用者)		• 如需 AWS CLI，請參閱AWS Command Line Interface 《使用者指南》中的設定 AWS CLI 要使用 AWS IAM Identity Center的 。 • AWS SDKs、工具和 AWS APIs，請參閱 AWS SDK 和工具參考指南中的 SDKsIAM Identity Center 身分驗證。
IAM	使用暫時登入資料簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	遵循《IAM 使用者指南》中將臨時登入資料與 AWS 資源搭配使用的指示。
IAM	(不建議使用) 使用長期憑證簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	請依照您要使用的介面所提供的指示操作。 • 如需 AWS CLI，請參閱AWS Command Line Interface 《使用者指南》中的使用 IAM 使用者憑證進行身分驗證。 • AWS SDKs和工具，請參閱 AWS SDKs和工具參考指南中的使用長期憑證進行身分驗證。 • 對於 AWS APIs，請參閱《IAM 使用者指南》中的管理 IAM 使用者的存取金鑰。

長期存取金鑰的替代方案

對於許多常見使用案例，有可以替代長期存取金鑰的方案。為了提高帳戶安全性，請考慮以下事項。

- 請勿在應用程式程式碼或程式碼儲存庫中嵌入長期存取金鑰和私密存取金鑰 – 反之，使用 AWS Secrets Manager或其他私密管理解決方案，因此您不必以純文字硬式編碼金鑰。然後，應用程式或用戶端可以在需要時擷取密碼。如需詳細資訊，請參閱什麼是 AWS Secrets Manager？ AWS Secrets Manager 《 使用者指南》中的 。

- 盡可能使用 IAM 角色產生暫時安全憑證 - 務必盡可能使用機制頒發暫時安全憑證，而非長期存取金鑰。暫時安全憑證更加安全，因為它們不會與使用者一起儲存，但會動態產生並在請求時提供給使用者。臨時安全憑證的生命週期有限，因此您不必對其進行管理或更新。提供暫時存取金鑰的機制包含 IAM 角色或 IAM Identity Center 使用者的身分驗證。對於在 外部執行的機器 AWS ，您可以使用 AWS Identity and Access Management Roles Anywhere。
- 針對 AWS Command Line Interface (AWS CLI) 或 **`aws-shell`** 使用長期存取金鑰的替代方案 – 替代方案包含以下內容。
  - 使用主控台登入資料登入進行 AWS 本機開發。您可以使用 第 2 AWS CLI 版和 `aws login`命令來產生短期登入資料，以使用您的主控台登入資料執行 AWS CLI 命令。若要進一步了解，請參閱AWS Command Line Interface 《 使用者指南》中的登入以進行 AWS 本機開發。
  - AWS CloudShell 是以瀏覽器為基礎的預先驗證 Shell，您可以直接從 啟動 AWS 管理主控台。您可以透過 AWS 服務 您偏好的 shell (Bash、Powershell 或 Z shell) 對 執行 AWS CLI 命令。當您這樣做時，無需下載或安裝命令列工具。如需詳細資訊，請參閱《 AWS CloudShell使用者指南》中的什麼是AWS CloudShell ？。
- 請勿為需要存取應用程式的人類使用者建立長期存取金鑰，或 AWS 服務 – IAM Identity Center 可以為您的外部 IdP 使用者產生臨時存取憑證以供其存取 AWS 服務。這樣就不需要在 IAM 中建立和管理長期憑證。在 IAM Identity Center 中，建立可授予外部 IdP 使用者存取權的 IAM Identity Center 許可集。然後將群組從 IAM Identity Center 指派給所選 中的許可集 AWS 帳戶。如需詳細資訊，請參閱AWS IAM Identity Center 《 使用者指南》中的什麼是 AWS IAM Identity Center、連接至外部身分提供者和許可集。
- 請勿在 AWS 運算服務中存放長期存取金鑰 – 請改為指派 IAM 角色來運算資源。這會自動提供暫時憑證以授予存取權。例如，當您建立連接至 Amazon EC2 執行個體的執行個體描述檔時，您可以將 AWS 角色指派給執行個體，並將其提供給其所有應用程式。執行個體設定檔包含該角色，並且可讓 Amazon EC2 執行個體上執行的程式取得臨時憑證。若要進一步了解，請參閱利用 IAM 角色來授予許可給 Amazon EC2 執行個體上執行的應用程式。

AWS 安全稽核準則

定期稽核您的安全組態，確保其符合您目前的業務需求。稽核的目的是讓您有機會移除不需要的 IAM 使用者、角色、群組和政策，確保使用者和軟體沒有過多的許可。

以下是系統性檢閱和監控 AWS 資源以取得安全最佳實務的指導方針。

> **Tip**
>
> 透過使用 AWS Security Hub CSPM 監視您 IAM 的使用狀況，因為它關係到安全最佳實務。Security Hub CSPM 使用安全控制來評估資源組態和安全標準，以協助您遵守各種合規架構。如需使用 Security Hub CSPM 評估 IAM 資源的詳細資訊，請參閱 AWS Security Hub CSPM 《使用者指南》中的 AWS Identity and Access Management 控制項。

目錄

- 何時執行安全性稽核
- 稽核準則
- 檢閱 AWS 您的帳戶登入資料
- 檢閱 IAM 使用者
- 檢閱 IAM 群組
- 檢閱 IAM 角色
- 檢閱 SAML 和 OpenID Connect (OIDC) 的 IAM 提供者
- 檢閱行動應用程式
- 檢閱 IAM 政策的要訣

何時執行安全性稽核

在下列情況下稽核安全組態：

- 定期。定期執行本文件中所述的步驟，做為確保安全性的最佳實務。
- 如果組織內部有所變化，例如人員離職。
- 如果您已停止使用一或多個個別 AWS 服務，以確認您已移除帳戶中使用者不再需要的許可。
- 如果您已在帳戶中新增或移除軟體，例如 Amazon EC2 執行個體上的應用程式、 OpsWorks 堆疊、CloudFormation 範本等。
- 如果您懷疑未經授權的人員可能存取了您的帳戶。

稽核準則

當您檢閱帳戶的安全組態時，請依照以下指導方針：

- 完整徹底。查看安全組態的所有層面，包括很少使用的部分。
- 絕不假設。如果您不熟悉安全組態的某些部分 (例如，特定政策或角色存在背後的原因)，請查清楚業務的需要，直到理解潛在的風險為止。
- 保持簡單。為了讓稽核 (和管理) 能更輕鬆執行，請使用 IAM 群組、IAM 角色、一致的命名機制和直截了當的政策。

檢閱 AWS 您的帳戶登入資料

當您稽核 AWS 帳戶登入資料時，請採取下列步驟：

1. 如果您有未使用根帳戶的存取金鑰，可移除它們。我們強烈建議您不要將根存取金鑰用於日常使用 AWS，而是使用具有臨時登入資料的使用者，例如 AWS IAM Identity Center 的使用者。
2. 如果您需要帳戶的存取金鑰，請確保在需要時更新這些金鑰。

檢閱 IAM 使用者

在稽核現有的 IAM 使用者時，請採取以下步驟：

1. 列出您的使用者，然後刪除不需要的使用者。
2. 從不需要存取權的群組中移除使用者。
3. 檢閱使用者所在群組中所連接的政策。請參閱 檢閱 IAM 政策的要訣。
4. 刪除使用者不需要，或者可能已公開的安全登入資料。例如，用於應用程式的 IAM 使用者不需要密碼（只需要登入 AWS 網站）。同樣地，如果使用者不使用存取金鑰，也就沒有理由具備存取金鑰。如需詳細資訊，請參閱管理 IAM 使用者的密碼和管理 IAM 使用者的存取金鑰。

   您可以產生並下載「憑證報告」，其中會列出帳戶中的所有 IAM 使用者，及其各種憑證的狀態，包括密碼、存取金鑰和 MFA 裝置。對於密碼和存取金鑰、憑證報告會顯示密碼或存取金鑰最近使用的日期和時間。請考慮從您的帳戶中移除最近未使用的憑證。(請勿移除您的緊急存取使用者。) 如需詳細資訊，請參閱取得 AWS 您帳戶的登入資料報告。
5. 對於需要長期憑證的使用案例，請視需要更新密碼和存取金鑰。如需詳細資訊，請參閱管理 IAM 使用者的密碼和管理 IAM 使用者的存取金鑰。

6. 最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 使用臨時憑證存取 。如果可能，請從 IAM 使用者轉換為聯合身分主體，例如 IAM Identity Center 中的使用者。保留應用程式所需的最少 IAM 使用者數量。

檢閱 IAM 群組

在稽核 IAM 群組時，請採取以下步驟：

1. 列出您的群組，然後刪除未使用的群組。
2. 檢閱每個群組中的使用者，並移除不屬於該群組的使用者。
3. 檢閱群組連接的政策。請參閱 檢閱 IAM 政策的要訣。

檢閱 IAM 角色

在稽核 IAM 角色時，請採取以下步驟：

1. 列出您的角色，然後刪除未使用的角色。
2. 檢閱角色的信任政策。確認您知道委託人是誰，而且了解為什麼該帳戶或使用者必須要能擔任該角色。
3. 檢閱角色的存取政策，確認授與適當的權限給能擔任該角色的人，請參閱 檢閱 IAM 政策的要訣。

檢閱 SAML 和 OpenID Connect (OIDC) 的 IAM 提供者

如果您為了建立 SAML 或 OIDC 身分提供者 (IdP) 的信任而建立了 IAM 實體，請採取下列步驟：

1. 刪除未使用的供應商。
2. 下載並檢閱每個 SAML IdP 的 AWS 中繼資料文件，並確保這些文件反映您目前的業務需求。
3. 從 SAML IdP 取得最新的中繼資料文件，並更新 IAM 中的提供者。

檢閱行動應用程式

如果您已建立向 提出請求的行動應用程式 AWS，請採取下列步驟：

1. 確認行動應用程式不包含內嵌的存取金鑰，即使位於加密儲存裝置中也是如此。
2. 使用專屬用途的 API 取得應用程式的暫時憑證。

> **Note**
>
> 建議您使用 Amazon Cognito 來管理應用程式中的使用者身分。此服務可讓您使用 Login with Amazon、Facebook、Google 或任何與身分提供者相容的 OpenID Connect (OIDC)，驗證使用者的身分。如需詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的Amazon Cognito 身分集區。

檢閱 IAM 政策的要訣

政策是功能強大而且微妙的工具，因此請務必研究並了解每個政策所授予的許可。檢閱政策時，請使用以下指導方針：

- 將政策連接至群組或角色，而非個別使用者。如果個別使用者具有政策，請確認您了解該使用者需要政策的原因。
- 確保 IAM 使用者、群組和角色具有所需的許可，且沒有任何其他許可。
- 使用 IAM 政策模擬器測試連接到使用者或群組的政策。
- 請記住，使用者的許可是所有適用政策的結果：身分型政策（在使用者、群組或角色上）和資源型政策（在 Amazon S3 儲存貯體、Amazon SQS 佇列、Amazon SNS 主題和 AWS KMS 金鑰等資源上）。請務必檢查套用到使用者的所有政策，並了解授予個別使用者的完整權限集。
- 請注意，讓使用者建立 IAM 使用者、群組、角色或政策，並將政策連接至主體實體，即表示對該使用者有效授予帳戶中所有資源的所有許可。可以建立政策並將它們連接到使用者、群組或角色的使用者也會授予他們本身任何許可。一般而言，只要是您不信任的使用者或角色，就不要授予能完全存取帳戶中資源的 IAM 許可。執行安全稽核時，請確認將下列 IAM 許可授予受信任身分：
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`
  - `iam:PutUserPolicy`
  - `iam:CreatePolicy`
  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- 確認政策不會對您沒在使用的服務授予許可。例如，如果您使用 AWS 受管政策，請確保帳戶中正在使用的 AWS 受管政策適用於您實際使用的 服務。若要了解您的帳戶正在使用哪些 AWS 受管政

策，請使用 IAM GetAccountAuthorizationDetails API (AWS CLI 命令：aws iam get-account-authorization-details)。

- 如果政策授予使用者許可，以啟動 Amazon EC2 執行個體，也可能會允許 iam:PassRole 動作，但若是如此，應該明確列出使用者可傳遞到 Amazon EC2 執行個體的角色。
- 檢查 Action 或 Resource 元素中包含 * 的任何值。如果可能，請授予使用者需要的個別動作和資源的 Allow 存取權。不過，以下原因說明為何在政策中適合使用 *：
  - 此政策旨在授予管理層級的許可。
  - 基於便利性考量，萬用字元會用於一組類似的動作 (例如，Describe*)，而且您放心以此方式參考的完整動作清單。
  - 萬用字元會用來以指示一個類別的資源或資源路徑 (例如，arn:aws:iam::*account-id*:users/division_abc/*)，而且您放心在該類別或路徑中授予所有資源的存取權。
  - 服務動作不支援資源層級權限，而資源的唯一選擇是 *。
- 檢查政策名稱，確認其反映政策的功能。例如，雖然政策名稱中可能會有「唯讀」兩字，但是實際上該政策可能會授予寫入或變更的許可。

如需有關規劃安全性稽核的詳細資訊，請參閱《AWS 架構中心》中的安全性、身分與合規的最佳實務。

中的資料保護 AWS Identity and Access Management

AWS 共同責任模型適用於 中的資料保護 AWS Identity and Access Management。如此模型所述，AWS 負責保護執行所有 的全域基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。 如需歐洲資料保護的相關資訊，請參閱「一般資料保護法規 (GDPR) 中心」。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 CloudTrail 追蹤。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。

- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-3。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 IAM 或使用主控台、API AWS CLI或其他 AWS 服務 AWS SDKs 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

IAM 和 中的資料加密 AWS STS

資料加密通常分為兩類：靜態加密和傳輸中加密。

靜態加密

由 IAM 收集和儲存的資料都會進行靜態加密。

- IAM – IAM 內收集並存放的資料包括 IP 地址、客戶帳戶中繼資料，以及包含密碼的客戶識別資料。系統會透過使用 AES 256 或使用 SHA 256 進行雜湊，對客戶帳戶中繼資料和客戶識別資料進行靜態加密。
- AWS STS – AWS STS 不會收集客戶內容，除了記錄成功、錯誤和錯誤的服務請求的服務日誌。

傳輸中加密

客戶識別資料 (包括密碼) 會在傳輸過程中使用 TLS 1.2 和 1.3 進行加密。所有 AWS STS 端點都支援 HTTPS 加密傳輸中的資料。如需 AWS STS 端點的清單，請參閱 AWS STS 區域和端點。

IAM 和 中的金鑰管理 AWS STS

您無法使用 IAM 或 管理加密金鑰 AWS STS。如需加密金鑰的詳細資訊，請參閱什麼是 AWS KMS？《 AWS Key Management Service 開發人員指南》中的

IAM 和 中的網際網路流量隱私權 AWS STS

對 IAM 的請求必須使用 Transport Layer Security 通訊協定 (TLS) 提出。您可以使用 VPC 端點保護 AWS STS 對服務的連線。如需進一步了解，請參閱介面 VPC 端點。

在 中記錄和監控 AWS Identity and Access Management

監控是維護 AWS Identity and Access Management (IAM)、 AWS Security Token Service (AWS STS) 和其他 AWS 解決方案的可靠性、可用性和效能的重要部分。 AWS 提供數種工具來監控您的 AWS 資源並回應潛在事件：

- AWS CloudTrail 會將 IAM 和 的所有 API 呼叫擷取 AWS STS 為事件，包括來自主控台的呼叫和 API 呼叫。若要進一步了解如何搭配 IAM 和 使用 CloudTrail AWS STS，請參閱 使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail。如需有關 CloudTrail 的相關資訊，請參閱 AWS CloudTrail 使用者指南。
- AWS Identity and Access Management 和 Access Analyzer 可協助您識別組織和帳戶中與外部實體共用的資源，例如 Amazon S3 儲存貯體或 IAM 角色。這有助於您發現非預期存取資源和資料的情況，避免產生安全性風險。若要進一步了解，請參閱 什麼是 IAM Access Analyzer？。
- Amazon CloudWatch AWS 會即時監控您的 AWS 資源和您在 上執行的應用程式。您可以收集和追蹤指標、建立自訂儀板表，以及設定警示，在特定指標達到您指定的閾值時通知您或採取動作。例如，您可以讓 CloudWatch 追蹤 CPU 使用量或其他 Amazon EC2 執行個體指標，並在需要時自動啟動新的執行個體。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南。
- Amazon CloudWatch Logs 可讓您監控、存放和存取來自 Amazon EC2 執行個體、CloudTrail 及其他來源的日誌檔案。CloudWatch Logs 可監控日誌檔案中的資訊，並在達到特定閾值時通知您。您也可以將日誌資料存檔在高耐用性的儲存空間。如需詳細資訊，請參閱 Amazon CloudWatch Logs 使用者指南。

如需適用於 IAM 的其他資源和安全最佳實務，請參閱 中的安全最佳實務和使用案例 AWS Identity and Access Management。

主題

- 使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail
- 在 中追蹤特殊權限任務 AWS CloudTrail

使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail

IAM 和 AWS STS 與 整合 AWS CloudTrail，此服務提供 IAM 使用者或角色所採取動作的記錄。CloudTrail 會將 IAM 和 的所有 API 呼叫擷取 AWS STS 為事件，包括來自主控台和 API 呼叫的呼叫。若您建立追蹤，便可將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的 Event history (事件歷史記錄) 檢視最新事件。您可

以使用 CloudTrail 來取得對 IAM 或 提出之請求的相關資訊 AWS STS。例如，您可以檢視發出請求的 IP 地址、發出請求的人員及時間，以及其他詳細資訊。

若要進一步了解 CloudTrail，請參閱《AWS CloudTrail 使用者指南》。

主題

- CloudTrail 中的 IAM 和 AWS STS 資訊
- 記錄 IAM 和 AWS STS API 請求
- 記錄對其他服務的 API 請求 AWS
- 記錄使用者登入事件
- 記錄暫時憑證的登入事件
- CloudTrail 日誌中的 IAM API 事件範例
- CloudTrail 日誌中的範例 AWS STS API 事件
- CloudTrail 日誌中的範例登入事件
- IAM 角色信任政策行為

CloudTrail 中的 IAM 和 AWS STS 資訊

當您建立帳戶 AWS 帳戶 時，您的 上會啟用 CloudTrail。當活動在 IAM 或 中發生時 AWS STS，該活動會與事件歷史記錄中的其他服務 AWS 事件一起記錄在 CloudTrail 事件中。您可以在 中檢視、搜尋和下載最近的事件 AWS 帳戶。如需詳細資訊，請參閱《使用 CloudTrail 事件歷史記錄檢視事件》https://docs.aws.amazon.com/awscloudtrail/latest/userguide/view-cloudtrail-events.html。

若要持續記錄 中的事件 AWS 帳戶，包括 IAM 和 的事件 AWS STS，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。根據預設，當您在主控台建立線索時，線索會套用到所有 區域。線索會記錄 AWS 分割區中所有區域的事件，並將日誌檔案交付至您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析和處理 CloudTrail 日誌中所收集的事件資料。如需詳細資訊，請參閱：

- 建立追蹤的概觀
- CloudTrail 支援的服務和整合
- 設定 CloudTrail 的 Amazon SNS 通知
- 從多個區域接收 CloudTrail 日誌檔案，以及從多個帳戶接收 CloudTrail 日誌檔案

CloudTrail 會記錄所有 IAM 和 AWS STS 動作，並記錄在 IAM API 參考和 AWS Security Token Service API 參考中。

記錄 IAM 和 AWS STS API 請求

CloudTrail 會將所有已驗證的 API 請求記錄到 IAM 和 AWS STS API 操作。CloudTrail 也會將未經驗證的請求記錄到 AWS STS 動作 `AssumeRoleWithSAML`和 `AssumeRoleWithWebIdentity`，並記錄身分提供者提供的資訊。不過，某些未驗證的 AWS STS 請求可能不會記錄，因為它們不符合充分有效以信任為合法請求的最低期望。CloudTrail 不會針對透過 `AssumeRoot` API 啟動的跨帳戶角色假設和特權工作階段，在目標帳戶中記錄拒絕的 AWS STS 請求。

您可以使用記錄的資訊，將具有所擔任角色的 OIDC 或 SAML 聯合身分主體發出的呼叫映射回最初的外部聯合身分呼叫者。如果是 `AssumeRole`，您可以將呼叫對應回原始 AWS 服務或原始使用者的帳戶。CloudTrail 日誌項目中 JSON 資料的 `userIdentity` 部分包含將 AssumeRole* 請求映射至特定工作階段主體所需的資訊。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 CloudTrail userIdentity 元素。

AWS CloudTrail 當 IAM 使用者使用 MFA 登入時， 日誌將包含 MFA 資訊。如果 IAM 使用者擔任 IAM 角色，CloudTrail 也會將 `mfaAuthenticated: true` 記錄在使用該擔任角色執行的動作的 `sessionContext` 屬性中。不過，CloudTrail 記錄與使用擔任角色的憑證進行 API 呼叫時 IAM 所需的記錄是分開的。如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

例如，對 IAM `CreateUser`、`DeleteRole`、`ListGroups` 和其他 API 操作的呼叫都是由 CloudTrail 記錄。

本主題稍後會提供此類日誌項目的範例。

記錄對其他服務的 API 請求 AWS

CloudTrail 會記錄對其他服務 AWS API 操作的已驗證請求，而且這些日誌項目包含產生請求者的相關資訊。

例如：假設您發出請求，要求列出 Amazon EC2 執行個體或建立 AWS CodeDeploy 部署群組。發出請求的人員或服務相關詳細資訊都會包含在該請求的日誌項目中。此資訊可協助您判斷請求是由 AWS 帳戶根使用者、IAM 使用者、角色或其他 AWS 服務提出。

有關 CloudTrail 日誌項目中使用者身分資訊的更多詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 userIdentity 元素。

記錄使用者登入事件

CloudTrail 會將登入事件記錄到 AWS 管理主控台、本機開發工具，例如 AWS CLI 和 SDK、 AWS 討論論壇，以及 AWS Marketplace。 SDKs CloudTrail 會記錄 IAM 使用者、SAML 和 OIDC 聯合身分主體，以及 AWS STS 聯合身分使用者主體的成功和失敗登入嘗試。

若要檢視成功和失敗根使用者登入的 CloudTrail 事件範例，請參閱《AWS CloudTrail 使用者指南》中的根使用者的範例事件記錄。

作為安全最佳實務，當登入失敗是由不正確的使用者名稱造成時， AWS 不會記錄輸入的 IAM 使用者名稱文字。使用者名稱文字是由值 `HIDDEN_DUE_TO_SECURITY_REASONS` 遮蓋的。如需此範例，請參閱本主題後述的因使用者名稱不正確以致登入失敗的事件範例。因為這類失敗可能是使用者錯誤所造成，所以會隱蔽使用者名稱文字。記錄這些錯誤可能會公開潛在的敏感資訊。例如：

- 您不小心在使用者名稱方塊中輸入密碼。
- 您可以選擇其中一個登入頁面的連結 AWS 帳戶，然後輸入另一個登入頁面的帳號 AWS 帳戶。
- 您忘記了正在登入的帳戶，並且無意中輸入了您的個人電子郵件帳戶的帳戶名稱，銀行登入識別碼或其他私有 ID。

記錄暫時憑證的登入事件

當主體請求暫時憑證時，主體類型會決定 CloudTrail 記錄事件的方式。當主體擔任另一個帳戶中的角色時，情況會很複雜。有多個 API 呼叫會執行與角色跨帳戶操作相關的操作。首先，委託人呼叫 AWS STS API 來擷取暫時登入資料。該操作會記錄在呼叫帳戶和執行 AWS STS 操作的帳戶中。然後，主體會使用該角色，在擔任角色的帳戶中執行其他 API 呼叫。

您可以使用角色信任政策中的 `sts:SourceIdentity` 條件金鑰，請求使用者在擔任角色時指定身分。例如，您可以請求 IAM 使用者將自己的使用者名稱指定為其來源身分。這可以協助您判斷哪位使用者在 AWS中執行了特定動作。如需詳細資訊，請參閱`sts:SourceIdentity`。您亦可以使用 `sts:RoleSessionName`，請求使用者在擔任角色時指定工作階段名稱。這可協助您在檢閱 AWS CloudTrail 日誌時，區分不同委託人使用之角色的角色工作階段。

下表顯示 CloudTrail 如何為產生臨時憑證的每個 AWS STS APIs 記錄不同的使用者身分資訊。

主體類型	STS API	呼叫者帳戶 CloudTrail 日誌中的使用者身分	所擔任角色帳戶 CloudTrail 日誌中的使用者身分	角色後續 API 呼叫之 CloudTrail 日誌中的使用者身分
AWS 帳戶根使用者 登入資料	GetSessionToken	根使用者身分	角色擁有者帳戶與呼叫帳戶相同	根使用者身分
AWS 帳戶根使用者 登入資料	AssumeRoot	根使用者工作階段	帳號和主體 ID (若為使用者)	根使用者工作階段
IAM 使用者	GetSessionToken	IAM 使用者身分	角色擁有者帳戶與呼叫帳戶相同	IAM 使用者身分
IAM 使用者	GetFederationToken	IAM 使用者身分	角色擁有者帳戶與呼叫帳戶相同	IAM 使用者身分
IAM 使用者	AssumeRole	IAM 使用者身分	帳號和委託人 ID （如果是使用者）或服務 AWS 委託人	僅限角色身分 (非使用者)
外部驗證的使用者	AssumeRoleWithSAML	N/A	SAML 使用者身分	僅限角色身分 (非使用者)
外部驗證的使用者	AssumeRoleWithWebIdentity	N/A	OIDC/Web 使用者身分	僅限角色身分 (非使用者)

如果動作對資源沒有任何變異效果，CloudTrail 會將其視為唯讀動作。記錄唯讀事件時，CloudTrail 會修訂日誌中的 `responseElements` 資訊。當 CloudTrail 記錄非唯讀的事件時，日誌項目中會顯示完整的 `responseElements`。對於 AWS STS APIs `AssumeRole`、`AssumeRoleWithSAML`和 `AssumeRoleWithWebIdentity`，即使它們記錄為唯讀，CloudTrail 仍會包含完整，`responseElements`但這些 APIs的日誌`secretAccessKey`中除外。

下表顯示 CloudTrail 如何記錄`responseElements`和產生臨時憑證之每個 AWS STS APIs`readOnly`的資訊。

STS API	回應元素資訊	唯讀
AssumeRole	包含	true
AssumeRoleWithSAML	包含	true
AssumeRoleWithWebIdentity	包含	true
AssumeRoot	包含	false
GetFederationToken	包含	false
GetSessionToken	包含	false

CloudTrail 日誌中的 IAM API 事件範例

CloudTrail 日誌檔案包含使用 JSON 格式化的事件。一個 API 事件代表單一 API 請求，並包含主體、請求動作、任何參數以及動作的日期和時間等資訊。

CloudTrail 日誌檔案中的 IAM API 事件範例

以下範例顯示針對 IAM `GetUserPolicy` 動作發出請求的 CloudTrail 日誌項目。


```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
```

```
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.101",
    "userAgent": "aws-cli/1.16.96 Python/2.7.8 Linux/10 botocore/1.12.86",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePoliccyName"
    },
    "responseElements": null,
    "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
    "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    }
}
```


從這個事件資訊，您可以判斷提出請求是為了取得使用者 ReadOnlyAccess-JaneDoe-201407151307 的一個名為 JaneDoe 的使用者政策，如 requestParameters 元素中所指定。您也可以看到該請求由名為 JaneDoe 的 IAM 使用者於 2014 年 7 月 15 日下午 9:40 (UTC) 提出。在這種情況下，請求源自 AWS 管理主控台，如您可以從 userAgent元素所說。

CloudTrail 日誌中的範例 AWS STS API 事件

CloudTrail 日誌檔案包含使用 JSON 格式化的事件。一個 API 事件代表單一 API 請求，並包含主體、請求動作、任何參數以及動作的日期和時間等資訊。

CloudTrail 日誌檔案中的跨帳戶 AWS STS API 事件範例

帳戶 777788889999 John中名為 的 IAM 使用者呼叫 AWS STS AssumeRole 動作，以擔任帳戶 111122223333 EC2-dev中的角色。帳戶管理員會要求使用者在擔任角色時，將來源身分設定為等於其使用者名稱。使用者傳入 John 的來源身分值。


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/John",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "John"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
    },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
  },
```

```
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

第二個範例顯示所擔任角色帳戶 (111122223333) 相同請求的 CloudTrail 日誌項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
    },
    "assumedRoleUser": {
```

```
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
      },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```

CloudTrail 日誌檔案中 AWS STS 的角色鏈結 API 事件範例

以下範例顯示 John Doe 在帳戶 111111111111 中請求的 CloudTrail 日誌項目。John 以前用過他的 John 使用者擔任 JohnRole1 角色。對此請求而言，他使用該角色的憑證擔任 JohnRole2 角色。這就是所謂的角色鏈接。他在擔任 John1 角色時所設定的來源身分將保留在請求中以擔任 JohnRole2。如果 John 嘗試在擔任角色時設定不同的來源身分，則請求將會遭到拒絕。John 將兩個工作階段標籤傳遞給請求。並將這兩個標籤設為轉移。因為 John 在擔任 Department 時將其設為轉移，所以此請求繼承 JohnRole1 標籤也為轉移。如需有關來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。如需有關角色鏈結中轉移索引鍵的詳細資訊，請參閱 使用工作階段標籤鏈結角色。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
            },
                "sessionIssuer": {
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "John"
```

```
            },
            "sourceIdentity": "John"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
 Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "John",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
            "expiration": "Oct 2, 2019, 11:12:29 PM",
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
```

```
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
        "sourceIdentity": "John",
        "sessionTokenSize": 1652,
        "sessionTokenUtilization": 40
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```

CloudTrail 日誌檔案中的範例 AWS 服務 AWS STS API 事件

下列範例顯示服務使用 AWS 來自服務角色的許可呼叫另一個服務 API 所提出請求的 CloudTrail 日誌項目。它會顯示帳戶 777788889999 所提出請求的 CloudTrail 日誌項目。

```
{
  "eventVersion": "1.04",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
```

```
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "amzn-s3-demo-bucket"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```

CloudTrail 日誌檔案中的範例 SAML AWS STS API 事件

下列範例顯示針對 AWS STS AssumeRoleWithSAML 動作提出之請求的 CloudTrail 日誌項目。此請求包含 SAML 屬性 `CostCenter` 和 `Project`，這些屬性透過 SAML 聲明以工作階段標籤的形式傳遞。這些標籤設為轉移，以便能在角色鏈接藍本中繼續。該請求包含選用的 API 參數 `DurationSeconds`，在 CloudTrail 日誌中表示為 `durationSeconds`，且設定為 1800 秒。此請求亦包含 SAML 屬性 `sourceIdentity`，會在 SAML 聲明中傳遞。如果有人使用產生的角色工作階段憑證來擔任另一個角色，則此來源身分會持續存在。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
```

```
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
    "eventTime": "2023-08-28T18:30:58Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 34,
        "sessionTokenSize": 1428,
        "sessionTokenUtilization": 34,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
```

```
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
    },
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```


CloudTrail 日誌檔案中的範例 OIDC AWS STS API 事件

下列範例顯示針對 AWS STS AssumeRoleWithWebIdentity 動作提出之請求的 CloudTrail 日誌項目。此請求包含屬性 `CostCenter` 和 `Project`，這些屬性透過 OpenID Connect (OIDC) 身分提供者 (IdP) 權杖以工作階段標籤的形式傳遞。這些標籤設為轉移，以便能在角色鏈接中繼續。請求包含從身分提供者權杖的 `sourceIdentity` 屬性。如果有人使用產生的角色工作階段憑證來擔任另一個角色，則此來源身分會持續存在。

CloudTrail 日誌項目也包含具有 `identityProviderConnectionVerificationMethod` 屬性的 `additionalEventData` 欄位。此屬性表示 AWS 用來驗證與 OIDC 提供者連線的方法。該屬性值將為 `IAMTrustStore` 或 `Thumbprint`。`IAMTrustStore` 值表示 使用我們的信任根憑證授權單位

(CA) 程式庫 AWS ，成功驗證與 OIDC IdP 的連線。 CAs Thumbprint 值表示 AWS 使用 IdP 組態中設定的憑證指紋來驗證 OIDC IdP 伺服器憑證。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>:<id of application>:<id of user>",
    "userName": "<id of user>",
    "identityProvider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>"
  },
  "eventTime": "2024-07-09T15:41:37Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/2.13.29 Python/3.11.6 Windows/10 exe/AMD64 prompt/off command/
sts.assume-role-with-web-identity",
  "requestParameters": {
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "<assigned role session name>",
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "principalTags": {
      "CostCenter": "24680",
      "Project": "Pegasus"
    },
    "transitiveTagKeys": [
      "CostCenter",
      "Project"
    ]
  },
  "responseElements": {
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "sessionToken": "<encoded session token blob>",
      "expiration": "Jul 9, 2024, 4:41:37 PM"
    },
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "assumedRoleUser": {
```

```
      "assumedRoleId": "AROA123456789EXAMPLE:<assigned role session name>",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/<assigned
 role session name>"
    },
    "provider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>",
    "audience": "<id of application>",
    "sessionTokenSize": 1536,
    "sessionTokenUtilization": 37
  },
  "additionalEventData": {
    "identityProviderConnectionVerificationMethod": "IAMTrustStore"
  },
  "requestID": "aEXAMPLE-0b26-40df-8973-c7012EXAMPLE",
  "eventID": "aEXAMPLE-ee29-4ac0-a0ed-3f5c5EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "444455556666",
      "type": "AWS::IAM::Role",
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "444455556666",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_128_GCM_SHA256",
    "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
  }
}
```

CloudTrail 日誌檔案中使用全域端點的範例 AWS STS API 事件

對於對 AWS Security Token Service (AWS STS) 全域端點 (`https://sts.amazonaws.com`) 的請求， AWS STS 包含其他 AWS CloudTrail 日誌欄位： `endpointType`和 `awsServingRegion`。這些欄位會出現在 `additionalEventDataRequestDetails`元素下方，以記錄正在呼叫的服務 AWS 區域 和端點類型。`endpointType` 欄位的值可以是 `global` 或 `regional`，指示處理請求的全域端點的類型。如需 AWS STS 全域端點變更的詳細資訊，請參閱 AWS STS 區域和端點。

> **Note**
>
> AWS CloudTrail 向 AWS STS 全域端點發出的請求日誌將傳送至美國東部（維吉尼亞北部）區域。 AWS STS 區域端點所提供請求的 CloudTrail 日誌將繼續記錄在 CloudTrail 中的其各自區域。

下列範例顯示對源自歐洲（斯德哥爾摩https://sts.amazonaws.com) 區域 - eu-north-1 的全域端點 () 提出 AWS STS 請求的 CloudTrail 日誌項目。欄位endpointType值 global表示 AWS STS 請求是由歐洲（斯德哥爾摩）區域中的全域端點提供服務。


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:48Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
```

```
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:48 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "awsServingRegion": "eu-north-1",
            "endpointType": "global"
        }
    },
    "requestID": "EXAMPLE7-2497-457a-9586-f21feEXAMPLE",
    "eventID": "EXAMPLEc-3d26-4c3a-9c94-722a9EXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "sts-global.eu-north-1.amazonaws.com"
    }
}
```

為了進行比較，以下範例顯示對歐洲（斯德哥爾摩https://sts.us-west-2.amazonaws.com) 區域端點 - eu-north-1 的區域端點 () 發出的 AWS STS 請求的 CloudTrail 日誌項目。endpointType 欄位值 regional表示 AWS STS 請求是由歐洲（斯德哥爾摩）區域中的全域端點提供服務。


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:30Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "eu-north-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:30 PM"
```

```
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "endpointType": "regional",
            "awsServingRegion": "eu-north-1"
        }
    },
    "requestID": "EXAMPLEd-2116-4cd7-bd72-9f72fEXAMPLE",
    "eventID": "EXAMPLEd-219a-48ed-bc54-00e3cEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "sts.eu-north-1.amazonaws.com"
    }
}
```

CloudTrail 日誌中的範例登入事件

CloudTrail 日誌檔案包含使用 JSON 格式化的事件。登入事件代表單一登入請求，並包含登入主體、區域以及動作的日期和時間等資訊。

CloudTrail 日誌檔案中的範例登入成功事件

以下範例顯示成功登入事件的 CloudTrail 日誌項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/John",
    "accountId": "111122223333",
    "userName": "John"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```

下列範例顯示成功授權碼請求的 CloudTrail 日誌項目。

```
{
"eventVersion": "1.11",
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:thesjain-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/thesjain-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
```

```
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-17T22:50:14Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-17T22:51:32Z",
"eventSource": "signin.amazonaws.com",
"eventName": "AuthorizeOAuth2Access",
"awsRegion": "us-east-1",
"sourceIPAddress": "52.94.133.136",
"userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36",
"requestParameters": {
"scope": "openid",
"redirect_uri": "http://127.0.0.1:53037/oauth/callback",
"code_challenge_method": "SHA-256",
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "e2854c76-1cba-4360-9fd1-5037b591466b",
"eventID": "59e1720d-3deb-44ff-933d-6828be2a860a",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```

下列範例顯示成功建立 OAuth2 權杖請求的 CloudTrail 日誌項目。


```
{
"eventVersion": "1.11",
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:jacobjoj-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/jacobjoj-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-18T20:38:10Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-18T20:38:44Z",
"eventSource": "signin.amazonaws.com",
"eventName": "CreateOAuth2Token",
"awsRegion": "us-east-1",
"sourceIPAddress": "15.248.6.6",
"userAgent": "aws-cli/2.32.0 md/awscrt#0.28.4 ua/2.1 os/macos#24.6.0 md/arch#arm64
 lang/python#3.13.9 md/pyimpl#CPython m/b,AA,Z,E cfg/retry-mode#standard md/
installer#exe sid/35033f4ca1bd md/prompt#off md/command#login",
"requestParameters": {
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "94562943-c85b-4dc1-bf72-43b0fd42d6de",
"eventID": "0b338fac-6a10-4740-b34d-1bb6923e799e",
"readOnly": true,
"eventType": "AwsApiCall",
```

```
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```

如需有關 CloudTrail 日誌檔案中包含資訊的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 CloudTrail 事件參考。

CloudTrail 日誌檔案中的登入失敗事件範例

以下範例顯示失敗登入事件的 CloudTrail 日誌項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
```

```
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

根據此資訊，您可以判斷嘗試登入的人是名為 JaneDoe 的 IAM 使用者 ，如 userIdentity 元素所示。您還可以看到登入嘗試失敗，如 responseElements 元素所示。您可以看到 JaneDoe 於 UTC 時間 2014 年 7 月 8 日下午 5:35 嘗試登入 Amazon SNS 主控台。

因使用者名稱不正確以致登入失敗的事件範例

下列範例顯示因使用者輸入不正確的使用者名稱而導致登入事件失敗的 CloudTrail 日誌項目。 會向 AWS 遮罩userName文字HIDDEN_DUE_TO_SECURITY_REASONS，以協助防止暴露潛在的敏感資訊。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
```

```
}
```

IAM 角色信任政策行為

在 2022 年 9 月 21 日，對 IAM 角色信任政策行為 AWS 進行了變更，以在角色擔任自己時要求角色信任政策中明確允許 。舊版行為允許清單中的 IAM 角色針對 AssumeRole 事件的 explicitTrustGrant 提供了 additionalEventData 欄位。當舊版允許清單上的角色使用舊版行為擔任自己時，`explicitTrustGrant` 的值為 false。當舊版允許清單上的角色擔任自己，但角色信任政策行為已更新為明確允許角色擔任自己時，`explicitTrustGrant` 的值為 true。

只有極少數的 IAM 角色位於舊版行為的允許清單上，並且此欄位只有當角色擔任自己時，才會出現在相應的 CloudTrail 日誌中。在大多數情況下，IAM 角色不需要擔任本身。 AWS 建議更新您的程序、程式碼或組態，以移除此行為或更新您的角色信任政策，以明確允許此行為。如需詳細資訊，請參閱 Announcing an update to IAM role trust policy behavior。

在 中追蹤特殊權限任務 AWS CloudTrail

IAM 的 AWS Organizations 管理帳戶或委派管理員帳戶可以使用短期根存取權在成員帳戶上執行一些根使用者任務。短期特權工作階段會為您提供臨時憑證，以便您可以確定相關範圍，對組織中的成員帳戶採取特權動作。您可以使用下列步驟來識別管理帳戶或委派管理員在 `sts:AssumeRoot` 工作階段期間所採取的動作。

> **Note**
>
> `sts:AssumeRoot` 不支援全域端點。CloudTrail 會記錄為端點指定的區域中的 `ConsoleLogin` 事件。

若要在 CloudTrail 日誌中追蹤特權工作階段所執行的動作

1. 在 CloudTrail 日誌中尋找 `AssumeRoot` 事件。當您的管理帳戶或委派的 IAM 管理員從 `sts:AssumeRoot` 取得一組短期憑證時，就會產生此事件。

   在以下範例中，AssumeRoot 的 CloudTrail 事件會記錄在 `eventName` 欄位中。

   ```
   {
       "eventVersion": "1.08",
       "userIdentity": {
           "type": "AssumedRole",
           "principalId": "AIDACKCEVSQ6C2EXAMPLE:JohnRole1",
   ```

```
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/John",
                "accountId": "111111111111",
                "userName": "Admin2"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-10-25T20:45:28Z",
                "mfaAuthenticated": "false"
            },
            "assumedRoot": "true"
        }
    },
    "eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
            "expiration": "Oct 25, 2024, 9:07:11 PM"
        }
    }
}
```

如需存取 CloudTrail 日誌的步驟，請參閱 AWS CloudTrail User Guide 中的 Getting and viewing your CloudTrail log files。

2. 在 CloudTrail 事件日誌中，找到指定所執行動作的目標成員帳戶的 `targetPrincipal`，以及 `AssumeRoot` 工作階段獨有的 `accessKeyId`。

   在以下範例中，`targetPrincipal` 為 222222222222，`accessKeyId` 為 ASIAIOSFODNN7EXAMPLE。

```
"eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
            "expiration": "Oct 25, 2024, 9:07:11 PM"
    }
}
```

3. 在目標主體的 CloudTrail 日誌中，搜尋對應至 `AssumeRoot` 事件的 `accessKeyId` 值的存取金鑰 ID。使用 `eventName` 欄位值來判斷在 `AssumeRoot` 工作階段期間執行的特權任務。在單一工作階段中可能會執行多個特權任務。`AssumeRoot` 的工作階段持續時間上限為 900 秒 (15 分鐘) 。

   在以下範例中，管理帳戶或委派管理員刪除了一個 Amazon S3 儲存貯體的資源型政策。

```
{
    "eventVersion": "1.10",
    "userIdentity": {
        "type": "Root",
        "principalId": "222222222222",
        "arn": "arn:aws:iam::222222222222:root",
        "accountId": "222222222222",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "creationDate": "2024-10-25T20:52:11Z",
```

```
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-10-25T20:53:47Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "DeleteBucketPolicy",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "bucketName": "resource-policy-John",
        "Host": "resource-policy-John.s3.amazonaws.com",
        "policy": ""
    },
    "responseElements": null,
    "requestID": "1234567890abcdef0",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "readOnly": false,
    "resources": [
        {
            "accountId": "222222222222",
            "type": "AWS::S3::Bucket",
            "ARN": "arn:aws:s3:::resource-policy-John"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "222222222222",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "resource-policy-John.s3.amazonaws.com"
    }
}
```


的合規驗證 AWS Identity and Access Management

在多個合規計畫中，第三方稽核人員會評估 AWS Identity and Access Management (IAM) 的安全與 AWS 合規。這些計劃包括 SOC、PCI、FedRAMP、ISO 等等。

我們的新 AWS 註冊體驗並非針對受管制的工作負載而設計。如果您正在使用我們新的 AWS 註冊體驗，但想要將 AWS 用於受管制的工作負載，您可以註冊 AWS （進階） 或為您的 AWS 環境啟用進階功能。

若要了解 是否 AWS 服務 在特定合規計劃範圍內，請參閱AWS 服務 合規計劃範圍內然後選擇您感興趣的合規計劃。如需一般資訊，請參閱AWS 合規計劃。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱下載報告下載 AWS Artifact。

您使用 時的合規責任 AWS 服務 取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。如需使用 時合規責任的詳細資訊 AWS 服務，請參閱 AWS 安全文件。

中的彈性 AWS Identity and Access Management

AWS 全球基礎設施是以 AWS 區域和可用區域為基礎建置。 AWS 區域具有多個實體分隔和隔離的可用區域，這些可用區域與低延遲、高輸送量和高備援聯網連接。如需 AWS 區域和可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

AWS Identity and Access Management (IAM) 和 AWS Security Token Service (AWS STS) 是可全域使用的自我維持、以區域為基礎的服務。

IAM 至關重要 AWS 服務。在 中執行的每個操作 AWS 都必須經過 IAM 驗證和授權。IAM 會根據 IAM 中儲存的身分和政策檢查每個請求，以判斷是否允許或拒絕請求。IAM 設計有一個單獨的控制平面和資料平面，以便服務即使在意外故障期間也會進行驗證。授權中使用的 IAM 資源 (例如角色和政策) 會儲存在控制平面中。IAM 客戶可以使用諸如 `DeletePolicy` 和 `AttachRolePolicy` 等 IAM 操作來變更這些資源的組態。這些組態變更請求會移至控制平面。所有商業都有一個 IAM 控制平面 AWS 區域，位於美國東部 （維吉尼亞北部） 區域。然後，IAM 系統會將組態變更傳播至每個已啟用 AWS 區域 的 IAM 資料平面。IAM 資料平面基本上是 IAM 控制平面組態資料的唯讀複本。每個 AWS 區域都有一個完全獨立的 IAM 資料平面執行個體，其會針對來自相同區域的請求執行身分驗證和授權。在每個區域中，IAM 資料平面至少分佈在三個可用區域，並具有足夠的容量來容忍可用區域的損失，而不會造成任何客戶損失。IAM 控制平面和資料平面都是針對零計劃停機而構建，以客戶看不到的方式執行所有軟體更新和擴展操作。

在預設啟用的區域中，對 AWS STS 全域端點的請求會在請求產生的相同區域中自動提供。在選擇加入區域中，全域 AWS STS 端點的請求由單一 AWS 區域美國東部 （維吉尼亞北部） 提供。您可以使用區域 AWS STS 端點來降低延遲，或為您的應用程式提供額外的備援。如需詳細資訊，請參閱 在 AWS STS 中管理 AWS 區域。

某些事件可能會中斷 AWS 區域 透過網路在 之間的通訊。不過，即使您無法與全域 IAM 端點通訊， 仍然 AWS STS 可以驗證 IAM 主體，IAM 可以授權您的請求。中斷通訊之事件的特定詳細資訊將決定您

存取 AWS 服務的能力。在大多數情況下，您可以繼續使用 AWS 環境中的 IAM 登入資料。下列情況可能適用於中斷通訊的事件。

IAM 使用者的存取金鑰

使用長期 IAM 使用者存取金鑰，您可以在區域中無限期地進行驗證。當您使用 AWS Command Line Interface 和 APIs 時，您可以提供 AWS 存取金鑰，讓 在程式設計請求中 AWS 驗證您的身分。

> **Important**
>
> 作為最佳實務，我們建議您的使用者使用暫時性憑證而非長期存取金鑰進行登入。

臨時憑證

您可以使用 AWS STS 區域服務端點請求新的臨時登入資料至少 24 小時。下列 API 操作會產生暫時性憑證。

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAML
- GetFederationToken
- GetSessionToken

主體和許可

- 您可能無法在 IAM 中新增、修改或移除主體或許可。
- 您的憑證可能不會反映您最近在 IAM 中套用的許可變更。如需詳細資訊，請參閱我所做的變更不一定都會立刻生效。

AWS 管理主控台

- 您可能可以使用區域登入端點以 IAM 使用者身分登入 AWS 管理主控台 。區域登入端點具有下列 URL 格式。

  ```
  https://{Account ID}.signin.aws.amazon.com/console?region={Region}
  ```

  範例：https://111122223333.signin.aws.amazon.com/console?region=us-west-2
- 您可能無法完成 Universal 2nd Factor (U2F) 多重要素驗證 (MFA)。

IAM 彈性的最佳實務

AWS 已在 AWS 區域 和可用區域中建置彈性。當您在與環境互動的系統中觀察下列 IAM 最佳實務時，可以充分利用該彈性。

1. 使用 AWS STS 區域服務端點，而非預設的全域端點。
2. 檢閱環境的組態，尋找定期建立或修改 IAM 資源的重要資源，並準備使用現有 IAM 資源的備用解決方案。

中的基礎設施安全 AWS Identity and Access Management

作為受管服務， AWS Identity and Access Management 受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及如何 AWS 保護基礎設施的資訊，請參閱AWS 雲端安全。若要使用基礎設施安全的最佳實務來設計您的 AWS 環境，請參閱安全支柱 AWS Well-Architected Framework 中的基礎設施保護。

您可以使用 AWS 發佈的 API 呼叫，透過網路存取 IAM。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

您可以透過 IAM HTTPS API 直接向服務發出 HTTPS 請求，以程式設計方式來存取 IAM。查詢 API 會傳回安全憑證等敏感資訊，所以您必須對所有 API 請求使用 HTTPS。當您使用 HTTPS API 時，必須包含使用您的憑證來數位簽署請求的程式碼。

您可從任何網路位置呼叫這些 API 操作，而 IAM 確實可支援資源型存取政策，以供納入依來源 IP 地址為主的限制。您也可以使用 IAM 政策來控制從特定 Amazon Virtual Private Cloud (Amazon VPC) 的端點或特定 VPC的存取。實際上，這只會隔離網路中特定 VPC 對指定 IAM 資源 AWS 的網路存取。

中的組態和漏洞分析 AWS Identity and Access Management

AWS 處理基本安全任務，例如訪客作業系統 (OS) 和資料庫修補、防火牆組態和災難復原。這些程序已由適當的第三方進行檢閱並認證。如需詳細資訊，請參閱以下資源：

- 共同的責任模型
- Amazon Web Services：安全程序概觀 (白皮書)

以下資源也可用來解決 AWS Identity and Access Management (IAM) 中的組態和漏洞分析問題：

- 的合規驗證 AWS Identity and Access Management
- 中的安全最佳實務和使用案例 AWS Identity and Access Management

AWS 的 受管政策 AWS Identity and Access Management、IAM Access Analyzer 和帳戶存取管理員

AWS 受管政策是由 AWS AWS 受管政策建立和管理的獨立政策旨在為許多常用案例提供許可，以便您可以開始將許可指派給使用者、群組和角色。

請記住， AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為這些許可可供所有 AWS 客戶使用。我們建議您定義特定於使用案例的客戶管理政策，以便進一步減少許可。

您無法變更 AWS 受管政策中定義的許可。如果 AWS 更新受 AWS 管政策中定義的許可，則更新會影響政策連接的所有主體身分（使用者、群組和角色）。當新的 AWS 服務 啟動或新的 API 操作可用於現有服務時， AWS 最有可能更新 AWS 受管政策。

如需詳細資訊，請參閱 IAM 使用者指南中的 AWS 受管政策。

IAMReadOnlyAccess

若要允許 IAM 資源的唯讀存取，請使用 `IAMReadOnlyAccess` 受管政策。此政策授予取得和列出所有 IAM 資源的許可。它允許檢視使用者、群組、角色、政策、身分提供者和 MFA 裝置的詳細資訊與活動報告。它不包括建立或刪除資源或存取 IAM Access Analyzer 資源的能力。查看政策以了解此政策支援之服務和動作的完整清單。

IAMUserChangePassword

使用 `IAMUserChangePassword` 受管政策可允許 IAM 使用者變更自己的密碼。

您可以設定 IAM 帳戶設定與密碼政策，以允許 IAM 使用者變更其 IAM 帳戶密碼。如果您允許此動作，IAM 會將政策連接至每位使用者。

若要檢視 JSON 政策，請參閱《 AWS 受管政策參考指南》中的 IAMUserChangePassword。

IAMAccessAnalyzerFullAccess

使用 `IAMAccessAnalyzerFullAccess` AWS 受管政策，讓您的管理員存取 IAM Access Analyzer。

許可群組

此政策會根據提供的許可集分組到陳述式中。

- IAM Access Analyzer – 允許對 IAM Access Analyzer 中所有資源的完整管理許可。
- 建立服務連結角色 – 允許管理員建立服務連結角色，允許 IAM Access Analyzer 代表您分析其他服務中的資源。此許可允許建立僅供 IAM Access Analyzer 使用的服務連結角色。
- AWS Organizations – 允許管理員在 AWS Organizations中針對組織使用 IAM Access Analyzer。在中啟用 IAM Access Analyzer 的受信任存取後 AWS Organizations，管理帳戶的成員可以檢視整個組織的調查結果。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

IAMAccessAnalyzerReadOnlyAccess

使用 `IAMAccessAnalyzerReadOnlyAccess` AWS 受管政策允許唯讀存取 IAM Access Analyzer。

若要允許 的 IAM Access Analyzer 唯讀存取 AWS Organizations，請建立客戶受管政策，以允許從IAMAccessAnalyzerFullAccess AWS 受管政策描述和列出動作。

服務層級許可

此政策提供 IAM Access Analyzer 的唯讀存取權。此政策中不包含任何其他服務許可。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
```

```
        "access-analyzer:CheckAccessNotGranted",
        "access-analyzer:CheckNoNewAccess",
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRolePolicy

您無法將 AccessAnalyzerServiceRolePolicy 連接至 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 IAM Access Analyzer 代表您執行動作。如需詳細資訊，請參閱使用適用於 的服務連結角色 AWS Identity and Access Management 和 Access Analyzer。

許可群組

此政策允許存取 IAM Access Analyzer，分析來自多個 AWS 服務的資源中繼資料。

- Amazon DynamoDB：允許檢視 DynamoDB 串流和資料表的許可。
- Amazon Elastic Compute Cloud — 允許描述 IP 地址、快照和 VPC 的許可。
- Amazon Elastic Container Registry：允許描述映像儲存庫、擷取帳戶設定以及擷取登錄和儲存庫政策的許可。
- Amazon Elastic File System — 允許檢視 Amazon EFS 檔案系統的說明，並檢視 Amazon EFS 檔案系統資源層級政策的許可。
- AWS Identity and Access Management — 允許擷取有關指定角色的資訊，並列出具有指定路徑前置詞的 IAM 角色的許可。允許擷取有關使用者、IAM 群組、登入設定檔、存取金鑰和服務上次存取資料資訊的許可。
- AWS Key Management Service — 允許檢視有關 KMS 金鑰及其金鑰政策和授予詳細資訊的許可。
- AWS Lambda — 允許檢視 Lambda 別名、函數、層和別名相關資訊的許可。
- AWS Organizations – 允許 的許可， AWS Organizations 並允許在 AWS 組織內建立分析器做為信任區域。
- Amazon Relational Database Service — 允許檢視有關 Amazon RDS 資料庫快照和 Amazon RDS 資料庫叢集快照詳細資訊的許可。

- Amazon Simple Storage Service – 允許檢視有關 Amazon S3 存取點、儲存貯體、Amazon S3 目錄儲存貯體存取點及目錄儲存貯體的詳細資訊的許可。
- AWS Secrets Manager — 允許檢視有關附加至秘密的秘密和資源政策詳細資訊的許可。
- Amazon Simple Notification Service — 允許檢視有關主題詳細資訊的許可。
- Amazon Simple Queue Service — 允許檢視有關指定佇列詳細資訊的許可。

若要檢視 JSON 政策，請參閱《 AWS 受管政策參考指南》中的 AccessAnalyzerServiceRolePolicy。

IAMAuditRootUserCredentials

當您在 AWS Organizations 成員帳戶上執行特殊權限任務以稽核成員帳戶的根使用者憑證狀態時，請使用 IAMAuditRootUserCredentials AWS 受管政策來縮小許可範圍。您可以列出或取得個別根使用者憑證資訊，例如：

- 根使用者密碼是否存在
- 根使用者是否擁有存取金鑰以及上次使用時間
- 根使用者是否具有相關聯的簽署憑證
- 根使用者關聯的 MFA 裝置
- 合併的根使用者憑證狀態清單

您不得將 IAMAuditRootUserCredentials 連接到 IAM 實體。此政策會連接至 AssumeRoot，以對您組織中的成員帳戶執行特權任務。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices",
            "iam:GetAccountSummary",
            "iam:GetUser",
```

```
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyAuditingCredentialsOnNonRootUserResource",
         "Action": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices" ,
            "iam:GetUser",
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

許可群組

此政策會根據提供的許可集分組到陳述式中。

- DenyAllOtherActionsOnAnyResource：拒絕存取所有資源的憑證。
- DenyAuditingCredentialsOnNonRootUserResource：拒絕存取所有非根使用者資源的憑證。

IAMCreateRootUserPassword

當您在 AWS Organizations 成員帳戶上執行特殊權限任務時，請使用 `IAMCreateRootUserPassword` AWS 受管政策來縮小許可範圍，以允許在沒有根使用者登入資料的成員帳戶的密碼復原。

您不得將 `IAMCreateRootUserPassword` 連接到 IAM 實體。此政策會連接至 AssumeRoot，以對您組織中的成員帳戶執行特權任務。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。

JSON

```
{
```

```
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:CreateLoginProfile",
            "iam:GetLoginProfile"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyCreatingPasswordOnNonRootUserResource",
         "Action": [
            "iam:CreateLoginProfile",
            "iam:GetLoginProfile"
         ],
         "Effect": "Deny",
         "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

許可群組

此政策會根據提供的許可集分組到陳述式中。

- DenyAllOtherActionsOnAnyResource：拒絕存取以取得或建立所有資源的密碼。
- DenyCreatingPasswordOnNonRootUserResource：拒絕存取以取得或建立所有非根使用者資源的密碼。

IAMDeleteRootUserCredentials

當您在 AWS Organizations 成員帳戶上執行特殊權限任務時，請使用 `IAMDeleteRootUserCredentials` AWS 受管政策來縮小許可範圍，以移除根使用者憑證，包括密碼、存取金鑰、簽署憑證和停用 MFA。此特權動作需要其他許可，因此您可以檢視上次使用的憑證資訊、驗證成員帳戶根使用者的上次使用資訊，以及列出要刪除的所有根使用者憑證的許可。

您不得將 `IAMDeleteRootUserCredentials` 連接到 IAM 實體。此政策會連接至 AssumeRoot，以對您組織中的成員帳戶執行特權任務。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "DenyAllOtherActionsOnAnyResource",
   "Effect": "Deny",
   "NotAction": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
    "iam:GetLoginProfile",
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "Resource": "*"
  },
  {
   "Sid": "DenyDeletingRootUserCredentialsOnNonRootUserResource",
   "Effect": "Deny",
   "Action": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
    "iam:GetLoginProfile",
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "NotResource": "arn:aws:iam::*:root"
  }
 ]
```

```
}
```

許可群組

此政策會根據提供的許可集分組到陳述式中。

- DenyAllOtherActionsOnAnyResource：拒絕存取以取得或刪除所有資源的憑證。
- DenyDeletingRootUserCredentialsOnNonRootUserResource：拒絕存取以取得或刪除所有非根使用者資源的憑證。

S3UnlockBucketPolicy

當您在 AWS Organizations 成員帳戶上執行特殊權限任務時，請使用 S3UnlockBucketPolicy AWS 受管政策來縮小許可範圍，以移除設定錯誤的儲存貯體政策，拒絕所有主體存取 Amazon S3 儲存貯體。

您不得將 S3UnlockBucketPolicy 連接到 IAM 實體。此政策會連接至 AssumeRoot，以對您組織中的成員帳戶執行特權任務。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "s3:DeleteBucketPolicy",
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyManagingBucketPolicyForNonRootCallers",
         "Action": [
            "s3:DeleteBucketPolicy",
```

```
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
         ],
         "Effect": "Deny",
         "Resource": "*",
         "Condition" : {
            "ArnNotLike" : {
               "aws:PrincipalArn" : "arn:aws:iam::*:root"
            }
         }
      }
   ]
}
```

許可群組

此政策會根據提供的許可集分組到陳述式中。

- DenyAllOtherActionsOnAnyResource：拒絕存取所有資源的儲存貯體政策。
- DenyManagingBucketPolicyForNonRootCallers：拒絕存取所有非根使用者資源的儲存貯體政策。

SQSUnlockQueuePolicy

當您在 AWS Organizations 成員帳戶上執行特殊權限任務時，請使用 SQSUnlockQueuePolicy AWS 受管政策來縮小許可範圍，以刪除拒絕所有主體存取 Amazon SQS 佇列的 Amazon Simple Queue Service 資源型政策。

您不得將 SQSUnlockQueuePolicy 連接到 IAM 實體。此政策會連接至 AssumeRoot，以對您組織中的成員帳戶執行特權任務。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。

許可群組

此政策會根據提供的許可集分組到陳述式中。

- DenyAllOtherActionsOnAnyResource：拒絕存取所有資源的 Amazon SQS 動作。
- DenyGettingQueueAttributesOnNonOwnQueue：拒絕存取另一個帳戶所擁有佇列的 Amazon SQS 佇列屬性。
- DenyActionsForNonRootUser：拒絕存取所有非根使用者資源的 Amazon SQS 動作。

AccountAccessManagerServiceRolePolicy

您不得將 AccountAccessManagerServiceRolePolicy 連接到 IAM 實體。此政策會連接到服務連結角色，允許帳戶存取管理員代表您執行動作。如需詳細資訊，請參閱the section called “安全”。

服務層級許可

此政策允許帳戶存取管理員擷取 Organizations 帳戶的相關資訊，以維持存取管理的準確帳戶成員資格。帳戶存取管理員使用這些許可來偵測帳戶何時離開您的組織、驗證是否已啟用受信任的服務存取，以及在管理權利之前確認帳戶狀態。

- AWS Organizations – 允許列出組織中帳戶的許可、驗證是否已啟用受信任的服務存取，以及描述帳戶狀態。

若要檢視 JSON 政策，請參閱《 AWS 受管政策參考指南》中的 AccountAccessManagerServiceRolePolicy。

AWS 受管政策的 IAM、IAM Access Analyzer 和帳戶存取管理員更新

檢視自服務開始追蹤這些變更以來，IAM 和 AWS 受管政策更新的詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 IAM 和 IAM Access Analyzer 文件歷程記錄頁面上的 RSS 摘要。

變更	描述	日期
AccountAccessManagerServiceRolePolicy – 新增受管政策	帳戶存取管理員新增了服務連結角色政策，允許服務擷取 Organizations 帳戶的相關資訊。	2026 年 7 月 31 日
IAMUserChangePassword – 新增了許可	IAM 已將許可新增至 IAMUserChangePassword ，以允許路徑中指定的使用者。	2025 年 5 月 28 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 向 AccessAnalyzerServ	2025 年 5 月 12 日

變更	描述	日期
	iceRolePolicy 的服務層級許可新增了 iam:GetAccountAuthorizationDetails 。	
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 向 AccessAnalyzerServiceRolePolicy 的服務層級許可新增了對 Amazon S3 目錄儲存貯體存取點的支援。	2025 年 3 月 31 日
IAMDeleteRootUserCredentials – 移除了許可	IAM 從受管政策中移除了 iam:DeleteVirtualMFADevice 許可。	2025 年 1 月 7 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 新增了許可支援，可將 Amazon ECR 帳戶設定和登錄政策的相關資訊擷取至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2024 年 12 月 10 日
IAMAuditRootUserCredentials：已新增受管政策	IAM 新增了受管政策，以集中管理成員帳戶的根存取權，限制您可以對 AWS Organizations 成員帳戶執行的特權任務。	2024 年 11 月 14 日
IAMCreateRootUserPassword：已新增受管政策	IAM 新增了受管政策，以集中管理成員帳戶的根存取權，限制您可以對 AWS Organizations 成員帳戶執行的特權任務。	2024 年 11 月 14 日

變更	描述	日期
IAMDeleteRootUserCredentials：已新增受管政策	IAM 新增了受管政策，以集中管理成員帳戶的根存取權，限制您可以對 AWS Organizations 成員帳戶執行的特權任務。	2024 年 11 月 14 日
S3UnlockBucketPolicy：已新增受管政策	IAM 新增了受管政策，以集中管理成員帳戶的根存取權，限制您可以對 AWS Organizations 成員帳戶執行的特權任務。	2024 年 11 月 14 日
SQSUnlockQueuePolicy：已新增受管政策	IAM 新增了受管政策，以集中管理成員帳戶的根存取權，限制您可以對 AWS Organizations 成員帳戶執行的特權任務。	2024 年 11 月 14 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 新增了許可支援，可將 IAM 使用者和角色標籤的相關資訊擷取至 `AccessAnalyzerServiceRolePolicy` 的服務層級許可。	2024 年 10 月 29 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 新增了許可支援，可將 IAM 使用者和角色政策的相關資訊擷取至 `AccessAnalyzerServiceRolePolicy` 的服務層級許可。	2024 年 5 月 30 日

變更	描述	日期
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 新增了許可支援，可將 Amazon EC2 快照的封鎖公開存取的目前狀態擷取至 `AccessAnalyzerServiceRolePolicy` 的服務層級許可。	2024 年 1 月 23 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 已將對 DynamoDB 串流和資料表的支援新增至 `AccessAnalyzerServiceRolePolicy` 的服務層級許可。	2024 年 1 月 11 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 已將對 Amazon S3 目錄儲存貯體的支援新增至 `AccessAnalyzerServiceRolePolicy` 的服務層級許可。	2023 年 12 月 1 日
IAMAccessAnalyzerReadOnlyAccess - 已新增許可	IAM Access Analyzer 新增了許可，可用來檢查政策更新是否授予其他存取權。 IAM Access Analyzer 需要此許可，才能對您的政策執行政策檢查。	2023 年 11 月 26 日

變更	描述	日期
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 為 AccessAnalyzerServiceRolePolicy 服務層級許可新增了 IAM 動作，以支援下列動作： • 列出政策的實體 • 產生上次存取的服務詳細資訊 • 列出存取金鑰資訊	2023 年 11 月 26 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 在 AccessAnalyzerServiceRolePolicy 服務層級許可中已新增對下列資源類型的支援： • Amazon EBS 磁碟區快照 • Amazon ECR 儲存庫 • Amazon EFS 檔案系統 • Amazon RDS 資料庫快照 • Amazon RDS 資料庫叢集快照 • Amazon SNS 主題	2022 年 10 月 25 日
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 對 AccessAnalyzerServiceRolePolicy 的服務層級許可新增了 lambda:GetFunctionUrlConfig 動作。	2022 年 4 月 6 日

變更	描述	日期
AccessAnalyzerServiceRolePolicy - 已新增許可	IAM Access Analyzer 新增了新的 Amazon S3 動作，以分析與多區域存取點相關聯的中繼資料。	2021 年 9 月 2 日
IAMAccessAnalyzerReadOnlyAccess - 已新增許可	IAM Access Analyzer 新增了新的動作，以授予 `ValidatePolicy` 許可，以允許您使用政策檢查進行驗證。 IAM Access Analyzer 需要此許可，才能對您的政策執行政策檢查。	2021 年 3 月 16 日
IAM Access Analyzer 開始追蹤變更	IAM Access Analyzer 開始追蹤其 AWS 受管政策的變更。	2021 年 3 月 1 日

IAM 外部的安全功能

您可以使用 IAM 來控制對使用 AWS 管理主控台、 AWS 命令列工具或使用 AWS SDKs 的服務 API 操作所執行之任務的存取。部分 AWS 產品還有其他方法來保護其資源。以下列表提供了一些範例，但仍有未能列出者。

Amazon EC2

在 Amazon Elastic Compute Cloud 中，需要使用金鑰對 (對於 Linux 執行個體) 或使用使用者名稱和密碼 (對於 Windows 執行個體) 來登入執行個體。

如需詳細資訊，請參閱下列 文件：

- Amazon EC2 使用者指南中的 Amazon EC2 Linux 執行個體入門
- Amazon EC2 使用者指南中的 Amazon EC2 Windows 執行個體入門

Amazon RDS

在 Amazon Relational Database Service 中，需要使用與資料庫綁定的使用者名稱和密碼來登入資料庫引擎。

如需詳細資訊，請參閱 Amazon RSD 使用者指南中的 Amazon RDS 入門。

Amazon EC2 和 Amazon RDS

在 Amazon EC2 與 Amazon RDS 中，需要使用安全群組來控制發送到執行個體或資料庫的流量。

如需詳細資訊，請參閱下列 文件：

- Amazon EC2 使用者指南中的適用於 Linux 執行個體的 Amazon EC2 安全群組
- Amazon EC2 使用者指南中的適用於 Windows 執行個體的 Amazon EC2 安全群組
- Amazon RDS 使用者指南中的 Amazon RDS 安全群組

WorkSpaces

在 Amazon WorkSpaces 中，使用者以使用者名稱和密碼登入桌面。

如需詳細資訊，請參閱 Amazon WorkSpaces 管理指南中的 WorkSpaces 入門。

這些存取控制方法不屬於 IAM。經由 IAM 可以控制管理這些 AWS 產品的方式，包括建立或終止 Amazon EC2 執行個體、設定新 WorkSpaces 桌面等。也就是說，IAM 可幫助您控制對 Amazon Web Services 提出請求來執行的任務，並且可幫助您控制對 AWS 管理主控台的存取。不過，IAM 無法協助您管理登入作業系統 (Amazon EC2)、資料庫 (Amazon RDS) 或桌面 (Amazon WorkSpaces) 這類任務的安全性。

當您使用特定 AWS 產品時，請務必閱讀 文件，以了解屬於該產品之所有資源的安全選項。

使用 AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer 提供下列功能：

- IAM Access Analyzer 外部存取權分析器可協助您識別組織與帳戶中與外部實體共用的資源。
- IAM Access Analyzer 內部存取分析器可協助在組織與帳戶中識別所選資源的內部存取權。
- IAM Access Analyzer 未使用的存取權分析器可協助您在組織與帳戶中識別未使用的存取權。
- IAM Access Analyzer 會根據政策文法和最佳實務來驗證 IAM 政策。 AWS
- IAM Access Analyzer 自訂政策檢查可協助根據您指定的安全標準驗證 IAM 政策。
- IAM Access Analyzer 會根據日誌中的存取活動產生 IAM 政策。 AWS CloudTrail

識別與外部實體共用的資源

IAM Access Analyzer 可協助您識別與外部實體共用的組織和帳戶中資源，例如 Amazon S3 儲存貯體或 IAM 角色。這有助於您識別非預期存取資源和資料的情況，避免產生安全性風險。IAM Access Analyzer 會使用邏輯式推理來分析 AWS 環境中以資源為基礎的政策，藉此識別與外部主體共用的資源。針對帳戶外部共用資源的每一個執行個體，IAM Access Analyzer 都會產生一份問題清單。調查結果包括存取權及其被授與存取的外部主體的資訊。您可以檢閱問題清單，判斷此為有意為之且安全的存取，還是非預期且有安全風險的存取。除了協助您識別與外部實體共用的資源外，IAM Access Analyzer 也可讓您使用 IAM Access Analyzer 的問題清單，在部署資源使用權限之前預覽您的政策對您資源的公有和跨帳戶存取權的影響。系統會在視覺化摘要儀表板中整理歸納調查結果。儀表板會反白公有與跨帳戶存取權調查結果之間的分隔線，並提供依資源類型分類的調查結果明細。若要深入了解儀表板，請參閱：檢視 IAM Access Analyzer 調查結果儀表板。

> **Note**
>
> 外部實體可以是另一個 AWS 帳戶、根使用者、IAM 使用者或角色、聯合身分使用者、匿名使用者或可用來建立篩選條件的另一個實體。如需詳細資訊，請參閱 AWS JSON 政策元素：主體。

如需入門的說明，請參閱 建立 IAM Access Analyzer 外部存取權分析器。建立分析器之後，您可以 管理 IAM Access Analyzer 外部存取分析器。

啟用 IAM Access Analyzer 後，您可以為整個組織或帳戶建立分析器。您選擇的組織或帳戶也稱為分析器的信任區域。分析器會監控您信任區域內所有支援的資源。在信任區域內主體對資源的任何存取權都會被視為受信任。啟用後，IAM Access Analyzer 會分析在您信任區域中套用至所有支援之資源的政策。在第一次分析後，IAM Access Analyzer 會定期分析這些政策。如果新增政策或變更現有政策，則 IAM Access Analyzer 會在大約 30 分鐘內分析新的或更新後的政策。

分析政策時，如果 IAM Access Analyzer 識別到會將存取權授與不在信任區域內的外部主體的政策，就會產生問題清單。每個問題清單都包含資源、具有該資源存取權的外部實體，以及授與之許可的詳細資訊，以便您可以採取適當的動作。您可以檢視問題清單中包含的詳細資訊，以判斷資源存取權是有意為之的，還是您應解決的潛在風險。當您將政策新增至資源或更新現有政策時，IAM Access Analyzer 會分析該政策。IAM Access Analyzer 也會定期分析所有以資源為基礎的政策。

在符合某些條件的極少數情況下，IAM Access Analyzer 不會接收可能會導致生成的調查結果延遲的新增或更新政策的通知。若您建立或刪除與 Amazon S3 儲存貯體相關聯的多區域存取點，或更新多區域存取點的政策，IAM Access Analyzer 最多可能需要 6 小時才能產生或解決問題清單。此外，如果 AWS CloudTrail 日誌交付或資源控制政策 (RCP) 限制變更發生交付問題，政策變更不會觸發重新掃描調查結果中報告的資源。發生這種情況時，IAM Access Analyzer 會在下次定期掃描 (24 小時內) 期間分析該新增或更新的政策。若要確認對政策所做的變更是否可以解決調查結果中報告的存取問題，可以重新掃描調查結果中報告的資源，方法為使用調查結果詳細資訊頁面中的重新掃描連結，或使用 IAM Analyzer API 的 `StartResourceScan` 操作。如需詳細資訊，請參閱 解決 IAM Access Analyzer 調查結果。

> **Important**
>
> 對於外部存取，IAM Access Analyzer 只會分析套用至相同 AWS 區域 (已在其中啟用) 中資源的政策。若要監控 AWS 環境中的所有資源，您必須建立外部存取分析器，以便在您使用支援 AWS 資源的每個區域中啟用 IAM Access Analyzer。
>
> 針對未使用的存取權，分析器的調查結果不會因區域而變更。不需要在每個擁有資源的區域中都建立未使用的存取分析器。

IAM Access Analyzer 會針對外部存取分析下列資源類型：

- Amazon Simple Storage Service 儲存貯體
- Amazon Simple Storage Service 目錄儲存貯體
- AWS Identity and Access Management 角色
- AWS Key Management Service 金鑰

- AWS Lambda 函數和圖層
- Amazon Simple Queue Service 佇列
- AWS Secrets Manager 秘密
- Amazon Simple Notification Service 主題
- Amazon Elastic Block Store 磁碟區快照
- Amazon Relational Service 資料庫快照
- Amazon Relational Database Service 資料庫叢集快照
- Amazon Elastic Container Registry 儲存庫
- Amazon Elastic File System 檔案系統
- Amazon DynamoDB Streams
- Amazon DynamoDB 資料表

識別對業務關鍵型資源的內部存取

對於所選的業務關鍵型資源，IAM Access Analyzer 可協助您識別組織或帳戶中的哪些主體可以存取這些資源。此分析透過確保您指定的資源僅可由組織內的預期主體存取，來支援實作最低權限原則。

如需入門的說明，請參閱 建立 IAM Access Analyzer 內部存取分析器。建立分析器之後，您可以 管理 IAM Access Analyzer 內部存取分析器。

內部存取權分析可協助您：

- 決定帳戶或組織內的哪些 IAM 使用者或角色可以存取指定的資源
- 了解 AWS 環境內主體與資源之間的存取路徑
- 確認存取控制是否如預期運作
- 檢閱調查結果，判斷此為有意為之且安全的存取，還是非預期且構成安全風險的存取
- 識別並修復組織內的非預期存取

IAM Access Analyzer 會針對內部存取分析下列資源類型：

- Amazon Simple Storage Service 儲存貯體
- Amazon Simple Storage Service 目錄儲存貯體
- Amazon Relational Service 資料庫快照
- Amazon Relational Database Service 資料庫叢集快照

- Amazon DynamoDB Streams
- Amazon DynamoDB 資料表

識別授予 IAM 使用者和角色的未使用存取權

IAM Access Analyzer 可協助您識別和檢閱 AWS 組織和帳戶中未使用的存取權。IAM Access Analyzer 會持續監控 AWS 組織和帳戶中的所有 IAM 角色和使用者，並針對未使用的存取權產生調查結果。調查結果會反白未使用的角色、IAM 使用者未使用的存取金鑰，以及 IAM 使用者未使用的密碼。調查結果可讓您了解作用中 IAM 角色和使用者未使用的服務和動作。

如需建立未使用的存取分析器和管理其調查結果的說明，請參閱 管理 IAM Access Analyzer 未使用的存取權分析器。

IAM Access Analyzer 會檢閱 AWS 組織和帳戶中所有角色的上次存取資訊，以協助您識別未使用的存取。IAM 動作上次存取的資訊可協助識別 AWS 帳戶中的角色未使用的動作。如需詳細資訊，請參閱AWS 使用上次存取的資訊在 中精簡許可。

外部、內部及未使用的存取分析器調查結果，會在視覺化摘要儀表板中整理歸納。儀表板會反白您的 AWS 資源和調查結果最多 AWS 帳戶 的資源，並依類型提供調查結果明細。如需儀表板中的詳細資訊，請參閱檢視 IAM Access Analyzer 調查結果儀表板。

根據 AWS 最佳實務驗證政策

您可以使用 IAM Access Analyzer 政策驗證功能提供的基本政策檢查，根據 IAM 政策文法和 AWS 最佳實務來驗證您的政策。您可以使用 IAM 主控台中的 AWS CLI、 AWS API 或 JSON 政策編輯器來建立或編輯政策。您可以檢視政策驗證檢查問題清單，包含安全警告、錯誤、一般警告和政策的建議。這些調查結果提供可行的建議，協助您撰寫功能正常且符合 AWS 最佳實務的政策。若要進一步了解如何使用政策驗證來驗證政策，請參閱：使用 IAM Access Analyzer 來驗證政策。如需 IAM Access Analyzer 執行的政策檢查清單，請參閱 驗證政策檢查。

根據您指定的安全標準驗證政策

您可以使用 IAM Access Analyzer 自訂政策檢查，根據您指定的安全標準來驗證 IAM 政策。您可以使用 IAM 主控台中的 AWS CLI、 AWS API 或 JSON 政策編輯器來建立或編輯政策。您可以透過主控台檢查與現有版本相比，更新版政策是否授予新的存取權。透過 AWS CLI 和 AWS API，您也可以檢查政策不允許您認為重要的特定 IAM 動作。這些檢查會反白授予新存取權的政策陳述式。您可以更新政策陳述式並重新執行檢查，直到政策符合安全標準為止。若要進一步了解如何使用自訂政策檢查來驗證政策，請參閱：使用 IAM Access Analyzer 自訂政策檢查來驗證政策。

產生政策

IAM Access Analyzer 會分析您的 AWS CloudTrail 日誌，以識別 IAM 實體（使用者或角色）在指定日期範圍內使用的動作和服務。然後它會產生以該存取活動為基礎的 IAM 政策。您可以使用產生的政策，將其連接至 IAM 使用者或角色，以進一步調整該實體的許可。若要進一步了解如何使用 IAM Access Analyzer 來產生政策，請參閱 產生 IAM Access Analyzer 政策。

IAM Access Analyzer 定價

IAM Access Analyzer 會根據每月每個分析器所分析的 IAM 角色和使用者數量，收取未使用的存取權分析費用。

- 您需為建立的每個未使用的存取權分析器支付費用。
- 若跨多個區域建立未使用的存取權分析器，您將須為每個分析器支付費用。
- 系統不會針對未使用的存取活動分析服務連結角色，也不會將這些角色包含在已分析的 IAM 角色總數中。

IAM Access Analyzer 會根據每個內部存取分析器每個月監控的資源數量，收取內部存取分析費用。

IAM Access Analyzer 會根據向 IAM Access Analyzer 發出的檢查新存取權 API 請求數量，來收取自訂政策檢查費用。

如需 IAM Access Analyzer 價格和費用的完整清單，請參閱 IAM Access Analyzer 定價。

若要查看您的帳單，請前往 AWS 帳單與成本管理 主控台中的帳單與成本管理儀表板。您的帳單內含用量報告的連結，可提供帳單的詳細資訊。若要進一步了解 AWS 帳戶 帳單，請參閱 AWS Billing 使用者指南

如果您對 AWS 帳單、帳戶和事件有任何疑問，請聯絡 支援 。

IAM Access Analyzer 調查結果

IAM Access Analyzer 會在您的 AWS 帳戶 或組織中產生外部存取、內部存取和未使用存取的問題清單。

針對外部存取權，若執行個體採用的資源類型政策會將信任區域內的資源存取權授予信任區域以外的主體，則 IAM Access Analyzer 會為每個執行個體產生調查結果。當您建立外部存取分析器時，您可以選擇組織或 AWS 帳戶 進行分析。該分析器會將所選組織或帳戶中的任何主體認定為可信任的。由於相

同組織或帳戶中的主體是可信任的，分析器的信任區域就會包含該組織或帳戶內的資源和主體。在信任區域內共享的任何項目都是安全的，因此 IAM Access Analyzer 不會產生問題清單。例如，如果您選擇組織做為分析器的信任區域，則該組織中的所有資源和主體都會在信任區域內。若您將其中一個組織成員帳戶中的 Amazon S3 儲存貯體許可授予另一個組織成員帳戶中的主體，IAM Access Analyzer 並不會產生調查結果。但若將許可授與非組織成員帳戶中的主體，IAM Access Analyzer 就會產生問題清單。

對於內部存取，IAM Access Analyzer 會在您組織內的 IAM 角色或使用者與指定資源之間存在可能的存取路徑時，產生調查結果。與外部存取分析類似，您選擇的範圍 (組織或帳戶) 會決定哪些存取是內部存取。如果您選擇組織作為分析範圍，IAM Access Analyzer 將會針對組織內的主體與資源之間的存取路徑產生調查結果。如果您選擇帳戶作為分析範圍，則會針對該特定帳戶內的存取路徑產生調查結果。IAM Access Analyzer 使用自動推理來評估所有 IAM 政策，以監控誰有權存取您的資源。

將外部和內部存取調查結果與相同信任區域相結合，可以對特定資源的所有可能存取進行全面分析，包括來自於定義的信任界限內外的存取。

對於未使用的存取，IAM Access Analyzer 會針對 AWS 組織和帳戶中授予的未使用存取產生調查結果。當您建立未使用的存取分析器時，IAM Access Analyzer 會持續監控 AWS 組織和帳戶中的所有 IAM 角色和使用者，並產生未使用的存取問題清單。IAM Access Analyzer 會針對未使用的存取權產生下列類型的調查結果：

- 未使用的角色：在指定使用期間內沒有存取活動的角色。
- 未使用的 IAM 使用者存取金鑰和密碼 – 屬於 IAM 使用者的登入資料，該登入資料尚未用於在指定的使用時段 AWS 帳戶 中存取您的 。
- 未使用的許可：指定使用期間內角色未使用的服務層級和動作層級許可。IAM Access Analyzer 會使用連接至角色的身分型政策，來判斷這些角色可存取的服務和動作。IAM Access Analyzer 支援檢閱所有服務層級許可的未使用許可。如需未使用的存取權調查結果支援的動作層級許可完整清單，請參閱：IAM 動作最近存取的資訊服務和動作。

> **Note**
>
> IAM Access Analyzer 免費提供外部存取調查結果。對於未使用的存取調查結果，會根據每個分析器每月所分析的 IAM 角色和使用者數量收取費用。根據每個分析器每月監控 AWS 的資源數量，內部存取調查結果也會產生費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

主題

- 了解 IAM Access Analyzer 調查結果的運作方式
- 入門 AWS Identity and Access Management Access Analyzer
- 檢視 IAM Access Analyzer 調查結果儀表板
- 檢閱 IAM Access Analyzer 調查結果
- 篩選 IAM Access Analyzer 調查結果
- 封存 IAM Access Analyzer 調查結果
- 解決 IAM Access Analyzer 調查結果
- IAM Access Analyzer 錯誤調查結果
- IAM Access Analyzer 外部和內部存取支援的資源類型
- IAM Access Analyzer 的委派管理員
- 封存規則
- AWS Identity and Access Management Access Analyzer 使用 Amazon EventBridge 進行監控
- 將 IAM Access Analyzer 與 整合 AWS Security Hub CSPM
- 使用 記錄 IAM Access Analyzer API 呼叫 AWS CloudTrail
- IAM Access Analyzer 篩選鍵
- 使用 的服務連結角色 AWS Identity and Access Management Access Analyzer

了解 IAM Access Analyzer 調查結果的運作方式

本主題說明 IAM Access Analyzer 中使用的概念和術語，協助您熟悉 IAM Access Analyzer 如何監控對 AWS 資源的存取。

外部存取權調查結果

若某資源在您的信任區域之外共用，其每個執行個體都會產生一份外部存取權調查結果。每次修改資源型政策後，IAM Access Analyzer 都會分析該政策。若更新後政策所共用的資源已被問題清單辨識，但具備不同許可或條件，則該資源共用的執行個體會產生新的問題清單。對影響資源控制政策 (RCP) 限制的資源控制政策的變更，也會產生新的調查結果。IAM Access Analyzer 還會評估由宣告式政策建立的存取控制組態。若第一份調查結果的存取權已移除，該調查結果的狀態會更新為已解決。

所有調查結果的狀態會保持為作用中，直到您將其封存，或者移除產生該調查結果的存取權。移除存取權時，調查結果狀態會更新為已解決。

> **Note**
>
> 修改政策後，IAM Access Analyzer 最多可能需要 30 分鐘才能分析資源並更新外部存取權調查結果。資源控制政策 (RCP) 的變更不會觸發調查結果中所回報資源的重新掃描。IAM Access Analyzer 會在下次定期掃描 (24 小時內) 期間分析該新增或更新的政策。

IAM Access Analyzer 產生外部存取調查結果的運作方式

AWS Identity and Access Management Access Analyzer 使用名為 Zelkova 的技術來分析 IAM 政策，並識別對 資源的外部存取。

Zelkova 會將 IAM 政策轉換為等效的邏輯陳述式，並針對它們執行一套通用和專用的邏輯求解器 (可滿足性模數理論)。IAM Access Analyzer 會將 Zelkova 重複套用至政策，使用越來越具體的查詢來表徵政策根據其內容允許的訪問權類型。如需可滿足性模數理論的詳細資訊，請參閱可滿足性模數理論。

針對外部存取權分析器，IAM Access Analyzer 不會檢查存取日誌，來判斷外部實體是否實際存取過您信任區域內的資源。相反，當資源型政策允許存取某資源時，不論外部實體是否存取了該資源，都會產生調查結果。

此外，IAM Access Analyzer 在作出判斷時也不考慮任何外部帳戶的狀態。如果它指出帳戶 111122223333 可以存取您的 Amazon S3 儲存貯體，表示它對該帳戶中的使用者、角色、服務控制政策 (SCP) 或其他相關組態的資訊一無所知。這是考量到客戶的隱私權，因為 IAM Access Analyzer 不知道誰擁有另一個帳戶。這也是為了安全起見，即使目前沒有可以使用該存取的作用中主體，了解潛在的外部存取至關重要。

IAM Access Analyzer 僅考慮外部使用者無法直接影響或對授權具有影響力的某些 IAM 條件索引鍵。如需條件金鑰 IAM Access Analyzer 考慮的範例，請參閱 IAM Access Analyzer 篩選金鑰。

IAM Access Analyzer 目前不會報告 AWS 服務 委託人或內部服務帳戶的調查結果。在其無法完全判斷政策陳述式是否會將存取權授予外部實體的極少數情況下，則會在宣告誤報調查結果時發生錯誤。這是因為 IAM Access Analyzer 旨在提供在您帳戶中共享的全方位資源檢視，並將漏報降到最低。

內部存取調查結果

若要使用內部存取分析，必須先選取要監控的特定資源來設定分析器。設定完成後，當組織或帳戶內的主體 (IAM 使用者或角色) 可以存取您選取的資源時，就會產生內部存取調查結果。當分析器下次掃描指定的資源並識別出可存取這些資源的主體時，就會產生新的調查結果。若更新後的政策允許已在調查結果中識別出的主體，但該主體具備不同的許可或條件，則會針對該主體和資源的執行個體產生新

的調查結果。此更新後的政策可以是資源型政策、身分型政策、服務控制政策 (SCP) 或資源控制政策 (RCP)。IAM Access Analyzer 還會評估由宣告式政策建立的存取控制組態。

> **Note**
>
> 內部存取調查結果只能透過 ListFindingsV2 API 動作來取得。

IAM Access Analyzer 如何產生內部存取調查結果

若要分析內部存取，即使您已經建立分析器來產生外部存取調查結果或未使用的存取調查結果，還是須建立單獨的分析器來取得資源的內部存取調查結果。

建立內部存取分析器之後，IAM Access Analyzer 會評估指定帳戶或組織內的所有資源型政策、身分型政策、服務控制政策 (SCP)、資源控制政策 RCP) 和許可界限。

透過建立專用於內部存取所選資源的分析器，您可以識別：

- 組織或帳戶中的主體何時可以存取您選取的資源
- 根據所有適用政策的交集，授予主體的所有有效許可
- 複雜存取路徑，主體在其中根據身分政策和資源政策組合取得存取權

> **Note**
>
> IAM Access Analyzer 無法為包含超過 7 萬個主體 (IAM 使用者和角色合計) 的組織產生內部存取調查結果。

未使用的存取權調查結果

系統會根據建立分析器時指定的天數，針對選定帳戶或組織內的 IAM 實體 (主體) 產生未使用的存取調查結果。如果符合下列其中一個條件，分析器下次掃描實體時就會產生新的調查結果：

- 角色在指定天數內處於非作用狀態。
- 未使用的許可、未使用的使用者密碼或未使用的使用者存取金鑰超過指定天數。

> **Note**
>
> 未使用的存取權調查結果只能透過 ListFindingsV2 API 動作來使用。

IAM Access Analyzer 如何產生未使用的存取權調查結果

若要分析未使用的存取，即使您已經建立分析器來為資源產生外部或內部存取調查結果，還是須建立單獨的分析器來取得角色的未使用存取調查結果。

建立未使用的存取權分析器之後，IAM Access Analyzer 會檢閱存取權活動，以識別未使用的存取權。IAM Access Analyzer 會檢查 AWS 組織和帳戶中所有 IAM 使用者、IAM 角色的上次存取資訊，包括服務角色、使用者存取金鑰和使用者密碼。這可協助您識別未使用的存取權。

> **Note**
>
> 服務連結角色是一種特殊類型的服務角色，連結至 AWS 服務 且由服務擁有。未使用的存取分析器不會分析服務連結角色。

對於有效 IAM 角色和使用者，IAM Access Analyzer 會使用 IAM 服務和動作的上次存取資訊來識別未使用的許可。這可讓您在 AWS 組織和帳戶層級擴展審核程序。您也可以使用動作上次存取資訊，對個別角色進行更深入的調查。這可提供更精細的洞察，了解哪些特定的許可未被使用。

透過建立專用於未使用存取的分析器，您可以全面檢閱和識別整個 AWS 環境中未使用的存取，補充現有外部存取分析器所產生的問題清單。

入門 AWS Identity and Access Management Access Analyzer

藉由此主題內的資訊，了解使用及管理 AWS Identity and Access Management Access Analyzer的必備需求。

如果您在 AWS Organizations 組織內使用 IAM Access Analyzer，您可以 新增 IAM Access Analyzer 的委派管理員。如需服務限制的相關資訊，請參閱 IAM Access Analyzer 配額。

使用 IAM Access Analyzer 的必要許可

若要成功設定和使用 IAM Access Analyzer，您使用的帳戶必須獲得必要的許可。

AWS IAM Access Analyzer 的 受管政策

AWS Identity and Access Management Access Analyzer 提供 AWS 受管政策，協助您快速入門。

- IAMAccessAnalyzerFullAccess：允許管理員完整存取 IAM Access Analyzer。此政策也允許建立服務連結的角色，這些角色允許 IAM Access Analyzer 分析您帳戶或 AWS 組織中的資源。
- IAMAccessAnalyzerReadOnlyAccess：允許對 IAM Access Analyzer 的唯讀存取權。您必須將其他政策新增到 IAM 身分 (使用者、使用者群組或角色)，以允許他們檢視其問題清單。

IAM Access Analyzer 定義的資源

若要檢視 IAM Access Analyzer 定義的資源，請參閱服務授權參考中的 IAM Access Analyzer 定義的資源類型。

必要的 IAM Access Analyzer 服務許可

IAM Access Analyzer 會使用名為 `AWSServiceRoleForAccessAnalyzer` 的服務連結角色 (SLR)。此 SLR 授予服務唯讀存取權，以使用 AWS 資源型政策分析資源，並代表您分析未使用的存取權。此服務會在以下情境中為您的帳戶建立角色：

- 您以自己的帳戶為信任區域建立外部存取權分析器。
- 您以自己的帳戶為選定帳戶來建立未使用的存取權分析器。
- 您以自己的帳戶為信任區域建立內部存取分析器。

如需詳細資訊，請參閱使用 的服務連結角色 AWS Identity and Access Management Access Analyzer。

> **Note**
>
> IAM Access Analyzer 是區域性的。針對外部和內部存取，必須在每個區域中分別啟用 IAM Access Analyzer。
> 針對未使用的存取權，分析器的調查結果不會因區域而變更。不需要在每個您擁有資源的區域中都建立分析器。

某些情況下，在 IAM Access Analyzer 中建立分析器後，系統會載入調查結果頁面或儀表板，但不含任何調查結果或摘要。這可能是因為主控台在填入您的問題清單時出現延遲。您可能須手動重新整理

瀏覽器，或稍後再回來檢視調查結果或摘要。若仍沒有看到外部存取權分析器的調查結果，是因為您的帳戶沒有外部實體可存取的支援資源。若資源套用的政策會將存取權授與外部實體，則 IAM Access Analyzer 會產生問題清單。

> **Note**
>
> 對於外部存取分析器，在修改政策後，IAM Access Analyzer 最多可能需要 30 分鐘才能分析資源並產生新的調查結果，或更新資源存取的現有調查結果。
> 建立內部存取分析器後，可能需要幾分鐘或幾小時才能取得調查結果。初次掃描後，IAM Access Analyzer 會每 24 小時自動重新掃描所有政策。
> 對於所有類型的存取分析器，調查結果的更新內容可能不會立即反映在儀表板中。

檢視調查結果儀表板所需的 IAM Access Analyzer 許可

若要檢視 IAM Access Analyzer 調查結果儀告板，您使用的帳戶必須擁有存取權，才能執行以下必要動作：

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

若要檢視 IAM Access Analyzer 定義的所有動作，請參閱服務授權參考中的 IAM Access Analyzer 定義的動作。

IAM Access Analyzer 狀態

若要檢視分析器的狀態，請選擇 Analyzers (分析器)。為組織或帳戶建立的分析器可能具備下列狀態：

狀態	描述
作用中	對於外部和內部存取分析器，分析器會主動監控信任區域內的資源。分析器會主動產生新的問題清單，並更新現有的問題清單。 對於未使用的存取分析器，分析器會主動監控所選組織或 AWS 帳戶 指定追蹤期間內未使用的存

狀態	描述
	取。分析器會主動產生新的問題清單，並更新現有的問題清單。
正在建立	分析器仍在建立中。建立完成後，分析器就會變成作用中狀態。
已停用	由於 AWS Organizations 管理員採取的動作，分析器已停用。例如，移除身分為 IAM Access Analyzer 委派管理員的分析器帳戶。當分析器處於停用狀態時，不會產生新的調查結果或更新現有的調查結果。
失敗	由於組態發生問題，分析器建立失敗。分析器不會產生任何問題清單。請刪除該分析器並建立新的分析器。

建立 IAM Access Analyzer 外部存取權分析器

若要在區域中啟用外部存取分析器，您必須在該區域中建立分析器。您想監控資源存取權的每個區域，都必須建立外部存取權分析器。

> **Note**
>
> 建立或更新分析器之後，可能需要一段時間才能取得調查結果。

使用 AWS 帳戶 作為信任區域建立外部存取分析器

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 選擇 Create analyzer (建立分析器)。
4. 在分析區段中，選擇資源分析 – 外部存取。
5. 在分析器詳細資訊區段中，確認顯示的區域是您要啟用 IAM Access Analyzer 的區域。
6. 輸入分析器的名稱。
7. 選擇目前帳戶作為分析器的信任區域。

> **Note**
>
> 如果您的帳戶不是 AWS Organizations 管理帳戶或委派管理員帳戶，您只能使用您的帳戶建立一個分析器做為信任區域。

8. 選用。新增您要套用至分析器的標籤。
9. 選擇 Create analyzer (建立分析器)。

建立外部存取權分析器來啟用 IAM Access Analyzer 時，您的帳戶會建立名為 `AWSServiceRoleForAccessAnalyzer` 的服務連結角色。

以組織為信任區域建立外部存取權分析器

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 選擇 Create analyzer (建立分析器)。
4. 在分析區段中，選擇資源分析 – 外部存取。
5. 在分析器詳細資訊區段中，確認顯示的區域是您要啟用 IAM Access Analyzer 的區域。
6. 輸入分析器的名稱。
7. 選擇目前組織做為分析器的信任區域。
8. 選用。新增您要套用至分析器的標籤。
9. 選擇提交。

當您建立以組織做為信任區域的外部存取權分析器時，組織的每個帳戶都會建立名為 `AWSServiceRoleForAccessAnalyzer` 的服務連結角色。

管理 IAM Access Analyzer 外部存取分析器

若要在區域中啟用外部存取分析器，您必須在該區域中建立分析器。您想監控資源存取權的每個區域，都必須建立外部存取權分析器。

> **Note**
>
> 建立或更新分析器之後，可能需要一段時間才能取得調查結果。

更新外部存取分析器

使用下列程序來更新外部存取分析器。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 在分析器區段中，選擇要管理之外部存取分析器的名稱。
4. 在封存規則索引標籤上，可以建立、編輯或刪除分析器的封存規則。如需詳細資訊，請參閱封存規則。
5. 在標籤索引標籤上，可以管理和建立分析器的索引標籤。如需詳細資訊，請參閱AWS Identity and Access Management 資源的標籤。

刪除外部存取分析器

使用下列程序刪除外部存取分析器。刪除分析器後，系統將不再監控資源，也不會產生新的調查結果。分析器產生的所有調查結果都會刪除。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 在分析器區段中，選擇要刪除之外部存取分析器的名稱。
4. 選擇刪除分析器。
5. 輸入 delete，然後選擇刪除以確認刪除分析器。

建立 IAM Access Analyzer 內部存取分析器

若要在某個區域中啟用內部存取權分析器，必須在該區域中建立分析器。您必須在要監控資源存取的每個區域中，建立內部存取分析器。

IAM Access Analyzer 會根據每個分析器每個月監控的資源數量，收取內部存取分析的費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

> **Note**
>
> 建立或更新分析器之後，可能需要一段時間才能取得調查結果。
> IAM Access Analyzer 無法為包含超過 7 萬個主體 (IAM 使用者和角色合計) 的組織產生內部存取調查結果。

您只能在 AWS 組織中建立一個組織層級的內部存取分析器。

使用 AWS 帳戶 作為信任區域來建立內部存取分析器

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 選擇 Create analyzer (建立分析器)。
4. 在分析區段中，選擇資源分析 – 內部存取。
5. 在分析器詳細資訊區段中，確認顯示的區域是您要啟用 IAM Access Analyzer 的區域。
6. 輸入分析器的名稱。
7. 選擇目前帳戶作為分析器的信任區域。

> **Note**
>
> 如果您的帳戶不是 AWS Organizations 管理帳戶或委派管理員帳戶，您只能使用您的帳戶建立一個分析器做為信任區域。

8. 在要分析的資源區段中，新增分析器要監控的資源。
   - 若要依帳戶新增資源，請選擇新增 > 從所選帳戶新增資源。
     1. 選擇所有支援的資源類型，或選擇定義特定資源類型，然後從資源類型清單中選取資源類型。

        內部存取分析器支援下列資源類型：

        - Amazon Simple Storage Service 儲存貯體
        - Amazon Simple Storage Service 目錄儲存貯體
        - Amazon Relational Service 資料庫快照
        - Amazon Relational Database Service 資料庫叢集快照
        - Amazon DynamoDB Streams
        - Amazon DynamoDB 資料表
     2. 選擇新增資源。
   - 若要依 Amazon Resource Name (ARN) 新增資源，請選擇新增資源 > 透過貼上資源 ARN 新增資源。

> **Note**
>
> ARN 必須完全相符 – 不支援萬用字元。對於 Amazon S3，僅支援儲存貯體 ARN。不支援 Amazon S3 物件 ARN 和字首。

1. 對於每個資源 ARN，請輸入帳戶擁有者 ID 和資源 ARN (以逗號分隔)。每行輸入一個帳戶擁有者 ID 和資源 ARN。
2. 選擇新增資源。

- 若要依 CSV 檔案新增資源，請選擇新增資源 > 透過上傳 CSV 檔案新增資源。

  您可以使用 AWS 資源總管 來搜尋帳戶中的資源，並匯出 CSV 檔案。然後，您可以上傳 CSV 檔案來設定分析器要監控的資源。

  1. 選擇選擇檔案，然後從您的電腦中選取 CSV 檔案。
  2. 選擇新增資源。

9. 選用。新增您要套用至分析器的標籤。
10. 選擇 Create analyzer (建立分析器)。

建立內部存取分析器來啟用 IAM Access Analyzer 時，帳戶中會建立名為 `AWSServiceRoleForAccessAnalyzer` 的服務連結角色。

以組織為信任區域建立內部存取分析器

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 選擇 Create analyzer (建立分析器)。
4. 在分析區段中，選擇資源分析 – 內部存取。
5. 在分析器詳細資訊區段中，確認顯示的區域是您要啟用 IAM Access Analyzer 的區域。
6. 輸入分析器的名稱。
7. 選擇整個組織作為分析器的信任區域。
8. 在要分析的資源區段中，新增分析器要監控的資源。

   - 若要為帳戶新增資源，請選擇新增資源 > 從所選帳戶新增資源。

1. 選擇所有支援的資源類型，或選擇定義特定資源類型，然後從資源類型清單中選取資源類型。

   內部存取分析器支援下列資源類型：

   - Amazon Simple Storage Service 儲存貯體
   - Amazon Simple Storage Service 目錄儲存貯體
   - Amazon Relational Service 資料庫快照
   - Amazon Relational Database Service 資料庫叢集快照
   - Amazon DynamoDB Streams
   - Amazon DynamoDB 資料表

2. 若要從您的組織中選取帳戶，請選擇從組織中選取。在選取帳戶區段中，選擇階層以依組織結構選取帳戶，或選擇清單以從組織中的所有帳戶清單中選取帳戶。

   若要從組織手動輸入帳戶，請選擇輸入 AWS 帳戶 ID。在帳戶 AWS 帳戶 IDs欄位中輸入以逗號分隔的AWS 一或多個 ID。

3. 選擇新增資源。

- 若要依 Amazon Resource Name (ARN) 新增資源，請選擇新增資源 > 透過貼上資源 ARN 新增資源。

  > **Note**
  >
  > ARN 必須完全相符 – 不支援萬用字元。對於 Amazon S3，僅支援儲存貯體 ARN。不支援 Amazon S3 物件 ARN 和字首。

  1. 對於每個資源 ARN，請輸入帳戶擁有者 ID 和資源 ARN (以逗號分隔)。每行輸入一個帳戶擁有者 ID 和資源 ARN。
  2. 選擇新增資源。

- 若要依 CSV 檔案新增資源，請選擇新增資源 > 透過上傳 CSV 檔案新增資源。

  您可以使用 AWS 資源總管 來搜尋帳戶中的資源，並匯出 CSV 檔案。然後，您可以上傳 CSV 檔案來設定分析器要監控的資源。

  1. 選擇選擇檔案，然後從您的電腦中選取 CSV 檔案。
  2. 選擇新增資源。

9. 選用。新增您要套用至分析器的標籤。

10. 選擇提交。

當您建立以組織作為信任區域的內部存取分析器時，組織的每個帳戶中會建立一個名為 `AWSServiceRoleForAccessAnalyzer` 的服務連結角色。

管理 IAM Access Analyzer 內部存取分析器

若要在某個區域中啟用內部存取權分析器，必須在該區域中建立分析器。您必須在要監控資源存取的每個區域中，建立內部存取分析器。

> **Note**
>
> 建立或更新分析器之後，可能需要一段時間才能取得調查結果。

更新內部存取分析器

使用下列程序更新內部存取分析器。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 在分析器區段中，選擇要管理之內部存取分析器的名稱。
4. 在封存規則索引標籤上，可以建立、編輯或刪除分析器的封存規則。如需詳細資訊，請參閱封存規則。
5. 在標籤索引標籤上，可以管理和建立分析器的索引標籤。如需詳細資訊，請參閱AWS Identity and Access Management 資源的標籤。
6. 在資源索引標籤上，選擇要分析的資源區段中的編輯。

    a. 若要依帳戶新增資源，請選擇新增資源 > 從所選帳戶新增資源。

        1. 選擇所有支援的資源類型，或選擇定義特定資源類型，然後從資源類型清單中選取資源類型。

            內部存取分析器支援下列資源類型：

            - Amazon Simple Storage Service 儲存貯體
            - Amazon Simple Storage Service 目錄儲存貯體
            - Amazon Relational Service 資料庫快照

- Amazon Relational Database Service 資料庫叢集快照
- Amazon DynamoDB Streams
- Amazon DynamoDB 資料表

2. 選擇新增資源。

b. 若要依 Amazon Resource Name (ARN) 新增資源，請選擇新增資源 > 透過貼上資源 ARN 新增資源。

> **Note**
>
> ARN 必須完全相符 – 不支援萬用字元。對於 Amazon S3，僅支援儲存貯體 ARN。不支援 Amazon S3 物件 ARN 和字首。

1. 對於每個資源 ARN，請輸入帳戶擁有者 ID 和資源 ARN (以逗號分隔)。每行輸入一個帳戶擁有者 ID 和資源 ARN。
2. 選擇新增資源。

c. 若要依 CSV 檔案新增資源，請選擇新增資源 > 透過上傳 CSV 檔案新增資源。

您可以使用 AWS 資源總管 來搜尋帳戶中的資源，並匯出 CSV 檔案。然後，您可以上傳 CSV 檔案來設定分析器要監控的資源。

1. 選擇選擇檔案，然後從您的電腦中選取 CSV 檔案。
2. 選擇新增資源。

d. 若要從分析器中移除資源，請選取要移除的資源旁的核取方塊，然後選擇移除。

e. 選擇儲存變更。

> **Note**
>
> 分析器的任何更新都會在 24 小時內於下一次自動重新掃描時進行評估。

刪除內部存取分析器

使用下列程序刪除內部存取分析器。刪除分析器後，系統將不再監控資源，也不會產生新的調查結果。分析器產生的所有調查結果都會刪除。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 在分析器區段中，選擇要刪除之內部存取分析器的名稱。
4. 選擇刪除分析器。
5. 輸入 delete，然後選擇刪除以確認刪除分析器。

建立 IAM Access Analyzer 未使用的存取權分析器

為目前帳戶建立未使用的存取權分析器

使用下列程序，為單一 AWS 帳戶建立未使用的存取權分析器。針對未使用的存取權，分析器的調查結果不會因區域而變更。不需要在每個您擁有資源的區域中都建立分析器。

IAM Access Analyzer 會根據每月每個分析器分析的 IAM 角色和使用者數量，針對未使用的存取權分析收費。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

> **Note**
>
> 建立或更新分析器之後，可能需要一段時間才能取得調查結果。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 選擇 Create analyzer (建立分析器)。
4. 在分析區段中，選擇主體分析 – 未使用的存取。
5. 輸入分析器的名稱。
6. 在追蹤期間欄位中輸入分析天數。分析器僅會評估在整個追蹤期間一直存在於所選帳戶中的 IAM 實體的許可。例如，如果您將追蹤期間設定為 90 天，則只會對存在至少 90 天的許可進行分析，而且如果在此期間這些許可沒有顯示用量，則會產生調查結果。您可以輸入 1 到 365 天之間的數值。
7. 在分析器詳細資訊區段中，確認顯示的區域是您要啟用 IAM Access Analyzer 的區域。
8. 在選取的帳戶欄位中選擇目前帳戶。

> **Note**
>
> 如果您的帳戶不是 AWS Organizations 管理帳戶或委派管理員帳戶，您只能使用您的帳戶建立一個分析器做為選取的帳戶。

9. 選用。在排除具有標籤的 IAM 使用者和角色區段中，您可以為 IAM 使用者和角色指定索引鍵值對，以將其從未使用的存取權分析中排除。不會為符合索引鍵值對的已排除 IAM 使用者和角色產生調查結果。針對標籤索引鍵，輸入長度為 1 到 128 個字元，而且不以 `aws:` 開頭的值。對於值，可以輸入長度為 0 到 256 個字元的值。如果您未輸入值，則規則會套用至具有指定標籤索引鍵的所有主體。選擇新增排除，以新增要排除的其他索引鍵值對。
10. 選用。新增您要套用至分析器的標籤。
11. 選擇 Create analyzer (建立分析器)。

建立未使用的存取權分析器來啟用 IAM Access Analyzer 時，您的帳戶會建立名為 `AWSServiceRoleForAccessAnalyzer` 的服務連結角色。

使用目前組織建立未使用的存取權分析器

使用下列程序為組織建立未使用的存取分析器，以集中檢閱 AWS 帳戶 組織中的所有 。針對未使用的存取權分析，分析器的調查結果不會因區域而變更。不需要在每個您擁有資源的區域中都建立分析器。

IAM Access Analyzer 會根據每月每個分析器分析的 IAM 角色和使用者數量，針對未使用的存取權分析收費。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

> **Note**
>
> 如果從組織中移除成員帳戶，未使用的存取權分析器將在 24 小時後停止產生新的調查結果，並更新該帳戶的現有調查結果。與從組織中移除的成員帳戶相關聯的調查結果，將在 90 天後永久移除。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 選擇 Create analyzer (建立分析器)。
4. 在分析區段中，選擇主體分析 – 未使用的存取。
5. 輸入分析器的名稱。

6. 在追蹤期間欄位中輸入分析天數。分析器僅會評估在整個追蹤期間一直存在於所選組織帳戶中的 IAM 實體的許可。例如，如果您將追蹤期間設定為 90 天，則只會對存在至少 90 天的許可進行分析，而且如果在此期間這些許可沒有顯示用量，則會產生調查結果。您可以輸入 1 到 365 天之間的數值。
7. 在分析器詳細資訊區段中，確認顯示的區域是您要啟用 IAM Access Analyzer 的區域。
8. 在選取的帳戶欄位中選擇目前組織。
9. 選用。在 AWS 帳戶 從分析中排除區段 AWS 帳戶 中，您可以在組織中選擇要從未使用的存取分析中排除。不會為排除的帳戶產生調查結果。

   a. 若要指定要排除的個別帳戶 ID，請選擇指定 AWS 帳戶 ID，然後在 AWS 帳戶 ID 欄位中輸入以逗號分隔的帳戶 ID。選擇排除。然後，帳戶會列在要排除的AWS 帳戶 資料表中。

   b. 若要從組織內的帳戶清單中選擇要排除的帳戶，請選擇從組織中選擇。

      i. 您可以在從組織排除帳戶欄位中，依名稱、電子郵件和帳戶 ID 搜尋帳戶。

      ii. 選擇階層依組織單位檢視您的帳戶，或選擇清單以檢視組織中所有個別帳戶的清單。

      iii. 選擇排除所有目前帳戶以排除組織單位中的所有帳戶，或選擇排除以排除個別帳戶。

   然後，帳戶會列在要排除的AWS 帳戶 資料表中。

   > **Note**
   >
   > 排除的帳戶不能包含組織分析器擁有者帳戶。將新帳戶新增至您的組織時，不會將其排除在分析之外，即使您先前已排除組織單位中的所有目前帳戶。如需建立未使用的存取權分析器後排除帳戶的詳細資訊，請參閱管理 IAM Access Analyzer 未使用的存取權分析器。

10. 選用。在排除具有標籤的 IAM 使用者和角色區段中，您可以為 IAM 使用者和角色指定索引鍵值對，以將其從未使用的存取權分析中排除。不會為符合索引鍵值對的已排除 IAM 使用者和角色產生調查結果。針對標籤索引鍵，輸入長度為 1 到 128 個字元，而且不以 `aws:` 開頭的值。對於值，可以輸入長度為 0 到 256 個字元的值。如果您未輸入值，則規則會套用至具有指定標籤索引鍵的所有主體。選擇新增排除，以新增要排除的其他索引鍵值對。
11. 選用。新增您要套用至分析器的標籤。
12. 選擇 Create analyzer (建立分析器)。

建立未使用的存取權分析器來啟用 IAM Access Analyzer 時，您的帳戶會建立名為 `AWSServiceRoleForAccessAnalyzer` 的服務連結角色。

管理 IAM Access Analyzer 未使用的存取權分析器

使用本主題中的資訊，了解如何更新或刪除現有的未使用存取權分析器。

> **Note**
>
> 建立或更新分析器之後，可能需要一段時間才能取得調查結果。

更新未使用的存取權分析器

使用下列程序來更新未使用的外部存取權分析器。

IAM Access Analyzer 會根據每月每個分析器分析的 IAM 角色和使用者數量，針對未使用的存取權分析收費。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access Analyzer 下，選擇分析器設定。
3. 在分析器區段中，選擇要管理之未使用的存取分析器的名稱。
4. 在排除索引標籤上，如果建立的分析器以組織作為分析範圍，請選擇已排除 AWS 帳戶區段中的管理。

   a. 若要指定要排除的個別帳戶 ID，請選擇指定 AWS 帳戶 ID，然後在 AWS 帳戶 ID 欄位中輸入以逗號分隔的帳戶 ID。選擇排除。然後，帳戶會列在要排除的AWS 帳戶 資料表中。

   b. 若要從組織內的帳戶清單中選擇要排除的帳戶，請選擇從組織中選擇。

      i. 您可以在從組織排除帳戶欄位中，依名稱、電子郵件和帳戶 ID 搜尋帳戶。

      ii. 選擇階層依組織單位檢視您的帳戶，或選擇清單以檢視組織中所有個別帳戶的清單。

      iii. 選擇排除所有目前帳戶以排除組織單位中的所有帳戶，或選擇排除以排除個別帳戶。

      然後，帳戶會列在要排除的AWS 帳戶 資料表中。

   c. 若要移除要排除的帳戶，請選擇要排除的AWS 帳戶 資料表旁的移除。

   d. 選擇儲存變更。

   > **Note**
   >
   > - 排除的帳戶不能包含組織分析器擁有者帳戶。

- 將新帳戶新增至您的組織時，不會將其排除在分析之外，即使您先前已排除組織單位中的所有目前帳戶。
- 更新分析器的排除項目後，最多可能需要兩天的時間才能更新排除的帳戶清單。

5. 在排除索引標籤上，選擇排除具有標籤的 IAM 使用者和角色區段中的管理。

   a. 您可以為 IAM 使用者和角色指定索引鍵值對，以將其從未使用的存取權分析中排除。針對標籤索引鍵，輸入長度為 1 到 128 個字元，而且不以 `aws:` 開頭的值。對於值，可以輸入長度為 0 到 256 個字元的值。如果您未輸入值，則規則會套用至具有指定標籤索引鍵的所有主體。

   b. 選擇新增排除，以新增要排除的其他索引鍵值對。

   c. 若要移除要排除的索引鍵值對，請選擇索引鍵值對旁的移除。

   d. 選擇儲存變更。

6. 在封存規則索引標籤上，可以建立、編輯或刪除分析器的封存規則。如需詳細資訊，請參閱封存規則。

7. 在標籤索引標籤上，可以管理和建立分析器的索引標籤。如需詳細資訊，請參閱AWS Identity and Access Management 資源的標籤。

刪除未使用的存取權分析器

使用下列程序來刪除未使用的外部存取權分析器。刪除分析器後，系統將不再監控資源，也不會產生新的調查結果。分析器產生的所有調查結果都會刪除。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在存取權分析器下，選擇未使用的存取權。
3. 在 Access Analyzer 下，選擇分析器設定。
4. 在分析器區段中，選擇要刪除之未使用的存取分析器的名稱。
5. 選擇刪除分析器。
6. 輸入 delete，然後選擇刪除以確認刪除分析器。

檢視 IAM Access Analyzer 調查結果儀表板

AWS Identity and Access Management Access Analyzer 會將外部、內部和未使用的存取調查結果組織到視覺化摘要儀表板中。儀表板可協助您了解大規模有效使用許可，並識別需要注意的帳戶 AWS 和資源。您可以使用儀表板，依 AWS 組織、帳戶和問題清單類型檢閱問題清單。

對於外部和內部存取調查結果：

- 儀表板會反白顯示公開存取調查結果、外部存取調查結果和內部存取調查結果之間的分割。
- 儀表板會依資源類型提供調查結果的明細。

對於未使用的存取權調查結果：

- 儀表板 AWS 帳戶 會以最未使用的存取調查結果反白 。
- 儀表板會依類型提供調查結果的明細。

建立任意類型的分析器之後，IAM Access Analyzer 會在相關儀表板中自動新增新的調查結果。這可讓您識別安全問題最多的區域並排定優先順序。

摘要儀表板可讓您全面檢視 IAM Access Analyzer 在 AWS 環境中偵測到的存取問題。然後，您可以向下鑽研個別調查結果以進一步調查，並採取適當的動作來解決這些調查結果。

檢視外部和內部存取分析器摘要儀表板

> **Note**
>
> 建立或更新分析器之後，摘要儀表板可能需要一些時間才會反映調查結果的更新內容。

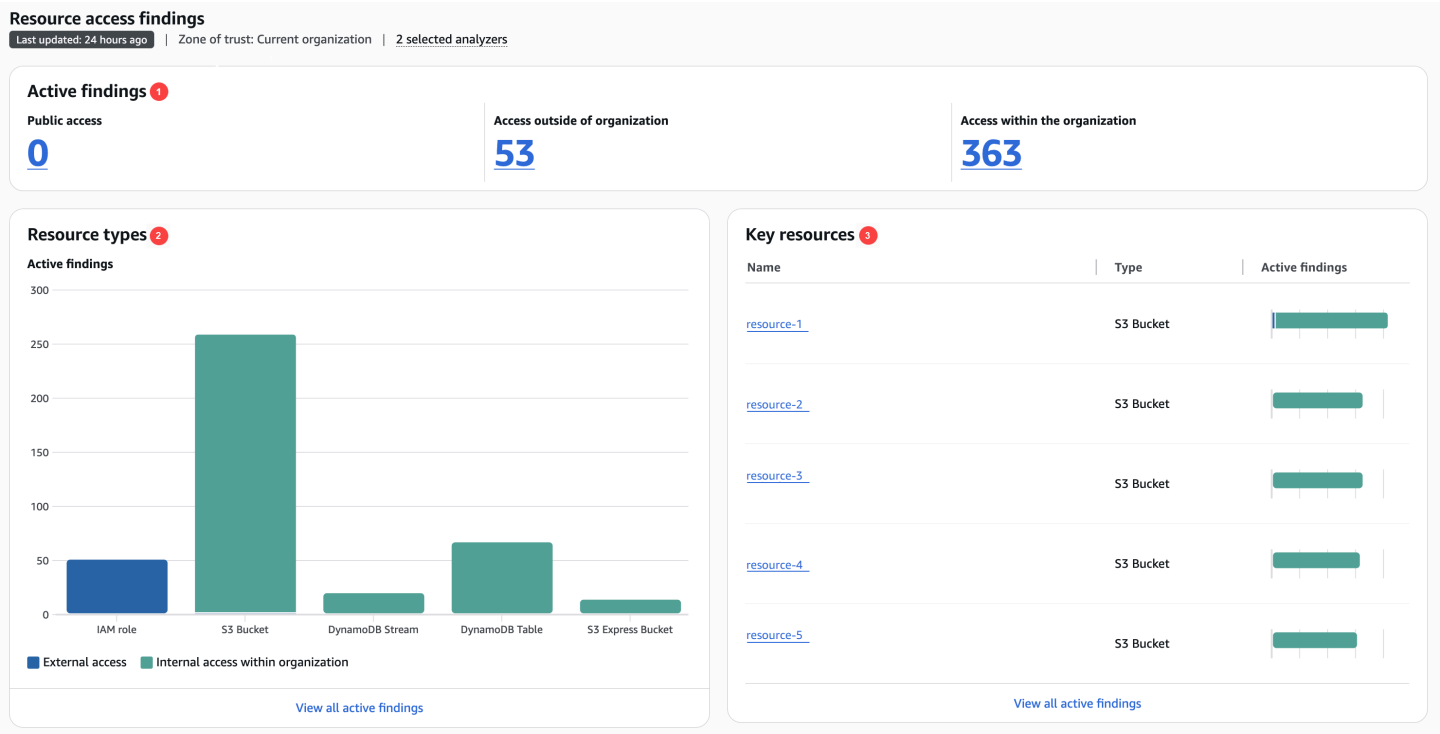
1. 前往網址 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇存取分析器。隨即會顯示摘要視窗。
3. 選擇選取分析器。
4. 在選取分析器視窗中，選擇信任區域的組織或帳戶。

   > **Note**
   >
   > 只有 AWS Organizations 管理帳戶或委派管理員可以選擇組織作為信任區域。

5. 從資源存取分析器下拉式清單中選擇外部和內部存取分析器。

   > **Note**
   >
   > 外部存取分析器和內部存取分析器均最多可以選擇一個。

6. 選擇更新。所選外部和內部存取分析器的調查結果摘要會顯示在資源存取權調查結果區段中。



在上圖中，資源調查結果儀表板顯示在摘要頁面中。

1. 作用中調查結果區段包含所選分析器的可供公開存取的作用中調查結果數量、可在帳戶或組織外部存取的作用中調查結果數量及作用中內部存取調查結果數量。選擇一個數字即可列出每個類型的所有作用中調查結果。
2. 資源類型區段包含資源類型明細，其中包含所選分析器的作用中調查結果。選擇檢視所有作用中調查結果，以取得所選分析器的作用中調查結果完整清單。
3. 主要資源區段包含具有作用中調查結果的主要資源摘要。此資訊可協助您排定業務關鍵型資源的調查結果的優先順序。選擇檢視所有作用中調查結果，以取得所選分析器的作用中調查結果完整清單。

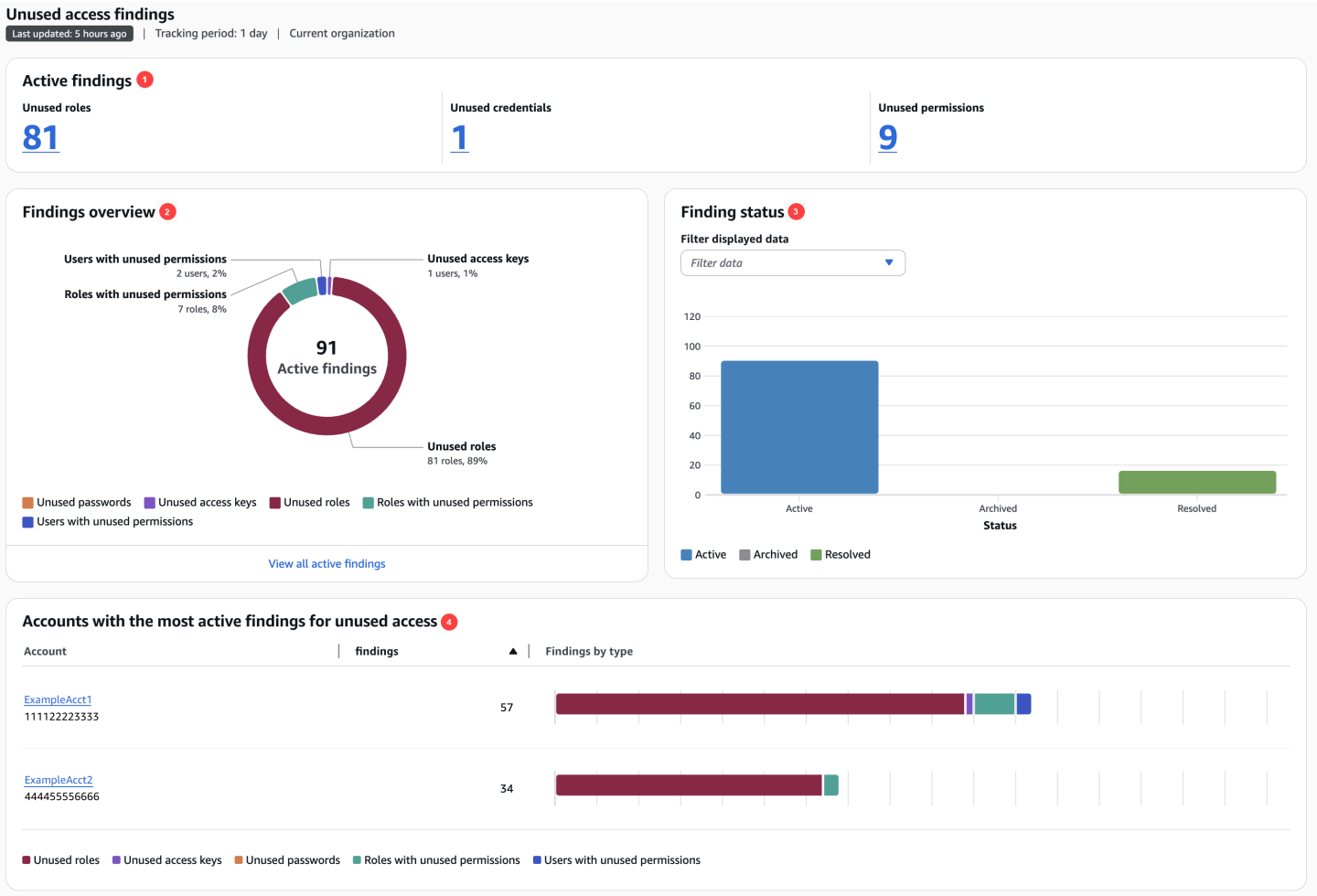
檢視未使用的存取分析器摘要儀表板

> **Note**
>
> 根據使用者和角色的數量建立或更新分析器之後，摘要儀表板可能需要一些時間才會反映調查結果的更新內容。

1. 前往網址 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇存取分析器。存取分析器摘要視窗會隨即顯示。
3. 選擇選取分析器。
4. 在選取分析器視窗中，選擇信任區域的組織或帳戶。

> **Note**
>
> 只有 AWS Organizations 管理帳戶或委派管理員可以選擇組織作為信任區域。

5. 從未使用的存取分析器下拉式清單中選擇未使用的存取分析器。
6. 選擇更新摘要。所選未使用的存取分析器的調查結果摘要會顯示在未使用的存取調查結果區段中。



在上圖中，未使用的存取調查結果儀表板顯示在摘要頁面中。

1. 作用中調查結果區段中，包含帳戶或組織中未使用角色、未使用憑證和未使用的許可作用中調查結果數。未使用的憑證包括未使用的存取金鑰和未使用的密碼調查結果。未使用的許可包括具有未使用許可的使用者和角色。選擇一個數字即可列出每個類型的所有作用中調查結果。
2. 調查結果概觀區段包含作用中調查結果類型的明細。選擇檢視所有作用中調查結果，以取得分析器帳戶或組織的作用中調查結果完整清單。
3. 調查結果狀態區段包含帳戶或組織的調查結果狀態明細 (作用中、已封存及已解決)。您可以在篩選顯示的資料下拉式清單中選取要顯示的調查結果狀態。
4. 只有在未使用的存取權分析器的選定帳戶為組織層級時，才會顯示具有最多未使用存取權調查結果的帳戶區段。此區段包含組織中具有最多作用中調查結果的帳戶明細。這並非貴組織中每個帳戶的詳盡清單。您的分析器作用中調查結果可能包含本區段沒有列出的其他帳戶。

檢閱 IAM Access Analyzer 調查結果

啟用 IAM Access Analyzer 後，下一個步驟是檢閱問題清單，藉以判定其中的存取權為預期之內或之外。您也可檢閱調查結果來決定類似的存取權調查結果是否在預期之內，之後即可建立封存規則，自動將這些調查結果封存。您也可檢閱已存檔與已解決的問題清單。

您應檢閱帳戶內的所有調查結果，藉以判定外部、內部或未使用的存取是否在預期內並經過核准。如果調查結果中識別的存取在預期內，則可將該調查結果封存。封存的調查結果狀態會變更為已封存，並從作用中調查結果清單中移除。但不會刪除該問題清單。您可隨時檢視已存檔的問題清單。請仔細閱讀帳戶內的所有問題清單，直到作用中問題清單數量為零。調查結果數量變成零後，即可知道任何新的作用中調查結果，都是環境內最近的變更所產生。

> **Tip**
>
> 如果您有大量問題清單需要檢閱，請使用 IAM Access Analyzer API 或 AWS CLI 命令以程式設計方式擷取和管理問題清單。

檢閱所有類型存取分析器的作用中調查結果

1. 前往網址 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇 Access analyzer (存取分析器)。此時會顯示調查結果儀表板。
3. 選擇選取分析器。
4. 在選取分析器視窗中，從資源存取分析器下拉式清單中選擇最多一個外部存取分析器和最多一個內部存取分析器。從未使用的存取分析器下拉式清單中選擇未使用的存取分析器。

5. 選擇更新摘要。儀表板上會顯示所選存取分析器的作用中調查結果摘要。在資源存取調查結果或未使用的存取調查結果區段中選擇調查結果類型，以檢視所選類型的所有作用中調查結果。

   如需檢視調查結果儀表板的詳細資訊，請參閱：檢視 IAM Access Analyzer 調查結果儀表板。

> **Note**
>
> 只有在您擁有分析器問題清單的檢視許可時，才會顯示問題清單。

外部和內部存取調查結果

> **Note**
>
> IAM Access Analyzer 會根據每個區域每個月監控的資源數量，收取內部存取分析的費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

1. 在存取分析器下，選擇資源分析。
2. 選擇選取分析器。
3. 在選取分析器視窗中，從資源存取分析器下拉式清單中選擇最多一個外部存取分析器和最多一個內部存取分析器。
4. 選擇更新摘要。

   資源分析頁面會顯示下列有關資源的詳細資訊，這些資源包含所選存取分析器的作用中調查結果：

名稱

具有作用中調查結果的資源的名稱。

類型

資源的類型。

擁有者帳戶

僅當您將某個組織用作一個或多個所選分析器的信任區域時，才會顯示此資料欄。問題清單會回報組織中擁有該資源的帳戶。

作用中調查結果

資源的作用中調查結果數量和類型的視覺化呈現。將滑鼠暫留在此欄位上，即可顯示有關資源調查結果的詳細資訊。

公開存取

指示資源的任何調查結果是否允許公開存取。

未使用的存取權調查結果

> **Note**
>
> IAM Access Analyzer 會根據每月分析的 IAM 角色和使用者數量，針對未使用的存取權分析收費。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

1. 在存取分析器下，選擇未使用的存取。
2. 選擇選取分析器。
3. 在選取分析器視窗中，從未使用的存取分析器下拉式清單中選擇未使用的存取分析器。
4. 選擇更新摘要。

未使用的存取頁面會顯示下列有關針對所選存取分析器產生調查結果的 IAM 實體的詳細資訊：

問題清單 ID

指派給問題清單的唯一 ID。選擇調查結果 ID 來顯示產生調查結果之 IAM 實體的其他詳細資訊。

調查結果類型

未使用的存取權調查結果類型：未使用的存取金鑰、未使用的密碼、未使用的許可或未使用的角色。

IAM 實體

調查結果中報告的 IAM 實體。可以是 IAM 使用者或角色。

AWS 帳戶 ID

只有當您已為組織中的所有 AWS 帳戶 設定分析器時，此欄才會顯示。 AWS 帳戶 組織中擁有調查結果中報告之 IAM 實體的 。

上次更新

上次更新調查結果中所報告 IAM 實體的時間，或者如果未進行更新，則為實體的建立時間。

狀態

調查結果的狀態為作用中、已封存或已解決。

篩選 IAM Access Analyzer 調查結果

調查結果頁面的預設篩選結果會顯示所有作用中調查結果。若要檢視所有調查結果，請從狀態下拉式清單中選擇全部。若要檢視已封存的調查結果，請選擇已封存。若要檢視已解決的調查結果，請選擇已解決。首次使用 IAM Access Analyzer 不會有已存檔的問題清單。

使用篩選器只顯示符合指定屬性條件的調查結果。若要建立篩選條件，請選取要篩選的屬性，然後選擇屬性是等於還是包含值，然後輸入或選擇要篩選的屬性值。

如需可用來建立或更新存檔規則的篩選鍵清單，請參閱 IAM Access Analyzer 篩選鍵。

篩選具有作用中調查結果的資源

您可以依資源檢視和篩選最多一個外部存取分析器和最多一個內部存取分析器的作用中調查結果。

篩選具有作用中調查結果的資源

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇資源分析。
3. 若要依資源名稱篩選，請在搜尋方塊中輸入資源的完整或部分名稱。
4. 在篩選存取類型下拉式清單中，選擇存取類型：
   - 所有類型 – 顯示具有所有類型存取調查結果的資源。
   - 公開存取 – 僅顯示具有公開存取調查結果的資源。
   - 外部存取 – 僅顯示具有外部存取調查結果的資源。
   - 組織內的內部存取 – 僅顯示具有內部存取調查結果的資源。
5. 在篩選資源類型下拉式清單中，選擇資源類型以僅顯示所選類型的資源。

篩選外部存取權調查結果

篩選外部存取權調查結果

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇分析器設定，然後在分析器區段中選擇外部存取分析器。
3. 選擇檢視調查結果。
4. 選擇搜尋方塊以顯示可用屬性清單。
5. 選擇要用來篩選所顯示之問題清單的屬性。
6. 選擇該屬性須符合的值。將會顯示具備該值的問題清單。

   例如，選擇資源作為屬性，然後選擇資源：，輸入儲存貯體的部分或完整名稱，然後按 Enter 鍵。僅顯示符合篩選條件的儲存貯體相關調查結果。若要建立僅顯示允許公開存取權之資源的調查結果，您可以選擇公開存取權屬性，選擇公開存取權 =，然後選擇公開存取權 = true。

您可新增其他屬性，進一步篩選所顯示的問題清單。新增其他屬性時，只會顯示符合所有篩選條件的問題清單。問題清單的篩選條件，不支援符合單一屬性「或」另一屬性的這類定義。選擇清除篩選條件以清除已定義的任何篩選條件，並顯示具有指定分析器狀態的所有調查結果。

只有在您檢視以組織做為信任區域的分析器問題清單時，某些欄位才會顯示。

下列屬性可用於定義外部存取篩選條件：

- Public access (公有存取) – 若要依允許公有存取之資源的問題清單篩選，請依 Public access (公有存取) 篩選，然後選擇 Public access: true (公有存取：true)。
- Resource (資源) – 輸入資源的完整或部分名稱，即可依資源篩選。
- Resource Type (資源類型) – 從所顯示的清單內選擇類型，即可依資源類型篩選。
- 資源擁有者帳戶：使用此屬性可篩選組織中擁有調查結果所報告資源的帳戶。
- 資源控制政策限制：使用此屬性依組織資源控制政策 (RCP) 所套用的限制類型進行篩選。若要進一步了解，請參閱 AWS Organizations 《使用者指南》中的資源控制政策 (RCPs)。
  - 無法評估 RCP：評估 RCP 時發生錯誤。
  - 不適用：沒有 RCP 會限制此資源或主體。這也包括尚未支援 RCP 的資源。
  - 適用：您的組織管理員已透過會影響資源或資源類型的 RCP 設定限制。如需更多詳細資訊，請聯絡您的組織管理員。

- AWS 帳戶 – 使用此屬性來篩選政策陳述式的主體區段中 AWS 帳戶 授予存取權的 。若要依 篩選 AWS 帳戶，請輸入 12 位數 AWS 帳戶 ID 的全部或部分，或可存取目前帳戶中資源之外部 AWS 使用者或角色的全部或部分完整帳戶 ARN。
- 正式使用者：輸入為 Amazon S3 儲存貯體定義的正式使用者 ID，即可依正式使用者篩選。若要進一步了解，請參閱 AWS 帳戶識別碼。
- Federated User (聯合身分使用者) – 輸入聯合身分的完整或部分 ARN，即可依聯合身分使用者篩選。若要進一步了解，請參閱身分提供者與聯合。
- 調查結果 ID：請輸入全部或部分調查結果 ID，即可依調查結果 ID 篩選。
- 錯誤：若要依錯誤類型篩選，請選擇存取遭拒或內部錯誤。
- Principal ARN (主體 ARN) – 可使用此屬性來依 aws:PrincipalArn 條件索引鍵中使用的主體 (IAM 使用者、角色或群組) ARN 篩選。若要依主體 ARN 篩選，請從調查結果中 AWS 帳戶 報告的外部輸入 IAM 使用者、角色或群組的全部或部分 ARN。
- 委託人OrgID – 若要依委託人OrgID 篩選，請輸入與屬於調查結果中指定為條件之組織的外部委託人相關聯的全部或部分 AWS 組織 ID。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- Principal OrgPaths – 若要依 Principal OrgPaths 篩選，請輸入 AWS 組織或組織單位 (OU) 的全部或部分 ID，以允許存取屬於指定組織或 OU 帳戶成員的所有外部主體作為政策中的條件。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- 來源帳戶 – 若要篩選來源帳戶，請輸入與資源相關聯的全部或部分 AWS 帳戶 ID，如 中的某些跨服務許可中所使用 AWS。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- Source ARN (來源 ARN) – 輸入問題清單內列為條件的完整或部分 ARN，即可依來源 ARN 篩選。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- Source IP (來源 IP) – 輸入允許外部實體在使用特定 IP 地址時即可存取目前帳戶內資源的完整或部分 IP 地址，即可依來源 IP 篩選。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- 來源組織 ID：輸入與資源相關聯的組織完整或部分 ID，即可依來源組織 ID 篩選，如同 AWS中部分跨服務許可所使用。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- 來源組織路徑：輸入與資源相關聯的完整或部分組織單位 (OU)，即可依來源組織路徑篩選，如同 AWS中部分跨服務許可所使用。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- Source VPC (來源 VPC) – 輸入允許外部實體在使用特定 VPC 時即可存取目前帳戶內資源的完整或部分 VPC ID，即可依來源 VPC 篩選。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- 來源 VPC ARN – 若要依來源 VPC ARN 篩選，請輸入 VPC ARN 的全部或部分，允許外部實體在使用指定的 VPC 時存取目前帳戶中的資源。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- 來源 VPCE – 若要依來源 VPCE 篩選，請輸入完整或部分 VPC 端點 ID，其允許外部實體在使用指定 VPC 端點時存取目前帳戶中的資源。若要進一步了解，請參閱 AWS 全域條件內容鍵。

- VPCE 帳戶 – 若要依 VPCE 帳戶篩選，請輸入全部或部分擁有 VPC 端點外部實體的 12 位數 AWS 帳戶 ID，並允許外部實體存取 資源。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- VPCE OrgID – 若要依 VPCE OrgID 篩選，請輸入完整或部分組織 ID，其擁有 VPC 端點外部實體並允許外部實體存取資源。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- VPCE OrgPaths – 若要依 VPCE OrgPaths 篩選，請輸入完整或部分組織單位，其擁有 VPC 端點外部實體並允許外部實體存取資源。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- 使用者 ID – 若要依使用者 ID 篩選，請從 AWS 帳戶 允許存取目前帳戶中資源的外部輸入 IAM 使用者的全部或部分使用者 ID。若要進一步了解，請參閱 AWS 全域條件內容鍵。
- KMS 金鑰 ID：輸入目前帳戶內 AWS KMS加密 Amazon S3 物件存取指定為條件的 KMS 金鑰完整或部分金鑰 ID，即可依 KMS 金鑰 ID 篩選。
- 工作階段模式 – 若要依 Amazon S3 目錄儲存貯體 (`ReadOnly` 或 `ReadWrite`) 的工作階段模式進行篩選，請輸入工作階段模式的完整或部分名稱。如要進一步了解，請參閱 Amazon Simple Storage Service API Reference 中的 CreateSession。
- Google Audience (Google 對象) – 輸入目前帳戶內 IAM 角色存取指定為條件的完整或部分 Google 應用程式 ID，即可依 Google 對象篩選。若要進一步了解，請參閱 IAM 和 AWS STS 條件內容索引鍵。
- Cognito 對象：輸入目前帳戶內 IAM 角色存取指定為條件的完整或部分 Amazon Cognito 身分池 ID，即可依 Cognito 對象篩選。若要進一步了解，請參閱 IAM 和 AWS STS 條件內容索引鍵。
- 來電者帳戶 – 擁有或包含呼叫實體的帳戶 AWS 帳戶 ID，例如 IAM 角色、使用者或帳戶根使用者。這是供 服務呼叫使用 AWS KMS。若要依來電者帳戶篩選，請輸入全部或部分 AWS 帳戶 ID。
- Facebook App ID (Facebook 應用程式 ID) – 關於允許存取目前帳戶內 IAM 角色的 Login with Facebook (使用 Facebook 登入) 聯合身分，輸入列為其條件的完整或部分 Facebook 應用程式 ID (或網站 ID)，即可依 Facebook 應用程式 ID 篩選。若要進一步了解，請參閱 IAM 中的 ID 區段和條件內容索引鍵。 AWS STS
- Amazon App ID (Amazon 應用程式 ID) – 關於允許存取目前帳戶內 IAM 角色的 Login with Amazon 聯合身分，輸入列為其條件的完整或部分 Amazon 應用程式 ID (或網站 ID)，即可依 Amazon 應用程式 ID 篩選。若要進一步了解，請參閱 IAM 中的 ID 區段和條件內容索引鍵。 AWS STS
- Lambda Event Source Token (Lambda 事件來源字符) – 輸入完整或部分字符字串，即可依使用 Alexa 整合傳入的 Lambda 事件來源字符篩選。

篩選內部存取調查結果

篩選內部存取調查結果

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇分析器設定，然後在分析器區段中選擇內部存取分析器。
3. 選擇檢視調查結果。
4. 選擇搜尋方塊以顯示可用屬性清單。
5. 選擇要用來篩選所顯示之問題清單的屬性。
6. 選擇該屬性須符合的值。將會顯示具備該值的問題清單。

   例如，選擇資源作為屬性，然後選擇資源： ，輸入儲存貯體的部分或完整名稱，然後按 Enter 鍵。僅顯示符合篩選條件的儲存貯體相關調查結果。

您可新增其他屬性，進一步篩選所顯示的問題清單。新增其他屬性時，只會顯示符合所有篩選條件的問題清單。問題清單的篩選條件，不支援符合單一屬性「或」另一屬性的這類定義。選擇清除篩選條件以清除已定義的任何篩選條件，並顯示具有指定分析器狀態的所有調查結果。

只有在您檢視以組織做為信任區域的分析器問題清單時，某些欄位才會顯示。

只有在您檢視的調查結果來自監控內部存取的分析器時，才會顯示下列欄位：

- Resource (資源) – 輸入資源的完整或部分名稱，即可依資源篩選。
- Resource Type (資源類型) – 從所顯示的清單內選擇類型，即可依資源類型篩選。
- 資源擁有者帳戶：使用此屬性可篩選組織中擁有調查結果所報告資源的帳戶。
- 調查結果 ID：輸入完整或部分調查結果 ID，即可依調查結果 ID 篩選。

篩選未使用的存取權

篩選未使用的存取權調查結果

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇未使用的存取權，然後從檢視分析器下拉式清單中選擇分析器。
3. 選擇搜尋方塊以顯示可用屬性清單。
4. 選擇要用來篩選所顯示之問題清單的屬性。
5. 選擇該屬性須符合的值。將會顯示具備該值的問題清單。

例如，選擇調查結果類型作為屬性，然後依次選擇調查結果類型 = 和調查結果類型 = 未使用的角色。僅顯示類型為未使用的角色的調查結果。

您可新增其他屬性，進一步篩選所顯示的問題清單。新增其他屬性時，只會顯示符合所有篩選條件的問題清單。問題清單的篩選條件，不支援符合單一屬性「或」另一屬性的這類定義。選擇清除篩選條件以清除已定義的任何篩選條件，並顯示具有指定分析器狀態的所有調查結果。

只有在您檢視的調查結果來自監控未使用的存取權之分析器時，下列欄位才會顯示：

- 調查結果類型：依調查結果類型篩選，然後選擇調查結果類型，即可依調查結果類型篩選。
- Resource (資源) – 輸入資源的完整或部分名稱，即可依資源篩選。
- Resource Type (資源類型) – 從所顯示的清單內選擇類型，即可依資源類型篩選。
- 資源擁有者帳戶：使用此屬性可篩選組織中擁有調查結果所報告資源的帳戶。
- 調查結果 ID：輸入完整或部分調查結果 ID，即可依調查結果 ID 篩選。

封存 IAM Access Analyzer 調查結果

如果調查結果中的資源存取權是有意圖的，就可以封存該調查結果。例如，針對已核准的工作流程而存取的 Amazon S3 儲存貯體的外部或內部存取調查結果，或可能仍是必要的某個存取金鑰的未使用存取調查結果。當您封存調查結果時，該項目會從作用中調查結果清單中清除。存檔的問題清單不會刪除。您可篩選調查結果頁面來顯示已封存的調查結果，並隨時將其取消封存。

在 Findings (問題清單) 頁面將問題清單存檔

1. 選取要存檔之一個或多個調查結果旁的核取方塊。
2. 選擇動作，然後選擇封存。

   螢幕頂端會顯示確認訊息。

在調查結果詳細資訊頁面中封存調查結果。

1. 選擇欲存檔的 Finding ID (問題清單 ID)。
2. 選擇 Archive (封存)。

   螢幕頂端會顯示確認訊息。

若要取消問題清單的存檔，請重複上述步驟，但選擇 Unarchive (取消存檔)，不要選擇 Archive (存檔)。問題清單取消存檔後，狀態會變更為作用中。

解決 IAM Access Analyzer 調查結果

解決資源調查結果

若要解決意外存取產生的外部和內部存取調查結果，您應該修改政策陳述式，以移除允許存取所識別資源的許可。

關於與 Amazon S3 儲存貯體有關的調查結果，請使用 Amazon S3 主控台來設定該儲存貯體的許可。

若是 IAM 角色，請使用 IAM 主控台來修改所列出之 IAM 角色的信任政策。

關於其他支援的資源，請使用主控台來修改產生調查結果的政策陳述式。

進行變更以解決資源調查結果後 (例如修改套用至 IAM 角色的政策)，IAM Access Analyzer 會再次掃描資源。如果移除資源的存取權，該調查結果的狀態會變更為已解決。然後，系統會在已解決的調查結果清單，而不是作用中的調查結果清單中顯示調查結果。

> **Note**
>
> 上述內容不適用於錯誤調查結果。當 IAM Access Analyzer 無法分析資源時，就會產生錯誤調查結果。如果您解決了導致 IAM Access Analyzer 無法分析資源的問題，錯誤調查結果就會完全移除，而不是變更為已解決的調查結果。如需詳細資訊，請參閱IAM Access Analyzer 錯誤調查結果。

如果所作的變更導致以另一種方式對資源進行外部或內部存取 (例如使用不同的主體或以不同的許可)，則 IAM Access Analyzer 會解決原始調查結果並產生新的作用中調查結果。如果所作的變更導致內部錯誤或存取遭拒錯誤，則會解決與資源特定存取連結的所有作用中非錯誤調查結果，並產生新的錯誤調查結果。

> **Note**
>
> 對於外部存取分析器，在修改政策後，IAM Access Analyzer 最多可能需要 30 分鐘才能再次分析資源並更新調查結果。
> 對於內部存取分析器，IAM Access Analyzer 可能需要幾分鐘或幾小時才能再次分析資源並更新調查結果。IAM Access Analyzer 會每 24 小時自動重新掃描所有政策。
> 已解決的問題清單會在最後更新至問題清單狀態的 90 天後刪除。

解決未使用的存取調查結果

IAM Access Analyzer 會提供建議的步驟，根據調查結果類型來解決未使用的存取分析器調查結果。

在您進行變更以解決未使用的存取權調查結果之後，下次執行未使用的存取權分析器時，調查結果的狀態會變更為已解決。該調查結果不會再出現於作用中調查結果清單中，而是會顯示在已解決調查結果清單內。如果您進行的變更僅部分解決未使用的存取權調查結果，現有的調查結果會變更為已解決，但會產生新的調查結果。例如，如果您只移除調查結果中部分未使用的許可，而非所有未使用的許可。

IAM Access Analyzer 會根據每月分析的 IAM 角色和使用者數量，針對未使用的存取權分析收費。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

解決未使用的許可調查結果

對於未使用的許可調查結果，IAM Access Analyzer 可以建議從 IAM 使用者或角色中移除的政策，並提供新政策來取代現有的許可政策。下列案例不支援政策建議：

- 未使用的許可調查結果適用於在使用者群組中的 IAM 使用者。
- 未使用的許可調查結果適用於 IAM Identity Center 的 IAM 角色。
- 未使用的許可調查結果具有包含 `notAction` 元素的現有許可政策。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇未使用的存取。
3. 選擇調查結果類型為未使用的許可的調查結果。
4. 在建議區段中，如果有在建議的政策資料欄中列出的政策，請選擇預覽政策以檢視具有建議政策的現有政策，以取代現有政策。如果有多個建議的政策，您可以選擇下一個政策和上一個政策來檢視每個現有和建議的政策。
5. 選擇下載 JSON 以下載 .zip 檔案，其中包含所有建議政策的 JSON 檔案。
6. 建立建議的政策並將其連接至 IAM 使用者或角色。如需詳細資訊，請參閱變更使用者的許可 (主控台) 和修改角色許可政策 (主控台)。
7. 從 IAM 使用者或角色移除現有的許可政策資料欄中列出的政策。如需詳細資訊，請參閱從使用者移除許可 （ 主控台 ） 和修改角色許可政策 （ 主控台 ）。

解決未使用的角色調查結果

對於未使用的角色調查結果，IAM Access Analyzer 建議刪除未使用的 IAM 角色。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇未使用的存取。
3. 選擇一個調查結果類型為未使用的角色的調查結果。
4. 在建議區段中，檢閱 IAM 角色的詳細資訊。
5. 刪除 IAM 角色。如需詳細資訊，請參閱刪除 IAM 角色 (主控台)。

解決未使用的存取金鑰調查結果

對於未使用的存取金鑰調查結果，IAM Access Analyzer 建議停用或刪除未使用的存取金鑰。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇未使用的存取。
3. 選擇一個調查結果類型為未使用的存取金鑰的調查結果。
4. 在建議區段中，檢閱存取金鑰的詳細資訊。
5. 停用或刪除存取金鑰 如需詳細資訊，請參閱管理存取金鑰 (控制台)。

解決未使用的密碼調查結果

對於未使用的密碼調查結果，IAM Access Analyzer 建議刪除 IAM 使用者的未使用密碼。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇未使用的存取。
3. 選擇一個調查結果類型為未使用的密碼的調查結果。
4. 在建議區段中，檢閱 IAM 使用者的詳細資訊。
5. 刪除 IAM 使用者的密碼。如需詳細資訊，請參閱建立、變更或刪除 IAM 使用者密碼 (主控台)。

IAM Access Analyzer 錯誤調查結果

當 IAM Access Analyzer 分析資源時，通常會產生調查結果，這些調查結果會顯示誰有權存取您的資源。不過，在某些情況下，分析器可能會遇到導致其無法完成分析的問題。在這些情況下，IAM Access Analyzer 會改為產生錯誤調查結果。

錯誤調查結果表示 IAM Access Analyzer 無法完成對特定資源或特定主體-資源對的分析。這些調查結果可協助您識別可能需要關注的資源，以確保正確地對它們進行分析。

外部存取錯誤調查結果

外部存取分析器可識別在您帳戶或組織外部共用的資源，並能夠產生兩種類型的錯誤調查結果：

- INTERNAL_ERROR – 表示 IAM Access Analyzer 在分析資源時遇到內部問題。導致此錯誤調查結果的原因可能是服務限制或臨時問題。


```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
 ],
 "resource": "arn:aws:iam::941407043048:role/TestAccessAnalyzer",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR",
 "createdAt": "2022-07-14T01:31:43.085000+00:00",
 "resourceType": "AWS::IAM::Role",
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.109000+00:00",
 "id": "4b035c7d-b7d2-40e4-a6c3-9887d1a995df",
 "updatedAt": "2022-07-14T01:31:43.085000+00:00"
}
```


- ACCESS_DENIED – 表示 IAM Access Analyzer 沒有分析資源所需的許可。當 IAM Access Analyzer 的服務連結角色 (SLR) 無法存取資源時，通常會產生此錯誤調查結果。


```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
 ],
 "resource": "arn:aws:kms:us-west-2:941407043048:key/01cae123-
b7f2-4488-9a05-0070a072ea2c",
 "status": "ACTIVE",
 "error": "ACCESS_DENIED",
 "createdAt": "2022-07-14T01:31:43.104000+00:00",
 "resourceType": "AWS::KMS::Key",
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.090000+00:00",
```

```
 "id": "7ef6f04a-9d2c-4038-9cc0-2a5f00a4d8f8",
 "updatedAt": "2022-07-14T01:31:43.104000+00:00"
}
```

內部存取錯誤調查結果

內部存取分析器可識別您帳戶或組織內的存取，並能夠產生四種類型的錯誤調查結果：

- PRINCIPAL_LIMIT_EXCEEDED – 當超過 3,000 個主體有權存取關鍵資源時會產生此錯誤調查結果。此錯誤可協助您識別存取權限過於廣泛的資源，可能需要對這些資源加以限制。

  如果您變更環境中的資源或主體，使主體數量低於限制，則分析器將在下次掃描期間產生正常的調查結果，且此錯誤調查結果會標記為已解決。

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_LIMIT_EXCEEDED",
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- 資源層級錯誤 (INTERNAL_ERROR 或 ACCESS_DENIED) – 與外部存取錯誤類似，這些錯誤表示分析器因內部問題或許可問題而無法分析特定資源。發生資源層級錯誤時，分析器會針對資源產生單一錯誤調查結果，而非正常調查結果。

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
```

```
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- 主體層級錯誤 (INTERNAL_ERROR 或 ACCESS_DENIED) – 表示分析器無法分析特定主體對特定資源的存取。與資源層級錯誤不同，資源可以同時有兩種類型的調查結果，即針對某些主體的正常調查結果和針對其他主體的錯誤調查結果。

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {
    "principal": {
     "AWS": "arn:aws:iam::111122223333:role/MyRole_1"
    },
    "principalOwnerAccount": "111122223333",
    "principalType": "IAM_ROLE",
    "accessType": "INTRA_ACCOUNT"
   }
  }
 ]
}
```

- PRINCIPAL_ERRORS_LIMIT_EXCEEDED – 當單一資源有太多主體層級錯誤調查結果時會產生此錯誤調查結果。此錯誤調查結果是資源層級的錯誤調查結果，可能會與同一個資源的正常調查結果一起顯示。

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_ERRORS_LIMIT_EXCEEDED",
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "resourceControlPolicyRestriction": "NOT_APPLICABLE",
 "serviceControlPolicyRestriction": "NOT_APPLICABLE",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

解決錯誤調查結果

如果您解決了導致 IAM Access Analyzer 無法分析資源的問題，錯誤調查結果就會完全移除，而不是變更為已解決的調查結果。

若要解決錯誤調查結果，請根據錯誤類型考慮採取下列方法：

- 對於 ACCESS_DENIED 錯誤，請確認 IAM Access Analyzer 服務連結角色具有存取資源的必要許可。
- 對於 PRINCIPAL_LIMIT_EXCEEDED 錯誤，請檢閱資源的存取政策，並考慮限制對該資源的存取權限，減少有存取權限主體的數量。
- 對於 INTERNAL_ERROR 調查結果，您可能需要等待後續分析週期，如果問題仍然存在，請聯絡 AWS 支援。
- 對於 PRINCIPAL_ERRORS_LIMIT_EXCEEDED，請檢閱並盡可能簡化受影響資源的存取模式。

在進行變更以解決根本問題之後，IAM Access Analyzer 會在下次掃描週期中嘗試再次分析資源。

IAM Access Analyzer 外部和內部存取支援的資源類型

對於外部和內部存取分析器，IAM Access Analyzer 會分析套用至您啟用 IAM Access Analyzer 之區域中 AWS 資源的資源型政策。系統只會分析資源型政策。如需有關 IAM Access Analyzer 如何為每個資源類型產生調查結果的詳細資訊，請檢閱資源類型資訊。

> **Note**
>
> 列出的支援資源類型適用於外部和內部存取分析器。內部存取分析器不支援外部存取分析器支援的所有資源類型。未使用的存取權分析器僅支援 IAM 使用者和角色。如需詳細資訊，請參閱了解 IAM Access Analyzer 調查結果的運作方式。

外部存取支援的資源類型

- Amazon Simple Storage Service 儲存貯體
- Amazon Simple Storage Service 目錄儲存貯體
- AWS Identity and Access Management 角色
- AWS Key Management Service 金鑰
- AWS Lambda 函數和圖層
- Amazon Simple Queue Service 佇列
- AWS Secrets Manager 秘密
- Amazon Simple Notification Service 主題
- Amazon Elastic Block Store 磁碟區快照
- Amazon Relational Service 資料庫快照
- Amazon Relational Database Service 資料庫叢集快照
- Amazon Elastic Container Registry 儲存庫
- Amazon Elastic File System 檔案系統
- Amazon DynamoDB Streams
- Amazon DynamoDB 資料表

內部存取支援的資源類型

- Amazon Simple Storage Service 儲存貯體
- Amazon Simple Storage Service 目錄儲存貯體
- Amazon Relational Service 資料庫快照
- Amazon Relational Database Service 資料庫叢集快照
- Amazon DynamoDB Streams
- Amazon DynamoDB 資料表

Amazon Simple Storage Service 儲存貯體

當 IAM Access Analyzer 對外部存取分析器的 Amazon S3 儲存貯體進行分析時，若套用至儲存貯體的 Amazon S3 儲存貯體政策、存取控制清單 (ACL) 或存取點 (包括多區域存取點) 向外部實體授予存取權，則會產生調查結果。外部實體是您可用來建立不在信任區域內之篩選的主體或其他實體。例如，如果儲存貯體政策對另一個帳戶授與存取權或允許公有存取權，則 IAM Access Analyzer 會產生問題清單。不過，如果您在儲存貯體上啟用 Block Public Access (封鎖公有存取權)，則可以在帳戶層級或儲存貯體層級封鎖存取權。

對於內部存取分析器，IAM Access Analyzer 會在組織或帳戶中的主體 (使用者或角色) 有權存取指定的 Amazon S3 儲存貯體時，產生調查結果。

> **Note**
>
> IAM Access Analyzer 不會分析連接至跨帳戶存取點的存取點政策，因為存取點及其政策不在分析器帳戶以內。當儲存貯體委派對跨帳戶存取點的存取權，且儲存貯體或帳戶上未啟用「封鎖公開存取」時，IAM Access Analyzer 會產生公開問題清單。當您啟用「封鎖公開存取」時，公開問題清單得以解決，IAM Access Analyzer 將產生跨帳戶存取點的跨帳戶問題清單。

Amazon S3 封鎖公開存取設定會覆寫套用至儲存貯體的儲存貯體政策。這些設定也會覆寫套用至儲存貯體存取點的存取點政策。IAM Access Analyzer 會在政策變更時分析儲存貯體層級的「封鎖公開存取」設定。不過，它僅以 6 小時一次的頻率在帳戶層級評估「封鎖公開存取」設定。這意味著 IAM Access Analyzer 未對儲存貯體產生公有存取權問題清單或進行解決的時間可能長達 6 小時。例如，如果您有允許公有存取權的儲存貯體政策，則 IAM Access Analyzer 會產生該存取權的問題清單。如果您接著啟用「封鎖公開存取」以在帳戶層級封鎖對儲存貯體的所有公開存取權，即使對儲存貯體的所有公開存取權都遭到封鎖，IAM Access Analyzer 可能需要達 6 小時的時間來解決儲存貯體政策之問題

清單。一旦您在帳戶層級啟用「封鎖公開存取」，解決跨帳戶存取點的公開問題清單最多也可能需要 6 小時。對資源控制政策 (RCP) 所做的變更，若未變更儲存貯體政策，則不會觸發重新掃描調查結果中報告的儲存貯體。IAM Access Analyzer 會在下次定期掃描 (24 小時內) 期間分析該新增或更新的政策。

對於多區域存取點，IAM Access Analyzer 會使用已建立的政策來產生問題清單。IAM Access Analyzer 每 6 小時會評估一次對多區域存取點的變更。這表示即使您建立或刪除多區域存取點，或更新其政策，IAM Access Analyzer 對此不產生或解決問題清單的時間可達最多 6 小時。

Amazon Simple Storage Service 目錄儲存貯體

Amazon S3 目錄儲存貯體會以階層式儲存結構將資料整理到目錄中，而不同於一般用途儲存貯體所採用的扁平式儲存結構，建議將後者用於效能關鍵型工作負載或應用程式。對於外部存取分析器，IAM Access Analyzer 會分析允許外部實體存取目錄儲存貯體的目錄儲存貯體政策 (包括政策中的條件陳述式)。

對於內部存取分析器，IAM Access Analyzer 會在組織或帳戶中的主體 (使用者或角色) 有權存取指定的 Amazon S3 目錄儲存貯體時，產生調查結果。

Amazon S3 目錄儲存貯體還支援存取點，可對透過存取點向目錄儲存貯體發出的所有請求，強制執行不同的許可和網路控制。每個存取點都有存取點政策，該政策可與連接至基礎目錄儲存貯體的儲存貯體政策搭配使用。藉由目錄儲存貯體的存取點，您可以限制對特定字首、API 動作或虛擬私有雲端 (VPC) 的存取。

> **Note**
>
> IAM Access Analyzer 不會分析連接至跨帳戶存取點的存取點政策，因為存取點及其政策不在分析器帳戶以內。當儲存貯體委派對跨帳戶存取點的存取權，且儲存貯體或帳戶上未啟用「封鎖公開存取」時，IAM Access Analyzer 會產生公開問題清單。當您啟用「封鎖公開存取」時，公開問題清單得以解決，IAM Access Analyzer 將產生跨帳戶存取點的跨帳戶問題清單。

如需有關 Amazon S3 目錄儲存貯體的詳細資訊，請參閱 Amazon Simple Storage Service User Guide 中的 Working with directory buckets。

AWS Identity and Access Management 角色

對於 IAM 角色，IAM Access Analyzer 會分析信任政策。在角色信任政策中，您會定義您信任能夠擔任角色的主體。角色信任政策是在 IAM 中連接至角色的以資源為基礎的必要政策。IAM Access Analyzer 會產生信任區域內角色的問題清單，該問題清單可供位於信任區域外的外部實體存取。

> **Note**
>
> IAM 角色是全域資源。如果角色信任政策授與外部實體存取權，則 IAM Access Analyzer 會在每個啟用的區域中產生問題清單。

AWS Key Management Service 金鑰

對於 AWS KMS keys，IAM Access Analyzer 會分析套用至金鑰的金鑰政策和授予。如果金鑰政策或授與允許外部實體存取金鑰，IAM Access Analyzer 會產生問題清單。例如，如果您在政策陳述式中使用 kms:CallerAccount 條件索引鍵來允許存取特定 AWS 帳戶中的所有使用者，而且您指定目前帳戶（目前分析器的信任區域）以外的帳戶，IAM Access Analyzer 會產生問題清單。若要進一步了解 IAM 政策陳述式中的 AWS KMS 條件金鑰，請參閱AWS KMS 條件金鑰。

IAM Access Analyzer 分析 KMS 金鑰時，它會讀取金鑰中繼資料，例如金鑰政策和授與清單。如果金鑰政策不允許 IAM Access Analyzer 角色讀取金鑰中繼資料，就會產生 Access Denied (存取遭拒) 的錯誤問題清單。例如，若下列範例政策陳述式是金鑰中套用的唯一政策，則會在 IAM Access Analyzer 中產生 Access denied (存取遭拒) 的錯誤問題清單。

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

由於此陳述式僅允許帳戶 111122223333 中名為 Admin AWS 的角色存取金鑰，因此會產生存取遭拒錯誤清單，因為 IAM Access Analyzer 無法完整分析金鑰。錯誤問題清單在 Findings (問題清單) 資料表中會以紅色文字顯示。問題清單看起來類似下列。

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
```

```
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

當您建立 KMS 金鑰時，所授與可存取金鑰的許可取決於金鑰的建立方式。如果您收到金鑰資源的 Access Denied (存取遭拒) 錯誤問題清單，請將下列政策陳述式套用至金鑰資源，以授與 IAM Access Analyzer 存取金鑰的許可。

```
{
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

在收到 KMS 金鑰資源的 Access Denied (存取遭拒) 問題清單，然後藉由更新金鑰政策來解決問題清單後，即會將問題清單更新為已解決的狀態。如果有會將許可授與外部實體之金鑰的政策陳述式或金鑰授與，您可能會看到金鑰資源的其他問題清單。

AWS Lambda 函數和圖層

對於 AWS Lambda 函數，IAM Access Analyzer 會分析政策，包括政策中的條件陳述式，以將對函數的存取權授予外部實體。使用 Lambda，您可以將唯一的資源型政策連接到函式、版本、別名和分層。IAM Access Analyzer 會根據連接至函式和分層的資源型政策來報告外部存取權。IAM Access Analyzer 不會根據連接至別名的資源型政策和使用合格 ARN 叫用的特定版本來報告外部存取權。

如需詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的針對 Lambda 使用資源型政策以及使用版本。

Amazon Simple Queue Service 佇列

對於 Amazon SQS 佇列，IAM Access Analyzer 會分析允許外部實體存取佇列的政策 (包括政策中的條件陳述式)。

AWS Secrets Manager 秘密

對於 AWS Secrets Manager 秘密，IAM Access Analyzer 會分析允許外部實體存取秘密的政策，包括政策中的條件陳述式。

Amazon Simple Notification Service 主題

IAM Access Analyze 會分析附加至 Amazon SNS 主題的資源型政策，包括允許外部存取主題的政策中的條件陳述式。您可以允許外部帳戶執行 Amazon SNS 動作，例如透過資源型政策訂閱和發佈主題。如果來自信任區域之外的帳戶主體可以對該主題執行操作，則可從外部存取 Amazon SNS 主題。如果您建立 Amazon SNS 主題時在政策中選擇 `Everyone`，則表示該主題可供公眾存取。`AddPermission` 是向允許外部存取的 Amazon SNS 主題新增資源型政策的另一種方法。

Amazon Elastic Block Store 磁碟區快照

Amazon Elastic Block Store 磁碟區快照沒有資源型政策。快照是透過 Amazon EBS 共用許可的方式進行共用。對於 Amazon EBS 磁碟區快照，IAM Access Analyzer 會分析允許外部實體存取快照的存取控制清單。在加密時，Amazon EBS 磁碟區快照可與外部帳戶共用。未加密的磁碟區快照可與外部帳戶共用，並可授予公用存取權。共用設定位於快照的 `CreateVolumePermissions` 屬性中。當客戶預覽 Amazon EBS 快照的外部存取時，他們可以將加密金鑰指定為快照已加密的指標，類似於 IAM Access Analyzer 預覽處理 Secrets Manager 秘密的方式。

Amazon Relational Service 資料庫快照

Amazon RDS 資料庫快照沒有資源型政策。資料庫快照可透過 Amazon RDS 資料庫許可共用，且只能共用手動資料庫快照。對於外部存取分析器，IAM Access Analyzer 會分析允許外部實體存取 Amazon RDS DB 快照的存取控制清單。未加密的資料庫快照可以是公開的。加密的資料庫快照無法公開共用，但可與多達 20 個其他帳戶共用。如需詳細資訊，請參閱建立資料庫快照。IAM Access Analyzer 會將匯出資料庫手動快照 (例如，匯出至 Amazon S3 儲存貯體) 的能力視為受信任的存取。

對於內部存取分析器，IAM Access Analyzer 會在組織或帳戶中的主體 (使用者或角色) 有權存取指定的 Amazon RDS DB 快照時，產生調查結果。

> **Note**
>
> IAM Access Analyzer 不會識別直接在資料庫本身設定的公用或跨帳戶存取權。IAM Access Analyzer 僅識別 Amazon RDS 資料庫快照上設定的公用或跨帳戶存取權的問題清單。

Amazon Relational Database Service 資料庫叢集快照

Amazon RDS 資料庫叢集快照沒有資源型政策。可透過 Amazon RDS 資料庫叢集許可共用快照。對於外部存取分析器，IAM Access Analyzer 會分析允許外部實體存取 Amazon RDS DB 叢集快照的存取控制清單。未加密的叢集快照可以公開。而加密的叢集快照則無法公開共用。未加密和加密的叢集快照可與最多 20 個其他帳戶共享。如需詳細資訊，請參閱建立資料庫叢集快照。IAM Access Analyzer 會將匯出資料庫叢集快照 (例如，匯出至 Amazon S3 儲存貯體) 的能力視為受信任的存取。

對於內部存取分析器，IAM Access Analyzer 會在組織或帳戶中的主體 (使用者或角色) 有權存取指定的 Amazon RDS DB 叢集快照時，產生調查結果。

> **Note**
>
> IAM Access Analyzer 調查結果不包括監控 Amazon RDS 資料庫叢集的任何共用，以及與其他使用 AWS 帳戶 或組織的複製 AWS Resource Access Manager。IAM Access Analyzer 僅識別 Amazon RDS 資料庫叢集快照上設定的公用或跨帳戶存取權的問題清單。

Amazon Elastic Container Registry 儲存庫

對於 Amazon ECR 儲存庫，IAM Access Analyzer 會分析資源型政策，包括政策中的條件陳述式，且這些政策允許外部實體存取儲存庫 (類似於 Amazon SNS 主題和 Amazon EFS 檔案系統等其他資源類型)。對於 Amazon ECR 儲存庫，主體必須擁有透過身分型政策 `ecr:GetAuthorizationToken` 的許可，才能將其視為外部可用。

Amazon Elastic File System 檔案系統

對於 Amazon EFS 檔案系統，IAM Access Analyzer 會分析允許外部實體存取檔案系統的政策 (包括政策中的條件陳述式)。如果來自信任區域之外的帳戶主體可以在該檔案系統上執行操作，則可從外部存取 Amazon EFS 檔案系統。可透過使用 IAM 的檔案系統政策以及檔案系統掛載方式來定義存取權。例如，在其他帳戶掛載 Amazon EFS 檔案系統會被視為可從外部存取，除非該帳戶位於您的組織中，且您已將該組織定義為您的信任區域。如果您從具有公有子網路的虛擬私有雲端掛載檔案系統，則可從外部存取檔案系統。當您搭配 使用 Amazon EFS 時 AWS Transfer Family，如果檔案系統允許公開存取，則會封鎖從與檔案系統不同的帳戶所擁有的 Transfer Family 伺服器所收到的檔案系統存取請求。

Amazon DynamoDB Streams

對於外部存取分析器，如果 DynamoDB 政策允許至少一個跨帳戶動作，讓外部實體可以存取 DynamoDB 串流，則 IAM Access Analyzer 會產生調查結果。如需 DynamoDB 支援的跨帳戶動作的

詳細資訊，請參閱《Amazon DynamoDB 開發人員指南》中的 IAM actions supported by resource-based policies。

對於內部存取分析器，IAM Access Analyzer 會在組織或帳戶中的主體 (使用者或角色) 有權存取指定的 DynamoDB 串流時，產生調查結果。

Amazon DynamoDB 資料表

對於外部存取分析器，如果 DynamoDB 政策允許至少一個跨帳戶動作，讓外部實體可以存取 DynamoDB 資料表或索引，則 IAM Access Analyzer 會產生 DynamoDB 資料表調查結果。如需 DynamoDB 支援的跨帳戶動作的詳細資訊，請參閱《Amazon DynamoDB 開發人員指南》中的 IAM actions supported by resource-based policies。

對於內部存取分析器，IAM Access Analyzer 會在組織或帳戶中的主體 (使用者或角色) 有權存取指定的 DynamoDB 資料表時產生調查結果。

IAM Access Analyzer 的委派管理員

如果您在 AWS Identity and Access Management Access Analyzer AWS Organizations 管理帳戶中設定 ，您可以將組織中的成員帳戶新增為委派管理員，以管理組織的 IAM Access Analyzer。委派管理員擁有許可，有權建立及管理以組織做為信任區域的分析器。只有管理帳戶可以新增委派管理員。

IAM Access Analyzer 的委派管理員是組織內擁有許可的成員帳戶，有權建立及管理為整個組織分析存取權的分析器。只有管理帳戶可以新增、移除或變更委派管理員。

如果您新增委派管理員，您可以在稍後將委派管理員變更為不同帳戶。當您這麼做時，先前的委派管理員帳戶會失去許可，無法再管理使用該帳戶建立來為整個組織分析存取權的所有分析器。這類分析器會進入已停用狀態，且不會再產生新的問題清單或更新現有的問題清單。您也無法無法再存取這類分析器的現有問題清單。您未來可透過將該帳戶配置為委派管理員，而再次存取那些問題清單。如果您知道您不會使用該相同的帳戶作為委派管理員，請考慮在變更委派管理員之前刪除分析器。這會刪除所有產生的問題清單。當新的委派管理員建立新的分析器時，則會產生相同問題清單的新執行個體。你不會遺失任何問題清單，它們只是在不同的帳戶中為新的分析器而產生。而且您可以使用組織管理帳戶 (該帳戶也具有管理員許可) 繼續存取組織的問題清單。新委派的管理員必須為 IAM Access Analyzer 建立新的分析器，才能開始在組織中監控資源。

如果委派管理員離開 AWS 組織，委派的管理權限會從帳戶中移除。帳戶中所有以組織做為信任區域的分析器都會進入已停用狀態，您也無法無法再存取這類分析器的現有問題清單。

第一次在管理帳戶中設定分析器時，您可以在 IAM Access Analyzer 主控台的分析器設定頁面中，選擇新增委派管理員。

> **Note**
>
> IAM Access Analyzer 會根據每月每個分析器所分析的 IAM 角色和使用者數量，收取未使用的存取權分析器費用。如果您在管理帳戶和委派管理員帳戶中都建立了未使用的存取權分析器，將需支付兩個未使用的存取權分析器的費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

變更委派管理員後，新的管理員必須建立分析器，以開始在組織中監控資源的存取權。

新增 IAM Access Analyzer 的委派管理員

如果您在 AWS Identity and Access Management Access Analyzer AWS Organizations 管理帳戶中設定 ，您可以將組織中的成員帳戶新增為委派管理員，以管理組織的 IAM Access Analyzer。委派管理員擁有許可，有權建立及管理以組織做為信任區域的分析器。只有管理帳戶可以新增委派管理員。

使用主控台新增委派管理員

1. 使用組織的管理帳戶登入 AWS 主控台。
2. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
3. 在 Access Analyzer 下，選擇分析器設定。
4. 選擇 Add delegated administrator (新增委派管理員)。
5. 在委派管理員欄位中，輸入組織成員帳戶的 AWS 帳戶 號碼以設為委派管理員。

   此帳戶必須是您組織的成員。
6. 選擇儲存變更。

使用 AWS CLI 或 AWS SDKs 新增委派管理員

當您建立分析器以使用 AWS CLI、 AWS API （ 使用 AWS SDKs) 分析委派管理員帳戶中整個組織的存取權 CloudFormation，或者，您必須使用 AWS Organizations APIs為 IAM Access Analyzer 啟用服務存取權，並將成員帳戶註冊為委派管理員。

1. 在 中啟用 IAM Access Analyzer 的受信任服務存取 AWS Organizations。請參閱 AWS Organizations 《 使用者指南》中的如何啟用或停用受信任存取。
2. 使用 API 操作或 `register-delegated-administrator` AWS CLI 命令，將 AWS AWS Organizations `RegisterDelegatedAdministrator`組織的有效成員帳戶註冊為委派管理員。

封存規則

封存規則會自動封存符合您在建立規則時所定義條件的新問題清單。您也可以追溯套用封存規則，來封存符合封存規則條件的現有問題清單。例如，您可以建立封存規則，以針對您定期授予存取權的特定 Amazon S3 儲存貯體，自動封存任何調查結果。或者，如果您將多個資源的存取權授與特定主體，則可以建立規則，自動存檔任何新問題清單，這些問題清單都是為授與該主體的存取權而產生。這可讓您僅專注於可能表示安全風險的作用中問題清單。

當您建立封存規則時，只會自動封存符合規則條件的新調查結果。系統不會自動存檔現有的問題清單。建立規則時，您可以在規則中為每個條件包含最多 20 個值。如需可用來建立或更新存檔規則的篩選鍵清單，請參閱 IAM Access Analyzer 篩選鍵。

> **Note**
>
> 建立或編輯封存規則時，IAM Access Analyzer 不會驗證您在規則篩選條件中包含的值。例如，如果您新增規則以比對 AWS 帳戶，即使該值不是有效的 AWS 帳戶號碼，IAM Access Analyzer 仍會接受欄位中的任何值。

建立存檔規則

1. 前往網址 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇 Access Analyzer，然後選擇分析器設定。
3. 在分析器區段中，選擇您要建立封存規則的分析器。
4. 在封存規則索引標籤上，選擇建立封存規則。
5. 如果要變更預設名稱，請輸入規則的名稱。
6. Rule (規則) 區段中 Criteria (條件) 下，選取要與規則比對的屬性。
7. 選擇屬性值的條件，例如包含、是或不等於。

   可用的運算子取決於您選擇的屬性。
8. 為屬性新增其他值，或為規則新增其他條件 (非必要)。針對外部存取權調查結果，若要確保規則不會封存公開存取權的新調查結果，您也可以包含公開存取權條件並設為 false。

   若要為條件新增另一個值，請選擇 Add another value (新增另一個值)。若要為規則新增其他條件，請選擇新增。
9. 當您完成新增條件和值時，請選擇 Create rule (建立規則)，以將規則僅套用至新的問題清單。選擇 Create and archive active findings (建立並封存作用中的問題清單)，以根據規則條件來封存新

的和現有的問題清單。在 Results (結果) 區段中，您可以檢閱封存規則所套用之作用中問題清單的清單。

例如，若要建立為 Amazon S3 儲存貯體自動封存所有調查結果的規則：選擇資源類型，然後選擇是 條件。接下來，從值清單中選擇 S3 儲存貯體。

若要針對未使用的存取權調查結果建立規則，以自動封存特定帳戶的所有調查結果，請選擇資源擁有者帳戶，然後選擇等於條件。在值文字方塊中輸入 AWS 帳戶 ID。

繼續定義條件以自訂適合您環境的規則，然後選擇建立規則。

如果您要建立新規則並新增多個條件，您可以選擇移除此條件，從規則中移除單一條件。您可以透過選擇 Remove value (移除值) 來移除為條件新增的值。

編輯存檔規則

1. 在名稱欄中選擇要編輯的規則之名稱。

   您一次只能編輯一個存檔規則。
2. 為每個條件新增準則或移除現有準則和值。
3. 選擇 Save changes (儲存變更)，將規則僅套用至新的問題清單。選擇 Save and archive active findings (儲存並封存作用中的問題清單)，以根據規則條件來封存新的和現有的問題清單。

刪除存檔規則

1. 選擇您要刪除的規則之核取方塊。
2. 選擇 刪除。
3. 在 Delete archive rule (刪除存檔規則) 確認對話方塊中輸入 **`delete`**，然後選擇 Delete (刪除)。

系統僅會在目前區域中將這些規則從分析器中刪除。您必須為您在其他區域中建立的每個分析器分別刪除存檔規則。

AWS Identity and Access Management Access Analyzer 使用 Amazon EventBridge 進行監控

使用本主題中的資訊來了解如何使用 Amazon EventBridge 監控 IAM Access Analyzer 問題清單及存取預覽。EventBridge 是 Amazon CloudWatch Events 的新版本。

問題清單事件

IAM Access Analyzer 會為每個產生的問題清單、對現有問題清單的狀態所做的變更，以及在刪除問題清單時，將事件傳送至 EventBridge。若要接收問題清單的相關問題清單和通知，您必須在 Amazon EventBridge 中建立事件規則。建立事件規則時，您也可以根據規則指定要觸發的目標動作。例如，您可以建立事件規則，以便在從 IAM Access Analyzer 收到新問題清單的事件時觸發 Amazon SNS 主題。有關資源控制政策 (RCP) 的詳細資訊，請參閱事件詳細資訊一節。

存取預覽事件

IAM Access Analyzer 會將事件傳送到每個存取預覽的 EventBridge 並變更為其狀態。這包括第一次建立存取預覽 (狀態為「建立」)、存取預覽完成 (狀態為「已完成」) 或存取預覽建立失敗 (狀態為「失敗」) 時的事件。若要接收存取預覽的相關通知，您必須在 EventBridge 中建立事件規則。建立事件規則時，您可以根據規則指定要觸發的目標動作。例如，您可以建立事件規則，以便在從 IAM Access Analyzer 收到的已完成存取預覽時觸發 Amazon SNS 主題。

事件通知頻率

IAM Access Analyzer 會在您帳戶中發生事件後約一小時內，將新問題清單和包含狀態更新問題清單的事件傳送給 EventBridge。IAM Access Analyzer 也會在刪除已解析的問題清單時將事件傳送給 EventBridge，因為保留期間已過期。對於因產生問題清單的分析器遭到刪除而刪除的問題清單，系統會在分析器遭到刪除後約 24 小時將事件傳送到 EventBridge。問題清單遭刪除時，問題清單的狀態不會變更。相反地，`isDeleted` 屬性會設定為 `true`。IAM Access Analyzer 也會為新建立的存取預覽傳送事件，以及對 EventBridge 的存取預覽狀態變更。

外部存取權調查結果事件範例

以下為傳送至 EventBridge 的 IAM Access Analyzer 外部存取權調查結果事件範例。所列的 `id` 是 EventBridge 中事件的 ID。若要進一步了解，請參閱 EventBridge 中的事件和事件模式。

在 `detail` 物件中，`accountId` 和 `region` 屬性的值參考結果所報告的帳戶和區域。`isDeleted` 屬性會指出事件是否來自要刪除的問題清單。`id` 是結果 ID。`resources` 陣列是具有產生結果的分析器 ARN 的單例。

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
```

```
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
            "AWS": "999988887777"
        },
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

IAM Access Analyzer 也會將事件傳送至錯誤問題清單的 EventBridge。錯誤問題清單是 IAM Access Analyzer 無法分析資源時產生的問題清單。錯誤問題清單的事件包括下列範例所示的 `error` 屬性。

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

內部存取調查結果事件範例

以下為傳送至 EventBridge 的 IAM Access Analyzer 內部存取調查結果事件範例。所列的 `id` 是 EventBridge 中事件的 ID。若要進一步了解，請參閱 EventBridge 中的事件和事件模式。

在 `detail` 物件中，`accountId` 和 `principalOwnerAccount` 屬性的值指調查結果中報告的主體的帳戶。`isDeleted` 屬性會指出事件是否來自要刪除的問題清單。`id` 是結果 ID。`resource` 是產生調查結果的分析器的 ARN。

```
{
    "version": "0",
    "id": "b45c3678-c278-b593-6121-c155259ce1b5",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2025-04-08T19:42:49Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:access-analyzer:us-east-1:111122223333:analyzer/testAnalyzer"
    ],
    "detail": {
        "accessType": "INTRA_ACCOUNT",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2025-04-08T03:18:43.509465073Z",
        "condition": {},
```

```
        "createdAt": "2025-04-07T21:33:49.914099224Z",
        "id": "11111111-2222-4444-aaaa-333333333333",
        "isDeleted": false,
        "findingType": "InternalAccess",
        "principal": {
            "AWS": "arn:aws:iam::111122223333:role/MyRole_6"
        },
        "principalOwnerAccount": "111122223333",
        "principalType": "IAM_ROLE",
        "resource": "arn:aws:s3:::critical-data",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "accountId": "111122223333",
        "resourceType": "AWS::S3::Bucket",
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "status": "ACTIVE",
        "updatedAt": "2025-04-08T03:22:12.654688231Z",
        "version": "1.0"
    }
}
```


IAM Access Analyzer 也會將事件傳送至錯誤問題清單的 EventBridge。錯誤問題清單是 IAM Access Analyzer 無法分析資源時產生的問題清單。錯誤問題清單的事件包括下列範例所示的 `error` 屬性。


```
{
    "version": "0",
    "id": "5a94b99b-e87d-a6a7-58c7-f47871532860",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer-test",
    "account": "444455556666",
    "time": "2025-05-07T11:57:54Z",
    "region": "us-west-2",
    "resources": ["arn:aws:access-analyzer-beta:us-west-2:444455556666:analyzer/
example-analyzer"],
    "detail": {
        "analyzedAt": "2025-03-24T19:58:52.512329448Z",
        "createdAt": "2025-03-22T03:30:46.920200692Z",
        "id": "ef573afd-12a5-4095-87a6-bf2f25109895",
        "isDeleted": false,
        "findingType": "InternalAccess",
        "resource": "arn:aws:s3:::test-entity-88",
        "accountId": "111122223333",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "resourceType": "AWS::S3::Bucket",
```

```
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "error": "ACCESS_DENIED", // can be INTERNAL_ERROR and ACCESS_DENIED
        "status": "ACTIVE",
        "updatedAt": "2025-03-24T20:09:39.176075014Z",
        "version": "1.0"
    }
}
```

未使用的存取權調查結果相關事件範例

以下為傳送至 EventBridge 的 IAM Access Analyzer 未使用的存取權調查結果事件範例。所列的 `id` 是 EventBridge 中事件的 ID。若要進一步了解，請參閱 EventBridge 中的事件和事件模式。

在 `detail` 物件中，`accountId` 和 `region` 屬性的值參考結果所報告的帳戶和區域。`isDeleted` 屬性會指出事件是否來自要刪除的問題清單。`id` 是結果 ID。

```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
```

```
        }
    }
```

IAM Access Analyzer 也會將事件傳送至錯誤問題清單的 EventBridge。錯誤問題清單是 IAM Access Analyzer 無法分析資源時產生的問題清單。錯誤問題清單的事件包括下列範例所示的 `error` 屬性。

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
  }
```

存取預覽事件範例

下列範例會顯示您建立存取預覽時傳送至 EventBridge 第一個事件的資料。`resources` 陣列是存取預覽相關聯的分析器 ARN 的單例。在 `detail` 物件中，`id` 指的是存取預覽 ID，而 `configuredResources` 是指為其建立存取預覽的資源。`status` 為 `Creating`，並且是指存取預覽狀態。`previousStatus` 未指定，因為剛剛建立存取預覽。

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

下列範例顯示針對狀態從 Creating 變更為 Completed 的存取預覽傳送至 EventBridge 的事件資料。在詳細資料物件中，id 是指存取預覽 ID。status 和 previousStatus 是指存取預覽狀態，其中先前的狀態為 Creating，而目前的狀態為 Completed。

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
```

```
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

下列範例顯示針對狀態從 Creating 變更為 Failed 的存取預覽傳送至 EventBridge 的事件資料。在 detail 物件中，id 是指存取預覽 ID。status 和 previousStatus 是指存取預覽狀態，其中先前的狀態為 Creating，而目前的狀態為 Failed。statusReason 欄位會提供原因代碼，指出存取預覽因為無效的資源組態而失敗。

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

使用主控台建立事件規則

下列程序說明如何使用主控台建立事件規則。

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。
2. 使用下列值，建立 EventBridge 規則，以監視問題清單事件或存取預覽事件：
   - 針對規則類型，選擇具有事件模式的規則。
   - 在 Event source (事件來源) 中，選擇 Other (其他)。
   - 在 Event pattern (事件模式) 中，選擇 Custom patterns (JSON editor) (自訂模式 (JSON 編輯器))，並將下列事件模式範例貼到文字區域：
     - 若要根據任意 IAM Access Analyzer 事件建立規則，請使用下列模式範例：

```
{
  "source": [
    "aws.access-analyzer"
  ]
}
```

     - 若要根據外部存取、內部存取或未使用的存取調查結果事件建立規則，請依照下列模式範例操作：

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding",
    "Internal Access Finding",
    "Unused Access Finding for IAM entities"
  ]
}
```

     - 若要僅根據外部存取權調查結果事件建立規則，請使用下列模式範例：

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding"
```

```
  ]
}
```

- 若要僅根據內部存取調查結果事件建立規則，請依照下列模式範例操作：

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Internal Access Finding"
  ]
}
```

- 若要僅根據未使用的存取權調查結果事件建立規則，請使用下列模式範例：

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

- 若要根據存取預覽事件建立規則，請使用下列模式範例：

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- 對於目標類型，請選擇AWS 服務，對於選取目標，選擇目標，例如 Amazon SNS 主題或 AWS Lambda 函數。當接收到符合規則中定義之事件模式的事件時，就會觸發目標。

如需有關建立規則的詳細資訊，請參閱 Amazon EventBridge User Guide 中的 Creating Amazon EventBridge rules that react to events。

使用 CLI 建立事件規則

1. 使用下列項目來使用 AWS CLI為 Amazon EventBridge 建立規則。將規則名稱 *TestRule* 取代為您的規則名稱。

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"]}"
```

2. 您可以自訂規則，只針對產生的一部分問題清單 (例如具有特定屬性的問題清單) 觸發目標動作。下列範例示範如何建立規則，該規則會僅針對狀態為作用中的問題清單而觸發目標動作。

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":
[\"ACTIVE\"]}}"
```

   下列範例示範如何建立規則，該規則會僅針對狀態從 `Creating` 變更為 `Completed` 的存取預覽而觸發目標動作。

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. 若要將 Lambda 函數定義為您建立之規則的目標，請使用下列範例命令。將 ARN 中的區域和函數名稱取代為適合您環境的值。

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. 新增叫用規則目標所需的許可。下列範例示範在遵循上述範例後如何將許可授與 Lambda 函數。

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

將 IAM Access Analyzer 與 整合 AWS Security Hub CSPM

AWS Security Hub CSPM 可用於全面檢視跨 AWS的安全狀態。它可協助您評估環境是否符合安全業界標準和最佳實務。Security Hub CSPM 會從跨 AWS 帳戶、 服務和支援的第三方合作夥伴產品收集安全資料。然後，它會分析您的安全趨勢，並識別最高優先順序的安全問題。

當您將 IAM Access Analyzer 與 Security Hub CSPM 整合時，您可以將問題清單從 IAM Access Analyzer 傳送至 Security Hub CSPM。然後，Security Hub CSPM 可以在分析整體安全狀態時包含這些調查結果。

內容

- IAM Access Analyzer 如何將調查結果傳送至 Security Hub CSPM
  - IAM Access Analyzer 傳送的調查結果類型
  - 傳送問題清單延遲
  - 無法使用 Security Hub CSPM 時重試
  - 更新 Security Hub CSPM 中的現有調查結果
- 在 Security Hub CSPM 中檢視 IAM Access Analyzer 調查結果
  - 解譯 Security Hub CSPM 中的 IAM Access Analyzer 調查結果名稱
- IAM Access Analyzer 的典型調查結果
- 啟用與設定整合
- 如何停止傳送問題清單

IAM Access Analyzer 如何將調查結果傳送至 Security Hub CSPM

在 Security Hub CSPM 中，將安全問題做為調查結果進行追蹤。有些問題清單來自其他 AWS 服務 或第三方合作夥伴偵測到的問題。Security Hub CSPM 也有一組規則，可用來偵測安全問題並產生問題清單。

Security Hub CSPM 提供用來跨所有這些來源管理調查結果的工具。您可以檢視並篩選調查結果列表，並檢視每項調查結果的相關詳細資訊。如需詳細資訊，請參閱《AWS Security Hub 使用者指南》中的檢視問題清單。您也可以追蹤調查結果的調查狀態。如需詳細資訊，請參閱《AWS Security Hub 使用者指南》中的針對問題清單採取動作。

Security Hub CSPM 中的所有問題清單都使用稱為 AWS 安全問題清單格式 (ASFF) 的標準 JSON 格式。ASFF 包含問題來源、受影響的資源以及問題清單目前狀態的詳細資訊。請參閱《AWS Security Hub 使用者指南》中的 AWS 安全問題清單格式 (ASFF)。

AWS Identity and Access Management Access Analyzer 是將問題清單傳送至 Security Hub CSPM AWS 服務 的 之一。針對未使用的存取權，IAM Access Analyzer 會偵測授予 IAM 使用者或角色的未使用存取權，並為其中每一項產生一個調查結果。然後，IAM Access Analyzer 會將這些調查結果傳送至 Security Hub CSPM。

針對外部存取權，IAM Access Analyzer 會偵測授予外部主體對您組織或帳戶中支援的資源的公開存取權或跨帳戶存取權的政策陳述式。IAM Access Analyzer 會產生公開存取的問題清單，並將其傳送至 Security Hub CSPM。對於跨帳戶存取，IAM Access Analyzer 會一次傳送一個外部主體的單一調查結果給 Security Hub CSPM。如果 IAM Access Analyzer 中有多個跨帳戶調查結果，您必須先解決單一外部主體的 Security Hub CSPM 調查結果，IAM Access Analyzer 才會提供下一個跨帳戶調查結果。如需分析器信任區域外具有跨帳戶存取權的外部主體完整清單，您必須檢視 IAM Access Analyzer 中的調查結果。有關資源控制政策 (RCP) 的詳細資訊，請參閱資源詳細資訊一節。

IAM Access Analyzer 傳送的調查結果類型

IAM Access Analyzer 會使用安全調查結果AWS 格式 (ASFF) 將調查結果傳送至 Security Hub CSPM。在 ASFF 中，Types 欄位提供問題清單類型。來自 IAM Access Analyzer 的調查結果可能具有以下 Types 值。

- 外部存取權調查結果：效果/資料暴露/授予的外部存取權
- 外部存取調查結果 – 軟體和組態檢查/AWS 安全性最佳實務/授予外部存取
- 未使用的存取調查結果 – 軟體和組態檢查/AWS 安全性最佳實務/未使用的許可
- 未使用的存取調查結果 – 軟體和組態檢查/AWS 安全性最佳實務/未使用的 IAM 角色
- 未使用的存取調查結果 – 軟體和組態檢查/AWS 安全性最佳實務/未使用的 IAM 使用者密碼
- 未使用的存取調查結果 – 軟體和組態檢查/AWS 安全性最佳實務/未使用的 IAM 使用者存取金鑰

傳送問題清單延遲

當 IAM Access Analyzer 建立新的問題清單時，通常會在 30 分鐘內傳送至 Security Hub CSPM。不過，在極少數情況下，IAM Access Analyzer 可能不會收到關於政策變更的通知。例如：

- 對 Amazon S3 帳戶層級封鎖公開存取權設定所做的變更，最多可能需要 12 小時才會反映在 IAM Access Analyzer 中。
- 對資源控制政策 (RCP) 所做的變更，若未變更資源型政策，則不會觸發重新掃描調查結果中報告的資源。IAM Access Analyzer 會在下次定期掃描 (24 小時內) 期間分析該新增或更新的政策。
- 如果 AWS CloudTrail 日誌交付發生交付問題，政策變更可能不會觸發重新掃描調查結果中報告的資源。

在這類情況下，IAM Access Analyzer 會在下次定期掃描期間分析該新增或更新的政策。

無法使用 Security Hub CSPM 時重試

如果 Security Hub CSPM 無法使用，IAM Access Analyzer 會定期重試傳送問題清單。

更新 Security Hub CSPM 中的現有調查結果

將調查結果傳送至 Security Hub CSPM 後，IAM Access Analyzer 會繼續傳送更新，以反映對 Security Hub CSPM 調查結果活動的任何其他觀察。這些更新會反映在同一個調查結果中。

對於外部存取權調查結果，IAM Access Analyzer 會為每個資源分組這些調查結果。在 Security Hub CSPM 中，如果該資源的至少一個問題清單在 IAM Access Analyzer 中處於作用中狀態，則資源的問題清單會保持作用中狀態。如果資源的 IAM Access Analyzer 中的所有問題清單都已封存或解析，則 Security Hub CSPM 問題清單也會封存。當政策存取在公有和跨帳戶存取之間變更時，Security Hub CSPM 調查結果會更新。此更新可以包含對問題清單類型、標題、描述和嚴重性的變更。

對於未使用的存取權調查結果，IAM Access Analyzer 不會依資源進行分組。相反地，如果在 IAM Access Analyzer 中解析未使用的存取調查結果，則對應的 Security Hub CSPM 調查結果也會解析。當您更新產生未使用存取調查結果的 IAM 使用者或角色時，Security Hub CSPM 調查結果會更新。

在 Security Hub CSPM 中檢視 IAM Access Analyzer 調查結果

若要在 Security Hub CSPM 中檢視 IAM Access Analyzer 調查結果，請選擇摘要頁面的 ：IAM Access Analyzer 區段中的調查結果。 AWS或者，您可以從導覽面板中選擇 Findings (問題清單)。然後，您可以選擇 產品名稱： 欄位的值為 ，篩選問題清單，以僅顯示 AWS Identity and Access Management Access Analyzer 問題清單**`IAM Access Analyzer`**。

解譯 Security Hub CSPM 中的 IAM Access Analyzer 調查結果名稱

AWS Identity and Access Management Access Analyzer 會使用安全調查結果格式 (ASFF) 將調查結果傳送至 AWS Security Hub CSPM。在 ASFF 中，Types (類型) 欄位提供問題清單類型。ASFF 類型使用的命名機制與 不同 AWS Identity and Access Management Access Analyzer。下表包含與 AWS Identity and Access Management Access Analyzer 問題清單相關聯的所有 ASFF 類型的詳細資訊，這些類型會顯示在 Security Hub CSPM 中。

ASFF 問題清單類型	Security Hub CSPM 問題清單標題	Description
效果/資料曝光/已授予的外部存取	<resource ARN> 允許公有存取	連接至資源的以資源為基礎的政策對所有外部主體允許資源的公有存取。

ASFF 問題清單類型	Security Hub CSPM 問題清單標題	Description
軟體和組態檢查/AWS 安全性最佳實務/授予外部存取	<resource ARN> 允許跨帳戶存取	連接至資源的以資源為基礎的政策對分析器信任區域外的外部主體允許跨帳戶存取。
軟體和組態檢查/AWS 安全性最佳實務/未使用的許可	<資源 ARN> 包含未使用的許可	使用者或角色包含未使用的服務和動作許可。
軟體和組態檢查/AWS 安全性最佳實務/未使用的 IAM 角色	<資源 ARN> 包含未使用的 IAM 角色	使用者或角色包含未使用的 IAM 角色。
軟體和組態檢查/AWS 安全性最佳實務/未使用的 IAM 使用者密碼	<資源 ARN> 包含未使用的 IAM 使用者密碼	使用者或角色包含未使用的 IAM 使用者密碼。
軟體和組態檢查/AWS 安全性最佳實務/未使用的 IAM 使用者存取金鑰	<資源 ARN> 包含未使用的 IAM 使用者存取金鑰	使用者或角色包含未使用的 IAM 使用者存取金鑰。

IAM Access Analyzer 的典型調查結果

IAM Access Analyzer 會使用安全調查結果AWS 格式 (ASFF) 將調查結果傳送至 Security Hub CSPM。

這是來自 IAM Access Analyzer 的典型外部存取權調查結果範例。

```
{
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::amzn-s3-demo-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
```

```
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-account
 access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-
account access from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
    "Resources": [
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
    "RecordState": "ACTIVE"
}
```

這是來自 IAM Access Analyzer 的典型未使用的存取權調查結果範例。

```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
```

```
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren’t required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
      "ProductFields": {
      "numberOfUnusedActions": "256",
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
```

```
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
    "Types": [
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```

啟用與設定整合

若要使用與 Security Hub CSPM 的整合，您必須啟用 Security Hub CSPM。如需如何啟用 Security Hub CSPM 的資訊，請參閱AWS Security Hub 《 使用者指南》中的設定 Security Hub。

當您同時啟用 IAM Access Analyzer 和 Security Hub CSPM 時，會自動啟用整合。IAM Access Analyzer 會立即開始將問題清單傳送至 Security Hub CSPM。

如何停止傳送問題清單

若要停止將調查結果傳送至 Security Hub CSPM，您可以使用 Security Hub CSPM 主控台或 API。

請參閱 AWS Security Hub 使用者指南 中的停用和啟用整合中的問題清單流程 (主控台) 或停用來自整合的問題清單流程 (Security Hub API、 AWS CLI)。

使用 記錄 IAM Access Analyzer API 呼叫 AWS CloudTrail

IAM Access Analyzer 已與 服務整合 AWS CloudTrail，此服務可提供使用者、角色或 AWS 服務在 IAM Access Analyzer 中採取之動作的記錄。CloudTrail 會將 IAM Access Analyzer 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自 IAM Access Analyzer 主控台的呼叫，以及對 IAM Access Analyzer API 操作進行的程式碼呼叫。

若您建立追蹤，便可將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 IAM Access Analyzer 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。

您可以利用 CloudTrail 所收集的資訊來判斷向 IAM Access Analyzer 發出的請求，以及發出請求的 IP 地址、人員、時間和其他詳細資訊。

若要進一步了解 CloudTrail，請參閱「AWS CloudTrail 使用者指南」。

CloudTrail 中的 IAM Access Analyzer 資訊

當您建立 AWS 帳戶時，會在您的帳戶上啟用 CloudTrail。當 IAM Access Analyzer 中發生活動時，該活動會記錄在 CloudTrail 事件中，其他 AWS 服務事件則記錄於 Event history (事件歷史記錄) 中。您可以在 AWS 帳戶中檢視、搜尋和下載最近的事件。如需詳細資訊，請參閱《使用 CloudTrail 事件歷史記錄檢視事件》https://docs.aws.amazon.com/awscloudtrail/latest/userguide/view-cloudtrail-events.html。

若要持續記錄您 AWS 帳戶中的事件，包括 IAM Access Analyzer 的事件，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。根據預設，當您在主控台中建立線索時，線索會套用至所有 AWS 區域。線索會記錄 AWS 分割區中所有區域的事件，並將日誌檔案傳送到您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析和處理 CloudTrail 日誌中收集的事件資料。如需詳細資訊，請參閱下列內容：

- 建立追蹤的概觀
- CloudTrail 支援的服務和整合
- 設定 CloudTrail 的 Amazon SNS 通知
- 從多個區域接收 CloudTrail 日誌檔案，以及從多個帳戶接收 CloudTrail 日誌檔案

CloudTrail 會記錄所有 IAM Access Analyzer 動作，並記錄在 IAM Access Analyzer API 參考中。例如，對 `CreateAnalyzer`、`CreateArchiveRule` 和 `ListFindings` 動作發出的呼叫會在 CloudTrail 記錄檔案中產生專案。

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 是否使用根或 AWS Identity and Access Management (IAM) 使用者登入資料提出請求。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 請求是否由其他 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

了解 IAM Access Analyzer 日誌檔案項目

追蹤是一種組態，能讓事件以日誌檔案的形式交付到您指定的 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含一或多個日誌專案。一個事件為任何來源提出的單一請求，並包含請求動作、請求的日期和時間、請求參數等資訊。CloudTrail 日誌檔並非依公有 API 呼叫的堆疊追蹤排序，因此不會以任何特定順序出現。

下列範例顯示的 CloudTrail 日誌項目會示範「2021 年 6 月 14 日」名為 `Alice-tempcreds` 的擔任角色工作階段所執行的 `CreateAnalyzer` 操作。角色工作階段是由名為 `admin-tempcreds` 的角色發行。

```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
  "eventTime": "2021-06-14T22:57:36Z",
```

```
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
            "tagkey1": "tagvalue1"
        }
  },
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

IAM Access Analyzer 篩選鍵

您可以使用下列篩選鍵來定義封存規則 (CreateArchiveRule)、更新封存規則 (UpdateArchiveRule)、擷取調查結果列表 (ListFindings 和 ListFindingsV2) 或擷取資源的存取預覽調查結果列表 (ListAccessPreviewFindings)。使用 IAM API 和 CloudFormation 設定封存規則之間沒有差異。

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
資源	Resource	ARN 唯一識別外部主體可存取的	String	✓ 是	✓ 是	✓ 是	外部

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
		資源。若要進一步了解，請參閱 Amazon 資源名稱 (ARN)。					內部 (Internal) 未使用的

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
resourceType AWS::S3 Bucket | AWS::IAM :Role | AWS::SQS :Queue | AWS::Lambda::Function | AWS::Lambda::LayerVersion |AWS::KMS :Key | AWS::SecretsManager::Secret | AWS::EFS :FileSystem | AWS::EC2 :Snapshot | AWS::ECR :Repositi	資源類型	外部主體可存取的資源類型。 **Note** 內部存取分析器不支援外部存取分析器支援的所有資源類型。未使用的存取權分析器僅支援 IAM 使用者和角色。如需詳細資訊，請參閱IAM Access Analyzer 外部和內部存取支援的資源類型。	String	是	是	是	外部 內部 (Internal) 未使用的

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
ry | AWS::RDS :DBSnapshot | AWS::RDS :DBClusterSnapsho | AWS::SNS :Topic | AWS::S3Express::DirectoryBucket | AWS::DynamoDB::Table | AWS::DynamoDB::Stream | AWS::IAM :User							
resourceOwnerAccount	資源擁有者帳戶	擁有資源的 12 位數 AWS 帳戶 ID。若要進一步了解，請參閱 AWS 帳戶識別碼。	String	✓ 是	✓ 是	✓ 是	外部 內部 (Internal) 未使用的

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
isPublic	公用存取	指出問題清單是否回報一具有允許公有存取之政策的資源。	Boolean	✓ 是	✓ 是	✓ 是	外部
findingType ExternalAccess | UnusedIAMRole | UnusedIAMUserAccessKey | UnusedIAMUserPassword | UnusedPermission | InternalAccess	調查結果類型	問題清單的類型。對於外部存取分析器，類型為 ExternalAccess 。對於未使用的存取分析器，類型可以是 UnusedIAMRole 、UnusedIAMUserAccessKey 、UnusedIAMUserPassword 或 UnusedPermission 。對於內部存取分析器，類型為 InternalAccess 。	String	✓ 是	✓ 是	✓ 是	外部 內部 (Internal) 未使用的

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
resourceControlPolicyRestriction APPLIED | APPLICABLE | FAILED_TO_EVALUATE_RCP | NOT_APPLICABLE	資源控制政策 (RCP) 限制	資源擁有者使用 Organizations 資源控制政策 (RCP) 套用的限制類型。如需有關此篩選金鑰的值的詳細資訊，請參閱 IAM Access Analyzer API Reference 中的 ExternalAccessDetails 和 InternalAccessDetails。	String	✓ 是	✓ 是	✓ 是	外部 內部 (Internal)
serviceControlPolicyRestriction APPLIED | APPLICABLE | FAILED_TO_EVALUATE_SCP | NOT_APPLICABLE	服務控制政策 (SCP) 限制	Organizations 服務控制政策 (SCP) 套用的限制類型。如需有關此篩選金鑰的值的詳細資訊，請參閱 IAM Access Analyzer API Reference 中的 InternalAccessDetails。	String	✓ 是	✓ 是	✓ 是	內部 (Internal)

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
status `ACTIVE` | `ARCHIVED` | `RESOLVED`	狀態	問題清單的目前狀態。	String	⊗ 否	⊙ 是	⊙ 是	外部 內部 (Internal) 未使用的
error	錯誤	指出針對問題清單所報告的錯誤。	String	⊙ 是	⊙ 是	⊙ 是	外部 內部 (Internal)
principal.AWS	AWS 帳戶	已授與問題清單 `Principal` 欄位中資源存取的帳戶。輸入 12 位數 AWS 的帳戶 ID 或外部 AWS 使用者或角色的 ARN。若要進一步了解，請參閱 AWS 帳戶識別碼。	String	⊙ 是	⊙ 是	⊙ 是	外部
principal.Federated	聯合身分使用者	可存取問題清單中資源的聯合身分 ARN。若要進一步了解，請參閱身分身分提供者與聯合	String	⊙ 是	⊙ 是	⊙ 是	外部

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
condition.aws:PrincipalArn	主體 ARN	表示為資源存取條件的主體 (IAM 使用者、角色或群組) 之 ARN。若要進一步了解，請參閱 AWS 全域條件內容鍵。	String	✓ 是	✓ 是	✓ 是	外部
condition.aws:PrincipalOrgID	主體 OrgID	表示為資源存取條件的主體組織識別碼。若要進一步了解，請參閱 AWS 全域條件內容金鑰。	String	✓ 是	✓ 是	✓ 是	外部
condition.aws:PrincipalOrgPaths	主體 OrgPaths	表示為資源存取條件的組織或組織單位 (OU) ID。若要進一步了解，請參閱 AWS 全域條件內容金鑰。	String	✓ 是	✓ 是	✓ 是	外部
condition.aws:SourceIp	來源 IP	使用指定的 IP 地址時，允許主體存取資源的 IP 地址。若要進一步了解，請參閱 AWS 全域條件內容金鑰。	IP 地址	✓ 是	✓ 是	✓ 是	外部

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
condition .aws:SourceVpc	來源 VPC	使用指定 VPC 時，允許主體存取資源的 VPC ID。若要進一步了解，請參閱 AWS 全域條件內容金鑰。	String	✓ 是	✓ 是	✓ 是	外部
condition .aws:UserId	使用者 ID	表示為存取資源條件且來自外部帳戶的 IAM 使用者之使用者 ID。若要進一步了解，請參閱 AWS 全域條件內容金鑰。	String	✓ 是	✓ 是	✓ 是	外部
condition .aws:VpceAccount	VPCE 帳戶	允許主體存取資源的 VPC 端點的帳戶 ID。若要進一步了解，請參閱 AWS 全域條件內容鍵。	String	✓ 是	✓ 是	✓ 是	外部 內部 (Internal)
condition .aws：SourceVpcArn	來源 VPC Arn	允許主體在使用指定的 VPC 時存取資源的 VPC ARN。若要進一步了解，請參閱 AWS 全域條件內容鍵。	ARN	✓ 是	✓ 是	✓ 是	外部

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
condition .aws:Vpce OrgID	VPCE OrgID	允許主體存取資源的 VPC 端點的組織 ID。若要進一步了解，請參閱 AWS 全域條件內容鍵。	String	✓ 是	✓ 是	✓ 是	外部 內部 (Internal)
condition .aws:Vpce OrgPaths	VPCE OrgPaths	允許主體存取資源的 VPC 端點的組織單位。若要進一步了解，請參閱 AWS 全域條件內容鍵。	字串 (清單)	✓ 是	✓ 是	✓ 是	外部 內部 (Internal)
condition .cognito-identity. amazonav .com:aud	Cognito 對象	指定為問題清單中 IAM 角色存取條件的 Amazon Cognito 身分集區 ID。若要進一步了解，請參閱 IAM 和 AWS STS 條件內容索引鍵。	String	✓ 是	✓ 是	✓ 是	外部
condition .graph.fa cebook.cc m:app_id	Facebook 應用程式 ID	指定為允許以 Facebook 聯合身分存取問題清單中 IAM 角色之條件的 Facebook 應用程式 ID (或網站 ID)。若要進一步了解，請參閱 IAM 和 AWS STS 條件內容索引鍵。	String	✓ 是	✓ 是	✓ 是	外部

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
condition.accounts.google.com:aud	Google 對象	指定為存取 IAM 角色之條件的 Google 應用程式 ID。若要進一步了解，請參閱 IAM 和 AWS STS 條件內容索引鍵。	String	✓ 是	✓ 是	✓ 是	外部
condition.kms:CallerAccount	KMS 金鑰 ID	擁有 服務呼叫所使用的呼叫實體 (IAM 使用者、角色或帳戶根使用者） AWS 的帳戶 ID AWS KMS。若要進一步了解，請參閱 的條件索引鍵 AWS Key Management Service。	String	✓ 是	✓ 是	✓ 是	外部
condition.www.amazon.com:app_id	Amazon 應用程式 ID	指定為允許使用 Login with Amazon 聯合存取角色之條件的 Amazon 應用程式 ID (或網站 ID)。如需進一步了解，請參閱	String	✓ 是	✓ 是	✓ 是	外部

Criterion	AWS 管理主控台 欄位	Description (描述)	類型	封存規則	列出問題清單	列出存取預覽問題清單	支援的分析器類型
id	問題清單 ID	問題清單的 ID。	String	⊗ 否	⊙ 是	⊙ 是	外部 內部 (Internal) 未使用的
ChangeType CHANGED | NEW | UNCHANGED		提供對存取預覽問題清單與 IAM Access Analyzer 中所識別的現有存取進行比較的內容。	String	⊗ 否	⊗ 否	⊙ 是	外部
existingFindingId		IAM Access Analyzer 中問題清單的現有 ID，僅針對存取預覽中現有的問題清單提供。	String	⊗ 否	⊗ 否	⊙ 是	外部
existingFindingStatus		Access Analyzer 中問題清單的現有狀態，僅針對存取預覽中現有的問題清單提供。	String	⊗ 否	⊗ 否	⊙ 是	外部

使用 的服務連結角色 AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer 使用 IAM 服務連結角色。服務連結角色是直接連結至 IAM Access Analyzer 的一種特殊 IAM 角色類型。服務連結角色由 IAM Access Analyzer 預先定義，並包含該功能代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓設定 IAM Access Analyzer 更為簡單，因為您不必手動新增必要的許可。IAM Access Analyzer 會定義其服務連結角色的許可，除非另有定義，否則僅有 IAM Access Analyzer 可以擔任其角色。定義的許可包括信任政策和許可政策，並且該許可政策不能連接到任何其他 IAM 實體。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，並尋找 Service-Linked Role (服務連結角色) 欄顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

的服務連結角色許可 AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer 使用名為 AWSServiceRoleForAccessAnalyzer 的服務連結角色 – 允許 Access Analyzer 分析外部存取的資源中繼資料，並分析活動以識別未使用的存取。

AWSServiceRoleForAccessAnalyzer 服務連結角色信任下列服務擔任該角色：

- `access-analyzer.amazonaws.com`

名為 `AccessAnalyzerServiceRolePolicy` 的角色許可政策允許 IAM Access Analyzer 對特定資源完成動作。

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立 IAM Access Analyzer 的服務連結角色

您不需要手動建立服務連結角色，當您在 AWS 管理主控台 或 AWS API 中啟用 Access Analyzer 時，IAM Access Analyzer 會為您建立服務連結角色。您啟用 IAM Access Analyzer 的所有區域中都會使用相同的服務連結角色。外部存取和未使用的存取權調查結果都使用相同的服務連結角色。

> **Note**
>
> IAM Access Analyzer 是區域性的。您必須在每個區域中個別啟用 IAM Access Analyzer。

如果您刪除此服務連結角色，在下一次建立分析器時，IAM Access Analyzer 會重新建立角色。

您也可以使用 IAM 主控台透過 Access Analyzer 使用案例以建立一個服務連結角色。在 AWS CLI 或 AWS API 中，使用服務名稱建立`access-analyzer.amazonaws.com`服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的建立服務連結角色。如果您刪除此服務連結角色，您可以使用此相同的程序以再次建立該角色。

編輯 IAM Access Analyzer 的服務連結角色

IAM Access Analyzer 不允許您編輯 AWSServiceRoleForAccessAnalyzer 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

刪除 IAM Access Analyzer 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，在手動刪除服務連結角色之前，您必須先清除資源。

如果在您的一或多個區域中啟用 IAM Access Analyzer AWS Organizations，您必須先刪除組織所有區域中的所有分析器，然後再嘗試刪除此角色。

> **Note**
>
> 如果 IAM Access Analyzer 在您試圖刪除資源時正在使用該角色，刪除動作可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

刪除 AWSServiceRoleForAccessAnalyzer 角色使用的 IAM Access Analyzer 資源

1. 前往網址 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 Access reports (存取報告) 區段的 Access analyzer (存取分析器) 底下，選擇 Analyzers (分析器)。
3. 選擇要從中刪除連接至服務連結角色之 IAM Access Analyzer 資源的分析器。
4. 選擇 刪除。

5. 若要確認您要刪除分析器，請輸入 **delete**，然後選擇 Delete (刪除)。

使用 IAM 手動刪除服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除 AWSServiceRoleForAccessAnalyzer 服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

IAM Access Analyzer 服務連結角色的支援區域

IAM Access Analyzer 支援在所有提供服務的區域中使用服務連結角色。如需詳細資訊，請參閱 AWS 區域與端點。

預覽存取

除了協助您識別與外部實體共用的資源外， AWS IAM Access Analyzer 還會在部署資源使用許可之前為您顯示 IAM Access Analyzer 調查結果的預覽，以便驗證您的政策變更是否僅授予資源的預期公有和跨帳戶存取權。這可協助您從預期的外部存取資源開始。

您可以在 Amazon S3 主控台中預覽和驗證對 Amazon S3 儲存貯體的公有和跨帳戶存取權。您也可以使用 IAM Access Analyzer APIs，為您的 資源提供建議許可，以預覽 Amazon S3 儲存貯體、 AWS KMS 金鑰、IAM 角色、Amazon SQS 佇列和 Secrets Manager 秘密的公有和跨帳戶存取。

主題

- 預覽 Amazon S3 主控台中的存取
- 使用 IAM Access Analyzer API 預覽存取

預覽 Amazon S3 主控台中的存取

在 Amazon S3 主控台中完成儲存貯體政策後，您可以選擇預覽對 Amazon S3 儲存貯體的公有和跨帳戶存取。您可以驗證您的政策變更僅授與預定的外部存取，然後再選擇儲存變更。此選用步驟可讓您預覽您儲存貯體的 AWS Identity and Access Management Access Analyzer 問題清單。您可以驗證政策變更是否引進新的問題清單，或是解決外部存取的現有問題清單。您可以隨時略過此驗證步驟並儲存 Amazon S3 儲存貯體政策。

若要預覽儲存貯體的外部存取，您必須在儲存貯體區域中擁有作用中的帳戶分析器，並將帳戶做為信任區域。您也必須具有使用 IAM Access Analyzer 和預覽存取所需的許可。如需啟用 IAM Access Analyzer 和所需許可的詳細資訊，請參閱 入門 AWS Identity and Access Management Access Analyzer。

建立或編輯儲存貯體政策時，預覽 Amazon S3 儲存貯體的存取

1. 完成建立或編輯儲存貯體政策後，請確保您的政策是有效的 Amazon S3 儲存貯體政策。政策 ARN 必須符合儲存貯體 ARN 且政策元素必須有效。
2. 在政策之下的預覽外部存取，選擇作用中的帳戶分析器，然後選擇預覽。IAM Access Analyzer 問題清單的預覽會針對您的儲存貯體產生。預覽會分析顯示的 Amazon S3 儲存貯體政策以及現有的儲存貯體許可。這包括儲存貯體和帳戶 BPA 設定、儲存貯體 ACL、連接至儲存貯體的 Amazon S3 存取點和多區域存取點，以及其政策和 BPA 設定。
3. 當存取預覽完成時，隨即顯示 IAM Access Analyzer 問題清單的預覽。儲存政策後，每個問題清單都會報告可存取您儲存貯體的帳戶以外的主體執行個體。您可以檢閱每個問題清單來驗證儲存貯體的存取。問題清單標頭會提供存取的摘要，您可以展開問題清單以檢閱問題清單詳細資訊。問題清單徽章提供了關於保存儲存貯體政策如何變更對儲存貯體存取的內容。例如，其可協助您驗證政策變更是否引進新的問題清單，或是解決外部存取的現有問題清單：
   a. 新的 – 表示政策將引進的新外部存取問題清單。
   b. 已解決 – 表示政策將移除之現有外部存取的問題清單。
   c. 已封存 – 表示新外部存取的問題清單會根據分析器的封存規則自動封存，該分析器定義何時應將問題清單標示為預期。
   d. 現有的 – 表示外部存取的現有問題清單會維持不變。
   e. 公有 – 如果問題清單是用於公有存取該資源，它除了上述其中一個徽章之外，還將具有公有徽章。
4. 如果您識別不打算引進或移除的外部存取，您可以修訂政策，然後再次選擇預覽，直到您達到您想要的外部存取為止。如果您具有標籤為公有的問題清單，建議您修改政策以移除公有存取，然後再選擇儲存變更。預覽存取是選擇性的步驟，您隨時可以選擇儲存變更。

使用 IAM Access Analyzer API 預覽存取

您可以使用 IAM Access Analyzer APIs 預覽 Amazon S3 儲存貯體、 AWS KMS 金鑰、IAM 角色、Amazon SQS 佇列和 Secrets Manager 秘密的公有和跨帳戶存取。您可以為您擁有的現有資源或要部署的新資源提供建議的許可來預覽存取。

若要預覽資源的外部存取，您必須擁有資源帳戶和區域的作用中帳戶分析器。您也必須具有使用 IAM Access Analyzer 和預覽存取所需的許可。如需啟用 IAM Access Analyzer 和所需許可的詳細資訊，請參閱 入門 AWS Identity and Access Management Access Analyzer。

若要預覽資源的存取，您可以使用 CreateAccessPreview 操作，並提供資源的分析器 ARN 和存取控制組態。服務會傳回存取預覽的唯一 ID，您可以使用透過 GetAccessPreview 操作該 ID 來檢查存取預覽的狀態。當狀態為 Completed 時，您可以使用 ListAccessPreviewFindings 操作來擷取針對存取預覽產生的問題清單。GetAccessPreview 和 ListAccessPreviewFindings 操作會擷取約 24 小時內建立的存取預覽和問題清單。

擷取的每個問題清單都包含描述存取的問題清單詳細資訊。說明問題清單在許可部署之後是否為 Active、Archived 或 Resolved 以及 changeType 的問題清單預覽狀態。changeType 提供對存取預覽問題清單與 IAM Access Analyzer 中所識別的現有存取進行比較的內容：

- 新的 – 問題清單是針對新引進的存取。
- 未變更 – 預覽問題清單是保持不變的現有問題清單。
- 已變更 – 預覽問題清單是狀態變更的現有問題清單。

status 與 changeType 可協助您了解資源組態如何變更現有資源存取。如果 changeType 是 Unchanged 或已變更，則問題清單也會包含 IAM Access Analyzer 中問題清單的現有 ID 和狀態。例如，包含預覽狀態 Resolved 和現有狀態 Active 的 Changed 問題清單表示資源的現有 Active 問題清單將成為 Resolved，做為提議許可變更的結果。

您可以使用 ListAccessPreviews 操作來擷取指定分析器的存取預覽清單。此操作會擷取約一小時內建立的存取預覽相關資訊。

一般而言，如果存取預覽是針對現有資源而您未指定組態選項，則依預設，存取預覽將使用現有的資源組態。如果存取預覽是針對新資源而您未指定組態選項，則存取預覽將根據資源類型使用預設值。如需每種資源類型的組態案例，請參閱下列項目。

預覽 Amazon S3 儲存貯體的存取

如需為新的 Amazon S3 儲存貯體或您擁有的現有 Amazon S3 儲存貯體建立存取預覽，您可以指定連接到儲存貯體的 Amazon S3 儲存貯體政策、儲存貯體 ACL、儲存貯體 BPA 設定和 Amazon S3 存取點 (包括多區域存取點) 來建議儲存貯體組態。

> **Note**
>
> 在嘗試為新儲存貯體建立存取預覽之前，建議您先呼叫 Amazon S3 HeadBucket 操作來檢查命名儲存貯體是否已經存在。此操作對於判斷儲存貯體是否存在以及您是否有許可加以存取很有用。

儲存貯體政策 – 如果組態適用於現有 Amazon S3 儲存貯體，而您未指定 Amazon S3 儲存貯體政策，則存取預覽會使用附加到儲存貯體的現有政策。如果存取預覽適用於新資源，且您未指定 Amazon S3 儲存貯體政策，則存取預覽會假設沒有政策的儲存貯體。若要提議刪除現有儲存貯體政策，您可以指定空字串。如需受支援儲存貯體政策限制的詳細資訊，請參閱儲存貯體政策範例。

儲存貯體 ACL 授與 – 您可以提議每個儲存貯體最多 100 個 ACL 授與。如果提議的授與組態適用於現有儲存貯體，則存取預覽會使用提議的授與組態清單來取代現有的授與。否則，存取預覽會使用儲存貯體的現有授與。

儲存貯體存取點 – 分析支援每個儲存貯體最多 100 個存取點 (包括多區域存取點)，包括每個儲存貯體最多可提議的 10 個新存取點。如果提議的 Amazon S3 存取點組態適用於現有儲存貯體，則存取預覽會使用提議的存取點組態來取代現有的存取點。若要提議沒有政策的存取點，您可以提供空字串做為存取點政策。如需存取點政策限制的詳細資訊，請參閱存取點的法規與限制。

封鎖公有存取組態 – 如果提議的組態適用於現有 Amazon S3 儲存貯體，而您未指定組態，則存取預覽會使用現有的設定。如果提議的組態適用於新儲存貯體，且您未指定儲存貯體 BPA 組態，則存取預覽會使用 `false`。如果提議的組態是針對新的存取點或多區域存取點，而您未指定存取點 BPA 組態，則存取預覽會使用 `true`。

預覽對 AWS KMS 金鑰的存取

若要為新 AWS KMS 金鑰或您擁有的現有 AWS KMS 金鑰建立存取預覽，您可以透過指定 AWS KMS 金鑰政策和 AWS KMS 授予組態來提議金鑰組態。

AWS KMS 金鑰政策 – 如果組態適用於現有的金鑰，而您未指定金鑰政策，則存取預覽會使用該金鑰的現有政策。如果存取預覽是針對新資源，而您未指定金鑰政策，則存取預覽會使用預設金鑰政策。提議的金鑰政策不得為空字串。

AWS KMS 授予 – 此分析支援每個組態最多 100 個 KMS 授予*。* 如果提議的授予組態適用於現有的金鑰，則存取預覽會使用提議的授予組態清單來取代現有的授予。否則，存取預覽會使用金鑰的現有授與。

預覽您 IAM 角色的存取

若要建立新 IAM 角色的存取預覽或您擁有的現有 IAM 角色，您可以指定金鑰政策來提議一個 IAM 角色組態。

角色信任政策 – 如果組態適用於新的 IAM 角色，您必須指定信任政策。如果組態適用於您所擁有的現有 IAM 角色，且您未提議信任政策，則存取預覽會使用該角色的現有信任政策。提議的信任政策不得為空字串。

預覽您 Amazon SQS 佇列的存取

若要為新的 Amazon SQS 佇列或您擁有的現有 Amazon SQS 佇列建立存取預覽，您可以指定佇列的 Amazon SQS 政策來提議 Amazon SQS 佇列組態。

Amazon SQS 佇列政策 – 如果組態適用於現有 Amazon SQS 佇列，而您未指定 Amazon SQS 政策，則存取預覽會將現有的 Amazon SQS 政策用於佇列。如果存取預覽是針對新資源，而您未指定政策，則存取預覽會假設 Amazon SQS 佇列不包含政策。若要提議刪除現有的 Amazon SQS 佇列政策，您可以為 Amazon SQS 政策指定空字串。

預覽您 Secrets Manager 秘密的存取

若要為新的 Secrets Manager 秘密或您擁有的現有 Secrets Manager 秘密建立存取預覽，您可以透過指定秘密政策和選用 AWS KMS 加密金鑰來提議 Secrets Manager 秘密組態。

秘密政策 – 如果組態是針對現有秘密，而您未指定秘密政策，則存取預覽會針對秘密使用現有的政策。如果存取預覽是針對新資源，而您未指定政策，則存取預覽會假設秘密不包含政策。若要提議刪除現有政策，您可以指定空字串。

AWS KMS 加密金鑰 – 如果提議的組態適用於新的秘密，而且您未指定 AWS KMS 金鑰 ID，則存取預覽會使用 AWS 帳戶的預設 KMS 金鑰。如果您為 AWS KMS 金鑰 ID 指定空字串，則存取預覽會使用 AWS 帳戶的預設 KMS 金鑰。

驗證政策檢查

IAM Access Analyzer 提供政策檢查功能，協助您在將 IAM 政策連接到實體之前驗證這些政策。包括政策驗證提供的基本政策檢查，根據政策文法和 AWS 最佳實務來驗證您的政策。您可以檢視政策驗證檢查問題清單，包含安全警告、錯誤、一般警告和政策的建議。

您可以使用自訂政策檢查，根據您的安全標準來檢查是否有新的存取權。每次檢查新存取權都會收取費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

自訂政策檢查的運作方式

您可以使用 AWS Identity and Access Management Access Analyzer 自訂政策檢查，根據您指定的安全標準來驗證 IAM 政策。您可以執行以下類型的自訂政策檢查：

- 對照參考政策進行檢查：編輯政策時，您可以檢查更新版政策是否與參考政策 (例如現有政策版本) 相比，是否會授予新的存取權。您可以在使用 IAM 主控台中的 AWS Command Line Interface (AWS CLI)、IAM Access Analyzer API (API) 或 JSON 政策編輯器編輯政策時執行此檢查。

> **Note**
>
> IAM Access Analyzer 自訂政策檢查允許在 `Principal` 元素中使用萬用字元來參考資源政策。

- 根據 IAM 動作或資源清單進行檢查：您可以進行檢查以確保政策不允許特定的 IAM 動作或資源。如果只指定動作，則 IAM Access Analyzer 會檢查該動作對政策中所有資源的存取。如果只指定資源，則 IAM Access Analyzer 會檢查哪些動作可以存取指定的資源。如果同時指定了動作和資源，則 IAM Access Analyzer 會檢查哪些指定的動作可以存取指定的資源。您可以在使用 或 AWS CLI API 建立或編輯政策時執行此檢查。
- 檢查公開存取：您可以檢查資源政策是否可以將公開存取授予指定的資源類型。您可以在使用 或 AWS CLI API 建立或編輯政策時執行此檢查。這種類型的自訂政策檢查與預覽存取不同，因為該檢查不需要任何帳戶或外部存取分析器內容。存取預覽可讓您在部署資源許可之前預覽 IAM Access Analyzer 調查結果，而自訂檢查則會判斷政策是否可能授予公開存取。

每次自訂政策檢查都會收取費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

您可以對身分和資源型政策執行自訂政策檢查。自訂政策檢查不會依賴模式比對技術或檢查存取日誌，來判斷政策是否允許新存取權或指定存取權。與外部存取權調查結果類似，自訂政策檢查是以 Zelkova 為基礎。會將 IAM 政策轉換為等效的邏輯陳述式，並針對問題執行一套通用和專用的邏輯求解器 (可滿足性模數理論)。為了檢查是否有新的存取權或指定存取權，IAM Access Analyzer 會將 Zelkova 重複套用至政策。根據政策內容，查詢範圍會越來越限縮，以符合政策允許的行為類別特徵。如需可滿足性模數理論的詳細資訊，請參閱可滿足性模數理論。

在罕見情況下，IAM Access Analyzer 無法完全判斷政策陳述式是否會授予新的存取權或指定存取權。若發生這種情況，會錯誤地因為未通過自訂政策檢查而宣告誤判。IAM Access Analyzer 旨在提供全方位政策評估，並致力將誤判降到最低。此做法表示 IAM Access Analyzer 可充分保證，通過檢查就表示政策不會授予存取權。您可以檢視 IAM Access Analyzer 回應中報告的政策陳述式，以手動方式審視失敗的檢查。

檢查新存取權的參考政策範例

如需參考政策的範例，並了解如何設定和執行自訂政策檢查以檢查新存取權，請參閱 GitHub 上的 IAM Access Analyzer 自訂政策檢查範例儲存庫。

使用這些範例之前

使用這些參考政策範例前，請執行下列操作：

- 詳閱參考政策並根據您的獨特需求自訂內容。
- 在您的環境中搭配您使用的 AWS 服務 徹底測試參考政策。

  參考政策示範如何實作和使用自訂政策檢查。他們並非闡述為完全如圖所示實作的官方 AWS 建議或最佳實務。您有責任仔細測試任何參考政策是否適合解決您環境的安全需求。
- 自訂政策檢查分析與環境無關。分析僅考慮輸入政策中包含的資訊。例如，自訂政策檢查無法檢查帳戶是否為特定 AWS 組織的成員。因此，自訂政策檢查無法根據 `aws:PrincipalOrgId` 和 `aws:PrincipalAccount` 條件索引鍵的條件索引鍵值來比較新存取權。

檢查失敗的自訂政策檢查

自訂政策檢查失敗時，IAM Access Analyzer 的回應中會包含造成檢查失敗的政策陳述式之 陳述式 ID (Sid)。雖然陳述式 ID 是選用的政策元素，但建議您為每個政策陳述式都新增陳述式 ID。自訂政策檢查也會傳回陳述式索引，以協助識別檢查失敗的原因。陳述式索引遵循以零為基準的編號方式，第一個陳述式會參考為 0。有多個陳述式造成檢查失敗時，檢查一次只會傳回一個陳述式 ID。建議您修正原因中反白的陳述式，然後重新執行檢查，直到通過檢查為止。

使用 IAM Access Analyzer 來驗證政策

您可以使用政策驗證來驗證您的 AWS Identity and Access Management Access Analyzer 政策。您可以使用 IAM 主控台中的 AWS CLI、 AWS API 或 JSON 政策編輯器來建立或編輯政策。IAM Access Analyzer 會根據 IAM 政策文法和 AWS 最佳實務來驗證您的政策。您可以檢視政策驗證檢查問題清單，包含安全警告、錯誤、一般警告和政策的建議。這些問題清單提供可行的建議，協助您撰寫具有功能性且符合安全最佳實務的政策。若要檢視 IAM Access Analyzer 執行的基本政策檢查清單，請參閱：IAM 政策驗證檢查參考。

在 IAM (主控台) 中驗證政策

您在 IAM 主控台中建立或編輯受管政策時，可以檢視 IAM Access Analyzer 政策驗證產生的調查結果。您也可以檢視內嵌使用者或角色政策的這些問題清單。IAM Access Analyzer 不會為內嵌群組政策產生調查結果。

檢視由 IAM JSON 政策的政策檢查產生的問題清單

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 使用以下其中一個方法開始建立或編輯政策：
   a. 若要建立新的受管政策，請移至 Policies (政策) 頁面並建立新政策。如需詳細資訊，請參閱正在使用 JSON 編輯器建立政策。
   b. 若要檢視現有客戶管理政策的政策檢查，請移至政策頁面，選擇政策的名稱，然後選擇編輯。如需詳細資訊，請參閱編輯客戶受管政策 (主控台)。
   c. 若要檢視使用者或角色內嵌政策的政策檢查，請移至使用者或角色頁面，選擇使用者或角色的名稱，然後選擇許可索引標籤上的政策名稱，然後選擇編輯。如需詳細資訊，請參閱編輯內嵌政策 (主控台)。
3. 在政策編輯器中，選擇 JSON 標籤。
4. 在政策下方的政策驗證窗格中，選擇下列一或多個標籤。標籤名稱也會指出政策的每個問題清單類型數目。
   - 安全 – 如果您的政策允許因存取過於寬鬆而 AWS 被視為安全風險的存取，請檢視警告。
   - 錯誤 – 如果您的政策包含使政策無法運作的行，請檢視錯誤。
   - 警告 – 如果您的政策不符合最佳做法，但問題不屬於安全風險，請檢視警告。
   - 建議 – 如果 AWS 建議不影響政策之許可的改進功能，請檢視警告。
5. 檢閱 IAM Access Analyzer 政策檢查所提供的問題清單詳細資訊。每個問題清單都會指出報告問題的位置。若要深入了解造成問題的原因以及如何解決問題，請選擇報告問題旁的 Learn more (進一步了解) 連結。您也可以在 Access Analyzer policy checks (Access Analyzer 政策檢查) 參考頁面中搜尋與每個問題清單相關聯的政策檢查。
6. 選用。如果您正在編輯現有政策，可以執行自訂政策檢查，以判斷更新版政策與現有版本相比，是否會授予新的存取權。在政策下方的政策驗證窗格中，選擇檢查新的存取權索引標籤，然後選擇檢查政策。如果修改後的許可會授予新存取權，該陳述式會在政策驗證窗格中反白顯示。如果您不打算授予新的存取權，請更新政策陳述式並選擇檢查政策，直到沒有偵測到新的存取權為止。如需詳細資訊，請參閱使用 IAM Access Analyzer 自訂政策檢查來驗證政策。

> **Note**
>
> 每次檢查新存取權都會收取費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

7. 更新您的政策以解決問題清單。

> **Important**
>
> 在您的生產工作流程中實作新的或已編輯的政策之前，請先完整測試。

8. 完成時，選擇 Next (下一步)。政策驗證器會報告 IAM Access Analyzer 未報告的所有語法錯誤。

> **Note**
>
> 您可以隨時在視覺化和 JSON 索引標籤之間切換。不過，如果您進行變更或在視覺化索引標籤中選擇下一步，IAM 可能會調整您的政策結構，以針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

9. 若使用新政策，在檢閱與建立頁面上，為您正在建立的政策輸入政策名稱與描述 (選用)。檢閱此政策中定義的許可，來查看您的政策所授予的許可。然後選擇 Create policy (建立政策) 來儲存您的工作。

   若使用現有的政策，在檢閱與儲存頁面上，檢閱在此政策中定義的許可來查看您的政策所授予的許可。選擇將此新版本設定為預設值。核取方塊，將更新版政策儲存為預設版政策。選擇儲存變更以儲存編輯內容。

使用 IAM Access Analyzer (AWS CLI 或 AWS API) 驗證政策

您可以從 AWS Command Line Interface (AWS CLI) 檢視 IAM Access Analyzer 政策驗證所產生的調查結果。

檢視 IAM Access Analyzer 政策驗證 (AWS CLI 或 AWS API) 所產生的問題清單

請使用下列其中一個：

- AWS CLI：aws accessanalyzer validate-policy
- AWS API：ValidatePolicy

IAM 政策驗證檢查參考

您可以使用政策驗證來驗證您的 AWS Identity and Access Management Access Analyzer 政策。您可以使用 IAM 主控台中的 AWS CLI、 AWS API 或 JSON 政策編輯器來建立或編輯政策。IAM Access

Analyzer 會根據 IAM 政策文法和 AWS 最佳實務來驗證您的政策。您可以檢視政策驗證檢查問題清單，包含安全警告、錯誤、一般警告和政策的建議。這些問題清單提供可行的建議，協助您撰寫具有功能性且符合安全最佳實務的政策。IAM Access Analyzer 提供的基本政策檢查清單如下。執行政策驗證檢查不會額外收費。若要進一步了解如何使用政策驗證來驗證政策，請參閱：使用 IAM Access Analyzer 來驗證政策。

錯誤 – 不允許 ARN 帳戶

問題代碼：ARN_ACCOUNT_NOT_ALLOWED

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

解決錯誤

從資源 ARN 中移除帳戶 ID。某些 AWS 服務的資源 ARNs 不支援指定帳戶 ID。

例如，Amazon S3 不支援帳戶 ID 做為儲存貯體 ARN 中的命名空間。Amazon S3 儲存貯體名稱是全域唯一的，且命名空間由所有 AWS 帳戶共用。若要檢視 Amazon S3 中可用的所有資源類型，請參閱服務授權參考中的 Amazon S3 定義的資源類型。

相關用語

- 政策資源
- 帳戶識別符
- 資源 ARN
- AWS 具有 ARN 格式的服務資源

錯誤 – 不允許 ARN 區域

問題代碼：ARN_REGION_NOT_ALLOWED

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

解決錯誤

從資源 ARN 中移除區域。某些 AWS 服務的資源 ARNs 不支援指定區域。

例如，IAM 是全球服務。IAM 資源 ARN 的「區域」部分一律保持空白。IAM 資源是全域的，就像帳戶 AWS 現在一樣。例如，以 IAM 使用者身分登入後，您可以存取任何地理區域中的 AWS 服務。

- 政策資源
- 資源 ARN
- AWS 具有 ARN 格式的服務資源

錯誤 – 資料類型不符

問題代碼：DATA_TYPE_MISMATCH

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

解決錯誤

更新文字以使用支援的資料類型。

例如，Version 全域條件鍵需要 String 資料類型。如果您提供日期或整數，則資料類型將不相符。

相關用語

- 全域條件鍵
- IAM JSON 政策元素：條件運算子

錯誤 – 重複鍵使用不同的大小寫

問題代碼：DUPLICATE_KEYS_WITH_DIFFERENT_CASE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

解決錯誤

檢閱相同條件區塊中的類似條件鍵，並針對所有執行個體使用相同的大小寫。

條件區塊是政策聲明 Condition 元素內的文字。條件鍵 names 不區分大小寫。條件鍵 values 是否區分大小寫取決於您使用的條件運算子。如需條件鍵中區分大小寫的詳細資訊，請參閱 IAM JSON 政策元素：Condition。

相關用語

- 條件
- 條件區塊
- 全域條件鍵
- AWS 服務條件索引鍵

錯誤 - 無效的動作

問題代碼：INVALID_ACTION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

解決錯誤

您指定的動作無效。如果您輸入錯誤的服務字首或動作名稱，就會發生這種情況。對於某些常見問題，政策檢查會傳回建議的動作。

相關用語

- 政策動作
- AWS 服務動作

AWS 具有此錯誤的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

下列 AWS 受管政策在其政策陳述式中包含無效的動作。無效的動作並不會影響政策所授與的許可。使用 AWS 受管政策做為建立受管政策的參考時， AWS 建議您從政策中移除無效的動作。

- AmazonEMRFullAccessPolicy_v2
- CloudWatchSyntheticsFullAccess

錯誤 - 無效的 ARN 帳戶

問題代碼：INVALID_ARN_ACCOUNT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-
digit account ID."
```

解決錯誤

更新資源 ARN 中的帳戶 ID。帳戶 ID 是 12 位數的整數。若要了解如何檢視您的帳戶 ID，請參閱尋找 AWS 您的帳戶 ID。

相關用語

- 政策資源
- 帳戶識別符
- 資源 ARN
- AWS 具有 ARN 格式的服務資源

錯誤 - 無效的 ARN 字首

問題代碼：INVALID_ARN_PREFIX

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

解決錯誤

AWS 資源 ARNs 必須包含必要的`arn:`字首。

相關用語

- 政策資源
- 資源 ARN
- AWS 具有 ARN 格式的服務資源

錯誤 - 無效的 ARN 區域

問題代碼：INVALID_ARN_REGION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

解決錯誤

資源類型在指定區域中不支援。如需每個區域中支援 AWS 的服務資料表，請參閱區域資料表。

相關用語

- 政策資源
- 資源 ARN
- 區域名稱和代碼

錯誤 - 無效的 ARN 資源

問題代碼：INVALID_ARN_RESOURCE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

解決錯誤

資源 ARN 必須符合已知資源類型的規格。若要檢視服務的預期 ARN 格式，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，以檢視其資源類型和 ARN 格式。

相關用語

- 政策資源
- 資源 ARN
- AWS 具有 ARN 格式的 服務資源

錯誤 - 無效的 ARN 服務案例

問題代碼：INVALID_ARN_SERVICE_CASE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid ARN service case: Update the service name {{service}} in the resource ARN to
 use all lowercase letters.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Update the service name {{service}} in the resource ARN to use all
 lowercase letters."
```

解決錯誤

資源 ARN 中的服務必須符合服務字首的規格 (包括大小寫)。若要檢視服務的字首，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，然後在第一句中找到該服務的字首。

相關用語

- 政策資源
- 資源 ARN
- AWS 具有 ARN 格式的 服務資源

錯誤 - 無效的條件資料類型

問題代碼：INVALID_CONDITION_DATA_TYPE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

解決錯誤

條件鍵值組中的值必須符合條件鍵和條件運算子的資料類型。若要檢視服務的條件索引鍵資料類型，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，以檢視該服務的條件鍵。

例如，CurrentTime全域條件鍵支援 Date 條件運算子。如果您為條件區塊中的值提供字串或整數，則資料類型將不相符。

相關用語

- 條件
- 條件區塊
- IAM JSON 政策元素：條件運算子
- 全域條件鍵
- AWS 服務條件索引鍵

錯誤 - 無效的條件鍵格式

問題代碼：INVALID_CONDITION_KEY_FORMAT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

解決錯誤

條件鍵值組中的鍵必須符合服務的規格。若要檢視服務的條件索引鍵，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，以檢視該服務的條件鍵。

相關用語

- 條件
- 全域條件鍵
- AWS 服務條件索引鍵

錯誤 - 無效條件多重布林值

問題代碼：INVALID_CONDITION_MULTIPLE_BOOLEAN

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

解決錯誤

條件鍵值對中的鍵需要單一布林值。當您提供多個布林值時，條件比對可能不會傳回您預期的結果。

若要檢視服務的條件索引鍵，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，以檢視該服務的條件鍵。

- 條件
- 全域條件鍵
- AWS 服務條件索引鍵

錯誤 - 無效的條件運算子

問題代碼：INVALID_CONDITION_OPERATOR

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

解決錯誤

更新條件以使用支援的條件運算子。

相關用語

- IAM JSON 政策元素：條件運算子
- 條件元素
- JSON 政策概觀

錯誤 — 效果的無效

問題代碼：INVALID_EFFECT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

解決錯誤

更新 `Effect` 元素以使用有效的效果。`Effect` 的有效值為 **`Allow`** 和 **`Deny`**。

相關用語

- 效果元素
- JSON 政策概觀

錯誤 — 無效的全域條件鍵

問題代碼：INVALID_GLOBAL_CONDITION_KEY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

解決錯誤

更新條件鍵值組中的條件鍵，以使用支援的全域條件鍵。

全域條件索引鍵是具有 `aws:` 字首的條件索引鍵。 AWS services 可以支援全域條件索引鍵，或提供包含其服務字首的服務特定索引鍵。例如，IAM 條件鍵會包含 `iam:` 字首。如需詳細資訊，請參閱 AWS 服務的動作、資源和條件金鑰 ，然後選擇您要檢視其金鑰的服務。

相關用語

- 全域條件鍵

錯誤 — 無效的分割區

問題代碼：INVALID_PARTITION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

解決錯誤

更新資源 ARN 以包含支援的分割區。如果您包含支援的分割區，則服務或資源可能不支援您包含的分割區。

分割區是一組 AWS 區域。每個 AWS 帳戶的範圍都是一個分割區。在傳統區域中，使用 aws 分割區。在中國區域，使用 aws-cn。

相關用語

- Amazon 資源名稱 (ARN) - 分割區

錯誤 — 無效的政策元素

問題代碼：INVALID_POLICY_ELEMENT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid policy element: The policy element {{element}} is not valid.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The policy element {{element}} is not valid."
```

解決錯誤

更新政策以僅包含支援的 JSON 政策元素。

相關用語

- JSON 政策元素：

錯誤 — 無效的主體格式

問題代碼：INVALID_PRINCIPAL_FORMAT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid principal format: The Principal element contents are not valid. Specify a key-
value pair in the Principal element.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value
 pair in the Principal element."
```

解決錯誤

更新主體，以使用支援的鍵值組格式。

您可以在以資源為基礎的政策中指定主體，但不能在以身分為基礎的政策中指定。

例如，若要為 AWS 帳戶中的每個人定義存取權，請在政策中使用下列委託人：

```
"Principal": { "AWS": "123456789012" }
```

相關用語

- JSON 政策元素：主體

- 以身分為基礎的政策和以資源為基礎的政策

錯誤 - 無效的主體鍵

問題代碼：INVALID_PRINCIPAL_KEY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

解決錯誤

更新主體鍵值組中的鍵，以使用支援的主體鍵。以下是支援的主體鍵：

- AWS
- CanonicalUser
- 聯合
- 服務

相關用語

- 主體元素

錯誤 - 無效的區域

問題代碼：INVALID_REGION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 supported Region.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 supported Region."
```

解決錯誤

更新條件鍵值組的值，以包含支援的區域。如需每個區域中支援 AWS 的服務資料表，請參閱區域資料表。

相關用語

- 政策資源
- 資源 ARN
- 區域名稱和代碼

錯誤 - 無效的服務

問題代碼：INVALID_SERVICE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

解決錯誤

動作或條件鍵中的服務字首必須符合服務字首的規格 (包括大小寫)。若要檢視服務的字首，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，然後在第一句中找到該服務的字首。

相關用語

- 已知的服務及其動作、資源和條件鍵

錯誤 - 無效的服務條件鍵

問題代碼：INVALID_SERVICE_CONDITION_KEY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

解決錯誤

更新條件鍵值組中的鍵，以使用服務的已知條件鍵。全域條件鍵名稱是以 aws 字首為開頭。 AWS 服務可提供包含其服務字首的服務專屬鍵。若要檢視服務的字首，請參閱 AWS 服務的動作、資源和條件索引鍵。

相關用語

- 全域條件鍵
- 已知的服務及其動作、資源和條件鍵

錯誤 - 動作中的服務無效

問題代碼：INVALID_SERVICE_IN_ACTION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

解決錯誤

動作中的服務字首必須符合服務字首的規格 (包括大小寫)。若要檢視服務的字首，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，然後在第一句中找到該服務的字首。

相關用語

- 動作元素
- 已知的服務及其行動

錯誤 — 運算子的無效變數

問題代碼：INVALID_VARIABLE_FOR_OPERATOR

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

解決錯誤

您可以在 Resource 元素中使用政策變數，也可以在 Condition 元素中使用字串比較。當您使用字串運算子或 ARN 運算子時，條件支援變數。字串運算子包括 StringEquals、StringLike 和 StringNotLike。ARN 運算子包括 ArnEquals 和 ArnLike。您無法使用政策變數搭配其他運算子，例如 Numeric、Date、Boolean、Binary、IP Address 或 Null 運算子。

相關用語

- 在「條件」元素中使用政策變數
- 條件元素

錯誤 - 無效的版本

問題代碼：INVALID_VERSION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid version: The version {{version}} is not valid. Use one of the following
 versions: {{versions}}
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The version {{version}} is not valid. Use one of the following
 versions: {{versions}}"
```

解決錯誤

Version 政策元素指定 AWS 用於處理政策的語言語法規則。若要使用所有可用的政策功能，請在所有政策的 Statement 元素前面包含最新的 Version 元素。

```
"Version": "2012-10-17"
```

相關用語

- 版本元素

錯誤 – Json 語法錯誤

問題代碼：JSON_SYNTAX_ERROR

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

解決錯誤

您的政策包含語法錯誤。檢查您的 JSON 語法。

相關用語

- JSON 驗證器
- IAM JSON 政策元素參考
- JSON 政策概觀

錯誤 – Json 語法錯誤

問題代碼：JSON_SYNTAX_ERROR

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Json syntax error: Fix the JSON syntax error.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Fix the JSON syntax error."
```

解決錯誤

您的政策包含語法錯誤。檢查您的 JSON 語法。

相關用語

- JSON 驗證器
- IAM JSON 政策元素參考
- JSON 政策概觀

錯誤 - 缺少動作

問題代碼：MISSING_ACTION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing action: Add an Action or NotAction element to the policy statement.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

解決錯誤

AWS JSON 政策必須包含 `Action`或 `NotAction`元素。

相關用語

- 動作元素
- NotAction 元素
- JSON 政策概觀

錯誤 – 缺少 ARN 欄位

問題代碼：MISSING_ARN_FIELD

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

解決錯誤

資源 ARN 中的所有欄位必須符合已知資源類型的規格。若要檢視服務的預期 ARN 格式，請參閱 AWS 服務的動作、資源和條件索引鍵。選擇服務的名稱，以檢視其資源類型和 ARN 格式。

相關用語

- 政策資源
- 資源 ARN
- AWS 具有 ARN 格式的 服務資源

錯誤 – 缺少 ARN 區域

問題代碼：MISING_ARN_REGION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

解決錯誤

大多數 AWS 服務的資源 ARNs 需要您指定區域。如需每個區域中支援的 AWS 服務資料表，請參閱區域資料表。

相關用語

- 政策資源
- 資源 ARN
- 區域名稱和代碼

錯誤 – 缺少效果

問題代碼：MISSING_EFFECT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

解決錯誤

AWS JSON 政策必須包含值為 和 的 **`Allow`** `Effect`元素 **`Deny`**。

相關用語

- 效果元素

- JSON 政策概觀

錯誤 – 缺少主體

問題代碼：MISSING_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing principal: Add a Principal element to the policy statement.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a Principal element to the policy statement."
```

解決錯誤

以資源為基礎的政策必須包含 `Principal` 元素。

例如，若要為 AWS 帳戶中的每個人定義存取權，請在政策中使用下列委託人：

```
"Principal": { "AWS": "123456789012" }
```

相關用語

- 主體元素
- 以身分為基礎的政策和以資源為基礎的政策

錯誤 – 缺少限定詞

問題代碼：MISSING_QUALIFIER

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing qualifier: The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

解決錯誤

在 Condition 元素中，您所建置的表達式使用條件運算子 (例如等於或小於) 來比較政策中的條件鍵與值，以及請求內容中的鍵與值。對於包含單一條件鍵之多個值的請求，您必須像陣列一樣，在括號內括住條件 ("Key2":["Value2A", "Value2B"])。您還必須使用 ForAllValues 或 ForAnyValue 設定運算子搭配 StringLike 條件運算子。這些限定詞新增設定操作功能到條件運算子，以便您可以針對多個條件值測試多個請求值。

相關用語

- 多值內容鍵
- 條件元素

AWS 具有此錯誤的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

下列 AWS 受管政策在其政策陳述式中包含缺少條件索引鍵的限定詞。使用 AWS 受管政策做為建立客戶受管政策的參考時， AWS 建議您將 ForAllValues或 ForAnyValue條件金鑰限定詞新增至 Condition元素。

- AWSGlueConsoleSageMakerNotebookFullAccess

錯誤 – 缺少資源

問題代碼：MISSING_RESOURCE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

解決錯誤

角色信任政策以外的所有政策都必須包含 Resource 或 NotResource 元素。

相關用語

- 資源元素
- NotResource 元素
- 以身分為基礎的政策和以資源為基礎的政策
- JSON 政策概觀

錯誤 – 缺少陳述式

問題代碼：MISSING_STATEMENT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing statement: Add a statement to the policy
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a statement to the policy"
```

解決錯誤

JSON 政策必須包含陳述式。

相關用語

- JSON 政策元素：

錯誤 – 如果存在，則為空

問題代碼：NULL_WITH_IF_EXISTS

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

解決錯誤

除 `Null` 條件運算子外，您可在任何條件運算子名稱的結尾新增 `IfExists`。使用 `Null` 條件運算子檢查授權時是否有條件鍵。使用 `...ifExists` 來表示「如果請求的內容中存在政策鍵，則依照政策所述來處理鍵。如果該鍵不存在，則評估條件元素為 true。」

相關用語

- ...IfExists 條件運算子
- Null 條件運算子
- 條件元素

錯誤 – SCP 語法錯誤動作萬用字元

問題代碼：SCP_SYNTAX_ERROR_ACTION_WILDCARD

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

解決錯誤

AWS Organizations 服務控制政策 SCPs) 支援在 `Action`或 `NotAction`元素中指定值。不過，這些值只能在字串結尾包含萬用字元 (*)。這表示您可指定 `iam:Get*` 而不是 `iam:*role`。

若要指定多個動作， AWS 建議您個別列出這些動作。

相關用語

- SCP 動作與 NotAction 元素
- SCP 評估
- AWS Organizations 服務控制政策
- IAM JSON 政策元素：動作

錯誤 – SCP 語法錯誤主體

問題代碼：SCP_SYNTAX_ERROR_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

解決錯誤

AWS Organizations 服務控制政策 SCPs) 不支援 `Principal`或 `NotPrincipal`元素。

您可以使用 `Condition` 元素中的 `aws:PrincipalArn` 全域條件鍵來指定 Amazon Resource Name (ARN)。

相關用語

- SCP 語法
- 主體的全域條件鍵

錯誤 – 需要唯一 Sid

問題代碼：UNIQUE_SIDS_REQUIRED

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

解決錯誤

對於某些政策類型，陳述式 ID 必須是唯一的。`Sid` (陳述式 ID) 元素允許您輸入您為政策陳述式提供的可選識別符。您可以使用 `SID` 元素將陳述式 ID 值指派給陳述式陣列中的每個陳述式。在允許您指定 ID 元素的服務中 (例如 SQS 和 SNS)，`Sid` 值只是政策文件 ID 的子 ID。例如，在 IAM 中，`Sid` 值在 JSON 政策中必須是唯一的。

相關用語

- IAM JSON 政策元素：Sid

錯誤 - 政策中不支援的動作

問題代碼：UNSUPPORTED_ACTION_IN_POLICY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

解決錯誤

在連接至不同資源類型的資源型政策中，部分動作在其 `Action` 元素中不受支援。例如，Amazon S3 儲存貯體政策不支援 AWS Key Management Service 動作。指定連接至資源型政策的資源類型支援的動作。

相關用語

- JSON 政策元素：動作

錯誤 – 不支援的元素組合

問題代碼：UNSUPPORTED_ELEMENT_COMBINATION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported element combination: The policy elements {{element1}} and {{element2}} can
 not be used in the same statement. Remove one of these elements.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The policy elements {{element1}} and {{element2}} can not be used in
 the same statement. Remove one of these elements."
```

解決錯誤

JSON 政策元素的某些組合無法一起使用。例如，您不能在同一政策陳述式中同時使用 `Action` 與 `NotAction`。相互排斥的其他組合包括 `Principal/NotPrincipal` 和 `Resource/NotResource`。

相關用語

- IAM JSON 政策元素參考
- JSON 政策概觀

錯誤 – 不支援的全域條件鍵

問題代碼：UNSUPPORTED_GLOBAL_CONDITION_KEY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

解決錯誤

AWS 不支援使用指定的全域條件金鑰。視您的使用案例而定，您可以使用 `aws:PrincipalArn` 或 `aws:SourceArn` 全域條件鍵。例如，並非使用 `aws:ARN`，而是使用 `aws:PrincipalArn` 來將提出請求主體的 Amazon Resource Name (ARN)) 與您在政策中所指定的 ARN 進行比較。或者，使用 `aws:SourceArn` 全域條件鍵，將提出服務對服務請求之資源的 Amazon Resource Name (ARN) 與您在政策中所指定的 ARN 進行比較。

相關用語

- AWS 全域條件內容索引鍵

錯誤 – 不支援的主體

問題代碼：UNSUPPORTED_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported principal: The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element."
```

解決錯誤

`Principal` 元素會指定允許或拒絕存取資源的主體。您無法在以 IAM 身分為基礎的政策中使用 `Principal` 元素。您可以在 IAM 角色和以資源為基礎政策的信任政策中使用該元素。資源型政策是您直接內嵌在資源中的政策。例如，您可以在 Amazon S3 儲存貯體或 AWS KMS 金鑰中嵌入政策。

相關用語

- AWS JSON 政策元素：主體
- IAM 中的跨帳戶資源存取

錯誤 – 政策中不支援的資源 ARN

問題代碼：UNSUPPORTED_RESOURCE_ARN_IN_POLICY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-based policy attached to the resource type {{resourceType}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The resource ARN is not supported for the resource-based policy attached to the resource type {{resourceType}}."
```

解決錯誤

當資源型政策連接至不同的資源類型時，部分資源 ARN 在此政策的 `Resource` 元素中不受支援。例如，Amazon S3 儲存貯體政策的 `Resource`元素不支援 AWS KMS ARNs。指定連接至資源型政策的資源類型支援的資源 ARN。

相關用語

- JSON 政策元素：動作

錯誤 – 不支援的 Sid

問題代碼：UNSUPPORTED_SID

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported Sid: Update the characters in the Sid element to use one of the following character types: [a-z, A-Z, 0-9]
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Update the characters in the Sid element to use one of the following character types: [a-z, A-Z, 0-9]"
```

解決錯誤

`Sid` 元素支援大寫字母，小寫字母和數字。

相關用語

- IAM JSON 政策元素：Sid

錯誤 – 主體中不支援的萬用字元

問題代碼：UNSUPPORTED_WILDCARD_IN_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

解決錯誤

`Principal` 元素結構支援使用鍵值組。政策中指定的主體值包括萬用字元 (*)。您不能在指定的主體鍵中包含萬用字元。例如，當您在 `Principal` 元素中指定使用者時，您無法使用萬用字元來表示「所有使用者」。您必須命名特定的一個或多個使用者。同樣地，當您指定擔任的角色工作階段時，您無法使用萬用字元來表示「所有工作階段」。您必須命名特定工作階段。您同樣不能使用萬用字元來符合部分名稱或 ARN。

若要解決此問題清單，請移除萬用字元，並提供更具體的主體。

相關用語

- AWS JSON 政策元素：主體

錯誤 – 變數中缺少大括號

問題代碼：MISING_BRACE_IN_VARIABLE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

解決錯誤

政策變數結構支援使用 $ 字首，後面接著一對大括號 ({ })。在 ${ } 字元內，包含要在政策中想要使用的請求中的值的名稱。

若要解決此問題清單，請新增遺失的大括號，以確定一組完整的大括號開頭和結尾已存在。

相關用語

- IAM 政策元素：變數

錯誤 – 變數中缺少引號

問題代碼：MISSING_QUOTE_IN_VARIABLE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

解決錯誤

將變數新增至政策時，您可以指定變數的預設值。如果變數不存在， AWS 會使用您提供的預設文字。

若要將預設值新增到一個變數，請以單引號 (' ') 括住預設值，並使用逗號和空格 (,) 分隔變數文字和預設值。

例如，如果主體標記為 team=yellow，他們可以存取名為 amzn-s3-demo-bucket-yellow 的 amzn-s3-demo-bucket Amazon S3 儲存貯體。具有此資源的政策可讓團隊成員存取其自己的資源，但不能存取其他團隊的資源。對於沒有團隊標籤的使用者，您可以將預設值設定為 company-wide。這些使用者只能存取 amzn-s3-demo-bucket-company-wide 儲存貯體，其中他們可以檢視廣泛的資訊，例如加入團隊的指示。

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

相關用語

- IAM 政策元素：變數

錯誤 – 變數中不支援的空間

問題代碼：UNSUPPORTED_SPACE_IN_VARIABLE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

解決錯誤

政策變數結構支援使用 $ 字首，後面接著一對大括號 ({ })。在 ${ } 字元內，包含要在政策中想要使用的請求中的值的名稱。雖然您在指定預設變數時可以包含空格，但不能在變數名稱中包含空格。

相關用語

- IAM 政策元素：變數

錯誤 – 空的變數

問題代碼：EMPTY_VARIABLE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

解決錯誤

政策變數結構支援使用 $ 字首，後面接著一對大括號 ({ })。在 ${ } 字元內，包含要在政策中想要使用的請求中的值的名稱。

相關用語

- IAM 政策元素：變數

錯誤 – 元素中不支援變數

問題代碼：VARIABLE_UNSUPPORTED_IN_ELEMENT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

解決錯誤

您可以在 Resource 元素中使用政策變數，也可以在 Condition 元素中使用字串比較。

相關用語

- IAM 政策元素：變數

錯誤 – 版本中不支援變數

問題代碼：VARIABLE_UNSUPPORTED_IN_VERSION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

解決錯誤

若要使用政策變數，必須包含 Version 元素，並將其設定為支援政策變數的版本。變數已導入版本 2012-10-17。舊版的政策語言不支援政策變數。如果您未將 Version 設定為 2012-10-17 或更高版本，則 ${aws:username} 等變數會被視為政策中的常值字串。

Version 政策元素與政策版本不同。Version 政策元素是在政策內使用，並定義政策語言的版本。政策版本會在您在 IAM 中變更客戶受管政策時建立。變更的政策不會覆寫現有的政策。反而，IAM 會建立新版本的受管政策。

相關用語

- IAM 政策元素：變數
- IAM JSON 政策元素：版本

錯誤 – 私有 IP 地址

問題代碼：PRIVATE_IP_ADDRESS

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

解決錯誤

全域條件鍵 aws:SourceIp 僅適用於公有 IP 地址範圍。當您的政策只允許私有 IP 地址時，您會收到此錯誤。在這種情況下，條件永遠不會符合。

- aws:SourceIp 全域條件鍵

- IAM JSON 政策元素：條件

錯誤 – 私有 NotIpAddress

問題代碼：PRIVATE_NOT_IP_ADDRESS

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

解決錯誤

全域條件鍵 `aws:SourceIp` 僅適用於公有 IP 地址範圍。當您使用 `NotIpAddress` 條件運算子，並只列出私有 IP 位址時，您會收到這個錯誤。在此情況下，條件會一律符合且為無效。

- aws:SourceIp 全域條件鍵
- IAM JSON 政策元素：條件

錯誤 – 政策大小超過 SCP 配額

問題代碼：POLICY_SIZE_EXCEEDS_SCP_QUOTA

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

解決錯誤

AWS Organizations 服務控制政策 SCPs) 支援在 `Action`或 `NotAction`元素中指定值。不過，這些值只能在字串結尾包含萬用字元 (*)。這表示您可指定 `iam:Get*` 而不是 `iam:*role`。

若要指定多個動作， AWS 建議您個別列出這些動作。

相關用語

- AWS Organizations 配額
- AWS Organizations 服務控制政策

錯誤 — 無效的服務委託人格式

問題代碼：INVALID_SERVICE_PRINCIPAL_FORMAT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

解決錯誤

條件鍵值組中的值必須符合已定義的服務主體格式。

服務主體是用來將許可授予給服務的識別符。您可以在 Principal 元素中指定服務主體，或將其作為某些全域條件鍵和服務特定鍵的值。服務主體是由每個服務定義。

服務主體的識別符包含服務名稱，而且通常採用以下全小寫字母格式：

service-name.amazonaws.com

某些服務特定鍵可能會針對服務主體使用不同的格式。例如，kms:ViaService 條件鍵需要下列全小寫字母格式的服務主體：

service-name.AWS_region.amazonaws.com

相關用語

- 服務主體
- AWS 全域條件索引鍵
- kms:ViaService 條件鍵

錯誤 – 條件中缺少標籤鍵

問題代碼：MISING_TAG_KEY_IN_CONDITION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

解決錯誤

若要根據標籤控制存取，則在政策的條件元素中提供標籤資訊。

例如，若要控制對 AWS 資源的存取，請包含 aws:ResourceTag條件金鑰。此鍵需要格式 aws:ResourceTag/*tag-key*。如需指定條件中的標籤鍵 owner 和標籤值 JaneDoe，請使用下列格式。

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
}
```

相關用語

- 使用標籤控制存取權限
- 條件
- 全域條件鍵
- AWS 服務條件索引鍵

錯誤 - vpc 格式無效

問題代碼：INVALID_VPC_FORMAT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

解決錯誤

此 aws:SourceVpc 條件鍵必須使用字首 vpc-，後跟 8 或 17 個英數字元，例如 vpc-11223344556677889 或 vpc-12345678。

相關用語

- AWS 全域條件索引鍵：aws：SourceVpc

錯誤 - vpce 格式無效

問題代碼：INVALID_VPCE_FORMAT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

解決錯誤

此 aws:SourceVpce 條件鍵必須使用字首 vpce-，後跟 8 或 17 個英數字元，例如 vpce-11223344556677889 或 vpce-12345678。

相關用語

- AWS 全域條件索引鍵：aws：SourceVpce

錯誤 - 不支援聯合身分主體

問題代碼：FEDERATED_PRINCIPAL_NOT_SUPPORTED

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

解決錯誤

此 `Principal` 元素將聯合身分主體用於連接至 IAM 角色的信任政策，以透過聯合身分提供存取權。身分政策和其他資源型曾策不支援 `Principal` 元素中的聯合身分提供者。例如，您無法在 Amazon S3 儲存貯體政策中使用 SAML 主體。將 `Principal` 元素變更為支援的主體類型。

相關用語

- 為聯合身分建立角色
- JSON 政策元素：主體

錯誤 - 條件鍵不支援的動作

問題代碼：UNSUPPORTED_ACTION_FOR_CONDITION_KEY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
 actions. We recommend that you move these actions to a different statement without
 this condition key.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
```

```
recommend that you move these actions to a different statement without this condition
key."
```

解決錯誤

請確保政策陳述式的 Condition 元素中的條件鍵會套用至 Action 元素中的每個動作。若要確保政策有效地允許或拒絕您指定的動作，您應該將不支援的動作移至不同的陳述式，而不使用條件鍵。

> **Note**
>
> 如果 Action 元素具有包含萬用字元的動作，則 IAM Access Analyzer 不會評估這些動作是否存在此錯誤。

相關用語

- JSON 政策元素：動作

錯誤 - 政策中不支援的動作

問題代碼：UNSUPPORTED_ACTION_IN_POLICY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

解決錯誤

在連接至不同資源類型的資源型政策中，部分動作在其 Action 元素中不受支援。例如，Amazon S3 儲存貯體政策不支援 AWS Key Management Service 動作。指定連接至資源型政策的資源類型支援的動作。

相關用語

- JSON 政策元素：動作

錯誤 – 政策中不支援的資源 ARN

問題代碼：UNSUPPORTED_RESOURCE_ARN_IN_POLICY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-based policy attached to the resource type {{resourceType}}.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The resource ARN is not supported for the resource-based policy attached to the resource type {{resourceType}}."
```

解決錯誤

當資源型政策連接至不同的資源類型時，部分資源 ARN 在此政策的 `Resource` 元素中不受支援。例如，Amazon S3 儲存貯體政策的 `Resource`元素不支援 AWS KMS ARNs。指定連接至資源型政策的資源類型支援的資源 ARN。

相關用語

- JSON 政策元素：動作

錯誤 - 服務主體不支援的條件鍵

問題代碼：UNSUPPORTED_CONDITION_KEY_FOR_SERVICE_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

解決錯誤

您可以使用服務委託人在資源型政策的 `Principal`元素 AWS 服務 中指定 ，這是服務的識別符。您無法將某些條件鍵與特定服務主體搭配使用。例如，您無法將 `aws:PrincipalOrgID` 條件鍵與服務主體 `cloudfront.amazonaws.com` 搭配使用。您應該移除不適用於 `Principal` 元素中服務主體的條件鍵。

相關用語

- 服務主體
- JSON 政策元素：主體

錯誤 – 角色信任政策語法錯誤：NotPrincipal

問題代碼：ROLE_TRUST_POLICY_SYNTAX_ERROR_NOTPRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

解決錯誤

角色信任政策是連接至 IAM 角色的以資源為基礎的政策。信任政策會定義哪些主體實體 (帳戶、使用者、角色和聯合身分使用者) 可擔任該角色。角色信任原則不支援 NotPrincipal。更新政策以改為使用 Principal 元素。

相關用語

- JSON 政策元素：主體
- JSON 政策元素：NotPrincipal

錯誤 – 角色信任政策不支援在主體中使用萬用字元

問題代碼：ROLE_TRUST_POLICY_UNSUPPORTED_WILDCARD_IN_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

解決錯誤

角色信任政策是連接至 IAM 角色的以資源為基礎的政策。信任政策會定義哪些主體實體 (帳戶、使用者、角色和聯合身分使用者) 可擔任該角色。在角色信任政策的 Principal 元素中不支援 "Principal:" "*"。以有效的主體值取代萬用字元。

相關用語

- JSON 政策元素：主體

錯誤 – 角色信任政策語法錯誤：Resource

問題代碼：ROLE_TRUST_POLICY_SYNTAX_ERROR_RESOURCE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

解決錯誤

角色信任政策是連接至 IAM 角色的以資源為基礎的政策。信任政策會定義哪些主體實體 (帳戶、使用者、角色和聯合身分使用者) 可擔任該角色。角色信任政策會套用至它們所連接的角色。您不能在角色信任政策中指定 `Resource` 或 `NotResource` 元素。移除 `Resource` 或 `NotResource` 元素。

- JSON 政策元素：Resource
- JSON 政策元素：NotResource

錯誤 – 類型與 IP 範圍不符

問題代碼：TYPE_MISMATCH_IP_RANGE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

解決錯誤

更新文字以使用 CIDR 格式的 IP 地址條件運算子資料類型。

相關用語

- IP 地址條件運算子
- IAM JSON 政策元素：條件運算子

錯誤 – 缺少條件鍵動作

問題代碼：MISING_ACTION_FOR_CONDITION_KEY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

解決錯誤

除非指定的動作位於 `Action` 元素中，否則不會評估政策陳述式的 `Condition` 元素中的條件鍵。若要確保政策有效地允許或拒絕您指定的條件鍵，請將動作新增至 `Action` 元素。

相關用語

- JSON 政策元素：動作

錯誤 – 角色信任政策中的聯合主體語法無效

問題代碼：INVALID_FEDERATED_PRINCIPAL_SYNTAX_IN_ROLE_TRUST_POLICY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

解決錯誤

主體值指定與預期格式不相符的聯合主體。將聯合主體的格式更新為有效的網域名稱或 SAML 中繼資料 ARN。

相關用語

- 聯合身分使用者和角色

錯誤 – 主體的不相符動作

問題代碼：MISMATCHED_ACTION_FOR_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

解決錯誤

對於 `Principal` 元素中指定的主體，政策陳述式的 `Action` 元素中指定的動作無效。例如，您無法將 SAML 提供者主體與 `sts:AssumeRoleWithWebIdentity` 動作結合使用。您應該將 SAML 提供者主體與 `sts:AssumeRoleWithSAML` 動作結合使用，或者將 OIDC 提供者主體與 `sts:AssumeRoleWithWebIdentity` 動作結合使用。

相關用語

- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity

錯誤 – 缺少 Roles Anywhere 信任政策的動作

問題代碼：MISSING_ACTION_FOR_ROLES_ANYWHERE_TRUST_POLICY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

解決錯誤

為了讓 IAM Roles Anywhere 能夠擔任角色並提供臨時的 AWS 憑證，角色必須信任 IAM Roles Anywhere 服務主體。IAM Roles Anywhere 服務主體需要 `sts:AssumeRole`、`sts:SetSourceIdentity` 和 `sts:TagSession` 許可來擔任角色。如果缺少這些許可中的任何一個，您必須將它們新增至政策。

相關用語

- AWS Identity and Access Management Roles Anywhere 中的信任模型

錯誤 – 政策大小超過 RCP 配額

問題代碼：POLICY_SIZE_EXCEEDS_RCP_QUOTA

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Policy size exceeds RCP quota: The {{policySize}} characters in the resource control
 policy (RCP) exceed the {{policySizeQuota}} character maximum for RCPs. We recommend
 that you use multiple granular policies.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{policySize}} characters in the resource control policy (RCP)
 exceed the {{policySizeQuota}} character maximum for RCPs. We recommend that you use
 multiple granular policies."
```

解決錯誤

AWS Organizations 資源控制政策 RCPs) 支援在 `Action`元素中指定值。不過，這些值只能在字串結尾包含萬用字元 (*)。這表示您可指定 `s3:Get*` 而不是 `s3:*Object`。

若要指定多個動作， AWS 建議您個別列出這些動作。

相關用語

- 的配額 AWS Organizations
- AWS Organizations 資源控制政策

錯誤 – RCP 語法錯誤主體

問題代碼：RCP_SYNTAX_ERROR_PRINCIPAL

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
RCP syntax error principal: The Principal element contents are not valid. RCPs only
 support specifying all principals ("*") in the Principal element. The NotPrincipal
 element is not supported for RCPs.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The Principal element contents are not valid. RCPs only support
 specifying all principals ("*") in the Principal element. The NotPrincipal element is
 not supported for RCPs."
```

解決錯誤

AWS Organizations 資源控制政策 RCPs) 僅支援指定 `Principal`元素中的所有主體 ("*")。RCP 不支援 `NotPrincipal` 元素。

相關用語

- RCP 語法
- 主體的屬性

錯誤 – RCP 語法錯誤允許

問題代碼：RCP_SYNTAX_ERROR_ALLOW

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
RCP syntax error allow: RCPs only support specifying all principals ("*") in the
 Principal element, all resources ("*") in the Resource element, and no Condition
 element with an effect of Allow.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "RCPs only support specifying all principals ("*") in the Principal
 element, all resources ("*") in the Resource element, and no Condition element with an
 effect of Allow."
```

解決錯誤

AWS Organizations 資源控制政策 RCPs) 僅支援指定 `Principal`元素中的所有主體 ("*") 和 `Resource`元素中的所有資源 ("*")。RCP 不支援具有 `Allow` 效果的 `Condition` 元素。

相關用語

- RCP 語法
- 主體的屬性
- 資源的屬性

錯誤 – RCP 語法錯誤 NotAction

問題代碼：RCP_SYNTAX_ERROR_NOTACTION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
RCP syntax error NotAction: RCPs do not support the NotAction element. Update to use
 the Action element.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "RCPs do not support the NotAction element. Update to use the Action
 element."
```

解決錯誤

AWS Organizations 資源控制政策 RCPs) 不支援 `NotAction`元素。請使用 `Action` 元素。

相關用語

- RCP 語法
- IAM JSON 政策元素：動作
- IAM JSON 政策元素：NotAction

錯誤 – RCP 語法錯誤動作

問題代碼：RCP_SYNTAX_ERROR_ACTION

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
RCP syntax error action: RCPs only support specifying select service prefixes in the
 Action element. Learn more here.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "RCPs only support specifying select service prefixes in the Action
 element. Learn more here."
```

解決錯誤

AWS Organizations 資源控制政策 RCPs) 僅支援在 `Action`元素中指定選取服務字首。

相關用語

- RCP 語法

- AWS 服務 支援 RCPs 清單

錯誤 – Missing ARN account

問題代碼：MISSING_ARN_ACCOUNT

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing ARN account: The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id."
```

解決錯誤

在資源 ARN 中加入帳戶 ID。帳戶 ID 是 12 位數的整數。若要了解如何檢視您的帳戶 ID，請參閱尋找 AWS 您的帳戶 ID。

相關用語

- 政策資源
- 帳戶識別符
- 資源 ARN
- AWS 具有 ARN 格式的服務資源

錯誤 – Invalid kms key value

問題代碼：INVALID_KMS_KEY_VALUE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid kms key value: The {{key}} condition key value must be a valid KMS key ARN.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{key}} condition key value must be a valid KMS key ARN."
```

解決錯誤

AWS KMS key ARN (Amazon Resource Name) 是 KMS 金鑰的唯一完整識別符。金鑰 ARN 包含 AWS 帳戶、 區域和金鑰 ID。金鑰 ARN 遵循以下格式：

`arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`*

相關用語

- 尋找金鑰 ID 和金鑰 ARN
- 金鑰 ARN
- AWS 全域條件內容索引鍵

錯誤 – Variable usage too permissive

問題代碼：VARIABLE_USAGE_TOO_PERMISSIVE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Variable usage too permissive: Overly permissive use of policy variable for the {{key}}
 condition key. Use the policy variable preceded by 6 consecutive characters.
```

```
The policy variable is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and policy variables can be evaluated as effective wildcards. Therefore
 policy variables are not allowed to be used with sensitive keys. Refer to https://
docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-keys.html for the
 list of sensitive keys.
```

```
Overly permissive use of policy variable with aws:userID. Use the policy variable on
 the right side of colon or as the only character on the left side of a colon.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Overly permissive use of policy variable for the {{key}} condition
 key. Use the policy variable preceded by 6 consecutive characters."
```

```
"findingDetails": "The policy variable is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and policy variables can be evaluated as
 effective wildcards. Therefore policy variables are not allowed to be used
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of policy variable with aws:userID. Use the
 policy variable on the right side of colon or as the only character on the left side
 of a colon."
```

解決錯誤

此錯誤有三種可能的調查結果訊息。

- 對於第一種錯誤訊息，請修改政策變數用途，使變數用途更加具體。在政策變數之前新增至少 6 個連續字元，以縮小許可範圍。例如，不使用 `${aws:username}`，而是改用 `prefix-${aws:username}` 或 `myapp-${aws:username}`。這樣可以確保政策變數不會授予過於廣泛的存取權。
- 對於第二種錯誤訊息，請從指定的條件索引鍵中移除政策變數。政策變數可以充當有效的萬用字元，但基於安全考量，不允許將其與敏感條件索引鍵一起使用。請改用特定的靜態值，或考慮重組政策，以使用支援政策變數的非敏感條件索引鍵。
- 對於第三種錯誤訊息，請修改 `aws:userID` 政策變數用途，更加嚴格地限制其用途。將政策變數放在冒號的右側 (在帳戶 ID 之後)，或使其作為冒號左側的唯一字元。例如，使用 `AIDACKCEVSQ6C2EXAMPLE:${aws:userid}` 或 `${aws:userid}:*`，而不使用 `${aws:userid}`。

相關用語

- IAM 政策元素：變數和標籤

- IAM 政策元素：Condition
- 具有多個內容索引鍵或值的條件
- AWS 全域條件內容索引鍵

錯誤 – Wildcard usage too permissive

問題代碼：WILDCARD_USAGE_TOO_PERMISSIVE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Wildcard usage too permissive: Overly permissive use of wildcard for the {{key}}
 condition key. Use the wildcard preceded by 6 consecutive characters.
```

```
The wildcard is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and wildcards are not allowed to be used with sensitive keys. Refer
 to https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-
keys.html for the list of sensitive keys.
```

```
Overly permissive use of wildcard with aws:userID. Use the wildcard on the right side
 of colon or as the only character on the left side of a colon.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Overly permissive use of wildcard for the {{key}} condition key. Use
 the wildcard preceded by 6 consecutive characters."
```

```
"findingDetails": "The wildcard is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and wildcards are not allowed to be used
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of wildcard with aws:userID. Use the wildcard
 on the right side of colon or as the only character on the left side of a colon."
```

解決錯誤

此錯誤有三種可能的調查結果訊息。

- 對於第一種錯誤訊息，請在萬用字元之前新增至少 6 個連續字元，使萬用字元用途更加具體。例如，不使用 `*`，而是改用 `prefix-*` 或 `prefix-*-suffix`。這樣可以縮小條件的範圍，並遵循最低權限原則。
- 對於第二種錯誤訊息，請從指定的條件索引鍵中移除萬用字元。基於安全考量，不允許將萬用字元與敏感條件索引鍵一起使用。將萬用字元取代為符合預期存取模式的特定值，或考慮使用支援萬用字元的其他非敏感條件索引鍵。
- 對於第三種錯誤訊息，請修改 `aws:userID` 萬用字元用途，更加嚴格地限制其用途。將萬用字元放在冒號的右側 (在帳戶 ID 之後)，或使其作為冒號左側的唯一字元。例如，使用 `AIDACKCEVSQ6C2EXAMPLE:*` 或 `*:*`，而不使用 `*`。

相關用語

- IAM 政策元素：Condition
- 具有多個內容索引鍵或值的條件
- AWS 全域條件內容索引鍵
- IAM 識別符

錯誤 – DynamoDB 無效傳回耗用容量

問題代碼：DYNAMODB_INVALID_RETURN_CONSUMED_CAPACITY

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB invalid return consumed capacity: The value for
 dynamodb:ReturnConsumedCapacity is not valid. Update {{value}} to either TOTAL or
 NONE.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The value for dynamodb:ReturnConsumedCapacity is not valid. Update
 {{value}} to either TOTAL or NONE."
```

解決錯誤

將條件值更新為 DynamoDB 提供的兩個值之一： TOTAL或 NONE。值在 StringEquals和 下區分大小寫StringLike，因此 等小寫變體total無效。StringEqualsIgnoreCase 運算子接受任何大小寫。指定無效值的條件永遠不會符合任何請求，因此陳述式沒有效果。

錯誤 – DynamoDB 無效傳回值

問題代碼：DYNAMODB_INVALID_RETURN_VALUES

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB invalid return values: The value for dynamodb:ReturnValues is not valid.
 Update {{value}} to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The value for dynamodb:ReturnValues is not valid. Update {{value}}
 to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW."
```

解決錯誤

將條件值更新為 DynamoDB 提供的五個值之一：NONE、ALL_OLDUPDATED_OLD、ALL_NEW、或 UPDATED_NEW。在 StringEquals和 下，值區分大小寫StringLike，因此 等小寫變體updated_new無效。StringEqualsIgnoreCase 運算子接受任何大小寫。指定無效值的條件永遠不會符合任何請求，因此陳述式沒有效果。

錯誤 – DynamoDB 無效的選取值

問題代碼：DYNAMODB_INVALID_SELECT_VALUE

調查結果類型：ERROR

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB invalid select value: The value for dynamodb:Select is not valid. Update
 {{value}} to one of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or
 COUNT.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The value for dynamodb:Select is not valid. Update {{value}} to one
 of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or COUNT."
```

解決錯誤

將條件值更新為 DynamoDB 提供的四個值之一：ALL_ATTRIBUTES、SPECIFIC_ATTRIBUTES、ALL_PROJECTED_ATTRIBUTES或 COUNT。在 StringEquals和 下，值區分大小寫StringLike，因此 等小寫變體specific_attributes無效。StringEqualsIgnoreCase 運算子接受任何大小寫。指定無效值的條件永遠不會符合任何請求，因此陳述式沒有效果。

一般警告 – 使用 NotResource 建立 SLR

問題代碼：CREATE_SLR_WITH_NOT_RESOURCE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

解決一般警告

動作會iam:CreateServiceLinkedRole授予許可，以建立允許 AWS 服務代表您執行動作的 IAM 角色。在政策中使用 iam:CreateServiceLinkedRole 與 NotResource 元素可允許為多個資源建立非預期的服務連結角色。 AWS 建議您改為在 Resource 元素中指定允許的 ARN。

- CreateServiceLinkedRole 操作
- IAM JSON 政策元素：NotResource
- IAM JSON 政策元素：資源

一般警告 – 使用動作中的星號和 NotResource 建立 SLR

問題代碼：CREATE_SLR_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

解決一般警告

動作會`iam:CreateServiceLinkedRole`授予許可，以建立允許 AWS 服務代表您執行動作的 IAM 角色。`Action` 中包含萬用字元 (*) 與包含 `NotResource` 元素的政策可允許為多個資源建立非預期的服務連結角色。 AWS 建議您改為在 `Resource` 元素中指定允許的 ARN。

- CreateServiceLinkedRole 操作
- IAM JSON 政策元素：NotResource
- IAM JSON 政策元素：資源

一般警告 – 使用 NotAction 和 NotResource 建立 SLR

問題代碼：CREATE_SLR_WITH_NOT_ACTION_AND_NOT_RESOURCE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

解決一般警告

動作會`iam:CreateServiceLinkedRole`授予許可，以建立允許 AWS 服務代表您執行動作的 IAM 角色。使用 `NotAction` 元素與 `NotResource` 元素可允許為多個資源建立非預期的服務連結角色。 AWS 建議您重寫政策，以改為在 `Resource` 元素中允許有限 ARN 清單上的 `iam:CreateServiceLinkedRole`。您也可以將 `iam:CreateServiceLinkedRole` 新增到 `NotAction` 元素。

- CreateServiceLinkedRole 操作
- IAM JSON 政策元素：NotAction
- IAM JSON 政策元素：動作
- IAM JSON 政策元素：NotResource
- IAM JSON 政策元素：資源

一般警告 – 使用資源中的星號建立 SLR

問題代碼：CREATE_SLR_WITH_STAR_IN_RESOURCE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

解決一般警告

動作會`iam:CreateServiceLinkedRole`授予許可，以建立允許 AWS 服務代表您執行動作的 IAM 角色。在政策中使用 `iam:CreateServiceLinkedRole` 與 `Resource` 元素中的萬用字元 (*) 可允許為多個資源建立非預期的服務連結角色。 AWS 建議您改為在 `Resource` 元素中指定允許的 ARN。

- CreateServiceLinkedRole 操作
- IAM JSON 政策元素：資源

AWS 具有此一般警告的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

其中一些使用案例適用於您帳戶中的進階使用者。下列 AWS 受管政策提供進階使用者存取權，並授予許可，為任何 服務建立服務連結角色。 AWS 建議您僅將下列 AWS 受管政策連接到您認為進階使用者的 IAM 身分。 AWS

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy – 此 AWS 受管政策提供由 AWS Organizations 服務連結角色使用的許可。此角色可讓 Organizations 為 AWS 組織中的其他 服務建立額外的服務連結角色。

一般警告 – 使用動作中的星號和資源建立 SLR

問題代碼：CREATE_SLR_WITH_STAR_IN_ACTION_AND_RESOURCE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

解決一般警告

動作會`iam:CreateServiceLinkedRole`授予許可，以建立允許 AWS 服務代表您執行動作的 IAM 角色。`Action` 和 `Resource` 元素中包含萬用字元 (*) 的政策可允許為多個資源建立非預期的服務連結角色。這可讓您在指定 `"Action": "*"`、 `"Action": "iam:*"`或 時建立服務連結角色`"Action": "iam:Create*"`。 AWS 建議您改為在 `Resource`元素中指定允許的 ARNs。

- CreateServiceLinkedRole 操作
- IAM JSON 政策元素：動作
- IAM JSON 政策元素：資源

AWS 具有此一般警告的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

其中一些使用案例適用於您帳戶中的系統管理員。下列 AWS 受管政策提供管理員存取權，並授予許可，為任何 服務建立服務連結角色。 AWS 建議您僅將下列 AWS 受管政策連接到您認為是管理員的 IAM 身分。 AWS

- AdministratorAccess
- IAMFullAccess

一般警告 – 使用資源中的星號和 NotAction 建立 SLR

問題代碼：CREATE_SLR_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

解決一般警告

動作會`iam:CreateServiceLinkedRole`授予許可，以建立允許 AWS 服務代表您執行動作的 IAM 角色。在 `Resource` 元素中包含萬用字元 (*) 的政策中使用 `NotAction` 元素可允許為多個資源建立非預期的服務連結角色。 AWS 建議您改為在 `Resource` 元素中指定允許的 ARN。您也可以將 `iam:CreateServiceLinkedRole` 新增到 `NotAction` 元素。

- CreateServiceLinkedRole 操作
- IAM JSON 政策元素：NotAction
- IAM JSON 政策元素：動作
- IAM JSON 政策元素：資源

一般警告 — 已取代的全域條件鍵

問題代碼：DEPRECATED_GLOBAL_CONDITION_KEY

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

解決一般警告

政策包含已取代的全域條件鍵。更新條件鍵值組中的條件鍵，以使用支援的全域條件鍵。

- 全域條件鍵

一般警告 – 無效的日期值

問題代碼：INVALID_DATE_VALUE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

解決一般警告

Unix Epoch 時間描述了自 1970 年 1 月 1 日以來經過的時間點，減去閏秒。Epoch 時間可能無法解析為您預期的精確時間。 AWS 建議您針對日期和時間格式使用 W3C 標準。例如，您可以指定完整的日期，例如 YYYY-MM-DD (1997-07-16)，或者您也可以將時間附加到秒，例如 YYYY-MM-DDThh:mm:ssTZD (1997-07-16T19:20:30+01:00)。

- W3C 日期和時間格式

- IAM JSON 政策元素：版本
- aws:CurrentTime 全域條件鍵

一般警告 – 角色參考無效

問題代碼：INVALID_ROLE_REFERENCE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

解決一般警告

AWS 建議您為 IAM 角色指定 Amazon Resource Name (ARN)，而不是其主體 ID。當 IAM 儲存政策時，它會將 ARN 轉換為現有角色的委託人 ID。 AWS 包含安全預防措施。如果有人刪除並重新建立角色，則會有新的 ID，且政策不會符合新角色的 ID。

- 指定主體：IAM 角色
- IAM ARN
- IAM 唯一 ID

一般警告 – 無效的使用者參考

問題代碼：INVALID_USER_REFERENCE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

解決一般警告

AWS 建議您為 IAM 使用者指定 Amazon Resource Name (ARN)，而不是其主體 ID。當 IAM 儲存政策時，它會將 ARN 轉換為現有使用者的委託人 ID。 AWS 包含安全預防措施。如果有人刪除並重新建立使用者，則會有新的 ID，且政策不會符合新使用者的 ID。

- 指定主體：IAM 使用者
- IAM ARN
- IAM 唯一 ID

一般警告 – 遺失版本

問題代碼：MISING_VERSION

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

解決一般警告

AWS 建議您在政策中包含選用Version參數。如果您未包含 Version 元素，則值預設為 2012-10-17，但較新功能 (例如政策變數) 將無法使用您的政策。例如，${aws:username} 這類變數無法辨識為變數，而是視為政策中的文字字串。

- IAM JSON 政策元素：版本

一般警告 – 建議使用獨特 Sid

問題代碼：UNIQUE_SIDS_RECOMMENDED

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

解決一般警告

AWS 建議您使用唯一的陳述式 IDs。Sid (陳述式 ID) 元素允許您輸入您為政策陳述式提供的可選識別符。您可以使用 SID 元素將陳述式 ID 值指派給陳述式陣列中的每個陳述式。

相關用語

- IAM JSON 政策元素：Sid

一般警告 – 沒有類似運算子的萬用字元

問題代碼：WILDCARD_WITHOUT_LIKE_OPERATOR

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

解決一般警告

Condition 元素結構會要求您使用條件運算子和鍵值組。當您指定使用萬用字元 (*、?) 的條件值時，必須使用 Like 版本的條件運算子。例如，並非使用 StringEquals 字串條件運算子,而是使用 StringLike。

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- IAM JSON 政策元素：條件運算子
- IAM JSON 政策元素：條件

AWS 具有此一般警告的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

下列 AWS 受管政策在其條件值中包含萬用字元，而沒有包含 Like 用於模式比對的條件運算子。使用 AWS 受管政策做為建立客戶受管政策的參考時， AWS 建議您使用支援模式比對的條件運算子與萬用字元 (*、 ？)，例如 StringLike。

- AWSGlueConsoleSageMakerNotebookFullAccess

一般警告 – 政策大小超過身分政策配額

問題代碼：POLICY_SIZE_EXCEEDS_IDENTITY_POLICY_QUOTA

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
 maximum for inline and managed policies. We recommend that you use multiple granular
 policies.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

解決一般警告

您最多可以將 20 個受管政策連接至 IAM 身分（使用者、使用者群組或角色）。但是，每個受管政策的大小不可超過 6,144 個字元的預設配額。在對此配額計算政策的大小時，IAM 不會計算空格數。配額也稱為 中的限制 AWS，是帳戶中 AWS 資源、動作和項目的最大值。

此外，您可以新增任意數量的內嵌政策到 IAM 身分。不過，每個身分的所有內嵌政策大小總計不能超過指定的配額。

如果您的政策大於配額，您可以將政策組織成多個陳述式，並將這些陳述式群組成多個政策。

相關用語

- IAM 和 AWS STS 字元配額
- 多項陳述式和多個政策
- IAM 客戶受管政策
- JSON 政策概觀
- IAM JSON 政策文法

AWS 具有此一般警告的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

下列 AWS 受管政策會將許可授予多個 AWS 服務的動作，並超過政策大小上限。當您使用 AWS 受管政策做為建立受管政策的參考時，必須將政策分割為多個政策。

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

一般警告 – 政策大小超過資源政策配額

問題代碼：POLICY_SIZE_EXCEEDS_RESOURCE_POLICY_QUOTA

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

解決一般警告

資源型政策是連接到資源 (如 Amazon S3 儲存貯體) 的 JSON 政策文件。這些政策會授予指定的主體許可，允許在該資源上執行特定的動作，並且定義資源所適用的條件。資源型政策的大小不能超過為該資源設定的配額。配額也稱為 中的限制 AWS，是 AWS 帳戶中資源、動作和項目的最大值。

如果您的政策大於配額，您可以將政策組織成多個陳述式，並將這些陳述式群組成多個政策。

相關用語

- 資源型政策
- 使用 Amazon S3 儲存貯體政策
- 多項陳述式和多個政策
- JSON 政策概觀
- IAM JSON 政策文法

一般警告 – 類型不符

問題代碼：TYPE_MISMATCH

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

解決一般警告

更新文字以使用支援的條件運算子資料類型。

例如，aws:MultiFactorAuthPresent 全域條件鍵需要資料類型為 Boolean 的條件運算子。如果您提供日期或整數，則資料類型將不相符。

相關用語

- 全域條件鍵
- IAM JSON 政策元素：條件運算子

一般警告 – 類型不符布林值

問題代碼：TYPE_MISMATCH_BOOLEAN

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

解決一般警告

更新文字以使用布林值條件運算子資料類型，例如 `true` 或 `false`。

例如，`aws:MultiFactorAuthPresent` 全域條件鍵需要資料類型為 `Boolean` 的條件運算子。如果您提供日期或整數，則資料類型將不相符。

相關用語

- 布林值條件運算子
- IAM JSON 政策元素：條件運算子

一般警告 – 類型不符日期

問題代碼：TYPE_MISMATCH_DATE

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

解決一般警告

更新文字以使用 YYYY-MM-DD 或其他 ISO 8601 日期時間格式的日期條件運算子資料類型。

相關用語

- 日期條件運算子
- IAM JSON 政策元素：條件運算子

一般警告 – 類型不符號碼

問題代碼：TYPE_MISMATCH_NUMBER

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

解決一般警告

更新文字以使用數字條件運算子資料類型。

相關用語

- 數位條件運算子
- IAM JSON 政策元素：條件運算子

一般警告 – 類型不符字串

問題代碼：TYPE_MISMATCH_STRING

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

解決一般警告

更新文字以使用字串條件運算子資料類型。

相關用語

- 字串條件運算子
- IAM JSON 政策元素：條件運算子

一般警告 – 建議使用特定的 github 儲存庫和分支

問題代碼：SPECIFIC_GITHUB_REPO_AND_BRANCH_RECOMMENDED

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name. If the subject claim does not include the branch, specify
 it in a token.actions.githubusercontent.com:ref condition key.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value
 of token.actions.githubusercontent.com:sub with the repository and branch
 name. If the subject claim does not include the branch, specify it in a
 token.actions.githubusercontent.com:ref condition key."
```

解決一般警告

如果您使用 GitHub 做為 OIDC IdP，最佳實務是限制可擔任與 IAM IdP 相關聯角色的實體。當您在角色信任政策中包含 `Condition` 陳述式時，可以將角色限制到特定 GitHub 組織、儲存庫或分支。您可以使用條件鍵 `token.actions.githubusercontent.com:sub`

來限制存取。建議您將條件限制為一組特定的儲存庫或分支。如果您在 中使用萬用字元 (*)`token.actions.githubusercontent.com:sub`，則來自您控制範圍之外的組織或儲存庫的 GitHub 動作可以擔任與您 AWS 帳戶中的 GitHub IAM IdP 相關聯的角色。

在 中指定儲存庫`token.actions.githubusercontent.com:sub`。您可以在相同的值或單獨的`token.actions.githubusercontent.com:ref`條件索引鍵中指定分支。當主體宣告不包含分支時，例如命名GitHub環境時，請在 中指定分支`token.actions.githubusercontent.com:ref`。

相關用語

- 設定 GitHub OIDC 身分提供者的角色

一般警告 – 政策大小超過角色信任政策配額

問題代碼：POLICY_SIZE_EXCEEDS_ROLE_TRUST_POLICY_QUOTA

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
 role trust policy length quota increase using Service Quotas and AWS Support Center.
 If the quotas have already been increased, then you can ignore this warning.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

解決一般警告

IAM 和 AWS STS 具有限制角色信任政策大小的配額。角色信任政策中的字元 (不包括空白) 超過字元上限。我們建議您使用 Service Quotas 和 AWS Support Center Console要求增加角色信任政策長度配額。

相關用語

- IAM AWS STS 和配額、名稱要求和字元限制

一般警告 – RCP 缺少相關主體條件索引鍵

問題代碼：RCP_MISSING_RELATED_PRINCIPAL_CONDITION_KEY

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
RCP missing related principal condition key: RCPs impact IAM roles, users, and AWS
 service principals. To prevent unintended impact to services acting on your behalf
 using a service principal, an additional statement should be added to the Condition
 block "BoolIfExists": { "aws:PrincipalIsAWSService": "false"} whenever a principal key
 {{conditionKeyName}} is used.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to services acting on your behalf using a service principal,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "false"} whenever a principal key {{conditionKeyName}}
 is used."
```

解決一般警告

AWS Organizations 資源控制政策 RCPs) 可能會影響 IAM 角色、使用者和 AWS 服務 主體。若要防止使用服務主體代表您執行的服務受到意外影響，請將下列陳述式新增至您的 Condition 元素：

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "false"}
```

相關用語

- RCP 語法
- 主體的屬性

一般警告 – RCP 缺少相關服務主體條件索引鍵

問題代碼：RCP_MISSING_RELATED_SERVICE_PRINCIPAL_CONDITION_KEY

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
RCP missing related service principal condition key: RCPs impact IAM roles, users,
 and AWS service principals. To prevent unintended impact to your principals,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "true"} whenever the key {{conditionKeyName}} is used.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to your principals, an additional statement should be added to the
 Condition block "BoolIfExists": { "aws:PrincipalIsAWSService": "true"} whenever the
 key {{conditionKeyName}} is used."
```

解決一般警告

AWS Organizations 資源控制政策 RCPs) 可能會影響 IAM 角色、使用者和 AWS 服務 主體。若要避免您的主體受到意外影響，請將下列陳述式新增至您的 Condition 元素：

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "true"}
```

相關用語

- RCP 語法
- 主體的屬性

一般警告 – RCP 缺少服務條件索引鍵 Null 檢查

問題代碼：RCP_MISSING_SERVICE_CONDITION_KEY_NULL_CHECK

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
RCP missing service condition key null check: The specified service may have a service
 integration that does not require the use of the {{conditionKeyName}} condition
 key. To prevent unintended impact to services acting on your behalf using a service
 principal, an additional statement should be added to the Condition block "Null":
 { "aws:SourceAccount": "false"} or "Null": { "aws:SourceArn": "false"} whenever the
 key {{conditionKeyName}} is used.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The specified service may have a service integration that does
 not require the use of the {{conditionKeyName}} condition key. To prevent unintended
 impact to services acting on your behalf using a service principal, an additional
 statement should be added to the Condition block "Null": { "aws:SourceAccount":
 "false"} or "Null": { "aws:SourceArn": "false"} whenever the key {{conditionKeyName}}
 is used."
```

解決一般警告

AWS Organizations 資源控制政策 RCPs) 可能會影響 IAM 角色、使用者和 AWS 服務 主體。若要防止使用服務主體代表您執行的服務受到意外影響，每當使用指定索引鍵時，請將下列陳述式之一新增至您的 Condition 元素：

```
"Null": { "aws:SourceAccount": "false"}
```

或

```
"Null": { "aws:SourceArn": "false"}
```

相關用語

- RCP 語法
- 主體的屬性

一般警告 – 僅搭配支援的服務使用條件索引鍵

問題代碼：USE_CONDITION_KEY_ONLY_WITH_SUPPORTED_SERVICES

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Use condition key only with supported services: The condition key {{key}} works only
 with specific AWS services and must be scoped to supported services in your policies.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key {{key}} works only with specific AWS services and
 must be scoped to supported services in your policies.
```

解決一般警告

檢閱 AWS 文件以識別 AWS 服務 支援此條件金鑰的 。如果政策中的任何服務不支援 條件金鑰，請修改您的政策，將條件金鑰範圍限定為僅 AWS 服務 支援它的 。

相關用語

- aws:VpceAccount
- aws:VpceOrgID
- aws:VpceOrgPaths
- 的動作、資源和條件索引鍵 AWS 服務

一般警告 – DynamoDB 使用屬性取代操作

問題代碼：DYNAMODB_REPLACE_OPERATION_WITH_ATTRIBUTES

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB replace operation with attributes: The {{action}} action replaces the entire
 item, so dynamodb:Attributes cannot restrict access at the attribute level. That
 condition only applies to UpdateItem; use dynamodb:UpdateItem for attribute-level
 access control.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{action}} action replaces the entire item, so
 dynamodb:Attributes cannot restrict access at the attribute level. That condition only
 applies to UpdateItem; use dynamodb:UpdateItem for attribute-level access control."
```

解決一般警告

dynamodb:Attributes 條件不會限制 dynamodb:PutItem或 dynamodb:BatchWriteItem。這些動作會取代整個項目，因此呼叫者只要將其從寫入的項目中保留，就可以捨棄受限制的屬性。若要控制發起人可以變更哪些屬性，請dynamodb:UpdateItem改為授予 並保留該陳述式dynamodb:Attributes的條件。如果您不打算在此處進行屬性層級控制，請移除dynamodb:Attributes條件。

一般警告 – DynamoDB 掃描會略過主索引鍵

問題代碼：DYNAMODB_SCAN_BYPASSES_LEADING_KEYS

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB scan bypasses leading keys: The dynamodb:Scan operation returns all items
 in the table, bypassing the dynamodb:LeadingKeys restriction. Consider removing
 dynamodb:Scan.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The dynamodb:Scan operation returns all items in the table,
 bypassing the dynamodb:LeadingKeys restriction. Consider removing dynamodb:Scan."
```

解決一般警告

dynamodb:Scan 會讀取整個資料表，因此此陳述式中的 dynamodb:LeadingKeys條件不會限制發起人可以讀取的項目。移除 dynamodb:Scan，或在不依賴 的個別陳述式中授予它dynamodb:LeadingKeys。

一般警告 – 動作的 DynamoDB 錯誤條件

問題代碼：DYNAMODB_WRONG_CONDITION_FOR_ACTION

調查結果類型：GENERAL_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB wrong condition for action: The condition key {{conditionKey}} does
 not apply to the {{action}} action. Use dynamodb:Select with read actions and
 dynamodb:ReturnValues with write actions.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key {{conditionKey}} does not apply to the {{action}}
 action. Use dynamodb:Select with read actions and dynamodb:ReturnValues with write
 actions."
```

解決一般警告

此陳述式使用 dynamodb:Select搭配寫入動作，或使用 dynamodb:ReturnValues搭配讀取動作。DynamoDB dynamodb:Select 僅針對讀取動作提供 ，dynamodb:ReturnValues也僅針對寫入動作提供 ，因此此處的條件永遠不會生效，也不會保護任何內容。將每個金鑰與支援動作配對，或將其移除。此檢查同時適用於 Allow和 Deny陳述式。

安全警告 – 不可信任的條件索引鍵

問題代碼：UNTRUSTWORTHY_CONDITION_KEY

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Untrustworthy condition key: The {{key}} condition key is not recommended for access
 control as it can be spoofed/manipulated by the caller.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{key}} condition key is not recommended for access control as
 it can be spoofed/manipulated by the caller."
```

解決安全警告

請勿將此條件索引鍵用於存取控制。呼叫者可能會篡改或偽造索引鍵值，從而帶來安全風險。

相關用語

- AWS 全域條件內容索引鍵

安全性警告 – 允許使用 NotPrincipal

問題代碼：ALLOW_WITH_NOT_PRINCIPAL

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

解決安全警告

使用 `"Effect": "Allow"` 與 `NotPrincipal` 可能太過寬鬆。例如，這可以將許可授予匿名 principals. AWS recommends，讓您指定需要使用 `Principal`元素存取的主體。或者，您可以允許廣泛存取，然後新增另一個使用 `NotPrincipal` 元素與 `“Effect”: “Deny”` 的陳述式。

- AWS JSON 政策元素：主體
- AWS JSON 政策元素：NotPrincipal

安全性警告 – ForAllValues 具有單一數值金鑰

問題代碼：FORALLVALUES_WITH_SINGLE_VALUED_KEY

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

解決安全警告

AWS 建議您ForAllValues僅將 與多值條件搭配使用。ForAllValues 集合運算子會測試請求集每個成員的值是否為條件鍵集的子集。如果請求中每個鍵值至少符合政策中的一個值，則條件會傳回 true。如果請求中沒有鍵，或鍵值解析為 null 資料集 (例如空白字串)，則也會傳回 true。

若要了解條件是支援單一值還是多個值，請檢閱服務的動作、資源及條件鍵頁面。具有 ArrayOf 資料類型字首的條件鍵是多重值條件鍵。例如，Amazon SES 支援具有單一值 (String) 和 ArrayOfString 多重值資料類型的鍵。

- 多值內容鍵

安全性警告 – 使用 NotResource 傳遞角色

問題代碼：PASS_ROLE_WITH_NOT_RESOURCE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

解決安全警告

若要設定許多 AWS 服務，您必須將 IAM 角色傳遞給服務。若要允許這樣做，您必須授與 `iam:PassRole` 許可給身分 (使用者、使用者群組或角色)。在政策`iam:PassRole`中使用 搭配 `NotResource`元素，可讓您的委託人存取比您預期更多的服務或功能。 AWS 建議您改為在 `Resource`元素中指定允許的 ARNs。此外，您可以使用 `iam:PassedToService` 條件鍵將許可降低為單一服務。

- 將角色傳遞到服務
- iam:PassedToService
- IAM JSON 政策元素：NotResource
- IAM JSON 政策元素：資源

安全性警告 – 使用動作中的星號和 NotResource 傳遞角色

問題代碼：PASS_ROLE_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

解決安全警告

若要設定許多 AWS 服務，您必須將 IAM 角色傳遞給服務。若要允許這樣做，您必須授與 `iam:PassRole` 許可給身分 (使用者、使用者群組或角色)。在 中包含萬用字元 (*) `Action`且包含 `NotResource`元素的政策，可讓您的委託人存取比您預期更多的服務或功能。 AWS 建議您改為在 `Resource`元素中指定允許的 ARNs。此外，您可以使用 `iam:PassedToService` 條件鍵將許可降低為單一服務。

- 將角色傳遞到服務
- iam:PassedToService
- IAM JSON 政策元素：NotResource
- IAM JSON 政策元素：資源

安全性警告 – 使用 NotAction 和 NotResource 傳遞角色

問題代碼：PASS_ROLE_WITH_NOT_ACTION_AND_NOT_RESOURCE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

解決安全警告

若要設定許多 AWS 服務，您必須將 IAM 角色傳遞給服務。若要允許這樣做，您必須授與 `iam:PassRole` 許可給身分 (使用者、使用者群組或角色)。使用 `NotAction`元素並在 `NotResource`元素中列出一些資源，可讓您的委託人存取比您預期更多的服務或功能。 AWS 建議您改為在 `Resource`元素中指定允許的 ARNs。此外，您可以使用 `iam:PassedToService` 條件鍵將許可降低為單一服務。

- 將角色傳遞到服務
- iam:PassedToService
- IAM JSON 政策元素：NotAction
- IAM JSON 政策元素：動作
- IAM JSON 政策元素：NotResource
- IAM JSON 政策元素：資源

安全性警告 – 使用資源中的星號傳遞角色

問題代碼：PASS_ROLE_WITH_STAR_IN_RESOURCE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

解決安全警告

若要設定許多 AWS 服務，您必須將 IAM 角色傳遞給服務。若要允許這樣做，您必須授與 `iam:PassRole` 許可給身分 (使用者、使用者群組或角色)。允許 `iam:PassRole` 和 在 `Resource`元素中包含萬用字元 (*) 的政策，可讓您的委託人存取比您預期更多的服務或功能。 AWS 建議您改為在 `Resource`元素中指定允許的 ARNs。此外，您可以使用 `iam:PassedToService` 條件鍵將許可降低為單一服務。

有些 AWS 服務會在其角色的名稱中包含其服務命名空間。此政策檢查會在分析政策以產生問題清單時，將這些慣例納入考量。例如，下列資源 ARN 可能不會產生問題清單：

```
arn:aws:iam::*:role/Service*
```

- 將角色傳遞到服務
- iam:PassedToService
- IAM JSON 政策元素：資源

AWS 具有此安全性警告的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

這些使用案例其中一個適用於您帳戶中的系統管理員。下列 AWS 受管政策提供管理員存取權，並授予許可，以將任何 IAM 角色傳遞給任何 服務。 AWS 建議您僅將下列 AWS 受管政策連接到您認為是管理員的 IAM 身分。

- AdministratorAccess-Amplify

下列 AWS 受管政策包含 資源中`iam:PassRole`具有萬用字元 (*) 且位於棄用路徑的 許可。對於每個政策，我們更新了許可指導，例如建議支援使用案例的新 AWS 受管政策。若要檢視這些政策的替代方案，請參閱指南以瞭解每個服務。

- AWSElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess
- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOpsWorksRole
- AWSDataPipelineRole
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonEC2ContainerServiceFullAccess

下列 AWS 受管政策僅提供服務連結角色的許可，允許 AWS 服務代表您執行動作。您可以將這些政策連接到 IAM 身分。

- AWSServiceRoleForAmazonEKSNodegroup

安全性警告 – 使用動作中的星號和資源傳遞角色

問題代碼：PASS_ROLE_WITH_STAR_IN_ACTION_AND_RESOURCE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

解決安全警告

若要設定許多 AWS 服務，您必須將 IAM 角色傳遞給服務。若要允許這樣做，您必須授與 `iam:PassRole` 許可給身分 (使用者、使用者群組或角色)。在 `Action`和 `Resource`元素中使用萬用字元 (*) 的政策，可讓您的委託人存取比您預期更多的服務或功能。 AWS 建議您改為在 `Resource`元素中指定允許的 ARNs。此外，您可以使用 `iam:PassedToService` 條件鍵將許可降低為單一服務。

- 將角色傳遞到服務
- iam:PassedToService
- IAM JSON 政策元素：動作
- IAM JSON 政策元素：資源

AWS 具有此安全性警告的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

其中一些使用案例適用於您帳戶中的系統管理員。下列 AWS 受管政策提供管理員存取權，並授予許可，以將任何 IAM 角色傳遞給任何 AWS 服務。 AWS 建議您僅將下列 AWS 受管政策連接到您認為是管理員的 IAM 身分。

- AdministratorAccess
- IAMFullAccess

安全性警告 – 使用資源中的星號和 NotAction 傳遞角色

問題代碼：PASS_ROLE_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

解決安全警告

若要設定許多 AWS 服務，您必須將 IAM 角色傳遞給服務。若要允許這樣做，您必須授與 `iam:PassRole` 許可給身分 (使用者、使用者群組或角色)。在政策中使用 `NotAction`元素搭配 `Resource`元素中的萬用字元 (*)，可讓您的委託人存取比您預期更多的服務或功能。 AWS 建議您改為在 `Resource`元素中指定允許的 ARNs。此外，您可以使用 `iam:PassedToService` 條件鍵將許可降低為單一服務。

- 將角色傳遞到服務
- iam:PassedToService

- IAM JSON 政策元素：NotAction
- IAM JSON 政策元素：動作
- IAM JSON 政策元素：資源

安全性警告 – 遺失配對的條件鍵

問題代碼：MISSING_PAIRED_CONDITION_KEYS

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

解決安全警告

當與其他相關條件鍵配對時，某些條件鍵會更安全。 AWS 建議您在與現有條件鍵相同的條件區塊中包含相關的條件鍵。這會使透過政策授與的許可更安全。

例如，您可以使用 `aws:VpcSourceIp` 條件鍵，將從中提出請求的 IP 地址與您在政策中指定的 IP 地址進行比較。 AWS 建議您新增相關的 `aws:SourceVPC` 條件鍵。這會檢查請求是否來自您在政策中指定的 VPC 和您指定的 IP 地址。

相關用語

- `aws:VpcSourceIp` 全域條件鍵

- aws:SourceVPC 全域條件鍵
- 全域條件鍵
- 條件元素
- JSON 政策概觀

安全性警告 – 拒絕使用不支援的服務標籤條件鍵

問題代碼：DENY_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

解決安全警告

在政策的 Condition元素中使用不支援的標籤條件金鑰"Effect": "Deny"可能會過度寬鬆，因為該服務的條件會被忽略。 AWS 建議您移除不支援條件金鑰的服務動作，並建立另一個陳述式來拒絕存取這些動作的特定資源。

如果您使用 aws:ResourceTag 條件鍵，且其不受服務動作支援，則該鍵不會包含在要求內容中。在此案例中，Deny 陳述式中的條件一律會傳回 false，且絕不會拒絕該動作。即使已正確標記資源，也會發生這種情況。

當服務支援 aws:ResourceTag 條件鍵時，您可以使用標籤來控制對該服務之資源的存取。這稱為以屬性為基礎的存取控制 (ABAC)。不支援這些鍵的服務需要您使用以資源為基礎的存取控制 (RBAC) 來控制對資源的存取權。

> **Note**
>
> 某些服務允許支援其資源和動作子集的 aws:ResourceTag 條件鍵。IAM Access Analyzer 會傳回不支援服務動作的問題清單。例如，Amazon S3 支援 aws:ResourceTag 以取得其資源的子集。若要檢視 Amazon S3 中可用且支援 aws:ResourceTag 條件鍵的所有資源類型，請參閱服務授權參考中的 Amazon S3 定義的資源類型。

例如，假設您想要拒絕存取取消標記刪除標記為鍵值組 status=Confidential 的特定資源。也假設 AWS Lambda 允許您標記和取消標記資源，但不支援 aws:ResourceTag條件金鑰。若要拒絕 的刪除動作 AWS App Mesh ， AWS Backup 如果此標籤存在，請使用 aws:ResourceTag條件索引鍵。對於 Lambda，請使用資源命名慣例，其中包含 "Confidential" 字首。然後包含一個單獨的陳述式，以防止刪除具有該命名慣例的資源。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
            }
        },
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
```

```
    ]
}
```

> **Warning**
>
> 請勿使用 ...IfExists 版本的條件運算子做為此問題清單的解決方法。這表示「如果鍵存在於請求內容中，且值相符，則拒絕動作。否則，請拒絕動作。」 在上述範例中，在 DenyDeleteSupported 陳述式中包含 lambda:DeleteFunction 動作與 StringEqualsIfExists 運算子一律會拒絕該動作。對於該動作，鍵不存在於內容中，且每次嘗試刪除該資源類型都會遭到拒絕，無論該資源是否已加上標籤。

相關用語

- 全域條件鍵
- 比較 ABAC 與 RBAC
- IAM JSON 政策元素：條件運算子
- 條件元素
- JSON 政策概觀

安全性警告 - 拒絕不支援的服務標籤條件鍵的 NotAction

問題代碼：DENY_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
 because some service actions are not denied by this statement. This is because the
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

解決安全警告

在政策的 Condition 元素中使用標籤條件鍵與元素 NotAction 和 "Effect": "Deny" 可能太過寬鬆。對於不支援 condition key. AWS recommends 的服務動作，系統會忽略條件。建議您重寫邏輯以拒絕動作清單。

如果您使用 aws:ResourceTag 條件鍵與 NotAction，則不會拒絕任何不支援鍵的新或現有服務動作。 AWS 建議您明確列出您要拒絕的動作。IAM Access Analyzer 會針對所列出不支援 aws:ResourceTag 條件鍵的動作傳回個別的問題清單。如需詳細資訊，請參閱 安全性警告 – 拒絕使用不支援的服務標籤條件鍵。

當服務支援 aws:ResourceTag 條件鍵時，您可以使用標籤來控制對該服務之資源的存取。這稱為以屬性為基礎的存取控制 (ABAC)。不支援這些鍵的服務需要您使用以資源為基礎的存取控制 (RBAC) 來控制對資源的存取權。

相關用語

- 全域條件鍵
- 比較 ABAC 與 RBAC
- IAM JSON 政策元素：條件運算子
- 條件元素
- JSON 政策概觀

安全性警告 - 限制存取服務主體

問題代碼：RESTRICT_ACCESS_TO_SERVICE_PRINCIPAL

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

解決安全警告

您可以使用服務主體在資源型政策的 `Principal`元素 AWS 服務 中指定 ，這是服務的識別符。授予服務主體代表您採取行動的存取權時，會限制存取。您可以使用 `aws:SourceArn`、`aws:SourceOrgID`、 `aws:SourceAccount`或 `aws:SourceOrgPaths`條件索引鍵來限制對特定來源的存取，例如特定資源 ARN、組織 ID 或組織路徑 AWS 帳戶，以防止過度寬鬆的政策。限制存取可協助您避免名為混淆代理人問題的安全問題。

相關用語

- AWS 服務 主體
- AWS 全域條件索引鍵：aws：SourceAccount
- AWS 全域條件索引鍵：aws：SourceArn
- AWS 全域條件索引鍵：aws：SourceOrgId
- AWS 全域條件索引鍵：aws：SourceOrgPaths
- 混淆代理人問題

安全性警告 – 缺少 OIDC 主體的條件鍵

問題代碼：MISSING_CONDITION_KEY_FOR_OIDC_PRINCIPAL

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
```

```
federated OIDC principals to ensure that only the intended identity provider assumes
the role.
```

在以程式設計方式呼叫 AWS CLI 或 AWS API 時，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

解決安全警告

在沒有條件的情況下使用 Open ID Connect 主體可能會過於寬鬆。新增其字首與您的聯合 OIDC 主體相符的條件鍵，以確保只有預定的身分提供者擔任該角色。

相關用語

- 建立 Web 身分的角色或 OpenID Connect 聯合身分 (主控台)

安全性警告 – 缺少 github 儲存庫條件鍵

問題代碼：MISING_GITHUB_REPO_CONDITION_KEY

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
 Add the token.actions.githubusercontent.com:sub condition key and specify the branch
 and repository name in the value.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
 token.actions.githubusercontent.com:sub condition key and specify the branch and
 repository name in the value."
```

解決安全警告

如果您使用 GitHub 做為 OIDC IdP，最佳實務是限制可擔任與 IAM IdP 相關聯角色的實體。當您在角色信任政策中包含 Condition 陳述式時，可以將角色限制到特定 GitHub 組織、儲存庫或分支。您可以使用條件鍵 token.actions.githubusercontent.com:sub 來限制存取。建議您將條件限制為一組特定的儲存庫或分支。如果您未包含此條件，則來自您控制範圍之外的組織或儲存庫的 GitHub 動作可以擔任與您 AWS 帳戶中的 GitHub IAM IdP 相關聯的角色。

相關用語

- 設定 GitHub OIDC 身分提供者的角色

安全警告 – 類似具有 ARN 條件索引鍵的運算子的字串

問題代碼：STRING_LIKE_OPERATOR_WITH_ARN_CONDITION_KEYS

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
String like operator with ARN condition keys: Use the operator type {{allowed}} instead
 of operator {{operator}} for the condition key {{key}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

解決安全警告

AWS 建議您在比較 ARN 時使用 ARNs 運算子而非字串運算子，以確保根據 ARN 條件值進行適當的存取限制。每當使用指定的索引鍵時，將 StringLike 運算子更新為 Condition 元素中的 ArnLike 運算子。

這些 AWS 受管政策是此安全性警告的例外狀況：

- AmazonSecurityLakeAdministrator
- AWSCodePipeline_FullAccess
- AWSCodePipeline_ReadOnlyAccess
- S3UnlockBucketPolicy

- SQSUnlockQueuePolicy

相關用語

- Amazon Resource Name (ARN) 條件運算子
- 字串條件運算子
- AWS 受管政策

安全警告 – 未選取 DynamoDB 屬性

問題代碼：DYNAMODB_ATTRIBUTES_WITHOUT_SELECT

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB attributes without select: Restricting dynamodb:Attributes without also
 setting dynamodb:Select to SPECIFIC_ATTRIBUTES allows all attributes to be returned
 on read requests that omit a projection expression. We recommend that you also set
 dynamodb:Select to SPECIFIC_ATTRIBUTES.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Restricting dynamodb:Attributes without also setting dynamodb:Select
 to SPECIFIC_ATTRIBUTES allows all attributes to be returned on read requests that
 omit a projection expression. We recommend that you also set dynamodb:Select to
 SPECIFIC_ATTRIBUTES."
```

解決安全警告

在與 `StringEquals` 條件相同的陳述式`SPECIFIC_ATTRIBUTES`中，新增 `dynamodb:Select`設定為 `dynamodb:Attributes`的條件。DynamoDB `dynamodb:Attributes`只會評估指定要傳回之屬性的請求。如果沒有此條件，省略投影表達式的讀取請求會傳回整個項目。

相關用語

- 使用 IAM 政策條件進行精細存取控制
- IAM JSON 政策元素：條件運算子

安全警告 – 負面內容中的 S3 字首

問題代碼：S3_PREFIX_IN_NEGATIVE_CONTEXT

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
S3 prefix in negative context: Using the s3:prefix condition key in a Deny statement or
 with a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the s3:prefix condition key in a Deny statement or with
 a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike."
```

解決安全警告

若要將清單限制為特定字首，請使用下列其中一種模式：

- 允許使用 StringLike – 在`s3:prefix`列出您想要允許的字首時授予 `s3:ListBucket` `StringLike` 條件。此允許不會涵蓋提供不同字首的請求，也無法列出這些物件。
- 使用 StringNotLike 拒絕 – 拒絕，`s3:ListBucket`條件`StringNotLike`是`s3:prefix`列出您想要允許的字首。省略字首或提供較短字首的請求不符合允許的集合，且會遭到拒絕。

`Deny` 具有 的 `StringLike`或`Allow`具有 的 `StringNotLike`不會有效地限制存取，因為發起人可以透過指定不符合條件值的較短字首 （ 例如，`priv`而不是 `private/`) 來略過條件。

相關用語

- Amazon S3 條件金鑰範例
- IAM JSON 政策元素：條件運算子

- IAM 政策元素：Condition

安全警告 – 具有對象宣告類型的 ForAnyValue

問題代碼：FORANYVALUE_WITH_AUDIENCE_CLAIM_TYPE

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
ForAnyValue with audience claim type: Using ForAnyValue qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAnyValue:.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using ForAnyValue qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAnyValue:."
```

解決安全警告

AWS 建議您不要將ForAnyValue集合運算子與單值條件索引鍵搭配使用。只能將集合運算子與多值條件鍵搭配使用。請移除 ForAnyValue 集運算子。

相關用語

- 單一值與多值內容索引鍵
- 單一值內容金鑰政策範例

安全警告 – 沒有傳回值的 DynamoDB 屬性

問題代碼：DYNAMODB_ATTRIBUTES_WITHOUT_RETURN_VALUES

調查結果類型：SECURITY_WARNING

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
DynamoDB attributes without return values: Restricting dynamodb:Attributes without
 limiting dynamodb:ReturnValues allows write operations to return the entire item,
 which can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Restricting dynamodb:Attributes without limiting
 dynamodb:ReturnValues allows write operations to return the entire item, which
 can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW."
```

解決安全警告

在與條件相同的陳述式UPDATED_NEW中新增限制dynamodb:ReturnValues為 UPDATED_OLD、NONE或 dynamodb:Attributes的條件。如果沒有此限制，則寫入會dynamodb:ReturnValues設定為ALL_OLD或ALL_NEW傳回整個項目，包括您允許的屬性以外的屬性。

建議 – 空陣列動作

問題代碼：EMPTY_ARRAY_ACTION

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

解決建議

陳述式必須包含一個 Action 或 NotAction 元素，其中包括一組動作。當元素為空時，政策陳述式不會提供任何許可。指定 Action 元素中的動作。

- IAM JSON 政策元素：動作

建議 – 空陣列條件

問題代碼：EMPTY_ARRAY_CONDITION

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

解決建議

選擇性的 `Condition` 元素結構會要求您使用條件運算子和鍵值組。當條件值為空時，條件會傳回 `true`，且政策陳述式不會提供任何許可。指定條件值。

- IAM JSON 政策元素：條件

建議 – 空陣列條件 ForAllValues

問題代碼：EMPTY_ARRAY_CONDITION_FORALLVALUES

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
```

```
the request context is empty, we recommend that you use the Null condition operator
with the value of true instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

解決建議

Condition 元素結構會要求您使用條件運算子和鍵值組。ForAllValues 集合運算子會測試請求集每個成員的值是否為條件鍵集的子集。

當您使用 ForAllValues 與空條件鍵時，只有在請求中沒有鍵時才會符合條件。 AWS 建議如果您想要測試請求內容是否為空，請改用 Null 條件運算子。

- 多值內容鍵
- Null 條件運算子
- IAM JSON 政策元素：條件

建議 – 空陣列條件 ForAnyValue

問題代碼：EMPTY_ARRAY_CONDITION_FORANYVALUE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

解決建議

Condition 元素結構會要求您使用條件運算子和鍵值組。ForAnyValues 集合運算子會測試這組請求值是否至少有一個成員符合這組條件鍵值的至少一個成員。

當您使用 ForAnyValues 與空條件鍵時，條件一律不會相符。這表示陳述式不會影響 policy. AWS recommends 您重寫條件。

- 多值內容鍵
- IAM JSON 政策元素：條件

建議 – 空陣列條件 IfExists

問題代碼：EMPTY_ARRAY_CONDITION_IFEXISTS

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
 request context is empty, we recommend that you use the Null condition operator with
 the value of true instead.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

解決建議

...IfExists 後置詞會編輯條件運算子。這表示如果政策鍵中存在於請求的內容中，則依照政策所述來處理鍵。如果該鍵不存在，則評估條件元素為 true。

當您使用 ...IfExists 與空條件鍵時，只有在請求中沒有鍵時才會符合條件。 AWS 建議如果您想要測試請求內容是否為空，請改用 Null 條件運算子。

- ...IfExists 條件運算子
- IAM JSON 政策元素：條件

建議 – 空陣列主體

問題代碼：EMPTY_ARRAY_PRINCIPAL

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

解決建議

您必須在 IAM 角色和以資源為基礎的政策中使用 `Principal` 或 `NotPrincipal` 元素。資源型政策是您直接內嵌在資源中的政策。

當您在陳述式的 `Principal`元素中提供空陣列時，陳述式不會影響 policy. AWS recommends，建議您指定應可存取資源的主體。

- IAM JSON 政策元素：主體
- IAM JSON 政策元素：NotPrincipal

建議 – 空陣列資源

問題代碼：EMPTY_ARRAY_RESOURCE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

解決建議

陳述式必須包含 `Resource` 或 `NotResource` 元素。

當您在陳述式的資源元素中提供空陣列時，陳述式不會影響 policy. AWS recommends，您會為資源指定 Amazon Resource Name (ARNs)。

- IAM JSON 政策元素：資源
- IAM JSON 政策元素：NotResource

建議 – 空物件條件

問題代碼：EMPTY_OBJECT_CONDITION

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

解決建議

Condition 元素結構會要求您使用條件運算子和鍵值組。

當您在陳述式的條件元素中提供空物件時，陳述式對政策沒有任何影響。移除可選元素或指定條件。

- IAM JSON 政策元素：條件

建議 – 空物件主體

問題代碼：EMPTY_OBJECT_PRINCIPAL

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

解決建議

您必須在 IAM 角色和以資源為基礎的政策中使用 Principal 或 NotPrincipal 元素。資源型政策是您直接內嵌在資源中的政策。

當您在陳述式的 Principal元素中提供空物件時，陳述式不會影響 policy. AWS recommends，建議您指定應可存取資源的主體。

- IAM JSON 政策元素：主體
- IAM JSON 政策元素：NotPrincipal

建議 – 空 Sid 值

問題代碼：EMPTY_SID_VALUE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

解決建議

選擇性的 Sid (陳述式 ID) 元素允許您輸入您為政策陳述式提供的識別碼。您可以將 Sid 值指派給陳述式陣列中的每個陳述式。如果您選擇使用 Sid 元素，您必須提供字串值。

相關用語

- IAM JSON 政策元素：Sid

建議 – 等同於 null false

問題代碼：EQUIVALENT_TO_NULL_FALSE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Equivalent to null false: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition."
```

解決建議

將目前條件索引鍵取代為建議的索引鍵，並將索引鍵設為 `false`。此變更可改善政策清晰度，並確保條件評估更加可靠。更新條件區塊以使用 `{recommendedKey}: false`，而不使用目前的索引鍵-運算子組合。

相關用語

- IAM 政策元素：Condition
- 具有多個內容索引鍵或值的條件
- AWS 全域條件內容索引鍵

建議 – 等同於 null true

問題代碼：EQUIVALENT_TO_NULL_TRUE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Equivalent to null true: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition."
```

解決建議

將目前條件索引鍵取代為建議的索引鍵，並將索引鍵設為 `true`。此變更可改善政策清晰度，並確保條件評估更加可靠。更新條件區塊以使用 `{recommendedKey}: true`，而不使用目前的索引鍵-運算子組合。

相關用語

- IAM 政策元素：Condition

- 具有多個內容索引鍵或值的條件
- AWS 全域條件內容索引鍵

建議 – 改善 IP 範圍

問題代碼：IMPROVE_IP_RANGE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

解決建議

IP 地址條件必須採用標準 CIDR 格式，例如 203.0.113.0/24 或 2001:DB8:1234:5678::/64。當您在遮罩位元之後包含非零位元時，不會考慮使用 condition. AWS recommends 您使用訊息中包含的新地址。

- IP 地址條件運算子
- IAM JSON 政策元素：條件

建議 – Null 具有限定詞

問題代碼：NULL_WITH_QUALIFIER

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

解決建議

在 Condition 元素中，您所建置的表達式使用條件運算子 (例如等於或小於) 來比較政策中的條件鍵與值，以及請求內容中的鍵與值。針對包含多個值的單一條件鍵請求，您必須使用 ForAllValues 或 ForAnyValue 集合運算子。

當您使用 Null 條件運算子與 ForAllValues 時，陳述式一律會傳回 true。當您搭配 使用 Null 條件運算子時ForAnyValue， 陳述式一律會傳回 false. AWS recommends，告知您搭配這些集合運算子使用 StringLike條件運算子。

相關用語

- 多值內容鍵
- Null 條件運算子
- 條件元素

建議 – 私有 IP 地址子集

問題代碼：PRIVATE_IP_ADDRESS_SUBSET

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

解決建議

全域條件鍵 `aws:SourceIp` 僅適用於公有 IP 地址範圍。

當您的 `Condition` 元素包含私有和公有 IP 地址的混合時，陳述式可能沒有預期的效果。您可以使用 `aws:VpcSourceIP` 來指定私有 IP 地址。

> **Note**
>
> 只有在請求來自指定 IP 地址並且透過 VPC 端點時，全域條件鍵 `aws:VpcSourceIP` 才會相符。

- aws:SourceIp 全域條件鍵
- aws:VpcSourceIp 全域條件鍵
- IP 地址條件運算子
- IAM JSON 政策元素：條件

建議 – 私有 NotIpAddress 子集

問題代碼：PRIVATE_NOT_IP_ADDRESS_SUBSET

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

解決建議

全域條件鍵 aws:SourceIp 僅適用於公有 IP 地址範圍。

當您的 Condition 元素包含 NotIpAddress 條件運算子與私有和公有 IP 地址的混合時，陳述式可能沒有預期的效果。每個未在政策中指定的公有 IP 地址都會相符。沒有任何私有 IP 地址會相符。為了達到這個效果，您可以使用 NotIpAddress 與 aws:VpcSourceIP，然後指定不應相符的私有 IP 地址。

- aws:SourceIp 全域條件鍵
- aws:VpcSourceIp 全域條件鍵
- IP 地址條件運算子
- IAM JSON 政策元素：條件

建議 – 冗餘動作

問題代碼：REDUNDANT_ACTION

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

解決建議

當您在 Action元素中使用萬用字元 (*) 時，您可以包含備援 permissions. AWS recommends，以檢閱您的政策，並僅包含您需要的許可。這可協助您移除冗餘動作。

例如，下列動作包含 iam:GetCredentialReport 動作兩次。

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
    ],
```

在此範例中，會針對以 Get 或 List 為開頭的每個 IAM 動作定義許可。當 IAM 新增其他取得或清單操作時，此政策將允許這些操作。您可能想要允許這全部的唯讀動作。iam:GetCredentialReport 動作已包含在內做為 iam:Get*。若要移除重複的許可，您可以移除 iam:GetCredentialReport。

當動作的所有內容都是冗餘時，您會收到此政策檢查的問題清單。在此範例中，如果元素包含 iam:*CredentialReport，其不被認為是冗餘。其中包括 iam:GetCredentialReport (此為冗餘) 以及 iam:GenerateCredentialReport (此非為冗餘)。移除 iam:Get* 或 iam:*CredentialReport 會變更政策的許可。

- IAM JSON 政策元素：動作

AWS 具有此建議的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

冗餘動作並不會影響政策所授與的許可。使用 AWS 受管政策做為建立客戶受管政策的參考時， AWS 建議您從政策中移除備援動作。

建議 – 冗餘條件值編號

問題代碼：REDUNDANT_CONDITION_VALUE_NUM

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

解決建議

當您針對條件鍵中的類似值使用數值條件運算子時，您可以建立導致冗餘許可的重疊。

例如，下列 Condition 元素包含多個 aws:MultiFactorAuthAge 條件，具有 1200 秒的年齡重疊。

```
"Condition": {
        "NumericLessThan": {
          "aws:MultiFactorAuthAge": [
            "2700",
            "3600"
          ]
        }
      }
```

在此範例中，如果多重要素驗證 (MFA) 完成時少於 3600 秒 (1 小時) 以前，就會定義許可。您可以移除冗餘 2700 值。

- 數位條件運算子
- IAM JSON 政策元素：條件

建議 – 冗餘資源

問題代碼：REDUNDANT_RESOURCE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

解決建議

在 Amazon 資源名稱 (ARN) 中使用萬用字元 (*) 時，您可以建立冗餘資源許可。

例如，下列 `Resource` 元素包含多個 ARN 與冗餘許可。

```
"Resource": [
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

在此範例中，會針對任何名稱開頭為 `jane` 的角色定義許可。您可以移除冗餘 `jane-admin` 和 `jane-s3only` ARN 而不變更結果許可。這確實會使政策為動態。其將定義任何未來角色開頭為 `jane` 的許可。如果政策的目的是允許存取靜態數量的角色，則移除最後一個 ARN，並僅列出應定義的 ARN。

- IAM JSON 政策元素：資源

AWS 具有此建議的 受管政策

AWS 受管政策可讓您根據一般 AWS 使用案例指派許可 AWS ，以開始使用 。

冗餘資源並不會影響政策所授與的許可。使用 AWS 受管政策做為建立客戶受管政策的參考時， AWS 建議您從政策中移除備援資源。

建議 – 冗餘陳述式

問題代碼：REDUNDANT_STATEMENT

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

解決建議

Statement 元素是政策的主要元素。此元素為必要。Statement 元素可包含單一陳述式，或是個別陳述式的陣列。

當您在長政策中多次包含相同的陳述式時，陳述式為冗餘。您可以移除其中一個陳述式，而不會影響政策授與的許可。當有人編輯政策時，他們可能會變更其中一個陳述式，而不會更新複本。這可能會導致超過預期的許可。

- IAM JSON 政策元素：陳述式

建議 – 服務名稱中的萬用字元

問題代碼：WILDCARD_IN_SERVICE_NAME

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

解決建議

當您在政策中包含 AWS 服務的名稱時， AWS 建議您不要包含萬用字元 (*、？)。這可能會為您不想要的未來服務新增許可。例如，有十幾個 AWS 服務在名稱*code*中包含 單字。

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- IAM JSON 政策元素：資源

建議 - 使用服務不支援的標籤條件鍵允許

問題代碼：ALLOW_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
 are not allowed by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
 affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

解決建議

在政策的 Condition元素中使用不支援的標籤條件金鑰，"Effect": "Allow"不會影響政策授予的許可，因為該服務動作會忽略該條件。 AWS 建議您移除不支援條件金鑰之服務的動作，並建立另一個陳述式，以允許存取該服務中的特定資源。

如果您使用 aws:ResourceTag 條件鍵，且其不受服務動作支援，則該鍵不會包含在要求內容中。在此案例中，Allow 陳述式中的條件一律會傳回 false，且絕不會允許該動作。即使已正確標記資源，也會發生這種情況。

當服務支援 aws:ResourceTag 條件鍵時，您可以使用標籤來控制對該服務之資源的存取。這稱為以屬性為基礎的存取控制 (ABAC)。不支援這些鍵的服務需要您使用以資源為基礎的存取控制 (RBAC) 來控制對資源的存取權。

> **Note**
>
> 某些服務允許支援其資源和動作子集的 aws:ResourceTag 條件鍵。IAM Access Analyzer 會傳回不支援服務動作的問題清單。例如，Amazon S3 支援 aws:ResourceTag 以取得其資源的子集。若要檢視 Amazon S3 中可用且支援 aws:ResourceTag 條件鍵的所有資源類型，請參閱服務授權參考中的 Amazon S3 定義的資源類型。

例如，假設您想要允許團隊成員檢視標記為鍵值組 team=BumbleBee 的特定資源詳細資訊。也假設 AWS Lambda 允許您標記資源，但不支援 aws:ResourceTag條件金鑰。若要允許 AWS App Mesh 和 的檢視動作， AWS Backup 如果此標籤存在，請使用 aws:ResourceTag條件索引鍵。對於 Lambda，請使用資源命名慣例，其中包含團隊名稱做為字首。接著納入一個單獨的陳述式，以允許檢視採用該命名慣例的資源。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
            }
        },
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
```

```
    ]
}
```

> **Warning**
>
> 請勿使用 Not 版的條件運算子搭配 "Effect": "Allow" 做為此問題清單的解決方法。這些條件運算子提供否定相符。這表示評估條件後，結果受到否定。在上述範例中，在 AllowViewSupported 陳述式中包含 lambda:GetFunction 動作與 StringNotEquals 運算子一律允許動作，無論該資源是否已加上標籤。
>
> 請勿使用 ...IfExists 版本的條件運算子做為此問題清單的解決方法。這表示「如果鍵存在於請求內容中，且值相符，則允許動作。否則，允許動作。」 在上述範例中，在 AllowViewSupported 陳述式中包含 lambda:GetFunction 動作與 StringEqualsIfExists 運算子一律會允許該動作。對於該動作，鍵不存在於內容中，且每次嘗試檢視該資源類型都會允許，無論該資源是否已加上標籤。

相關用語

- 全域條件鍵
- IAM JSON 政策元素：條件運算子
- 條件元素
- JSON 政策概觀

建議 - 使用服務不支援的標籤條件鍵允許 NotAction

問題代碼：ALLOW_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

解決建議

在政策的 Condition 元素中使用不受支援的標籤條件鍵與元素 NotAction 和 "Effect": "Allow" 並不會影響政策所授與的許可。對於不支援 condition key. AWS recommends 的服務動作，系統會忽略條件。建議您重寫邏輯，以允許動作清單。

如果您使用 aws:ResourceTag 條件鍵與 NotAction，則不會允許任何不支援鍵的新或現有服務動作。 AWS 建議您明確列出您要允許的動作。IAM Access Analyzer 會針對所列出不支援 aws:ResourceTag 條件鍵的動作傳回個別的問題清單。如需詳細資訊，請參閱 建議 - 使用服務不支援的標籤條件鍵允許。

當服務支援 aws:ResourceTag 條件鍵時，您可以使用標籤來控制對該服務之資源的存取。這稱為以屬性為基礎的存取控制 (ABAC)。不支援這些鍵的服務需要您使用以資源為基礎的存取控制 (RBAC) 來控制對資源的存取權。

相關用語

- 全域條件鍵
- 比較 ABAC 與 RBAC
- IAM JSON 政策元素：條件運算子
- 條件元素
- JSON 政策概觀

建議 – 服務主體的建議條件鍵

問題代碼：RECOMMENDED_CONDITION_KEY_FOR_SERVICE_PRINCIPAL

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
```

```
aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
{{key}}.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

解決建議

您可以使用服務委託人在資源型政策的 `Principal`元素 AWS 服務 中指定 ，這是服務的識別符。在授予對服務主體的存取權時，您應該使用 `aws:SourceArn`、`aws:SourceAccount`、`aws:SourceOrgID` 或 `aws:SourceOrgPaths` 條件鍵，而不是其他條件鍵 (例如 `aws:Referer`)。這可協助您避免名為混淆代理人問題的安全問題。

相關用語

- AWS 服務 主體
- AWS 全域條件索引鍵：aws：SourceAccount
- AWS 全域條件索引鍵：aws：SourceArn
- AWS 全域條件索引鍵：aws：SourceOrgId
- AWS 全域條件索引鍵：aws：SourceOrgPaths
- 混淆代理人問題

建議 - 政策中的不相關條件鍵

問題代碼：IRRELEVANT_CONDITION_KEY_IN_POLICY

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

解決建議

部分條件鍵與資源型政策無關。例如，s3:ResourceAccount 條件鍵與連接至 Amazon S3 儲存貯體或 Amazon S3 存取點資源類型的資源型政策無關。

您應在身分型政策中使用條件鍵，以控制存取資源。

相關用語

- 以身分為基礎的政策和以資源為基礎的政策

建議 – 因條件中的萬用字元而導致冗餘索引鍵

問題代碼：REDUNDANT_KEY_DUE_TO_WILDCARD_IN_CONDITION

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Redundant key due to wildcard in condition: The key {{key}} in the condition block of
 {{operator}} is redundant because it is always matched. Remove this key to simplify
 the condition.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} is redundant
 because it is always matched. Remove this key to simplify the condition."
```

解決建議

從政策中移除冗餘的條件索引鍵。由於萬用字元模式，索引鍵一律相符，因此不需要此索引鍵。移除此索引鍵可以簡化條件區塊，同時保持相同的有效許可。

相關用語

- IAM 政策元素：Condition
- 具有多個內容索引鍵或值的條件
- AWS 全域條件內容索引鍵

建議 – 角色信任政策中的冗餘主體

問題代碼：REDUNDANT_PRINCIPAL_IN_ROLE_TRUST_POLICY

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

解決建議

如果您在政策的 `Principal` 元素中同時指定擔任角色的主體及其父角色，則它會不允許或拒絕任何不同的許可。例如，如果您使用下列格式指定 `Principal` 元素，則其為冗餘主體：

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

我們建議移除擔任角色的主體。

相關用語

- 角色工作階段主體

建議 – 因條件中的萬用字元而導致冗餘陳述式

問題代碼：REDUNDANT_STATEMENT_DUE_TO_WILDCARD_IN_CONDITION

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Redundant statement due to wildcard in condition: The key {{key}} in the condition
 block of {{operator}} does not match any values. Remove this key to simplify the
 condition.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} does not
 match any values. Remove this key to simplify the condition."
```

解決建議

移除與任何值均不相符的條件索引鍵。此索引鍵建立了一個永遠無法滿足的不可達條件，因此它是多餘的。透過移除此索引鍵來清理政策，從而改善可讀性和效能。

相關用語

- IAM 政策元素：Condition
- 具有多個內容索引鍵或值的條件
- AWS 全域條件內容索引鍵

建議 – 確認對象要求類型

問題代碼：CONFIRM_AUDIENCE_CLAIM_TYPE

調查結果類型：SUGGESTION

尋找詳細資訊

在 中 AWS 管理主控台，此檢查的調查結果包含下列訊息：

```
Confirm audience claim type: The "{{key}}" ({{audienceType}}) claim key identifies
 the recipients that the JSON web token is intended for. Because this claim is single-
valued, do not use a qualifier.
```

在對 AWS CLI 或 AWS API 的程式設計呼叫中，此檢查的問題清單包含下列訊息：

```
"findingDetails": "The "{{key}}" ({{audienceType}}) claim key identifies the recipients
 that the JSON web token is intended for. Because this claim is single-valued, do not
 use a qualifier."
```

解決建議

aud (對象) 要求鍵是您在向 IdP 註冊應用程式時向您發佈的應用程式的唯一識別符，可識別 JSON Web 符記適用的收件人。對象要求可以是多重值或單一值。如果要求是多重值，請使用 `ForAllValues` 或 `ForAnyValue` 條件集運算子。如果要求是單一值，請勿使用條件集運算子。

相關用語

- 建立 Web 身分的角色或 OpenID Connect 聯合身分 (主控台)
- 多值內容鍵
- 單一值與多重值的條件鍵

使用 IAM Access Analyzer 自訂政策檢查來驗證政策

您可以使用自訂政策檢查，根據您的安全標準來檢查是否有新的存取權。每次檢查新存取權都會收取費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

使用自訂政策檢查驗證政策 (主控台)

在 IAM 主控台中使用 JSON 政策編輯器編輯政策時，您可以執行自訂政策檢查，此為選用步驟。您可以檢查與現有版本相比，更新版政策是否授予新的存取權。

在編輯 IAM JSON 政策時檢查是否有新的存取權

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。

3. 在政策清單中，選擇您要編輯的政策之名稱。您可以使用搜尋方塊來篩選政策清單。
4. 選擇許可索引標籤，然後選擇編輯。
5. 選擇 JSON 選項並更新政策。
6. 在政策下方的政策驗證窗格中，選擇檢查新的存取權索引標籤，然後選擇檢查政策。如果修改後的許可會授予新存取權，該陳述式會在政策驗證窗格中反白顯示。
7. 如果您不打算授予新的存取權，請更新政策陳述式並選擇檢查政策，直到沒有偵測到新的存取權為止。

> **Note**
>
> 每次檢查新存取權都會收取費用。如需定價的詳細資訊，請參閱 IAM Access Analyzer 定價。

8. 選擇下一步。
9. 在檢視與儲存頁面上，檢視此政策中定義的許可，然後選擇儲存變更。

使用自訂政策檢查 (AWS CLI 或 API) 驗證政策

您可以從 AWS CLI 或 IAM Access Analyzer API 執行 IAM Access Analyzer 自訂政策檢查。

執行 IAM Access Analyzer 自訂政策檢查 (AWS CLI)

- 若要在與現有政策進行比較時，檢查更新版政策是否允許新存取權，請執行下列命令：check-no-new-access
- 若要檢查政策是否不允許指定的存取權，請執行下列命令：check-access-not-granted
- 若要檢查資源政策是否可以授予指定資源類型的公開存取權，請執行下列命令：check-no-public-access

執行 IAM Access Analyzer 自訂政策檢查 (API)

- 若要在與現有政策進行比較時，檢查更新版政策是否允許新存取權，請使用 CheckNoNewAccess API 操作。
- 若要檢查政策是否不允許指定的存取權，請使用 CheckAccessNotGranted API 操作。
- 若要檢查資源政策是否可以授予指定資源類型的公開存取權，請使用 CheckNoPublicAccess API 操作。

產生 IAM Access Analyzer 政策

身為管理員或開發人員，您可能會將超出其所需範圍的許可授予 IAM 實體 (使用者或角色)。IAM 提供數個選項，協助您精簡您授予的許可。

其中一種選擇是產生以實體存取活動基礎的 IAM 政策。IAM Access Analyzer 會檢閱您的 AWS CloudTrail 日誌，並產生政策範本，其中包含實體在指定日期範圍中使用的許可。您可以使用範本來建立具有精細許可的政策，這些許可只會授予支援特定使用案例所需的許可。

主題

- 政策產生的運作方式
- 服務與動作層級資訊
- 產生政策的注意事項
- 產生政策所需的許可
- 根據 CloudTrail 活動 (主控台) 產生政策
- 使用另一個帳戶中 AWS CloudTrail 的資料產生政策
- 根據 CloudTrail 活動 (AWS CLI) 產生政策
- 根據 CloudTrail 活動 (AWS API) 產生政策
- IAM Access Analyzer 政策產生服務

政策產生的運作方式

IAM Access Analyzer 會分析您的 CloudTrail 事件，以識別 IAM 實體 (使用者或角色) 所使用的動作和服務。然後它會產生以該活動為基礎的 IAM 政策。當您使用產生的政策取代連接至實體的廣泛許可政策時，您可以精簡實體的許可。以下是政策產生程序的高階概觀。

- 設定政策範本產生 – 您可以指定最長 90 天的期間，讓 IAM Access Analyzer 分析您的歷史 AWS CloudTrail 事件。您必須指定現有的服務角色或建立新的服務角色。服務角色可讓您 IAM Access Analyzer 存取 CloudTrail 線索和服務最近存取的資訊，以識別所使用的服務和動作。您必須先指定記錄帳戶事件的 CloudTrail 線索，才能產生政策。如需 CloudTrail 資料 IAM Access Analyzer 配額的詳細資訊，請參閱 IAM Access Analyzer 配額。
- 產生政策 – IAM Access Analyzer 會根據 CloudTrail 事件中的存取活動產生政策。
- 檢閱和自訂政策 – 產生政策之後，您可以檢閱實體在指定日期範圍內所使用的服務和動作。您可以透過新增或移除許可、指定資源，以及將條件新增至政策範本，進一步自訂政策。

- 建立並連接政策 – 您可以選擇透過建立受管政策來儲存產生的政策。您可以將建立的政策連接至使用其活動來產生政策的使用者或角色。

服務與動作層級資訊

IAM Access Analyzer 產生 IAM 政策時，系統會傳回資訊以協助您進一步自訂政策。產生政策時，可以傳回兩種類別的資訊：

- 具有動作層級資訊的政策 – 對於某些 AWS 服務，例如 Amazon EC2，IAM Access Analyzer 可以識別 CloudTrail 事件中找到的動作，並列出它產生的政策中使用的動作。如需支援服務的清單，請參閱 IAM Access Analyzer 政策產生服務。對於某些服務，IAM Access Analyzer 會提示您將服務的動作新增至產生的政策。
- 具有服務層級資訊 – 的政策 IAM Access Analyzer 會使用最近存取的資訊來建立包含最近使用之所有服務的政策範本。使用 時 AWS 管理主控台，我們會提示您檢閱服務並新增動作以完成政策。

如需每個服務的動作清單，請參閱服務授權參考中的AWS 服務的動作、資源和條件金鑰。

產生政策的注意事項

產生政策之前，請先檢閱下列重要詳細資訊。

- 啟用 CloudTrail 線索 – 您的帳戶必須啟用 CloudTrail 追蹤，才能根據存取活動產生政策。建立 CloudTrail 線索時，CloudTrail 會將與線索相關的事件傳送至您指定的 Amazon S3 儲存貯體。若要了解如何建立 CloudTrail 追蹤，請參閱AWS CloudTrail 《 使用者指南》中的為您的 AWS 帳戶建立追蹤。
- 資料事件無法使用 – IAM Access Analyzer 不會在產生的政策中識別資料事件 (例如 Amazon S3 資料事件) 的動作層級活動。
- PassRole – `iam:PassRole` 動作不會被 CloudTrail 追蹤，也不會包含在產生的政策中。
- 縮短政策產生時間 – 若要更快產生政策，請減少您在設定政策產生期間指定的日期範圍。
- 使用 CloudTrail 稽核 – 請勿將政策產生用於稽核用途；請改用 CloudTrail。如需使用 CloudTrail 的詳細資訊，請參閱使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail。
- 拒絕的操作 – 原則產生會審查所有 CloudTrail 事件，包括拒絕的操作。
- 一個政策 IAM 主控台 – 您可以在 IAM 主控台中一次產生一個政策。
- 在 IAM 主控台所產生政策的可用性 – 您可以在政策產生後最多 7 天內檢閱 IAM 主控台中產生的政策。7 天後，您必須產生新政策。

- 政策產生配額 – 如需有關 IAM Access Analyzer 政策產生配額的其他資訊，請參閱 IAM Access Analyzer 配額。
- 適用 Amazon S3 標準費率 - 當您使用政策產生功能時，IAM Access Analyzer 會檢閱 S3 儲存貯體中的 CloudTrail 日誌。為產生政策而存取 CloudTrail 日誌不會產生額外的儲存費用。 會針對存放在 S3 儲存貯體中的 CloudTrail 日誌的請求和資料傳輸 AWS 收取標準 Amazon S3 費率。 Amazon S3
- AWS Control Tower 支援 – 基於下列原因，政策產生不支援使用在產生政策時 AWS Control Tower 建立的 AWS CloudTrail 線索：
  - 來自於組織的 CloudTrail 資料記錄在另一個帳戶中，即 AWS Control Tower 日誌封存帳戶。
  - 由於 AWS Control Tower的服務控制政策 (SCP) 針對 S3 記錄儲存貯體設定了限制，因此無法重新設定儲存這些日誌的 S3 儲存貯體的許可。

產生政策所需的許可

您第一次產生政策所需的許可與您需要產生政策以供後續使用的許可不同。如需詳細資訊，請參閱入門 AWS Identity and Access Management Access Analyzer。

首次設定

第一次產生政策時，您必須在帳戶中選擇合適的現有服務角色，或建立新的服務角色。服務角色可讓您在帳戶中 IAM Access Analyzer 存取 CloudTrail 和服務最近存取的資訊。只有管理員應具有建立和設定角色所需的許可。因此，我們建議管理員在第一次設定期間建立服務角色。若要進一步了解建立服務角色所需的許可，請參閱建立角色以將許可委派給 AWS 服務。

服務角色所需的許可

當您建立服務角色時，您可以設定該角色的兩個政策。您可以將 IAM 許可政策連接至指定角色可執行內容的角色。您也可以將角色信任政策連接至指定可以使用該角色之主體的角色。

第一個範例政策顯示產生政策所需之服務角色的許可政策。第二個範例政策顯示服務角色所需的角色信任政策。當您使用 AWS API 或 產生政策時，您可以使用這些政策 AWS CLI 來協助您建立服務角色。當您使用 IAM 主控台建立服務角色作為政策產生程序的一部分時，我們會為您產生這些政策。

JSON

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

```
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

下列範例政策顯示具有允許 IAM Access Analyzer 擔任角色之許可的角色信任政策。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
```

```
  ]
}
```

後續使用

若要在 中產生政策 AWS 管理主控台，IAM 使用者必須擁有許可政策，允許其將用於產生政策的服務角色傳遞給 IAM Access Analyzer。 `iam:PassRole` 通常會隨附 ，`iam:GetRole`以便使用者可以取得要傳遞之角色的詳細資訊。在這個範例中，使用者可以僅傳遞存在於指定的帳戶，名稱開頭為 `AccessAnalyzerMonitorServiceRole*` 的角色。若要進一步了解如何將 IAM 角色傳遞至 AWS 服務，請參閱授予使用者將角色傳遞至 AWS 服務的許可。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

您還必須擁有下列 IAM Access Analyzer 許可，才能在 AWS 管理主控台、 AWS API 或 中產生政策，AWS CLI 如下列政策陳述式所示。

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
```

```
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
}
```

適用於首次及後續使用

當您使用 AWS 管理主控台 產生政策時，您必須具有列出帳戶中 CloudTrail 追蹤的`cloudtrail:ListTrails`許可，如下列政策陳述式所示。

```
{
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "cloudtrail:ListTrails"
  ],
  "Resource": "*"
}
```

根據 CloudTrail 活動 (主控台) 產生政策

您可以為 IAM 使用者或角色產生政策。

步驟 1：根據 CloudTrail 活動產生政策

下列程序說明如何使用 AWS 管理主控台產生角色的政策。

產生 IAM 角色的政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。

   > **Note**
   >
   > 根據 IAM 使用者之活動產生政策的步驟幾乎相同。若要這樣做，請選擇 Users (使用者) 而非 Roles (角色)。

3. 在帳戶中的角色清單中，選擇您要用來產生政策之活動的角色名稱。
4. 在 Permissions (許可) 索引標籤的 Generate policy based on CloudTrail events (根據 CloudTrail 事件產生政策) 區段中，選擇 Generate policy (產生政策)。

5. 在 Generate policy (產生政策) 頁面上，指定您要 IAM Access Analyzer 分析 CloudTrail 事件的時間間隔，以執行角色所採取的動作。您可以選擇最多 90 天的範圍。我們建議您選擇最短的時間間隔，以減少政策產生時間。
6. 在 CloudTrail 存取部分中，選擇適當的角色或新建角色 (如果適當的角色不存在)。角色授與代表您存取 CloudTrail 資料的 IAM Access Analyzer 許可，以檢閱存取活動來識別已使用的服務和動作。若要進一步了解此角色所需的許可，請參閱 產生政策所需的許可。
7. 在 CloudTrail trail to be analyzed (要分析的 CloudTrail 線索) 區段中，請指定記錄帳戶事件的 CloudTrail 線索。

   如果您選擇將記錄儲存在不同帳戶中的 CloudTrail 追蹤，則會顯示有關跨帳戶存取的資訊方塊。跨帳戶存取需要額外的設定。如需進一步了解，請參閱本主題稍後的 Choose a role for cross-account access。
8. 選擇 Generate policy (產生政策)。
9. 當正在產生政策時，您會返回 Permissions (許可) 索引標籤上的 Roles Summary (角色摘要) 頁面。等待 Policy request details (政策請求詳細資訊) 區段中的狀態顯示 Success (成功)，然後選擇 View generated policy (檢視產生的政策)。您可以在最多七天內檢視產生的政策。如果您產生另一個政策，則現有政策會被您產生的新政策取代。

步驟 2：檢閱許可並為所使用之服務新增動作

檢閱 IAM Access Analyzer 識別出之角色所使用的服務和動作。您可以針對所產生的政策範本中使用的任何服務新增動作。

1. 請檢閱下列各節：

   - 在 Review permissions (檢閱許可) 頁面上，檢閱 Actions included in the generated policy (產生政策中所包含的動作) 清單。清單會顯示 IAM Access Analyzer 識別的在指定日期範圍內由角色使用的服務和動作。
   - Services used (所使用的服務) 區段會顯示 IAM Access Analyzer 識別的在指定日期範圍中角色所使用的其他服務。使用哪些動作的相關資訊可能不適用於本節所列的服務。使用列出的每個服務的功能表，手動選擇您要包含在政策中的動作。
2. 完成新增動作後，請選擇 Next (下一步)。

步驟 3：進一步自訂產生的政策

您可以透過新增或移除許可或指定資源，進一步自訂政策。

自訂產生的政策

1. 更新政策範本。政策範本包含支援資源層級許可之動作的資源 ARN 預留位置，如下圖所示。資源層級許可能夠讓您指定使用者可執行動作的資源。建議您使用 ARN 在政策中為支援資源層級許可的動作指定個別資源。您可以將預留位置資源 ARN 取代為使用案例的有效資源 ARN。

   如果動作不支援資源層級許可，您必須使用萬用字元 (*) 來指定該動作可以影響所有資源。若要了解哪些 AWS 服務支援資源層級許可，請參閱AWS 使用 IAM 的 服務。如需每項服務中的動作清單，以及了解哪些動作支援資源層級許可，請參閱 AWS 服務的動作、資源和條件索引鍵。



2. (選用) 在範本中新增、修改或移除 JSON 政策陳述式。若要進一步了解如何撰寫 JSON 政策，請參閱建立 IAM 政策 (主控台)。
3. 完成自訂政策範本後，您有下列選項：

   - (選用) 您可以複製範本中的 JSON，以便在 Generated policy (產生的政策) 頁面之外獨立使用。例如，如果您想要使用 JSON 在不同的帳戶中建立政策。如果範本中的政策超過 JSON 政策的 6,144 個字元限制，則政策會分割成多個政策。
   - 選擇 Next (下一步) 以檢閱並在同一個帳戶中建立受管政策。

步驟 4：檢閱並建立受管政策

如果您有建立和連接 IAM 政策的許可，則可以從產生的政策中建立受管政策。然後，您可以將政策連接至帳戶中的使用者或角色。

檢閱和建立政策

1. 在 Review and create managed policy (檢閱並建立受管政策) 頁面上，為您正在建立的政策輸入 Name (名稱) 與 Description (描述) (選用)。
2. (選用) 在 Summary (摘要) 區段中，您可以檢閱將包含在政策中的許可。
3. (選用) 藉由連接標籤作為鍵值組，將中繼資料新增至政策。如需在 IAM 中使用標籤的詳細資訊，請參閱標記 IAM 資源。
4. 完成後，請執行下列其中一項動作：
   - 您可以將新政策直接連接至用來產生政策的角色。若要執行這項操作，請在頁面底部附近，選取將政策連接至 ***YourRoleName*** 旁的核取方塊。然後選擇 Create and attach policy (建立並連接政策)。
   - 否則，請選擇 Create policy (建立政策)。您可以在 IAM 主控台的 Policies (政策) 導覽窗格的政策清單中找到您建立的政策。
5. 您可以將您建立的政策連接至帳戶中的實體。連接政策之後，您可以移除可能連接至實體的任何其他過於廣泛的政策。若要了解如何連接受管政策，請參閱新增 IAM 身分許可 (主控台)。

使用另一個帳戶中 AWS CloudTrail 的資料產生政策

您可以建立將資料存放在中央帳戶中的 CloudTrail 追蹤，以簡化管理活動。例如，您可以使用 AWS Organizations 建立追蹤，記錄 AWS 帳戶 該組織中所有 的所有事件。這條追蹤屬於一個中央帳戶。

如果您想要為帳戶中的使用者或角色產生政策，而該帳戶與儲存 CloudTrail 記錄資料的帳戶不同，則必須授與跨帳戶存取權。若要執行此動作，您需要一個角色和儲存貯體政策，將 IAM Access Analyzer 許可授與 CloudTrail 日誌。如需建立 Organizations 追蹤的詳細資訊，請參閱為組織建立追蹤。

在此範例中，假設您想要為帳戶 A 中的使用者或角色產生政策。帳戶 A 中的 CloudTrail 追蹤會將 CloudTrail 記錄儲存在帳戶 B 的儲存貯體中。您必須先進行下列更新，才能產生政策：

1. 選擇現有角色，或建立新的服務角色，授與 IAM Access Analyzer 存取帳戶 B (儲存 CloudTrail 日誌的位置) 中的儲存貯體。
2. 在帳戶 B 中確認 Amazon S3 儲存貯體物件擁有權和儲存貯體許可政策，以便 IAM Access Analyzer 可存取儲存貯體中的物件。

步驟 1：選擇或建立跨帳戶存取的角色

- 在產生政策畫面中，如果您的帳戶中存在具有必要許可的角色，則會為您預先選取使用現有的角色選項。否則，請選擇建立並使用新的服務角色。新角色可用來授與 IAM Access Analyzer 存取帳戶 B 中 CloudTrail 記錄的存取權。

步驟 2：在帳戶 B 中確認或更新您的 Amazon S3 儲存貯體組態

1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/s3/ ：// 開啟 Amazon S3 主控台。
2. 在儲存貯體清單中，選擇儲存 CloudTrail 追蹤日誌的儲存貯體名稱。
3. 選擇 Permissions (許可) 索引標籤，然後轉至 Object Ownership (物件擁有權) 區段。

   使用 Amazon S3 物件擁有權儲存貯體設定，可控制您上傳至儲存貯體的物件的擁有權。根據預設，當其他 AWS 帳戶 上傳物件到您的儲存貯體時，上傳帳戶會擁有物件。若要產生政策，儲存貯體擁有者必須擁有儲存貯體中的所有物件。根據您的 ACL 使用案例，您可能需要變更儲存貯體的 Object Ownership (物件擁有權) 設定。將 Object Ownership (物件擁有權) 設定為下列其中一個選項。

   - Bucket owner enforced (儲存貯體擁有者強制執行) (建議)
   - Bucket owner preferred (儲存貯體擁有者偏好)

   > **Important**
   >
   > 若要成功產生政策，儲存貯體中的物件必須由儲存貯體擁有者擁有。如果您選擇使用儲存貯體擁有者偏好，您只能在物件擁有權變更後的期間內產生政策。

   如需有關 Amazon S3 中物件擁有權的詳細資訊，請參閱 Amazon S3 User Guide 中的 Controlling ownership of objects and disabling ACLs for your bucket。
4. 將許可新增到帳戶 B 中的 Amazon S3 儲存貯體政策，以允許存取帳戶 A 中的角色。

   下列範例政策允許名為 `amzn-s3-demo-bucket` 之儲存貯體的 `ListBucket` 和 `GetObject`。如果存取儲存貯體的角色屬於貴組織中的帳戶，且名稱開頭為 `AccessAnalyzerMonitorServiceRole`，則允許存取。使用 `aws:PrincipalArn` 做為 `Resource` 元素中的 `Condition` 可確保角色只能存取帳戶的活動 (如果其屬於帳戶 A)。您可以將

amzn-s3-demo-bucket 取代為您的儲存貯體名稱，將 optional-prefix 取代為可選的儲存貯體字首，以及將 organization-id 取代為貴組織 ID。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

5. 如果您使用 加密日誌 AWS KMS，請在存放 CloudTrail 日誌的帳戶中更新 AWS KMS 金鑰政策，以授予 IAM Access Analyzer 使用您的金鑰的存取權，如下列政策範例所示。將 CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN 取代為您追蹤的 ARN 以及將 organization-id 取代為貴組織 ID。

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

```
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
            "s3.*.amazonaws.com"
            ]
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
     }
    ]
}
```

根據 CloudTrail 活動 (AWS CLI) 產生政策

可以使用以下列命令藉由 AWS CLI產生政策。

產生政策

- aws accessanalyzer start-policy-generation

檢視產生的政策

- aws accessanalyzer get-generated-policy

取消政策產生請求

- aws accessanalyzer cancel-policy-generation

檢視政策產生請求的清單

- aws accessanalyzer list-policy-generations

根據 CloudTrail 活動 (AWS API) 產生政策

您可以使用下列操作，使用 AWS API 產生政策。

產生政策

- StartPolicyGeneration

檢視產生的政策

- GetGeneratedPolicy

取消政策產生請求

- CancelPolicyGeneration

檢視政策產生請求的清單

- ListPolicyGenerations

IAM Access Analyzer 政策產生服務

下表列出 IAM Access Analyzer 產生具有動作層級資訊之政策 AWS 的服務。如需每個服務的動作清單，請參閱《服務授權參考》中的AWS 服務的動作、資源和條件索引鍵。

AWS 提供 JSON 格式的政策產生資訊，以簡化政策管理工作流程的自動化。使用服務參考資訊，您可以從 AWS 服務 機器可讀取的檔案跨 存取政策產生資訊。如需詳細資訊，請參閱 Service Authorization Reference 中的 Simplified AWS 服務 information for programmatic access。

服務	服務前綴
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS 帳戶管理	帳戶
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	mplify
AWS Amplify UI Builder	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS 應用程式成本分析器	application-cost-profiler
Amazon CloudWatch Application Insights	applicationinsights
AWS App Mesh	appmesh
Amazon WorkSpaces 應用程式	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	自動擴展
AWS Marketplace	aws-marketplace
AWS Backup	備份

服務	服務前綴
AWS Batch	批次
Amazon Braket	braket
AWS Budgets	預算
AWS Cloud9	Cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
Amazon CodeGuru Reviewer	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar 通知	codestar-notifications
Amazon Cognito 身分	cognito-identity
Amazon Cognito 使用者集區	cognito-idp
Amazon Cognito Sync	cognito-sync

服務	服務前綴
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect 客戶	connect
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Amazon DocumentDB Elastic Clusters	docdb-elastic
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public

服務	服務前綴
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon EMR on EKS (EMR 容器)	emr-containers
Amazon EMR Serverless	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	events
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift 伺服器	gamelift
Amazon Location Service	geo
Amazon Glacier	glacier
Amazon Managed Grafana	grafana

服務	服務前綴
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
AWS 身分存放區	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Amazon Interactive Video Service Chat	ivschat

服務	服務前綴
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex V2	lex
AWS License Manager Linux 訂閱管理員	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	日誌
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaTailor	mediatailor

服務	服務前綴
Amazon MemoryDB	memorydb
AWS Transform MGN	mgn
AWS Migration Hub	mgh
AWS Migration Hub 策略建議	migration hub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	組織
AWS Performance Insights (績效詳情)	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Amazon Connect 客戶設定檔	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS 資源回收筒	rbin

服務	服務前綴
Amazon Relational Database Service	rds
Amazon Redshift	redshift
Amazon Redshift 資料 API	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
AWS 資源總管	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Roles Anywhere	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	s3
Amazon S3 on Outposts	s3-outposts
Amazon SageMaker AI 地理空間功能	sagemaker-geospatial

服務	服務前綴
Savings Plans	savingsplans
Amazon EventBridge Schemas	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub CSPM	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint 簡訊和語音服務	sms-voice
AWS Snowball Edge	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents

服務	服務前綴
適用於 SAP 的 AWS Systems Manager	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	標籤
Amazon Textract	textract
Amazon Timestream	timestream
AWS 電信網路建置器	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	傳輸
Amazon Translate	translate
連接客戶語音 ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Customer Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	工作區
AWS X-Ray	xray

政策產生動作資訊

下表會列出 IAM Access Analyzer 使用動作層級資訊為其產生政策的動作。

> **Important**
>
> `iam:UpdateAccountName` 動作將於 2026 年 4 月 22 日棄用。2026 年 4 月 22 日之後，僅 `account:PutAccountName` 許可能夠控制帳戶名稱更新存取權。我們強烈建議您更新所有控制帳戶名稱更新的服務控制政策 (SCP)，以便使用 `account:PutAccountName` 許可。

服務前綴	動作
access-analyzer	access-analyzer:ApplyArchiveRule
	access-analyzer:CancelPolicyGeneration
	access-analyzer:CheckAccessNotGranted
	access-analyzer:CheckNoNewAccess
	access-analyzer:CheckNoPublicAccess
	access-analyzer:CreateAccessPreview
	access-analyzer:CreateAnalyzer
	access-analyzer:CreateArchiveRule
	access-analyzer:DeleteAnalyzer
	access-analyzer:DeleteArchiveRule
	access-analyzer:GenerateFindingRecommendation
	access-analyzer:GetAccessPreview
	access-analyzer:GetAnalyzedResource
	access-analyzer:GetAnalyzer
	access-analyzer:GetArchiveRule

服務前綴	動作
	access-analyzer:GetFinding
	access-analyzer:GetFindingRecommendation
	access-analyzer:GetFindingsStatistics
	access-analyzer:GetGeneratedPolicy
	access-analyzer:ListAccessPreviewFindings
	access-analyzer:ListAccessPreviews
	access-analyzer:ListAnalyzedResources
	access-analyzer:ListAnalyzers
	access-analyzer:ListArchiveRules
	access-analyzer:ListFindings
	access-analyzer:ListPolicyGenerations
	access-analyzer:StartPolicyGeneration
	access-analyzer:StartResourceScan
	access-analyzer:UpdateAnalyzer
	access-analyzer:UpdateArchiveRule
	access-analyzer:UpdateFindings
	access-analyzer:ValidatePolicy

服務前綴	動作
帳戶	account:AcceptPrimaryEmailUpdate account:DeleteAlternateContact account:DisableRegion account:EnableRegion account:GetAccountInformation account:GetAlternateContact account:GetContactInformation account：GetGovCloudAccountInformation account:GetPrimaryEmail account:GetRegionOptStatus account:ListRegions account:PutAccountName account:PutAlternateContact account:PutContactInformation account:StartPrimaryEmailUpdate

服務前綴	動作
acm	acm：CreateAcmeEndpoint acm:DeleteCertificate acm:DescribeCertificate acm:ExportCertificate acm:GetAccountConfiguration acm:GetCertificate acm:ImportCertificate acm：ListAcmeEndpoints acm:ListCertificates acm:PutAccountConfiguration acm:RenewCertificate acm:RequestCertificate acm:ResendValidationEmail acm：SearchCertificates acm:UpdateCertificateOptions

服務前綴	動作
airflow	airflow:CreateCliToken airflow:CreateEnvironment airflow:CreateWebLoginToken airflow:DeleteEnvironment airflow:GetEnvironment airflow:ListEnvironments airflow:PublishMetrics airflow:UpdateEnvironment

服務前綴	動作
mplify	amplify:CreateApp
	amplify:CreateBackendEnvironment
	amplify:CreateBranch
	amplify:CreateDeployment
	amplify:CreateDomainAssociation
	amplify:CreateWebHook
	amplify:DeleteApp
	amplify:DeleteBackendEnvironment
	amplify:DeleteBranch
	amplify:DeleteDomainAssociation
	amplify:DeleteJob
	amplify:DeleteWebHook
	amplify:GenerateAccessLogs
	amplify:GetApp
	amplify:GetArtifactUrl
	amplify:GetBackendEnvironment
	amplify:GetBranch
	amplify:GetDomainAssociation
	amplify:GetJob
	amplify:GetWebHook
	amplify:ListApps

服務前綴	動作
	amplify:ListArtifacts
	amplify:ListBackendEnvironments
	amplify:ListBranches
	amplify:ListDomainAssociations
	amplify:ListJobs
	amplify:ListWebHooks
	amplify:StartDeployment
	amplify:StartJob
	amplify:StopJob
	amplify:UpdateApp
	amplify:UpdateBranch
	amplify:UpdateDomainAssociation
	amplify:UpdateWebHook

服務前綴	動作
amplifyuibuilder	amplifyuibuilder:CreateComponent amplifyuibuilder:CreateForm amplifyuibuilder:CreateTheme amplifyuibuilder:DeleteComponent amplifyuibuilder:DeleteForm amplifyuibuilder:DeleteTheme amplifyuibuilder:ExportComponents amplifyuibuilder:ExportThemes amplifyuibuilder:GetCodegenJob amplifyuibuilder:ListCodegenJobs amplifyuibuilder:ListComponents amplifyuibuilder:ListForms amplifyuibuilder:ListThemes amplifyuibuilder:ResetMetadataFlag amplifyuibuilder:StartCodegenJob amplifyuibuilder:UpdateComponent amplifyuibuilder:UpdateForm amplifyuibuilder:UpdateTheme

服務前綴	動作
app-integrations	app-integrations:CreateApplication app-integrations:CreateDataIntegration app-integrations:CreateDataIntegrationAssociation app-integrations:CreateEventIntegration app-integrations:DeleteApplication app-integrations:DeleteDataIntegration app-integrations:DeleteEventIntegration app-integrations:GetApplication app-integrations:GetDataIntegration app-integrations:GetEventIntegration app-integrations:ListApplicationAssociations app-integrations:ListApplications app-integrations:ListDataIntegrationAssociations app-integrations:ListDataIntegrations app-integrations:ListEventIntegrationAssociations app-integrations:ListEventIntegrations app-integrations:UpdateApplication app-integrations:UpdateDataIntegration app-integrations:UpdateDataIntegrationAssociation app-integrations:UpdateEventIntegration

服務前綴	動作
appconfig	appconfig:CreateApplication appconfig:CreateConfigurationProfile appconfig:CreateDeploymentStrategy appconfig:CreateEnvironment appconfig:CreateExtension appconfig:CreateExtensionAssociation appconfig:CreateHostedConfigurationVersion appconfig:DeleteApplication appconfig:DeleteConfigurationProfile appconfig:DeleteDeploymentStrategy appconfig:DeleteEnvironment appconfig:DeleteExtension appconfig:DeleteExtensionAssociation appconfig:DeleteHostedConfigurationVersion appconfig:GetAccountSettings appconfig:GetApplication appconfig:GetConfiguration appconfig:GetConfigurationProfile appconfig:GetDeployment appconfig:GetDeploymentStrategy appconfig:GetEnvironment

服務前綴	動作
	appconfig:GetExtension appconfig:GetExtensionAssociation appconfig:GetHostedConfigurationVersion appconfig:ListApplications appconfig:ListConfigurationProfiles appconfig:ListDeploymentStrategies appconfig:ListDeployments appconfig:ListEnvironments appconfig:ListExtensionAssociations appconfig:ListExtensions appconfig:ListHostedConfigurationVersions appconfig:StartDeployment appconfig:StopDeployment appconfig:UpdateAccountSettings appconfig:UpdateApplication appconfig:UpdateConfigurationProfile appconfig:UpdateDeploymentStrategy appconfig:UpdateEnvironment appconfig:UpdateExtension appconfig:UpdateExtensionAssociation appconfig:ValidateConfiguration

服務前綴	動作
appflow	appflow:CancelFlowExecutions
	appflow:CreateConnectorProfile
	appflow:CreateFlow
	appflow:DeleteConnectorProfile
	appflow:DeleteFlow
	appflow:DescribeConnector
	appflow:DescribeConnectorEntity
	appflow:DescribeConnectorProfiles
	appflow:DescribeConnectors
	appflow:DescribeFlow
	appflow:DescribeFlowExecutionRecords
	appflow:ListConnectorEntities
	appflow:ListConnectors
	appflow:ListFlows
	appflow:RegisterConnector
	appflow:ResetConnectorMetadataCache
	appflow:StartFlow
	appflow:StopFlow
	appflow:UnRegisterConnector
	appflow:UpdateConnectorProfile
	appflow:UpdateConnectorRegistration

服務前綴	動作
	appflow:UpdateFlow

服務前綴	動作
applicationinsights	applicationinsights:AddWorkload applicationinsights:CreateApplication applicationinsights:CreateComponent applicationinsights:CreateLogPattern applicationinsights:DeleteApplication applicationinsights:DeleteComponent applicationinsights:DeleteLogPattern applicationinsights:DescribeApplication applicationinsights:DescribeComponent applicationinsights:DescribeComponentConfiguration applicationinsights:DescribeComponentConfigurationRecommendation applicationinsights:DescribeLogPattern applicationinsights:DescribeObservation applicationinsights:DescribeProblem applicationinsights:DescribeProblemObservations applicationinsights:DescribeWorkload applicationinsights:ListApplications applicationinsights:ListComponents applicationinsights:ListConfigurationHistory applicationinsights:ListLogPatternSets

服務前綴	動作
	applicationinsights:ListLogPatterns applicationinsights:ListProblems applicationinsights:ListWorkloads applicationinsights:RemoveWorkload applicationinsights:UpdateApplication applicationinsights:UpdateComponent applicationinsights:UpdateComponentConfiguration applicationinsights:UpdateLogPattern applicationinsights:UpdateWorkload

服務前綴	動作
appmesh	appmesh:CreateGatewayRoute
	appmesh:CreateMesh
	appmesh:CreateRoute
	appmesh:CreateVirtualGateway
	appmesh:CreateVirtualNode
	appmesh:CreateVirtualRouter
	appmesh:CreateVirtualService
	appmesh:DeleteGatewayRoute
	appmesh:DeleteMesh
	appmesh:DeleteRoute
	appmesh:DeleteVirtualGateway
	appmesh:DeleteVirtualNode
	appmesh:DeleteVirtualRouter
	appmesh:DeleteVirtualService
	appmesh:DescribeGatewayRoute
	appmesh:DescribeMesh
	appmesh:DescribeRoute
	appmesh:DescribeVirtualGateway
	appmesh:DescribeVirtualNode
	appmesh:DescribeVirtualRouter
	appmesh:DescribeVirtualService

服務前綴	動作
	appmesh:ListGatewayRoutes
	appmesh:ListMeshes
	appmesh:ListRoutes
	appmesh:ListVirtualGateways
	appmesh:ListVirtualNodes
	appmesh:ListVirtualRouters
	appmesh:ListVirtualServices
	appmesh:StreamAggregatedResources
	appmesh:UpdateGatewayRoute
	appmesh:UpdateMesh
	appmesh:UpdateRoute
	appmesh:UpdateVirtualGateway
	appmesh:UpdateVirtualNode
	appmesh:UpdateVirtualRouter
	appmesh:UpdateVirtualService

服務前綴	動作
appstream	appstream:AssociateAppBlockBuilderAppBlock appstream:AssociateApplicationFleet appstream:AssociateApplicationToEntitlement appstream:AssociateFleet appstream：AssociateSoftwareToImageBuilder appstream:BatchAssociateUserStack appstream:BatchDisassociateUserStack appstream:CopyImage appstream:CreateAppBlock appstream:CreateAppBlockBuilder appstream:CreateAppBlockBuilderStreamingURL appstream:CreateApplication appstream:CreateDirectoryConfig appstream:CreateEntitlement appstream:CreateFleet appstream:CreateImageBuilder appstream:CreateImageBuilderStreamingURL appstream:CreateStack appstream:CreateStreamingURL appstream:CreateThemeForStack appstream:CreateUpdatedImage

服務前綴	動作
	appstream:CreateUsageReportSubscription appstream:CreateUser appstream:DeleteAppBlock appstream:DeleteAppBlockBuilder appstream:DeleteApplication appstream:DeleteDirectoryConfig appstream:DeleteEntitlement appstream:DeleteFleet appstream:DeleteImage appstream:DeleteImageBuilder appstream:DeleteImagePermissions appstream:DeleteStack appstream:DeleteThemeForStack appstream:DeleteUsageReportSubscription appstream:DeleteUser appstream:DescribeAppBlockBuilderAppBlockAssociations appstream:DescribeAppBlockBuilders appstream:DescribeAppBlocks appstream：DescribeAppLicenseUsage appstream:DescribeApplicationFleetAssociations appstream:DescribeApplications

服務前綴	動作
	appstream:DescribeDirectoryConfigs appstream:DescribeEntitlements appstream:DescribeFleets appstream:DescribeImageBuilders appstream:DescribeImagePermissions appstream:DescribeImages appstream:DescribeSessions appstream:DescribeStacks appstream:DescribeThemeForStack appstream:DescribeUsageReportSubscriptions appstream:DescribeUserStackAssociations appstream:DescribeUsers appstream:DisableUser appstream:DisassociateAppBlockBuilderAppBlock appstream:DisassociateApplicationFleet appstream:DisassociateApplicationFromEntitlement appstream:DisassociateFleet appstream：DisassociateSoftwareFromImageBuilder appstream:EnableUser appstream:ExpireSession appstream：GetExportImageTask

服務前綴	動作
	appstream:ListAssociatedFleets appstream:ListAssociatedStacks appstream:ListEntitledApplications appstream：ListExportImageTasks appstream:StartAppBlockBuilder appstream:StartFleet appstream:StartImageBuilder appstream：StartSoftwareDeploymentToImageBuilder appstream:StopAppBlockBuilder appstream:StopFleet appstream:StopImageBuilder appstream:UpdateAppBlockBuilder appstream:UpdateApplication appstream:UpdateDirectoryConfig appstream:UpdateEntitlement appstream:UpdateFleet appstream:UpdateImagePermissions appstream:UpdateStack appstream:UpdateThemeForStack

服務前綴	動作
appsync	appsync:AssociateApi
	appsync:AssociateMergedGraphqlApi
	appsync:AssociateSourceGraphqlApi
	appsync:CreateApi
	appsync:CreateApiCache
	appsync:CreateApiKey
	appsync:CreateChannelNamespace
	appsync:CreateDataSource
	appsync:CreateDomainName
	appsync:CreateFunction
	appsync:CreateGraphqlApi
	appsync:CreateResolver
	appsync:CreateType
	appsync:DeleteApi
	appsync:DeleteApiCache
	appsync:DeleteApiKey
	appsync:DeleteChannelNamespace
	appsync:DeleteDataSource
	appsync:DeleteDomainName
	appsync:DeleteFunction
	appsync:DeleteGraphqlApi

服務前綴	動作
	appsync:DeleteResolver
	appsync:DeleteType
	appsync:DisassociateApi
	appsync:DisassociateMergedGraphqlApi
	appsync:DisassociateSourceGraphqlApi
	appsync:EvaluateCode
	appsync:EvaluateMappingTemplate
	appsync:FlushApiCache
	appsync:GetApi
	appsync:GetApiAssociation
	appsync:GetApiCache
	appsync:GetChannelNamespace
	appsync:GetDataSource
	appsync:GetDataSourceIntrospection
	appsync:GetDomainName
	appsync:GetFunction
	appsync:GetGraphqlApi
	appsync:GetGraphqlApiEnvironmentVariables
	appsync:GetIntrospectionSchema
	appsync:GetResolver
	appsync:GetSchemaCreationStatus

服務前綴	動作
	appsync:GetSourceApiAssociation
	appsync:GetType
	appsync:ListApiKeys
	appsync:ListApis
	appsync:ListChannelNamespaces
	appsync:ListDataSources
	appsync:ListDomainNames
	appsync:ListFunctions
	appsync:ListGraphqlApis
	appsync:ListResolvers
	appsync:ListResolversByFunction
	appsync:ListSourceApiAssociations
	appsync:ListTypes
	appsync:ListTypesByAssociation
	appsync:PutGraphqlApiEnvironmentVariables
	appsync:StartDataSourceIntrospection
	appsync:StartSchemaCreation
	appsync:StartSchemaMerge
	appsync:UpdateApi
	appsync:UpdateApiCache
	appsync:UpdateApiKey

服務前綴	動作
	appsync:UpdateChannelNamespace appsync:UpdateDataSource appsync:UpdateDomainName appsync:UpdateFunction appsync:UpdateGraphqlApi appsync:UpdateResolver appsync:UpdateSourceApiAssociation appsync:UpdateType

服務前綴	動作
aps	aps:CreateAlertManagerDefinition aps：CreateAnomalyDetector aps:CreateLoggingConfiguration aps:CreateQueryLoggingConfiguration aps:CreateRuleGroupsNamespace aps:CreateWorkspace aps:DeleteAlertManagerDefinition aps：DeleteAnomalyDetector aps:DeleteLoggingConfiguration aps:DeleteQueryLoggingConfiguration aps:DeleteResourcePolicy aps:DeleteRuleGroupsNamespace aps:DeleteScraper aps：DeleteScraperLoggingConfiguration aps:DeleteWorkspace aps:DescribeAlertManagerDefinition aps：DescribeAnomalyDetector aps:DescribeLoggingConfiguration aps:DescribeQueryLoggingConfiguration aps:DescribeResourcePolicy aps:DescribeRuleGroupsNamespace

服務前綴	動作
	aps:DescribeScraper aps：DescribeScraperLoggingConfiguration aps:DescribeWorkspace aps:DescribeWorkspaceConfiguration aps:GetDefaultScraperConfiguration aps：ListAnomalyDetectors aps:ListRuleGroupsNamespaces aps:ListScrapers aps:ListWorkspaces aps:PutAlertManagerDefinition aps：PutAnomalyDetector aps:PutResourcePolicy aps:PutRuleGroupsNamespace aps:UpdateLoggingConfiguration aps:UpdateQueryLoggingConfiguration aps:UpdateScraper aps：UpdateScraperLoggingConfiguration aps:UpdateWorkspaceAlias aps:UpdateWorkspaceConfiguration

服務前綴	動作
athena	athena:BatchGetNamedQuery
	athena:BatchGetPreparedStatement
	athena:BatchGetQueryExecution
	athena:CancelCapacityReservation
	athena:CreateCapacityReservation
	athena:CreateDataCatalog
	athena:CreateNamedQuery
	athena:CreateNotebook
	athena:CreatePreparedStatement
	athena:CreatePresignedNotebookUrl
	athena:CreateWorkGroup
	athena:DeleteCapacityReservation
	athena:DeleteDataCatalog
	athena:DeleteNamedQuery
	athena:DeleteNotebook
	athena:DeletePreparedStatement
	athena:DeleteWorkGroup
	athena:ExportNotebook
	athena:GetCalculationExecution
	athena:GetCalculationExecutionCode
	athena:GetCalculationExecutionStatus

服務前綴	動作
	athena:GetCapacityAssignmentConfiguration
	athena:GetCapacityReservation
	athena:GetDataCatalog
	athena:GetDatabase
	athena:GetNamedQuery
	athena:GetNotebookMetadata
	athena:GetPreparedStatement
	athena:GetQueryExecution
	athena:GetQueryResults
	athena:GetQueryResultsStream
	athena:GetQueryRuntimeStatistics
	athena：GetResourceDashboard
	athena:GetSession
	athena：GetSessionEndpoint
	athena:GetSessionStatus
	athena:GetTableMetadata
	athena:GetWorkGroup
	athena:ImportNotebook
	athena:ListApplicationDPUSizes
	athena:ListCalculationExecutions
	athena:ListCapacityReservations

服務前綴	動作
	athena:ListDataCatalogs
	athena:ListDatabases
	athena:ListEngineVersions
	athena:ListExecutors
	athena:ListNamedQueries
	athena:ListNotebookMetadata
	athena:ListNotebookSessions
	athena:ListPreparedStatements
	athena:ListQueryExecutions
	athena:ListSessions
	athena:ListTableMetadata
	athena:ListWorkGroups
	athena:PutCapacityAssignmentConfiguration
	athena:StartCalculationExecution
	athena:StartQueryExecution
	athena:StartSession
	athena:StopCalculationExecution
	athena:StopQueryExecution
	athena:TerminateSession
	athena:UpdateCapacityReservation
	athena:UpdateDataCatalog

服務前綴	動作
	athena:UpdateNamedQuery
	athena:UpdateNotebook
	athena:UpdateNotebookMetadata
	athena:UpdatePreparedStatement
	athena:UpdateWorkGroup

服務前綴	動作
auditmanager	auditmanager:AssociateAssessmentReportEvidenceFolder auditmanager:BatchAssociateAssessmentReportEvidence auditmanager:BatchCreateDelegationByAssessment auditmanager:BatchDeleteDelegationByAssessment auditmanager:BatchDisassociateAssessmentReportEvidence auditmanager:BatchImportEvidenceToAssessmentControl auditmanager:CreateAssessment auditmanager:CreateAssessmentFramework auditmanager:CreateAssessmentReport auditmanager:CreateControl auditmanager:DeleteAssessment auditmanager:DeleteAssessmentFramework auditmanager:DeleteAssessmentFrameworkShare auditmanager:DeleteAssessmentReport auditmanager:DeleteControl auditmanager:DeregisterAccount auditmanager:DeregisterOrganizationAdminAccount auditmanager:DisassociateAssessmentReportEvidenceFolder auditmanager:GetAccountStatus auditmanager:GetAssessment auditmanager:GetAssessmentFramework

服務前綴	動作
	auditmanager:GetAssessmentReportUrl auditmanager:GetChangeLogs auditmanager:GetControl auditmanager:GetDelegations auditmanager:GetEvidence auditmanager:GetEvidenceByEvidenceFolder auditmanager:GetEvidenceFileUploadUrl auditmanager:GetEvidenceFolder auditmanager:GetEvidenceFoldersByAssessment auditmanager:GetEvidenceFoldersByAssessmentControl auditmanager:GetInsights auditmanager:GetInsightsByAssessment auditmanager:GetOrganizationAdminAccount auditmanager:GetServicesInScope auditmanager:GetSettings auditmanager:ListAssessmentControlInsightsByControlDomain auditmanager:ListAssessmentFrameworkShareRequests auditmanager:ListAssessmentFrameworks auditmanager:ListAssessmentReports auditmanager:ListAssessments auditmanager:ListControlDomainInsights

服務前綴	動作
	auditmanager:ListControlDomainInsightsByAssessment auditmanager:ListControlInsightsByControlDomain auditmanager:ListControls auditmanager:ListKeywordsForDataSource auditmanager:ListNotifications auditmanager:RegisterAccount auditmanager:RegisterOrganizationAdminAccount auditmanager:StartAssessmentFrameworkShare auditmanager:UpdateAssessment auditmanager:UpdateAssessmentControl auditmanager:UpdateAssessmentControlSetStatus auditmanager:UpdateAssessmentFramework auditmanager:UpdateAssessmentFrameworkShare auditmanager:UpdateAssessmentStatus auditmanager:UpdateControl auditmanager:UpdateSettings auditmanager:ValidateAssessmentReportIntegrity

服務前綴	動作
自動擴展	autoscaling:AttachInstances
	autoscaling:AttachLoadBalancerTargetGroups
	autoscaling:AttachLoadBalancers
	autoscaling:AttachTrafficSources
	autoscaling:BatchDeleteScheduledAction
	autoscaling:BatchPutScheduledUpdateGroupAction
	autoscaling:CancelInstanceRefresh
	autoscaling:CompleteLifecycleAction
	autoscaling:CreateAutoScalingGroup
	autoscaling:CreateLaunchConfiguration
	autoscaling:DeleteAutoScalingGroup
	autoscaling:DeleteLaunchConfiguration
	autoscaling:DeleteLifecycleHook
	autoscaling:DeleteNotificationConfiguration
	autoscaling:DeletePolicy
	autoscaling:DeleteScheduledAction
	autoscaling:DeleteWarmPool
	autoscaling:DescribeAccountLimits
	autoscaling:DescribeAdjustmentTypes
	autoscaling:DescribeAutoScalingGroups
	autoscaling:DescribeAutoScalingInstances

服務前綴	動作
	autoscaling:DescribeAutoScalingNotificationTypes
	autoscaling:DescribeInstanceRefreshes
	autoscaling:DescribeLaunchConfigurations
	autoscaling:DescribeLifecycleHookTypes
	autoscaling:DescribeLifecycleHooks
	autoscaling:DescribeLoadBalancerTargetGroups
	autoscaling:DescribeLoadBalancers
	autoscaling:DescribeMetricCollectionTypes
	autoscaling:DescribeNotificationConfigurations
	autoscaling:DescribePolicies
	autoscaling:DescribeScalingActivities
	autoscaling:DescribeScalingProcessTypes
	autoscaling:DescribeScheduledActions
	autoscaling:DescribeTerminationPolicyTypes
	autoscaling:DescribeTrafficSources
	autoscaling:DescribeWarmPool
	autoscaling:DetachInstances
	autoscaling:DetachLoadBalancerTargetGroups
	autoscaling:DetachLoadBalancers
	autoscaling:DetachTrafficSources
	autoscaling:DisableMetricsCollection

服務前綴	動作
	autoscaling:EnableMetricsCollection autoscaling:EnterStandby autoscaling:ExecutePolicy autoscaling:ExitStandby autoscaling:GetPredictiveScalingForecast autoscaling:PutLifecycleHook autoscaling:PutNotificationConfiguration autoscaling:PutScalingPolicy autoscaling:PutScheduledUpdateGroupAction autoscaling:PutWarmPool autoscaling:RecordLifecycleActionHeartbeat autoscaling:ResumeProcesses autoscaling:RollbackInstanceRefresh autoscaling:SetDesiredCapacity autoscaling:SetInstanceHealth autoscaling:SetInstanceProtection autoscaling:StartInstanceRefresh autoscaling:SuspendProcesses autoscaling:TerminateInstanceInAutoScalingGroup autoscaling:UpdateAutoScalingGroup
aws-marketplace	aws-marketplace:GetEntitlements

服務前綴	動作
備份	backup:CancelLegalHold backup:CreateBackupPlan backup:CreateBackupSelection backup:CreateBackupVault backup:CreateFramework backup:CreateLegalHold backup:CreateLogicallyAirGappedBackupVault backup:CreateReportPlan backup:CreateRestoreAccessBackupVault backup:CreateRestoreTestingPlan backup:CreateRestoreTestingSelection 備份：CreateTieringConfiguration backup:DeleteBackupPlan backup:DeleteBackupSelection backup:DeleteBackupVault backup:DeleteBackupVaultAccessPolicy backup:DeleteBackupVaultLockConfiguration backup:DeleteBackupVaultNotifications backup:DeleteFramework backup:DeleteRecoveryPoint backup:DeleteReportPlan

服務前綴	動作
	backup:DeleteRestoreTestingPlan
	backup:DeleteRestoreTestingSelection
	備份：DeleteTieringConfiguration
	backup:DescribeBackupJob
	backup:DescribeBackupVault
	backup:DescribeCopyJob
	backup:DescribeFramework
	backup:DescribeGlobalSettings
	backup:DescribeProtectedResource
	backup:DescribeRecoveryPoint
	backup:DescribeRegionSettings
	backup:DescribeReportJob
	backup:DescribeReportPlan
	backup:DescribeRestoreJob
	備份：DescribeScanJob
	backup:DisassociateRecoveryPoint
	backup:DisassociateRecoveryPointFromParent
	backup:ExportBackupPlanTemplate
	backup:GetBackupPlan
	backup:GetBackupPlanFromJSON
	backup:GetBackupPlanFromTemplate

服務前綴	動作
	backup:GetBackupSelection
	backup:GetBackupVaultAccessPolicy
	backup:GetBackupVaultNotifications
	backup:GetLegalHold
	backup:GetRecoveryPointRestoreMetadata
	backup:GetRestoreJobMetadata
	backup:GetRestoreTestingInferredMetadata
	backup:GetRestoreTestingPlan
	backup:GetRestoreTestingSelection
	backup:GetSupportedResourceTypes
	備份：GetTieringConfiguration
	backup:ListBackupJobSummaries
	backup:ListBackupJobs
	backup:ListBackupPlanTemplates
	backup:ListBackupPlanVersions
	backup:ListBackupPlans
	backup:ListBackupSelections
	backup:ListBackupVaults
	backup:ListCopyJobSummaries
	backup:ListCopyJobs
	backup:ListFrameworks

服務前綴	動作
	backup:ListIndexedRecoveryPoints
	backup:ListLegalHolds
	backup:ListProtectedResources
	backup:ListRecoveryPointsByBackupVault
	backup:ListRecoveryPointsByLegalHold
	backup:ListRecoveryPointsByResource
	backup:ListReportJobs
	backup:ListReportPlans
	backup:ListRestoreAccessBackupVaults
	backup:ListRestoreJobSummaries
	backup:ListRestoreJobs
	backup:ListRestoreJobsByProtectedResource
	backup:ListRestoreTestingPlans
	backup:ListRestoreTestingSelections
	備份：ListScanJobSummaries
	備份：ListScanJobs
	備份：ListTieringConfigurations
	backup:PutBackupVaultAccessPolicy
	backup:PutBackupVaultLockConfiguration
	backup:PutBackupVaultNotifications
	backup:PutRestoreValidationResult

服務前綴	動作
	backup:StartBackupJob
	backup:StartCopyJob
	backup:StartReportJob
	backup:StartRestoreJob
	backup:StopBackupJob
	backup:UpdateBackupPlan
	backup:UpdateFramework
	backup:UpdateGlobalSettings
	backup:UpdateRecoveryPointLifecycle
	backup:UpdateRegionSettings
	backup:UpdateReportPlan
	backup:UpdateRestoreTestingPlan
	backup:UpdateRestoreTestingSelection
	備份：UpdateTieringConfiguration

服務前綴	動作
批次	batch:CancelJob batch:CreateComputeEnvironment batch:CreateConsumableResource batch:CreateJobQueue batch：CreateQuotaShare batch:CreateSchedulingPolicy batch:CreateServiceEnvironment batch:DeleteComputeEnvironment batch:DeleteConsumableResource batch:DeleteJobQueue batch：DeleteQuotaShare batch:DeleteSchedulingPolicy batch:DeleteServiceEnvironment batch:DeregisterJobDefinition batch:DescribeComputeEnvironments batch:DescribeConsumableResource batch:DescribeJobDefinitions batch:DescribeJobQueues batch:DescribeJobs batch：DescribeQuotaShare batch:DescribeSchedulingPolicies

服務前綴	動作
	batch:DescribeServiceEnvironments batch:DescribeServiceJob batch:GetJobQueueSnapshot batch:ListConsumableResources batch:ListJobs batch:ListJobsByConsumableResource batch：ListQuotaShares batch:ListSchedulingPolicies batch:ListServiceJobs batch:RegisterJobDefinition batch:SubmitJob batch:SubmitServiceJob batch:TerminateJob batch:TerminateServiceJob batch:UpdateComputeEnvironment batch:UpdateConsumableResource batch:UpdateJobQueue batch：UpdateQuotaShare batch:UpdateSchedulingPolicy batch:UpdateServiceEnvironment batch：UpdateServiceJob

服務前綴	動作
braket	braket:CancelJob braket:CancelQuantumTask braket:CreateJob braket:CreateQuantumTask braket：CreateSpendingLimit braket:GetDevice braket:GetJob braket:GetQuantumTask braket:SearchDevices braket:SearchJobs braket:SearchQuantumTasks braket：SearchSpendingLimits

服務前綴	動作
預算	budgets:CreateBudgetAction budgets:DeleteBudgetAction budgets:DescribeBudgetAction budgets:DescribeBudgetActionHistories budgets:DescribeBudgetActionsForAccount budgets:DescribeBudgetActionsForBudget budgets:ExecuteBudgetAction budgets:ModifyBudget budgets:UpdateBudgetAction budgets:ViewBudget
Cloud9	cloud9:CreateEnvironmentEC2 cloud9:CreateEnvironmentMembership cloud9:DeleteEnvironment cloud9:DeleteEnvironmentMembership cloud9:DescribeEnvironmentMemberships cloud9:DescribeEnvironmentStatus cloud9:DescribeEnvironments cloud9:ListEnvironments cloud9:UpdateEnvironment cloud9:UpdateEnvironmentMembership

服務前綴	動作
cloudformation	cloudformation:BatchDescribeTypeConfigurations cloudformation:CancelUpdateStack cloudformation:ContinueUpdateRollback cloudformation:CreateChangeSet cloudformation:CreateGeneratedTemplate cloudformation:CreateStack cloudformation:CreateStackInstances cloudformation:CreateStackSet cloudformation:DeactivateType cloudformation:DeleteChangeSet cloudformation:DeleteGeneratedTemplate cloudformation:DeleteStack cloudformation:DeleteStackInstances cloudformation:DeleteStackSet cloudformation:DeregisterType cloudformation:DescribeAccountLimits cloudformation:DescribeChangeSet cloudformation:DescribeChangeSetHooks cloudformation：DescribeEvents cloudformation:DescribeGeneratedTemplate cloudformation:DescribeOrganizationsAccess

服務前綴	動作
	cloudformation:DescribePublisher cloudformation:DescribeResourceScan cloudformation:DescribeStackDriftDetectionStatus cloudformation:DescribeStackEvents cloudformation:DescribeStackInstance cloudformation:DescribeStackResource cloudformation:DescribeStackResourceDrifts cloudformation:DescribeStackResources cloudformation:DescribeStackSet cloudformation:DescribeStackSetOperation cloudformation:DescribeStacks cloudformation:DescribeType cloudformation:DescribeTypeRegistration cloudformation:DetectStackDrift cloudformation:DetectStackResourceDrift cloudformation:DetectStackSetDrift cloudformation:EstimateTemplateCost cloudformation:ExecuteChangeSet cloudformation:GetGeneratedTemplate cloudformation：GetHookResult cloudformation:GetStackPolicy

服務前綴	動作
	cloudformation:GetTemplate cloudformation:GetTemplateSummary cloudformation:ImportStacksToStackSet cloudformation:ListChangeSets cloudformation:ListExports cloudformation:ListGeneratedTemplates cloudformation:ListHookResults cloudformation:ListImports cloudformation:ListResourceScanRelatedResources cloudformation:ListResourceScanResources cloudformation:ListResourceScans cloudformation:ListStackInstanceResourceDrifts cloudformation:ListStackInstances cloudformation:ListStackRefactors cloudformation:ListStackResources cloudformation:ListStackSetAutoDeploymentTargets cloudformation:ListStackSetOperationResults cloudformation:ListStackSetOperations cloudformation:ListStackSets cloudformation:ListTypeRegistrations cloudformation:ListTypeVersions

服務前綴	動作
	cloudformation:ListTypes cloudformation:PublishType cloudformation:RecordHandlerProgress cloudformation:RegisterPublisher cloudformation:RegisterType cloudformation:RollbackStack cloudformation:SetStackPolicy cloudformation:SetTypeConfiguration cloudformation:SetTypeDefaultVersion cloudformation:SignalResource cloudformation:StartResourceScan cloudformation:StopStackSetOperation cloudformation:TestType cloudformation:UpdateGeneratedTemplate cloudformation:UpdateStack cloudformation:UpdateStackInstances cloudformation:UpdateStackSet cloudformation:UpdateTerminationProtection cloudformation:ValidateTemplate

服務前綴	動作
cloudfront	cloudfront:AssociateAlias
	cloudfront:AssociateDistributionTenantWebACL
	cloudfront:AssociateDistributionWebACL
	cloudfront:CreateCachePolicy
	cloudfront:CreateCloudFrontOriginAccessIdentity
	cloudfront：CreateConnectionFunction
	cloudfront:CreateContinuousDeploymentPolicy
	cloudfront:CreateDistributionTenant
	cloudfront:CreateFieldLevelEncryptionConfig
	cloudfront:CreateFieldLevelEncryptionProfile
	cloudfront:CreateFunction
	cloudfront:CreateInvalidation
	cloudfront:CreateKeyGroup
	cloudfront:CreateKeyValueStore
	cloudfront:CreateMonitoringSubscription
	cloudfront:CreateOriginAccessControl
	cloudfront:CreateOriginRequestPolicy
	cloudfront:CreatePublicKey
	cloudfront:CreateRealtimeLogConfig
	cloudfront:CreateResponseHeadersPolicy
	cloudfront：CreateTrustStore

服務前綴	動作
	cloudfront:DeleteAnycastIpList
	cloudfront:DeleteCachePolicy
	cloudfront:DeleteCloudFrontOriginAccessIdentity
	cloudfront：DeleteConnectionFunction
	cloudfront:DeleteConnectionGroup
	cloudfront:DeleteContinuousDeploymentPolicy
	cloudfront:DeleteDistribution
	cloudfront:DeleteDistributionTenant
	cloudfront:DeleteFieldLevelEncryptionConfig
	cloudfront:DeleteFieldLevelEncryptionProfile
	cloudfront:DeleteFunction
	cloudfront:DeleteKeyGroup
	cloudfront:DeleteKeyValueStore
	cloudfront:DeleteMonitoringSubscription
	cloudfront:DeleteOriginAccessControl
	cloudfront:DeleteOriginRequestPolicy
	cloudfront:DeletePublicKey
	cloudfront:DeleteRealtimeLogConfig
	cloudfront:DeleteResponseHeadersPolicy
	cloudfront:DeleteStreamingDistribution
	cloudfront：DeleteTrustStore

服務前綴	動作
	cloudfront:DeleteVpcOrigin
	cloudfront:DescribeFunction
	cloudfront:DescribeKeyValueStore
	cloudfront:DisassociateDistributionTenantWebACL
	cloudfront:DisassociateDistributionWebACL
	cloudfront:GetAnycastIpList
	cloudfront:GetCachePolicy
	cloudfront:GetCachePolicyConfig
	cloudfront:GetCloudFrontOriginAccessIdentity
	cloudfront:GetCloudFrontOriginAccessIdentityConfig
	cloudfront:GetContinuousDeploymentPolicy
	cloudfront:GetContinuousDeploymentPolicyConfig
	cloudfront:GetDistributionConfig
	cloudfront:GetFieldLevelEncryption
	cloudfront:GetFieldLevelEncryptionConfig
	cloudfront:GetFieldLevelEncryptionProfile
	cloudfront:GetFieldLevelEncryptionProfileConfig
	cloudfront:GetFunction
	cloudfront:GetInvalidation
	cloudfront:GetInvalidationForDistributionTenant
	cloudfront:GetKeyGroup

服務前綴	動作
	cloudfront:GetKeyGroupConfig
	cloudfront:GetMonitoringSubscription
	cloudfront:GetOriginAccessControl
	cloudfront:GetOriginAccessControlConfig
	cloudfront:GetOriginRequestPolicy
	cloudfront:GetOriginRequestPolicyConfig
	cloudfront:GetPublicKey
	cloudfront:GetPublicKeyConfig
	cloudfront:GetRealtimeLogConfig
	cloudfront:GetResponseHeadersPolicy
	cloudfront:GetResponseHeadersPolicyConfig
	cloudfront:GetStreamingDistribution
	cloudfront:GetStreamingDistributionConfig
	cloudfront:GetVpcOrigin
	cloudfront:ListAnycastIpLists
	cloudfront:ListCachePolicies
	cloudfront:ListCloudFrontOriginAccessIdentities
	cloudfront:ListConflictingAliases
	cloudfront：ListConnectionFunctions
	cloudfront:ListConnectionGroups
	cloudfront:ListContinuousDeploymentPolicies

服務前綴	動作
	cloudfront:ListDistributionTenants
	cloudfront:ListDistributionTenantsByCustomization
	cloudfront:ListDistributions
	cloudfront:ListDistributionsByAnycastIpListId
	cloudfront:ListDistributionsByCachePolicyId
	cloudfront:ListDistributionsByConnectionMode
	cloudfront:ListDistributionsByKeyGroup
	cloudfront:ListDistributionsByOriginRequestPolicyId
	cloudfront:ListDistributionsByRealtimeLogConfig
	cloudfront:ListDistributionsByResponseHeadersPolicyId
	cloudfront:ListDistributionsByVpcOriginId
	cloudfront:ListDistributionsByWebACLId
	cloudfront:ListFieldLevelEncryptionConfigs
	cloudfront:ListFieldLevelEncryptionProfiles
	cloudfront:ListFunctions
	cloudfront:ListInvalidations
	cloudfront:ListInvalidationsForDistributionTenant
	cloudfront:ListKeyGroups
	cloudfront:ListKeyValueStores
	cloudfront:ListOriginAccessControls
	cloudfront:ListOriginRequestPolicies

服務前綴	動作
	cloudfront:ListPublicKeys
	cloudfront:ListRealtimeLogConfigs
	cloudfront:ListResponseHeadersPolicies
	cloudfront:ListStreamingDistributions
	cloudfront：ListTrustStores
	cloudfront：PublishConnectionFunction
	cloudfront:PublishFunction
	cloudfront：TestConnectionFunction
	cloudfront:TestFunction
	cloudfront：UpdateAnycastIpList
	cloudfront:UpdateCachePolicy
	cloudfront:UpdateCloudFrontOriginAccessIdentity
	cloudfront：UpdateConnectionFunction
	cloudfront:UpdateConnectionGroup
	cloudfront:UpdateContinuousDeploymentPolicy
	cloudfront:UpdateDistribution
	cloudfront:UpdateDistributionTenant
	cloudfront:UpdateFieldLevelEncryptionConfig
	cloudfront:UpdateFieldLevelEncryptionProfile
	cloudfront:UpdateFunction
	cloudfront:UpdateKeyGroup

服務前綴	動作
	cloudfront:UpdateKeyValueStore cloudfront:UpdateOriginAccessControl cloudfront:UpdateOriginRequestPolicy cloudfront:UpdatePublicKey cloudfront:UpdateRealtimeLogConfig cloudfront:UpdateResponseHeadersPolicy cloudfront：UpdateTrustStore
cloudhsm	cloudhsm:CreateHsm cloudhsm:DeleteBackup cloudhsm:DeleteHsm cloudhsm:DeleteResourcePolicy cloudhsm:DescribeBackups cloudhsm:DescribeClusters cloudhsm:GetResourcePolicy cloudhsm:InitializeCluster cloudhsm:ModifyBackupAttributes cloudhsm:ModifyCluster cloudhsm:PutResourcePolicy cloudhsm:RestoreBackup

服務前綴	動作
cloudsearch	cloudsearch:BuildSuggesters
	cloudsearch:CreateDomain
	cloudsearch:DefineAnalysisScheme
	cloudsearch:DefineExpression
	cloudsearch:DefineIndexField
	cloudsearch:DefineSuggester
	cloudsearch:DeleteAnalysisScheme
	cloudsearch:DeleteDomain
	cloudsearch:DeleteExpression
	cloudsearch:DeleteIndexField
	cloudsearch:DeleteSuggester
	cloudsearch:DescribeAnalysisSchemes
	cloudsearch:DescribeAvailabilityOptions
	cloudsearch:DescribeDomainEndpointOptions
	cloudsearch:DescribeDomains
	cloudsearch:DescribeExpressions
	cloudsearch:DescribeIndexFields
	cloudsearch:DescribeScalingParameters
	cloudsearch:DescribeServiceAccessPolicies
	cloudsearch:DescribeSuggesters
	cloudsearch:IndexDocuments

服務前綴	動作
	cloudsearch:ListDomainNames
	cloudsearch:UpdateAvailabilityOptions
	cloudsearch:UpdateDomainEndpointOptions
	cloudsearch:UpdateScalingParameters
	cloudsearch:UpdateServiceAccessPolicies

服務前綴	動作
cloudtrail	cloudtrail:CancelQuery cloudtrail:CreateChannel cloudtrail:CreateDashboard cloudtrail:CreateEventDataStore cloudtrail:CreateTrail cloudtrail:DeleteChannel cloudtrail:DeleteDashboard cloudtrail:DeleteEventDataStore cloudtrail:DeleteResourcePolicy cloudtrail:DeleteTrail cloudtrail:DeregisterOrganizationDelegatedAdmin cloudtrail:DescribeQuery cloudtrail:DescribeTrails cloudtrail:DisableFederation cloudtrail:GenerateQuery cloudtrail:GetChannel cloudtrail:GetDashboard cloudtrail:GetEventConfiguration cloudtrail:GetEventDataStore cloudtrail:GetEventDataStoreData cloudtrail:GetEventSelectors

服務前綴	動作
	cloudtrail:GetImport cloudtrail:GetInsightSelectors cloudtrail:GetResourcePolicy cloudtrail:GetTrail cloudtrail:GetTrailStatus cloudtrail:ListChannels cloudtrail:ListDashboards cloudtrail:ListEventDataStores cloudtrail:ListImportFailures cloudtrail:ListImports cloudtrail：ListInsightsData cloudtrail:ListPublicKeys cloudtrail:ListQueries cloudtrail:ListTrails cloudtrail:LookupEvents cloudtrail:PutEventConfiguration cloudtrail:PutEventSelectors cloudtrail:PutInsightSelectors cloudtrail:PutResourcePolicy cloudtrail:RegisterOrganizationDelegatedAdmin cloudtrail:RestoreEventDataStore

服務前綴	動作
	cloudtrail:SearchSampleQueries cloudtrail:StartEventDataStoreIngestion cloudtrail:StartImport cloudtrail:StartLogging cloudtrail:StartQuery cloudtrail:StopEventDataStoreIngestion cloudtrail:StopImport cloudtrail:StopLogging cloudtrail:UpdateChannel cloudtrail:UpdateDashboard cloudtrail:UpdateEventDataStore cloudtrail:UpdateTrail

服務前綴	動作
cloudwatch	cloudwatch：DeleteAlarmMuteRule
	cloudwatch:DeleteAlarms
	cloudwatch:DeleteAnomalyDetector
	cloudwatch:DeleteDashboards
	cloudwatch:DeleteInsightRules
	cloudwatch:DeleteMetricStream
	cloudwatch:DescribeAlarmHistory
	cloudwatch:DescribeAlarms
	cloudwatch:DescribeAlarmsForMetric
	cloudwatch:DescribeAnomalyDetectors
	cloudwatch:DescribeInsightRules
	cloudwatch:DisableAlarmActions
	cloudwatch:DisableInsightRules
	cloudwatch:EnableAlarmActions
	cloudwatch:EnableInsightRules
	cloudwatch：GetAlarmMuteRule
	cloudwatch:GetDashboard
	cloudwatch：GetDataset
	cloudwatch:GetInsightRuleReport
	cloudwatch:GetMetricStatistics
	cloudwatch:GetMetricStream

服務前綴	動作
	cloudwatch：ListAlarmMuteRules
	cloudwatch:ListDashboards
	cloudwatch:ListManagedInsightRules
	cloudwatch:ListMetricStreams
	cloudwatch：PutAlarmMuteRule
	cloudwatch:PutAnomalyDetector
	cloudwatch:PutCompositeAlarm
	cloudwatch:PutDashboard
	cloudwatch:PutInsightRule
	cloudwatch：PutLogAlarm
	cloudwatch:PutManagedInsightRules
	cloudwatch:PutMetricAlarm
	cloudwatch:PutMetricStream
	cloudwatch:SetAlarmState
	cloudwatch:StartMetricStreams
	cloudwatch:StopMetricStreams

服務前綴	動作
codeartifact	codeartifact:AssociateExternalConnection codeartifact:CopyPackageVersions codeartifact:CreateDomain codeartifact:CreateRepository codeartifact:DeleteDomain codeartifact:DeleteDomainPermissionsPolicy codeartifact:DeletePackage codeartifact:DeletePackageVersions codeartifact:DeleteRepository codeartifact:DeleteRepositoryPermissionsPolicy codeartifact:DescribeDomain codeartifact:DescribePackage codeartifact:DescribePackageVersion codeartifact:DescribeRepository codeartifact:DisassociateExternalConnection codeartifact:DisposePackageVersions codeartifact:GetAssociatedPackageGroup codeartifact:GetAuthorizationToken codeartifact:GetDomainPermissionsPolicy codeartifact:GetPackageVersionAsset codeartifact:GetPackageVersionReadme

服務前綴	動作
	codeartifact:GetRepositoryEndpoint codeartifact:GetRepositoryPermissionsPolicy codeartifact:ListDomains codeartifact:ListPackageGroups codeartifact:ListPackageVersionAssets codeartifact:ListPackageVersionDependencies codeartifact:ListPackageVersions codeartifact:ListPackages codeartifact:ListRepositories codeartifact:ListRepositoriesInDomain codeartifact:PublishPackageVersion codeartifact:PutDomainPermissionsPolicy codeartifact:PutPackageMetadata codeartifact:PutPackageOriginConfiguration codeartifact:PutRepositoryPermissionsPolicy codeartifact:ReadFromRepository codeartifact:UpdatePackageVersionsStatus codeartifact:UpdateRepository

服務前綴	動作
codedeploy	codedeploy:BatchGetApplicationRevisions
	codedeploy:BatchGetApplications
	codedeploy:BatchGetDeploymentGroups
	codedeploy:BatchGetDeploymentInstances
	codedeploy:BatchGetDeploymentTargets
	codedeploy:BatchGetDeployments
	codedeploy:BatchGetOnPremisesInstances
	codedeploy:ContinueDeployment
	codedeploy:CreateApplication
	codedeploy:CreateDeployment
	codedeploy:CreateDeploymentConfig
	codedeploy:CreateDeploymentGroup
	codedeploy:DeleteApplication
	codedeploy:DeleteDeploymentConfig
	codedeploy:DeleteDeploymentGroup
	codedeploy:DeleteGitHubAccountToken
	codedeploy:DeleteResourcesByExternalId
	codedeploy:DeregisterOnPremisesInstance
	codedeploy:GetApplication
	codedeploy:GetApplicationRevision
	codedeploy:GetDeployment

服務前綴	動作
	codedeploy:GetDeploymentConfig
	codedeploy:GetDeploymentGroup
	codedeploy:GetDeploymentInstance
	codedeploy:GetDeploymentTarget
	codedeploy:GetOnPremisesInstance
	codedeploy:ListApplicationRevisions
	codedeploy:ListApplications
	codedeploy:ListDeploymentConfigs
	codedeploy:ListDeploymentGroups
	codedeploy:ListDeploymentInstances
	codedeploy:ListDeploymentTargets
	codedeploy:ListDeployments
	codedeploy:ListGitHubAccountTokenNames
	codedeploy:ListOnPremisesInstances
	codedeploy:PutLifecycleEventHookExecutionStatus
	codedeploy:RegisterApplicationRevision
	codedeploy:RegisterOnPremisesInstance
	codedeploy:SkipWaitTimeForInstanceTermination
	codedeploy:StopDeployment
	codedeploy:UpdateApplication
	codedeploy:UpdateDeploymentGroup

服務前綴	動作
codeguru-profiler	codeguru-profiler:AddNotificationChannels codeguru-profiler:BatchGetFrameMetricData codeguru-profiler:CreateProfilingGroup codeguru-profiler:DeleteProfilingGroup codeguru-profiler:DescribeProfilingGroup codeguru-profiler:GetFindingsReportAccountSummary codeguru-profiler:GetNotificationConfiguration codeguru-profiler:GetPolicy codeguru-profiler:GetProfile codeguru-profiler:GetRecommendations codeguru-profiler:ListFindingsReports codeguru-profiler:ListProfileTimes codeguru-profiler:ListProfilingGroups codeguru-profiler:PutPermission codeguru-profiler:RemoveNotificationChannel codeguru-profiler:RemovePermission codeguru-profiler:SubmitFeedback codeguru-profiler:UpdateProfilingGroup

服務前綴	動作
codeguru-reviewer	codeguru-reviewer:AssociateRepository codeguru-reviewer:CreateCodeReview codeguru-reviewer:DescribeCodeReview codeguru-reviewer:DescribeRecommendationFeedback codeguru-reviewer:DescribeRepositoryAssociation codeguru-reviewer:DisassociateRepository codeguru-reviewer:ListCodeReviews codeguru-reviewer:ListRecommendationFeedback codeguru-reviewer:ListRecommendations codeguru-reviewer:ListRepositoryAssociations codeguru-reviewer:PutRecommendationFeedback

服務前綴	動作
codepipeline	codepipeline:AcknowledgeJob codepipeline:AcknowledgeThirdPartyJob codepipeline:CreateCustomActionType codepipeline:CreatePipeline codepipeline:DeleteCustomActionType codepipeline:DeletePipeline codepipeline:DeleteWebhook codepipeline:DeregisterWebhookWithThirdParty codepipeline:GetActionType codepipeline:GetJobDetails codepipeline:GetPipeline codepipeline:GetPipelineExecution codepipeline:GetPipelineState codepipeline:GetThirdPartyJobDetails codepipeline:ListActionExecutions codepipeline:ListActionTypes codepipeline:ListPipelineExecutions codepipeline:ListPipelines codepipeline:ListRuleExecutions codepipeline:ListRuleTypes codepipeline:ListWebhooks

服務前綴	動作
	codepipeline:OverrideStageCondition codepipeline:PollForJobs codepipeline:PollForThirdPartyJobs codepipeline:PutActionRevision codepipeline:PutApprovalResult codepipeline:PutJobFailureResult codepipeline:PutJobSuccessResult codepipeline:PutThirdPartyJobFailureResult codepipeline:PutThirdPartyJobSuccessResult codepipeline:PutWebhook codepipeline:RegisterWebhookWithThirdParty codepipeline:RollbackStage codepipeline:StartPipelineExecution codepipeline:StopPipelineExecution codepipeline:UpdateActionType codepipeline:UpdatePipeline

服務前綴	動作
codestar	codestar:AssociateTeamMember
	codestar:CreateProject
	codestar:CreateUserProfile
	codestar:DeleteProject
	codestar:DeleteUserProfile
	codestar:DescribeProject
	codestar:DescribeUserProfile
	codestar:DisassociateTeamMember
	codestar:ListProjects
	codestar:ListResources
	codestar:ListTeamMembers
	codestar:ListUserProfiles
	codestar:UpdateProject
	codestar:UpdateTeamMember
	codestar:UpdateUserProfile

服務前綴	動作
codestar-notifications	codestar-notifications:CreateNotificationRule codestar-notifications:DeleteNotificationRule codestar-notifications:DeleteTarget codestar-notifications:DescribeNotificationRule codestar-notifications:ListEventTypes codestar-notifications:ListNotificationRules codestar-notifications:ListTargets codestar-notifications:Subscribe codestar-notifications:Unsubscribe codestar-notifications:UpdateNotificationRule

服務前綴	動作
cognito-identity	cognito-identity:CreateIdentityPool
	cognito-identity:DeleteIdentities
	cognito-identity:DeleteIdentityPool
	cognito-identity:DescribeIdentity
	cognito-identity:DescribeIdentityPool
	cognito-identity:GetIdentityPoolRoles
	cognito-identity:ListIdentities
	cognito-identity:ListIdentityPools
	cognito-identity:LookupDeveloperIdentity
	cognito-identity:MergeDeveloperIdentities
	cognito-identity:SetIdentityPoolRoles
	cognito-identity:UnlinkDeveloperIdentity
	cognito-identity:UpdateIdentityPool

服務前綴	動作
cognito-idp	cognito-idp:AddCustomAttributes cognito-idp：AddUserPoolClientSecret cognito-idp:AdminAddUserToGroup cognito-idp:AdminConfirmSignUp cognito-idp:AdminCreateUser cognito-idp:AdminDeleteUser cognito-idp:AdminDeleteUserAttributes cognito-idp:AdminDisableProviderForUser cognito-idp:AdminDisableUser cognito-idp:AdminEnableUser cognito-idp:AdminForgetDevice cognito-idp:AdminGetDevice cognito-idp:AdminGetUser cognito-idp：AdminGetUserAuthFactors cognito-idp:AdminInitiateAuth cognito-idp:AdminLinkProviderForUser cognito-idp:AdminListDevices cognito-idp:AdminListGroupsForUser cognito-idp:AdminListUserAuthEvents cognito-idp:AdminRemoveUserFromGroup cognito-idp:AdminResetUserPassword

服務前綴	動作
	cognito-idp:AdminRespondToAuthChallenge cognito-idp:AdminSetUserMFAPreference cognito-idp:AdminSetUserPassword cognito-idp:AdminSetUserSettings cognito-idp:AdminUpdateAuthEventFeedback cognito-idp:AdminUpdateDeviceStatus cognito-idp:AdminUpdateUserAttributes cognito-idp:AdminUserGlobalSignOut cognito-idp:AssociateSoftwareToken cognito-idp:ChangePassword cognito-idp:ConfirmDevice cognito-idp:ConfirmForgotPassword cognito-idp:ConfirmSignUp cognito-idp:CreateGroup cognito-idp:CreateIdentityProvider cognito-idp:CreateManagedLoginBranding cognito-idp:CreateResourceServer cognito-idp：CreateTerms cognito-idp:CreateUserImportJob cognito-idp:CreateUserPool cognito-idp:CreateUserPoolClient

服務前綴	動作
	cognito-idp:CreateUserPoolDomain cognito-idp：CreateUserPoolReplica cognito-idp:DeleteGroup cognito-idp:DeleteIdentityProvider cognito-idp:DeleteManagedLoginBranding cognito-idp:DeleteResourceServer cognito-idp：DeleteTerms cognito-idp:DeleteUser cognito-idp:DeleteUserAttributes cognito-idp:DeleteUserPool cognito-idp:DeleteUserPoolClient cognito-idp：DeleteUserPoolClientSecret cognito-idp:DeleteUserPoolDomain cognito-idp：DeleteUserPoolReplica cognito-idp:DescribeIdentityProvider cognito-idp:DescribeManagedLoginBranding cognito-idp:DescribeManagedLoginBrandingByClient cognito-idp:DescribeResourceServer cognito-idp:DescribeRiskConfiguration cognito-idp：DescribeTerms cognito-idp:DescribeUserImportJob

服務前綴	動作
	cognito-idp:DescribeUserPool cognito-idp:DescribeUserPoolClient cognito-idp:DescribeUserPoolDomain cognito-idp:ForgetDevice cognito-idp:ForgotPassword cognito-idp:GetCSVHeader cognito-idp:GetDevice cognito-idp:GetGroup cognito-idp:GetIdentityProviderByIdentifier cognito-idp:GetLogDeliveryConfiguration cognito-idp：GetProvisionedLimit cognito-idp:GetSigningCertificate cognito-idp:GetUICustomization cognito-idp:GetUser cognito-idp:GetUserAttributeVerificationCode cognito-idp:GetUserPoolMfaConfig cognito-idp:GlobalSignOut cognito-idp:InitiateAuth cognito-idp:ListDevices cognito-idp:ListGroups cognito-idp:ListIdentityProviders

服務前綴	動作
	cognito-idp:ListResourceServers cognito-idp：ListTerms cognito-idp:ListUserImportJobs cognito-idp：ListUserPoolClientSecrets cognito-idp:ListUserPoolClients cognito-idp：ListUserPoolReplicas cognito-idp:ListUserPools cognito-idp:ListUsers cognito-idp:ListUsersInGroup cognito-idp:ResendConfirmationCode cognito-idp:RespondToAuthChallenge cognito-idp:RevokeToken cognito-idp:SetLogDeliveryConfiguration cognito-idp:SetRiskConfiguration cognito-idp:SetUICustomization cognito-idp:SetUserMFAPreference cognito-idp:SetUserPoolMfaConfig cognito-idp:SetUserSettings cognito-idp:SignUp cognito-idp:StartUserImportJob cognito-idp:StopUserImportJob

服務前綴	動作
	cognito-idp:UpdateAuthEventFeedback cognito-idp:UpdateDeviceStatus cognito-idp:UpdateGroup cognito-idp:UpdateIdentityProvider cognito-idp：UpdateProvisionedLimit cognito-idp:UpdateResourceServer cognito-idp：UpdateTerms cognito-idp:UpdateUserAttributes cognito-idp:UpdateUserPool cognito-idp:UpdateUserPoolClient cognito-idp:UpdateUserPoolDomain cognito-idp：UpdateUserPoolReplica cognito-idp:VerifySoftwareToken cognito-idp:VerifyUserAttribute

服務前綴	動作
cognito-sync	cognito-sync:BulkPublish
	cognito-sync:DeleteDataset
	cognito-sync:DescribeDataset
	cognito-sync:DescribeIdentityPoolUsage
	cognito-sync:DescribeIdentityUsage
	cognito-sync:GetBulkPublishDetails
	cognito-sync:GetCognitoEvents
	cognito-sync:GetIdentityPoolConfiguration
	cognito-sync:ListDatasets
	cognito-sync:ListIdentityPoolUsage
	cognito-sync:ListRecords
	cognito-sync:RegisterDevice
	cognito-sync:SetCognitoEvents
	cognito-sync:SetIdentityPoolConfiguration
	cognito-sync:SubscribeToDataset
	cognito-sync:UnsubscribeFromDataset
	cognito-sync:UpdateRecords

服務前綴	動作
comprehendmedical	comprehendmedical:DescribeEntitiesDetectionV2Job comprehendmedical:DescribeICD10CMInferenceJob comprehendmedical:DescribePHIDetectionJob comprehendmedical:DescribeRxNormInferenceJob comprehendmedical:DescribeSNOMEDCTInferenceJob comprehendmedical:DetectEntitiesV2 comprehendmedical:DetectPHI comprehendmedical:InferICD10CM comprehendmedical:InferRxNorm comprehendmedical:InferSNOMEDCT comprehendmedical:ListEntitiesDetectionV2Jobs comprehendmedical:ListICD10CMInferenceJobs comprehendmedical:ListPHIDetectionJobs comprehendmedical:ListRxNormInferenceJobs comprehendmedical:ListSNOMEDCTInferenceJobs comprehendmedical:StartEntitiesDetectionV2Job comprehendmedical:StartICD10CMInferenceJob comprehendmedical:StartPHIDetectionJob comprehendmedical:StartRxNormInferenceJob comprehendmedical:StartSNOMEDCTInferenceJob comprehendmedical:StopEntitiesDetectionV2Job

服務前綴	動作
	comprehendmedical:StopICD10CMInferenceJob comprehendmedical:StopPHIDetectionJob comprehendmedical:StopRxNormInferenceJob comprehendmedical:StopSNOMEDCTInferenceJob

服務前綴	動作
compute-optimizer	compute-optimizer:DeleteRecommendationPreferences
	compute-optimizer:DescribeRecommendationExportJobs
	compute-optimizer:ExportAutoScalingGroupRecommendations
	compute-optimizer:ExportEBSVolumeRecommendations
	compute-optimizer:ExportEC2InstanceRecommendations
	compute-optimizer:ExportECSServiceRecommendations
	compute-optimizer:ExportIdleRecommendations
	compute-optimizer:ExportLambdaFunctionRecommendations
	compute-optimizer:ExportLicenseRecommendations
	compute-optimizer:ExportRDSDatabaseRecommendations
	compute-optimizer:GetEC2RecommendationProjectedMetrics
	compute-optimizer:GetECSServiceRecommendationProjectedMetrics
	compute-optimizer:GetEffectiveRecommendationPreferences
	compute-optimizer:GetEnrollmentStatus
	compute-optimizer:GetEnrollmentStatusesForOrganization
	compute-optimizer:GetRDSDatabaseRecommendationProjectedMetrics
	compute-optimizer:GetRecommendationPreferences
	compute-optimizer:GetRecommendationSummaries
	compute-optimizer:PutRecommendationPreferences

服務前綴	動作
	compute-optimizer:UpdateEnrollmentStatus

服務前綴	動作
config	config:BatchGetResourceConfig config:DeleteAggregationAuthorization config:DeleteConfigRule config:DeleteConfigurationAggregator config:DeleteConfigurationRecorder config:DeleteConformancePack config:DeleteDeliveryChannel config:DeleteEvaluationResults config:DeleteOrganizationConfigRule config:DeleteOrganizationConformancePack config:DeletePendingAggregationRequest config:DeleteRemediationConfiguration config:DeleteRemediationExceptions config:DeleteResourceConfig config:DeleteRetentionConfiguration config:DeleteStoredQuery config:DeliverConfigSnapshot config:DescribeAggregateComplianceByConfigRules config:DescribeAggregateComplianceByConformancePacks config:DescribeAggregationAuthorizations config:DescribeComplianceByConfigRule

服務前綴	動作
	config:DescribeComplianceByResource config:DescribeConfigRuleEvaluationStatus config:DescribeConfigRules config:DescribeConfigurationAggregatorSourcesStatus config:DescribeConfigurationAggregators config:DescribeConfigurationRecorderStatus config:DescribeConfigurationRecorders config:DescribeConformancePackCompliance config:DescribeConformancePackStatus config:DescribeConformancePacks config:DescribeDeliveryChannelStatus config:DescribeDeliveryChannels config:DescribeOrganizationConfigRuleStatuses config:DescribeOrganizationConfigRules config:DescribeOrganizationConformancePackStatuses config:DescribeOrganizationConformancePacks config:DescribePendingAggregationRequests config:DescribeRemediationConfigurations config:DescribeRemediationExceptions config:DescribeRemediationExecutionStatus config:DescribeRetentionConfigurations

服務前綴	動作
	config:GetComplianceDetailsByConfigRule config:GetComplianceDetailsByResource config:GetComplianceSummaryByConfigRule config:GetComplianceSummaryByResourceType config:GetConformancePackComplianceDetails config:GetConformancePackComplianceSummary config:GetCustomRulePolicy config:GetDiscoveredResourceCounts config:GetOrganizationConfigRuleDetailedStatus config:GetOrganizationConformancePackDetailedStatus config:GetOrganizationCustomRulePolicy config:GetResourceConfigHistory config:GetResourceEvaluationSummary config:GetStoredQuery config:ListConfigurationRecorders config:ListConformancePackComplianceScores config：ListConnectors config:ListDiscoveredResources config:ListResourceEvaluations config:ListStoredQueries config:PutConfigRule

服務前綴	動作
	config:PutConfigurationAggregator config:PutConfigurationRecorder config:PutConformancePack config：PutConnector config:PutDeliveryChannel config:PutEvaluations config:PutExternalEvaluation config:PutOrganizationConfigRule config:PutOrganizationConformancePack config:PutRemediationConfigurations config:PutRemediationExceptions config:PutResourceConfig config:PutRetentionConfiguration config:PutStoredQuery config:SelectResourceConfig config:StartConfigRulesEvaluation config:StartConfigurationRecorder config:StartRemediationExecution config:StartResourceEvaluation config:StopConfigurationRecorder

服務前綴	動作
connect	connect:ActivateEvaluationForm
	connect:AssociateAnalyticsDataSet
	connect:AssociateApprovedOrigin
	connect:AssociateBot
	connect：AssociateContactWithUser
	connect:AssociateDefaultVocabulary
	connect：AssociateEmailAddressAlias
	connect:AssociateFlow
	connect:AssociateInstanceStorageConfig
	connect:AssociateLambdaFunction
	connect:AssociateLexBot
	connect:AssociatePhoneNumberContactFlow
	connect：AssociateQueueEmailAddresses
	connect:AssociateQueueQuickConnects
	connect:AssociateRoutingProfileQueues
	connect:AssociateSecurityKey
	connect:AssociateUserProficiencies
	connect:BatchAssociateAnalyticsDataSet
	connect：BatchCreateDataTableValue
	connect：BatchDeleteDataTableValue
	connect：BatchDescribeDataTableValue

服務前綴	動作
	connect:BatchDisassociateAnalyticsDataSet
	connect:BatchGetFlowAssociation
	connect:BatchPutContact
	connect：BatchUpdateDataTableValue
	connect:ClaimPhoneNumber
	connect:CreateAgentStatus
	connect：CreateAttachedFile
	connect:CreateContact
	connect:CreateContactFlow
	connect:CreateContactFlowModule
	connect：CreateContactFlowModuleAlias
	connect：CreateContactFlowModuleVersion
	connect:CreateContactFlowVersion
	connect：CreateDataTable
	connect：CreateDataTableAttribute
	connect:CreateEmailAddress
	connect:CreateEvaluationForm
	connect:CreateHoursOfOperation
	connect:CreateInstance
	connect:CreateIntegrationAssociation
	connect:CreateParticipant

服務前綴	動作
	connect:CreatePersistentContactAssociation
	connect:CreatePredefinedAttribute
	connect:CreatePrompt
	connect:CreatePushNotificationRegistration
	connect:CreateQueue
	connect:CreateQuickConnect
	connect:CreateRoutingProfile
	connect:CreateRule
	connect:CreateSecurityProfile
	connect:CreateTaskTemplate
	connect:CreateTrafficDistributionGroup
	connect:CreateUseCase
	connect:CreateUser
	connect:CreateUserHierarchyGroup
	connect:CreateView
	connect:CreateViewVersion
	connect:CreateVocabulary
	connect：CreateWorkspace
	connect:DeactivateEvaluationForm
	connect:DeleteContactEvaluation
	connect:DeleteContactFlow

服務前綴	動作
	connect:DeleteContactFlowModule
	connect：DeleteContactFlowModuleAlias
	connect：DeleteContactFlowModuleVersion
	connect:DeleteContactFlowVersion
	connect：DeleteDataTable
	connect：DeleteDataTableAttribute
	connect:DeleteEmailAddress
	connect:DeleteEvaluationForm
	connect:DeleteHoursOfOperation
	connect:DeleteHoursOfOperationOverride
	connect:DeleteInstance
	connect:DeleteIntegrationAssociation
	connect：DeleteNotification
	connect:DeletePredefinedAttribute
	connect:DeletePrompt
	connect:DeletePushNotificationRegistration
	connect:DeleteQueue
	connect:DeleteQuickConnect
	connect:DeleteRoutingProfile
	connect:DeleteRule
	connect:DeleteSecurityProfile

服務前綴	動作
	connect:DeleteTaskTemplate
	connect:DeleteTrafficDistributionGroup
	connect:DeleteUseCase
	connect:DeleteUser
	connect:DeleteUserHierarchyGroup
	connect:DeleteView
	connect:DeleteVocabulary
	connect：DeleteWorkspace
	connect：DeleteWorkspaceMedia
	connect：DescribeAttachedFilesConfiguration
	connect:DescribeAuthenticationProfile
	connect：DescribeContactFlowModuleAlias
	connect：DescribeDataTableAttribute
	connect:DescribeHoursOfOperationOverride
	connect:DescribeInstanceAttribute
	connect:DescribeInstanceStorageConfig
	connect：DescribeNotification
	connect:DescribePhoneNumber
	connect:DescribeRule
	connect:DescribeTrafficDistributionGroup
	connect:DescribeUserHierarchyStructure

服務前綴	動作
	connect:DescribeVocabulary
	connect:DisassociateAnalyticsDataSet
	connect:DisassociateApprovedOrigin
	connect:DisassociateBot
	connect：DisassociateEmailAddressAlias
	connect:DisassociateFlow
	connect:DisassociateInstanceStorageConfig
	connect:DisassociateLambdaFunction
	connect:DisassociateLexBot
	connect:DisassociatePhoneNumberContactFlow
	connect：DisassociateQueueEmailAddresses
	connect:DisassociateQueueQuickConnects
	connect:DisassociateRoutingProfileQueues
	connect:DisassociateSecurityKey
	connect:DisassociateUserProficiencies
	connect:DismissUserContact
	connect：EvaluateDataTableValues
	connect:GetContactAttributes
	connect:GetContactMetrics
	connect:GetCurrentMetricData
	connect:GetCurrentUserData

服務前綴	動作
	connect:GetEffectiveHoursOfOperations
	connect：GetEvaluationFormValidation
	connect:GetFederationToken
	connect:GetFlowAssociation
	connect:GetMetricData
	connect:GetMetricDataV2
	connect:GetPromptFile
	connect:GetTaskTemplate
	connect:GetTrafficDistribution
	connect:ImportPhoneNumber
	connect：ImportWorkspaceMedia
	connect:ListAnalyticsDataAssociations
	connect:ListAnalyticsDataLakeDataSets
	connect:ListApprovedOrigins
	connect:ListAssociatedContacts
	connect：ListAttachedFilesConfigurations
	connect:ListAuthenticationProfiles
	connect:ListBots
	connect:ListContactEvaluations
	connect：ListContactFlowModuleAliases
	connect：ListContactFlowModuleVersions

服務前綴	動作
	connect:ListContactFlowModules
	connect:ListContactFlowVersions
	connect:ListContactFlows
	connect:ListContactReferences
	connect：ListDataTableAttributes
	connect：ListDataTablePrimaryValues
	connect：ListDataTableValues
	connect：ListDataTables
	connect:ListDefaultVocabularies
	connect:ListEvaluationFormVersions
	connect:ListEvaluationForms
	connect:ListFlowAssociations
	connect:ListHoursOfOperations
	connect:ListInstanceAttributes
	connect:ListInstanceStorageConfigs
	connect:ListIntegrationAssociations
	connect:ListLambdaFunctions
	connect:ListLexBots
	connect：ListNotifications
	connect:ListPhoneNumbers
	connect:ListPhoneNumbersV2

服務前綴	動作
	connect:ListPredefinedAttributes
	connect:ListPrompts
	connect：ListQueueEmailAddresses
	connect:ListQueueQuickConnects
	connect:ListQueues
	connect:ListQuickConnects
	connect:ListRealtimeContactAnalysisSegmentsV2
	connect：ListRoutingProfileManualAssignmentQueues
	connect:ListRoutingProfileQueues
	connect:ListRoutingProfiles
	connect:ListRules
	connect:ListSecurityKeys
	connect:ListSecurityProfileApplications
	connect：ListSecurityProfileFlowModules
	connect:ListSecurityProfilePermissions
	connect:ListSecurityProfiles
	connect:ListTaskTemplates
	connect:ListTrafficDistributionGroups
	connect:ListUseCases
	connect:ListUserHierarchyGroups
	connect：ListUserNotifications

服務前綴	動作
	connect:ListUsers
	connect:ListViewVersions
	connect:ListViews
	connect：ListWorkspaceMedia
	connect：ListWorkspacePages
	connect：ListWorkspaces
	connect:MonitorContact
	connect:PauseContact
	connect:PutUserStatus
	connect:ReleasePhoneNumber
	connect:ReplicateInstance
	connect:ResumeContact
	connect:ResumeContactRecording
	connect:SearchAgentStatuses
	connect:SearchAvailablePhoneNumbers
	connect：SearchContactEvaluations
	connect:SearchContactFlowModules
	connect:SearchContactFlows
	connect:SearchContacts
	connect：SearchDataTables
	connect:SearchEmailAddresses

服務前綴	動作
	connect：SearchEvaluationForms
	connect:SearchHoursOfOperations
	connect:SearchPredefinedAttributes
	connect:SearchPrompts
	connect:SearchQueues
	connect:SearchQuickConnects
	connect:SearchRoutingProfiles
	connect：SearchRules
	connect:SearchSecurityProfiles
	connect:SearchUserHierarchyGroups
	connect：SearchViews
	connect:SearchVocabularies
	connect：SearchWorkspaceAssociations
	connect：SearchWorkspaces
	connect:SendChatIntegrationEvent
	connect:SendOutboundEmail
	connect:StartChatContact
	connect：StartContactConversationalAnalyticsJob
	connect:StartContactEvaluation
	connect：StartContactMediaProcessing
	connect:StartContactRecording

服務前綴	動作
	connect:StartContactStreaming
	connect:StartEmailContact
	connect:StartOutboundChatContact
	connect:StartOutboundEmailContact
	connect:StartOutboundVoiceContact
	connect:StartScreenSharing
	connect:StartTaskContact
	connect:StartWebRTCContact
	connect:StopContact
	connect：StopContactMediaProcessing
	connect:StopContactRecording
	connect:StopContactStreaming
	connect:SubmitContactEvaluation
	connect:SuspendContactRecording
	connect:TransferContact
	connect:UpdateAgentStatus
	connect：UpdateAttachedFilesConfiguration
	connect:UpdateAuthenticationProfile
	connect:UpdateContact
	connect:UpdateContactAttributes
	connect:UpdateContactEvaluation

服務前綴	動作
	connect:UpdateContactFlowContent
	connect:UpdateContactFlowMetadata
	connect：UpdateContactFlowModuleAlias
	connect:UpdateContactFlowModuleContent
	connect:UpdateContactFlowModuleMetadata
	connect:UpdateContactFlowName
	connect:UpdateContactRoutingData
	connect:UpdateContactSchedule
	connect：UpdateDataTableAttribute
	connect：UpdateDataTableMetadata
	connect：UpdateDataTablePrimaryValues
	connect:UpdateEmailAddressMetadata
	connect:UpdateEvaluationForm
	connect:UpdateHoursOfOperation
	connect:UpdateHoursOfOperationOverride
	connect:UpdateInstanceAttribute
	connect:UpdateInstanceStorageConfig
	connect：UpdateNotificationContent
	connect:UpdateParticipantAuthentication
	connect:UpdateParticipantRoleConfig
	connect:UpdatePhoneNumber

服務前綴	動作
	connect:UpdatePhoneNumberMetadata
	connect:UpdatePredefinedAttribute
	connect:UpdatePrompt
	connect:UpdateQueueHoursOfOperation
	connect:UpdateQueueMaxContacts
	connect:UpdateQueueName
	connect:UpdateQueueOutboundCallerConfig
	connect:UpdateQueueOutboundEmailConfig
	connect:UpdateQueueStatus
	connect:UpdateQuickConnectConfig
	connect:UpdateQuickConnectName
	connect:UpdateRoutingProfileAgentAvailabilityTimer
	connect:UpdateRoutingProfileConcurrency
	connect:UpdateRoutingProfileDefaultOutboundQueue
	connect:UpdateRoutingProfileName
	connect:UpdateRoutingProfileQueues
	connect:UpdateRule
	connect:UpdateSecurityProfile
	connect:UpdateTaskTemplate
	connect:UpdateTrafficDistribution
	connect:UpdateUserHierarchy

服務前綴	動作
	connect:UpdateUserHierarchyGroupName connect:UpdateUserHierarchyStructure connect:UpdateUserIdentityInfo connect：UpdateUserNotificationStatus connect:UpdateUserPhoneConfig connect:UpdateUserProficiencies connect:UpdateUserRoutingProfile connect:UpdateUserSecurityProfiles connect:UpdateViewContent connect:UpdateViewMetadata connect：UpdateWorkspaceMetadata connect：UpdateWorkspaceTheme connect：UpdateWorkspaceVisibility
cur	cur:DeleteReportDefinition cur:DescribeReportDefinitions cur:ModifyReportDefinition cur:PutReportDefinition

服務前綴	動作
databrew	databrew:BatchDeleteRecipeVersion
	databrew:CreateDataset
	databrew:CreateProfileJob
	databrew:CreateProject
	databrew:CreateRecipe
	databrew:CreateRecipeJob
	databrew:CreateRuleset
	databrew:CreateSchedule
	databrew:DeleteDataset
	databrew:DeleteJob
	databrew:DeleteProject
	databrew:DeleteRecipeVersion
	databrew:DeleteRuleset
	databrew:DeleteSchedule
	databrew:DescribeDataset
	databrew:DescribeJob
	databrew:DescribeJobRun
	databrew:DescribeProject
	databrew:DescribeRecipe
	databrew:DescribeRuleset
	databrew:DescribeSchedule

服務前綴	動作
	databrew:ListDatasets
	databrew:ListJobRuns
	databrew:ListJobs
	databrew:ListProjects
	databrew:ListRecipeVersions
	databrew:ListRecipes
	databrew:ListRulesets
	databrew:ListSchedules
	databrew:PublishRecipe
	databrew:SendProjectSessionAction
	databrew:StartJobRun
	databrew:StartProjectSession
	databrew:StopJobRun
	databrew:UpdateDataset
	databrew:UpdateProfileJob
	databrew:UpdateProject
	databrew:UpdateRecipe
	databrew:UpdateRecipeJob
	databrew:UpdateRuleset
	databrew:UpdateSchedule

服務前綴	動作
dataexchange	dataexchange:AcceptDataGrant dataexchange:CancelJob dataexchange:CreateDataGrant dataexchange:CreateDataSet dataexchange:CreateEventAction dataexchange:CreateJob dataexchange:CreateRevision dataexchange:DeleteAsset dataexchange:DeleteDataGrant dataexchange:DeleteEventAction dataexchange:DeleteRevision dataexchange:GetDataGrant dataexchange:GetEventAction dataexchange:GetJob dataexchange:GetReceivedDataGrant dataexchange:ListDataGrants dataexchange:ListDataSetRevisions dataexchange:ListDataSets dataexchange:ListEventActions dataexchange:ListJobs dataexchange:ListReceivedDataGrants

服務前綴	動作
	dataexchange:ListRevisionAssets dataexchange:RevokeRevision dataexchange:SendDataSetNotification dataexchange:StartJob dataexchange:UpdateAsset dataexchange:UpdateDataSet dataexchange:UpdateEventAction dataexchange:UpdateRevision

服務前綴	動作
datapipeline	datapipeline:ActivatePipeline datapipeline:CreatePipeline datapipeline:DeactivatePipeline datapipeline:DeletePipeline datapipeline:DescribeObjects datapipeline:DescribePipelines datapipeline:EvaluateExpression datapipeline:GetPipelineDefinition datapipeline:ListPipelines datapipeline:PollForTask datapipeline:PutPipelineDefinition datapipeline:QueryObjects datapipeline:ReportTaskProgress datapipeline:ReportTaskRunnerHeartbeat datapipeline:SetStatus datapipeline:SetTaskStatus datapipeline:ValidatePipelineDefinition

服務前綴	動作
dax	dax:CreateCluster dax:DecreaseReplicationFactor dax:DeleteCluster dax:DeleteParameterGroup dax:DeleteSubnetGroup dax:DescribeClusters dax:DescribeDefaultParameters dax:DescribeEvents dax:DescribeParameterGroups dax:DescribeParameters dax:DescribeSubnetGroups dax:IncreaseReplicationFactor dax:RebootNode dax:UpdateCluster dax:UpdateParameterGroup dax:UpdateSubnetGroup

服務前綴	動作
devicefarm	devicefarm:CreateDevicePool
	devicefarm:CreateInstanceProfile
	devicefarm:CreateNetworkProfile
	devicefarm:CreateProject
	devicefarm:CreateRemoteAccessSession
	devicefarm:CreateTestGridProject
	devicefarm:CreateTestGridUrl
	devicefarm:CreateUpload
	devicefarm:CreateVPCEConfiguration
	devicefarm:DeleteDevicePool
	devicefarm:DeleteInstanceProfile
	devicefarm:DeleteNetworkProfile
	devicefarm:DeleteProject
	devicefarm:DeleteRemoteAccessSession
	devicefarm:DeleteRun
	devicefarm:DeleteTestGridProject
	devicefarm:DeleteUpload
	devicefarm:DeleteVPCEConfiguration
	devicefarm:GetAccountSettings
	devicefarm:GetDevice
	devicefarm:GetDeviceInstance

服務前綴	動作
	devicefarm:GetDevicePool
	devicefarm:GetDevicePoolCompatibility
	devicefarm:GetInstanceProfile
	devicefarm:GetJob
	devicefarm:GetNetworkProfile
	devicefarm:GetOfferingStatus
	devicefarm:GetProject
	devicefarm:GetRemoteAccessSession
	devicefarm:GetRun
	devicefarm:GetSuite
	devicefarm:GetTest
	devicefarm:GetTestGridProject
	devicefarm:GetTestGridSession
	devicefarm:GetUpload
	devicefarm:GetVPCEConfiguration
	devicefarm:ListArtifacts
	devicefarm:ListDeviceInstances
	devicefarm:ListDevicePools
	devicefarm:ListDevices
	devicefarm:ListInstanceProfiles
	devicefarm:ListJobs

服務前綴	動作
	devicefarm:ListNetworkProfiles
	devicefarm:ListOfferingPromotions
	devicefarm:ListOfferingTransactions
	devicefarm:ListOfferings
	devicefarm:ListProjects
	devicefarm:ListRemoteAccessSessions
	devicefarm:ListRuns
	devicefarm:ListSamples
	devicefarm:ListSuites
	devicefarm:ListTestGridProjects
	devicefarm:ListTestGridSessionActions
	devicefarm:ListTestGridSessionArtifacts
	devicefarm:ListTestGridSessions
	devicefarm:ListTests
	devicefarm:ListUniqueProblems
	devicefarm:ListUploads
	devicefarm:ListVPCEConfigurations
	devicefarm:PurchaseOffering
	devicefarm:RenewOffering
	devicefarm:ScheduleRun
	devicefarm:StopJob

服務前綴	動作
	devicefarm:StopRemoteAccessSession
	devicefarm:StopRun
	devicefarm:UpdateDeviceInstance
	devicefarm:UpdateDevicePool
	devicefarm:UpdateInstanceProfile
	devicefarm:UpdateNetworkProfile
	devicefarm:UpdateProject
	devicefarm:UpdateTestGridProject
	devicefarm:UpdateUpload
	devicefarm:UpdateVPCEConfiguration

服務前綴	動作
devops-guru	devops-guru:AddNotificationChannel devops-guru:DeleteInsight devops-guru:DescribeAccountHealth devops-guru:DescribeAccountOverview devops-guru:DescribeAnomaly devops-guru:DescribeEventSourcesConfig devops-guru:DescribeFeedback devops-guru:DescribeInsight devops-guru:DescribeOrganizationHealth devops-guru:DescribeOrganizationOverview devops-guru:DescribeOrganizationResourceCollectionHealth devops-guru:DescribeResourceCollectionHealth devops-guru:DescribeServiceIntegration devops-guru:GetCostEstimation devops-guru:GetResourceCollection devops-guru:ListAnomaliesForInsight devops-guru:ListAnomalousLogGroups devops-guru:ListEvents devops-guru:ListInsights devops-guru:ListMonitoredResources devops-guru:ListNotificationChannels

服務前綴	動作
	devops-guru:ListOrganizationInsights devops-guru:ListRecommendations devops-guru:PutFeedback devops-guru:RemoveNotificationChannel devops-guru:SearchInsights devops-guru:SearchOrganizationInsights devops-guru:StartCostEstimation devops-guru:UpdateEventSourcesConfig devops-guru:UpdateResourceCollection devops-guru:UpdateServiceIntegration

服務前綴	動作
directconnect	directconnect:AcceptDirectConnectGatewayAssociationProposal
	directconnect:AllocateConnectionOnInterconnect
	directconnect:AllocateHostedConnection
	directconnect:AllocatePrivateVirtualInterface
	directconnect:AllocatePublicVirtualInterface
	directconnect:AllocateTransitVirtualInterface
	directconnect:AssociateConnectionWithLag
	directconnect:AssociateHostedConnection
	directconnect:AssociateMacSecKey
	directconnect:AssociateVirtualInterface
	directconnect:ConfirmConnection
	directconnect:ConfirmCustomerAgreement
	directconnect:ConfirmPrivateVirtualInterface
	directconnect:ConfirmPublicVirtualInterface
	directconnect:ConfirmTransitVirtualInterface
	directconnect:CreateBGPPeer
	directconnect:CreateConnection
	directconnect:CreateDirectConnectGateway
	directconnect:CreateDirectConnectGatewayAssociation
	directconnect:CreateDirectConnectGatewayAssociationProposal
	directconnect:CreateInterconnect

服務前綴	動作
	directconnect:CreateLag
	directconnect:CreatePrivateVirtualInterface
	directconnect:CreatePublicVirtualInterface
	directconnect:CreateTransitVirtualInterface
	directconnect:DeleteBGPPeer
	directconnect:DeleteConnection
	directconnect:DeleteDirectConnectGateway
	directconnect:DeleteDirectConnectGatewayAssociation
	directconnect:DeleteDirectConnectGatewayAssociationProposal
	directconnect:DeleteInterconnect
	directconnect:DeleteLag
	directconnect:DeleteVirtualInterface
	directconnect:DescribeConnectionLoa
	directconnect:DescribeConnections
	directconnect:DescribeConnectionsOnInterconnect
	directconnect:DescribeCustomerMetadata
	directconnect:DescribeDirectConnectGatewayAssociationProposals
	directconnect:DescribeDirectConnectGatewayAssociations
	directconnect:DescribeDirectConnectGatewayAttachments
	directconnect:DescribeDirectConnectGateways
	directconnect:DescribeHostedConnections

服務前綴	動作
	directconnect:DescribeInterconnectLoa
	directconnect:DescribeInterconnects
	directconnect:DescribeLags
	directconnect:DescribeLoa
	directconnect:DescribeLocations
	directconnect:DescribeRouterConfiguration
	directconnect:DescribeVirtualGateways
	directconnect:DescribeVirtualInterfaces
	directconnect:DisassociateConnectionFromLag
	directconnect:DisassociateMacSecKey
	directconnect:ListVirtualInterfaceTestHistory
	directconnect:StartBgpFailoverTest
	directconnect:StopBgpFailoverTest
	directconnect:UpdateConnection
	directconnect:UpdateDirectConnectGateway
	directconnect:UpdateDirectConnectGatewayAssociation
	directconnect:UpdateLag
	directconnect:UpdateVirtualInterfaceAttributes

服務前綴	動作
dlm	dlm:CreateLifecyclePolicy dlm:DeleteLifecyclePolicy dlm:GetLifecyclePolicies dlm:GetLifecyclePolicy dlm:UpdateLifecyclePolicy

服務前綴	動作
dms	dms:ApplyPendingMaintenanceAction
	dms:AssociateExtensionPack
	dms:BatchStartRecommendations
	dms：CancelMetadataModelCreation
	dms:CancelReplicationTaskAssessmentRun
	dms:CreateDataProvider
	dms:CreateEndpoint
	dms:CreateEventSubscription
	dms:CreateInstanceProfile
	dms:CreateMigrationProject
	dms:CreateReplicationConfig
	dms:CreateReplicationInstance
	dms:CreateReplicationSubnetGroup
	dms:CreateReplicationTask
	dms:DeleteCertificate
	dms:DeleteConnection
	dms:DeleteDataMigration
	dms:DeleteDataProvider
	dms:DeleteEndpoint
	dms:DeleteEventSubscription
	dms:DeleteFleetAdvisorCollector

服務前綴	動作
	dms:DeleteFleetAdvisorDatabases
	dms:DeleteInstanceProfile
	dms:DeleteMigrationProject
	dms:DeleteReplicationConfig
	dms:DeleteReplicationInstance
	dms:DeleteReplicationSubnetGroup
	dms:DeleteReplicationTask
	dms:DeleteReplicationTaskAssessmentRun
	dms:DescribeAccountAttributes
	dms:DescribeApplicableIndividualAssessments
	dms:DescribeCertificates
	dms:DescribeConnections
	dms:DescribeDataMigrations
	dms:DescribeEndpointSettings
	dms:DescribeEndpointTypes
	dms:DescribeEndpoints
	dms:DescribeEngineVersions
	dms:DescribeEventCategories
	dms:DescribeEventSubscriptions
	dms:DescribeEvents
	dms:DescribeFleetAdvisorCollectors

服務前綴	動作
	dms:DescribeFleetAdvisorDatabases
	dms:DescribeFleetAdvisorLsaAnalysis
	dms:DescribeFleetAdvisorSchemaObjectSummary
	dms:DescribeFleetAdvisorSchemas
	dms：DescribeMetadataModel
	dms：DescribeMetadataModelChildren
	dms：DescribeMetadataModelCreations
	dms:DescribeMetadataModelImports
	dms:DescribeOrderableReplicationInstances
	dms:DescribePendingMaintenanceActions
	dms:DescribeRecommendationLimitations
	dms:DescribeRecommendations
	dms:DescribeRefreshSchemasStatus
	dms:DescribeReplicationConfigs
	dms:DescribeReplicationInstanceTaskLogs
	dms:DescribeReplicationInstances
	dms:DescribeReplicationSubnetGroups
	dms:DescribeReplicationTableStatistics
	dms:DescribeReplicationTaskAssessmentResults
	dms:DescribeReplicationTaskAssessmentRuns
	dms:DescribeReplicationTaskIndividualAssessments

服務前綴	動作
	dms:DescribeReplicationTasks
	dms:DescribeReplications
	dms:DescribeSchemas
	dms:DescribeTableStatistics
	dms:ExportMetadataModelAssessment
	dms:ImportCertificate
	dms:ListDataProviders
	dms:ListExtensionPacks
	dms:ListInstanceProfiles
	dms:ListMetadataModelAssessments
	dms:ListMetadataModelConversions
	dms:ListMetadataModelExports
	dms:ListMigrationProjects
	dms:ModifyDataMigration
	dms:ModifyEndpoint
	dms:ModifyEventSubscription
	dms:ModifyReplicationConfig
	dms:ModifyReplicationInstance
	dms:ModifyReplicationSubnetGroup
	dms:ModifyReplicationTask
	dms:MoveReplicationTask

服務前綴	動作
	dms:RebootReplicationInstance
	dms:RefreshSchemas
	dms:ReloadReplicationTables
	dms:ReloadTables
	dms:RunFleetAdvisorLsaAnalysis
	dms:StartMetadataModelAssessment
	dms:StartMetadataModelConversion
	dms：StartMetadataModelCreation
	dms:StartMetadataModelExportAsScripts
	dms:StartMetadataModelExportToTarget
	dms:StartRecommendations
	dms:StartReplication
	dms:StartReplicationTask
	dms:StartReplicationTaskAssessment
	dms:StopDataMigration
	dms:StopReplicationTask
	dms:TestConnection
	dms:UpdateConversionConfiguration
	dms:UpdateDataProvider
	dms:UpdateInstanceProfile
	dms:UpdateMigrationProject

服務前綴	動作
	dms:UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic:ApplyPendingMaintenanceAction docdb-elastic:CopyClusterSnapshot docdb-elastic:DeleteCluster docdb-elastic:DeleteClusterSnapshot docdb-elastic:GetCluster docdb-elastic:GetClusterSnapshot docdb-elastic:GetPendingMaintenanceAction docdb-elastic:ListClusterSnapshots docdb-elastic:ListClusters docdb-elastic:ListPendingMaintenanceActions docdb-elastic:RestoreClusterFromSnapshot docdb-elastic:StartCluster docdb-elastic:StopCluster docdb-elastic:UpdateCluster

服務前綴	動作
dynamodb	dynamodb:CreateBackup
	dynamodb:CreateGlobalTable
	dynamodb:CreateTable
	dynamodb:DeleteBackup
	dynamodb:DeleteTable
	dynamodb:DescribeBackup
	dynamodb:DescribeContinuousBackups
	dynamodb:DescribeContributorInsights
	dynamodb:DescribeEndpoints
	dynamodb:DescribeExport
	dynamodb:DescribeGlobalTable
	dynamodb:DescribeGlobalTableSettings
	dynamodb:DescribeImport
	dynamodb:DescribeKinesisStreamingDestination
	dynamodb:DescribeLimits
	dynamodb:DescribeStream
	dynamodb:DescribeTable
	dynamodb:DescribeTableReplicaAutoScaling
	dynamodb:DescribeTimeToLive
	dynamodb:DisableKinesisStreamingDestination
	dynamodb:EnableKinesisStreamingDestination

服務前綴	動作
	dynamodb:ExportTableToPointInTime
	dynamodb:GetResourcePolicy
	dynamodb:ImportTable
	dynamodb:ListBackups
	dynamodb:ListContributorInsights
	dynamodb:ListExports
	dynamodb:ListGlobalTables
	dynamodb:ListImports
	dynamodb:ListStreams
	dynamodb:ListTables
	dynamodb:RestoreTableFromBackup
	dynamodb:RestoreTableToPointInTime
	dynamodb:UpdateContinuousBackups
	dynamodb:UpdateContributorInsights
	dynamodb:UpdateGlobalTable
	dynamodb:UpdateGlobalTableSettings
	dynamodb:UpdateKinesisStreamingDestination
	dynamodb:UpdateTable
	dynamodb:UpdateTableReplicaAutoScaling
	dynamodb:UpdateTimeToLive

服務前綴	動作
ebs	ebs:CompleteSnapshot ebs:StartSnapshot

服務前綴	動作
ec2	ec2:AcceptAddressTransfer
	ec2:AcceptCapacityReservationBillingOwnership
	ec2:AcceptReservedInstancesExchangeQuote
	ec2：AcceptTransitGatewayClientVpnAttachment
	ec2:AcceptTransitGatewayMulticastDomainAssociations
	ec2:AcceptTransitGatewayPeeringAttachment
	ec2:AcceptTransitGatewayVpcAttachment
	ec2:AcceptVpcEndpointConnections
	ec2:AcceptVpcPeeringConnection
	ec2:AdvertiseByoipCidr
	ec2:AllocateAddress
	ec2:AllocateHosts
	ec2:AllocateIpamPoolCidr
	ec2:ApplySecurityGroupsToClientVpnTargetNetwork
	ec2:AssignIpv6Addresses
	ec2:AssignPrivateIpAddresses
	ec2:AssignPrivateNatGatewayAddress
	ec2:AssociateAddress
	ec2:AssociateCapacityReservationBillingOwner
	ec2:AssociateClientVpnTargetNetwork
	ec2:AssociateDhcpOptions

服務前綴	動作
	ec2:AssociateEnclaveCertificateIamRole
	ec2:AssociateIamInstanceProfile
	ec2:AssociateInstanceEventWindow
	ec2:AssociateIpamByoasn
	ec2:AssociateIpamResourceDiscovery
	ec2:AssociateNatGatewayAddress
	ec2:AssociateRouteServer
	ec2:AssociateRouteTable
	ec2:AssociateSecurityGroupVpc
	ec2:AssociateSubnetCidrBlock
	ec2:AssociateTransitGatewayMulticastDomain
	ec2:AssociateTransitGatewayPolicyTable
	ec2:AssociateTransitGatewayRouteTable
	ec2:AssociateTrunkInterface
	ec2:AssociateVpcCidrBlock
	ec2:AttachClassicLinkVpc
	ec2：AttachImageWatermark
	ec2:AttachInternetGateway
	ec2:AttachNetworkInterface
	ec2:AttachVerifiedAccessTrustProvider
	ec2:AttachVolume

服務前綴	動作
	ec2:AttachVpnGateway
	ec2:AuthorizeClientVpnIngress
	ec2:AuthorizeSecurityGroupEgress
	ec2:AuthorizeSecurityGroupIngress
	ec2:BundleInstance
	ec2:CancelBundleTask
	ec2:CancelCapacityReservation
	ec2:CancelCapacityReservationFleets
	ec2:CancelConversionTask
	ec2:CancelDeclarativePoliciesReport
	ec2:CancelExportTask
	ec2:CancelImageLaunchPermission
	ec2:CancelImportTask
	ec2:CancelReservedInstancesListing
	ec2:CancelSpotFleetRequests
	ec2:CancelSpotInstanceRequests
	ec2:ConfirmProductInstance
	ec2:CopyFpgaImage
	ec2:CopyImage
	ec2:CopySnapshot
	ec2：CopyVolumes

服務前綴	動作
	ec2：CreateCapacityManagerDataExport
	ec2:CreateCapacityReservation
	ec2:CreateCapacityReservationBySplitting
	ec2：CreateCapacityReservationCancellationQuote
	ec2:CreateCapacityReservationFleet
	ec2:CreateCarrierGateway
	ec2:CreateClientVpnEndpoint
	ec2:CreateClientVpnRoute
	ec2:CreateCoipCidr
	ec2:CreateCoipPool
	ec2:CreateCustomerGateway
	ec2:CreateDefaultSubnet
	ec2:CreateDefaultVpc
	ec2:CreateDelegateMacVolumeOwnershipTask
	ec2:CreateDhcpOptions
	ec2:CreateEgressOnlyInternetGateway
	ec2:CreateFleet
	ec2:CreateFlowLogs
	ec2:CreateFpgaImage
	ec2:CreateImage
	ec2：CreateImageUsageReport

服務前綴	動作
	ec2:CreateInstanceConnectEndpoint
	ec2:CreateInstanceEventWindow
	ec2:CreateInstanceExportTask
	ec2：CreateInternetGateway
	ec2：CreateInterruptibleCapacityReservationAllocation
	ec2:CreateIpam
	ec2:CreateIpamExternalResourceVerificationToken
	ec2：CreateIpamPolicy
	ec2:CreateIpamPool
	ec2：CreateIpamPrefixListResolver
	ec2：CreateIpamPrefixListResolverTarget
	ec2:CreateIpamResourceDiscovery
	ec2:CreateIpamScope
	ec2:CreateKeyPair
	ec2:CreateLaunchTemplateVersion
	ec2:CreateLocalGatewayRoute
	ec2:CreateLocalGatewayRouteTable
	ec2:CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:CreateLocalGatewayRouteTableVpcAssociation
	ec2:CreateLocalGatewayVirtualInterface

服務前綴	動作
	ec2:CreateLocalGatewayVirtualInterfaceGroup
	ec2:CreateMacSystemIntegrityProtectionModificationTask
	ec2:CreateManagedPrefixList
	ec2:CreateNatGateway
	ec2:CreateNetworkAcl
	ec2:CreateNetworkAclEntry
	ec2:CreateNetworkInsightsAccessScope
	ec2:CreateNetworkInsightsPath
	ec2:CreateNetworkInterface
	ec2:CreateNetworkInterfacePermission
	ec2:CreatePlacementGroup
	ec2:CreatePublicIpv4Pool
	ec2:CreateReplaceRootVolumeTask
	ec2:CreateReservedInstancesListing
	ec2:CreateRestoreImageTask
	ec2：CreateRoute
	ec2:CreateRouteServer
	ec2:CreateRouteServerEndpoint
	ec2:CreateRouteServerPeer
	ec2：CreateRouteTable
	ec2：CreateSecondaryNetwork

服務前綴	動作
	ec2：CreateSecondarySubnet
	ec2：CreateSecurityGroup
	ec2:CreateSnapshots
	ec2:CreateSpotDatafeedSubscription
	ec2:CreateStoreImageTask
	ec2：CreateSubnet
	ec2:CreateSubnetCidrReservation
	ec2:CreateTrafficMirrorFilter
	ec2:CreateTrafficMirrorFilterRule
	ec2:CreateTrafficMirrorSession
	ec2:CreateTrafficMirrorTarget
	ec2:CreateTransitGateway
	ec2:CreateTransitGatewayConnect
	ec2:CreateTransitGatewayConnectPeer
	ec2：CreateTransitGatewayMeteringPolicy
	ec2：CreateTransitGatewayMeteringPolicyEntry
	ec2:CreateTransitGatewayMulticastDomain
	ec2:CreateTransitGatewayPeeringAttachment
	ec2:CreateTransitGatewayPolicyTable
	ec2:CreateTransitGatewayPrefixListReference
	ec2:CreateTransitGatewayRoute

服務前綴	動作
	ec2:CreateTransitGatewayRouteTable
	ec2:CreateTransitGatewayRouteTableAnnouncement
	ec2:CreateTransitGatewayVpcAttachment
	ec2:CreateVerifiedAccessEndpoint
	ec2:CreateVerifiedAccessGroup
	ec2:CreateVerifiedAccessInstance
	ec2:CreateVerifiedAccessTrustProvider
	ec2:CreateVolume
	ec2：CreateVpc
	ec2:CreateVpcBlockPublicAccessExclusion
	ec2：CreateVpcEncryptionControl
	ec2:CreateVpcEndpoint
	ec2:CreateVpcEndpointConnectionNotification
	ec2:CreateVpcEndpointServiceConfiguration
	ec2:CreateVpcPeeringConnection
	ec2：CreateVpnConcentrator
	ec2:CreateVpnConnection
	ec2:CreateVpnConnectionRoute
	ec2:CreateVpnGateway
	ec2：DeleteCapacityManagerDataExport
	ec2:DeleteCarrierGateway

服務前綴	動作
	ec2:DeleteClientVpnEndpoint
	ec2:DeleteClientVpnRoute
	ec2:DeleteCoipCidr
	ec2:DeleteCoipPool
	ec2:DeleteCustomerGateway
	ec2:DeleteDhcpOptions
	ec2:DeleteEgressOnlyInternetGateway
	ec2:DeleteFleets
	ec2:DeleteFlowLogs
	ec2:DeleteFpgaImage
	ec2：DeleteImageUsageReport
	ec2:DeleteInstanceConnectEndpoint
	ec2:DeleteInstanceEventWindow
	ec2:DeleteInternetGateway
	ec2:DeleteIpam
	ec2:DeleteIpamExternalResourceVerificationToken
	ec2：DeleteIpamPolicy
	ec2:DeleteIpamPool
	ec2：DeleteIpamPrefixListResolver
	ec2：DeleteIpamPrefixListResolverTarget
	ec2:DeleteIpamResourceDiscovery

服務前綴	動作
	ec2:DeleteIpamScope
	ec2:DeleteKeyPair
	ec2:DeleteLaunchTemplate
	ec2:DeleteLaunchTemplateVersions
	ec2:DeleteLocalGatewayRoute
	ec2:DeleteLocalGatewayRouteTable
	ec2:DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:DeleteLocalGatewayRouteTableVpcAssociation
	ec2:DeleteLocalGatewayVirtualInterface
	ec2:DeleteLocalGatewayVirtualInterfaceGroup
	ec2:DeleteManagedPrefixList
	ec2:DeleteNatGateway
	ec2:DeleteNetworkAcl
	ec2:DeleteNetworkAclEntry
	ec2:DeleteNetworkInsightsAccessScope
	ec2:DeleteNetworkInsightsAccessScopeAnalysis
	ec2:DeleteNetworkInsightsAnalysis
	ec2:DeleteNetworkInsightsPath
	ec2:DeleteNetworkInterface
	ec2:DeleteNetworkInterfacePermission

服務前綴	動作
	ec2:DeletePlacementGroup
	ec2:DeletePublicIpv4Pool
	ec2:DeleteQueuedReservedInstances
	ec2:DeleteRoute
	ec2:DeleteRouteServer
	ec2:DeleteRouteServerEndpoint
	ec2:DeleteRouteServerPeer
	ec2:DeleteRouteTable
	ec2：DeleteSecondaryNetwork
	ec2：DeleteSecondarySubnet
	ec2:DeleteSecurityGroup
	ec2:DeleteSpotDatafeedSubscription
	ec2:DeleteSubnet
	ec2:DeleteSubnetCidrReservation
	ec2:DeleteTrafficMirrorFilter
	ec2:DeleteTrafficMirrorFilterRule
	ec2:DeleteTrafficMirrorSession
	ec2:DeleteTrafficMirrorTarget
	ec2:DeleteTransitGateway
	ec2：DeleteTransitGatewayClientVpnAttachment
	ec2:DeleteTransitGatewayConnect

服務前綴	動作
	ec2:DeleteTransitGatewayConnectPeer
	ec2：DeleteTransitGatewayMeteringPolicy
	ec2：DeleteTransitGatewayMeteringPolicyEntry
	ec2:DeleteTransitGatewayMulticastDomain
	ec2:DeleteTransitGatewayPeeringAttachment
	ec2:DeleteTransitGatewayPolicyTable
	ec2:DeleteTransitGatewayPrefixListReference
	ec2:DeleteTransitGatewayRoute
	ec2:DeleteTransitGatewayRouteTable
	ec2:DeleteTransitGatewayRouteTableAnnouncement
	ec2:DeleteTransitGatewayVpcAttachment
	ec2:DeleteVerifiedAccessEndpoint
	ec2:DeleteVerifiedAccessGroup
	ec2:DeleteVerifiedAccessInstance
	ec2:DeleteVerifiedAccessTrustProvider
	ec2:DeleteVolume
	ec2:DeleteVpc
	ec2:DeleteVpcBlockPublicAccessExclusion
	ec2：DeleteVpcEncryptionControl
	ec2:DeleteVpcEndpointConnectionNotifications
	ec2:DeleteVpcEndpointServiceConfigurations

服務前綴	動作
	ec2:DeleteVpcEndpoints
	ec2:DeleteVpcPeeringConnection
	ec2：DeleteVpnConcentrator
	ec2:DeleteVpnConnection
	ec2:DeleteVpnConnectionRoute
	ec2:DeleteVpnGateway
	ec2:DeprovisionByoipCidr
	ec2:DeprovisionIpamByoasn
	ec2:DeprovisionIpamPoolCidr
	ec2:DeprovisionPublicIpv4PoolCidr
	ec2:DeregisterImage
	ec2:DeregisterInstanceEventNotificationAttributes
	ec2:DeregisterTransitGatewayMulticastGroupMembers
	ec2:DeregisterTransitGatewayMulticastGroupSources
	ec2:DescribeAccountAttributes
	ec2:DescribeAddressTransfers
	ec2:DescribeAddresses
	ec2:DescribeAddressesAttribute
	ec2:DescribeAggregateIdFormat
	ec2：DescribeAvailabilityZones
	ec2:DescribeAwsNetworkPerformanceMetricSubscriptions

服務前綴	動作
	ec2:DescribeBundleTasks
	ec2:DescribeByoipCidrs
	ec2:DescribeCapacityBlockExtensionHistory
	ec2:DescribeCapacityBlockExtensionOfferings
	ec2:DescribeCapacityBlockStatus
	ec2:DescribeCapacityBlocks
	ec2：DescribeCapacityManagerDataExports
	ec2:DescribeCapacityReservationBillingRequests
	ec2：DescribeCapacityReservationCancellationQuotes
	ec2:DescribeCapacityReservationFleets
	ec2：DescribeCapacityReservationTopology
	ec2:DescribeCapacityReservations
	ec2:DescribeCarrierGateways
	ec2:DescribeClassicLinkInstances
	ec2:DescribeClientVpnAuthorizationRules
	ec2:DescribeClientVpnConnections
	ec2:DescribeClientVpnEndpoints
	ec2:DescribeClientVpnRoutes
	ec2:DescribeClientVpnTargetNetworks
	ec2:DescribeCoipPools
	ec2:DescribeConversionTasks

服務前綴	動作
	ec2:DescribeCustomerGateways
	ec2:DescribeDeclarativePoliciesReports
	ec2:DescribeDhcpOptions
	ec2:DescribeEgressOnlyInternetGateways
	ec2:DescribeElasticGpus
	ec2:DescribeExportImageTasks
	ec2:DescribeExportTasks
	ec2:DescribeFastLaunchImages
	ec2:DescribeFastSnapshotRestores
	ec2:DescribeFleetHistory
	ec2:DescribeFleetInstances
	ec2:DescribeFleets
	ec2:DescribeFlowLogs
	ec2:DescribeFpgaImageAttribute
	ec2:DescribeFpgaImages
	ec2:DescribeHostReservationOfferings
	ec2:DescribeHostReservations
	ec2:DescribeHosts
	ec2:DescribeIamInstanceProfileAssociations
	ec2:DescribeIdFormat
	ec2:DescribeIdentityIdFormat

服務前綴	動作
	ec2:DescribeImageAttribute
	ec2：DescribeImageReferences
	ec2：DescribeImageUsageReportEntries
	ec2：DescribeImageUsageReports
	ec2:DescribeImportImageTasks
	ec2:DescribeImportSnapshotTasks
	ec2:DescribeInstanceConnectEndpoints
	ec2:DescribeInstanceCreditSpecifications
	ec2:DescribeInstanceEventNotificationAttributes
	ec2:DescribeInstanceEventWindows
	ec2:DescribeInstanceImageMetadata
	ec2：DescribeInstanceSqlHaHistoryStates
	ec2：DescribeInstanceSqlHaStates
	ec2:DescribeInstanceTopology
	ec2:DescribeInstanceTypes
	ec2：DescribeInternetGateways
	ec2:DescribeIpamByoasn
	ec2:DescribeIpamExternalResourceVerificationTokens
	ec2：DescribeIpamPolicies
	ec2：DescribeIpamPoolAllocations
	ec2:DescribeIpamPools

服務前綴	動作
	ec2：DescribeIpamPrefixListResolverTargets
	ec2：DescribeIpamPrefixListResolvers
	ec2:DescribeIpamResourceDiscoveries
	ec2:DescribeIpamResourceDiscoveryAssociations
	ec2:DescribeIpamScopes
	ec2:DescribeIpams
	ec2:DescribeIpv6Pools
	ec2：DescribeKeyPairs
	ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	ec2:DescribeLocalGatewayRouteTableVpcAssociations
	ec2:DescribeLocalGatewayRouteTables
	ec2:DescribeLocalGatewayVirtualInterfaceGroups
	ec2:DescribeLocalGatewayVirtualInterfaces
	ec2:DescribeLocalGateways
	ec2:DescribeLockedSnapshots
	ec2:DescribeMacHosts
	ec2:DescribeMacModificationTasks
	ec2:DescribeManagedPrefixLists
	ec2:DescribeMovingAddresses
	ec2:DescribeNatGateways

服務前綴	動作
	ec2:DescribeNetworkAcls
	ec2:DescribeNetworkInsightsAccessScopeAnalyses
	ec2:DescribeNetworkInsightsAccessScopes
	ec2:DescribeNetworkInsightsAnalyses
	ec2:DescribeNetworkInsightsPaths
	ec2:DescribeNetworkInterfaceAttribute
	ec2:DescribeNetworkInterfacePermissions
	ec2:DescribeNetworkInterfaces
	ec2:DescribeOutpostLags
	ec2:DescribePlacementGroups
	ec2:DescribePrefixLists
	ec2:DescribePrincipalIdFormat
	ec2:DescribePublicIpv4Pools
	ec2:DescribeRegions
	ec2:DescribeReplaceRootVolumeTasks
	ec2:DescribeReservedInstances
	ec2:DescribeReservedInstancesListings
	ec2:DescribeReservedInstancesModifications
	ec2:DescribeReservedInstancesOfferings
	ec2:DescribeRouteServerEndpoints
	ec2:DescribeRouteServerPeers

服務前綴	動作
	ec2:DescribeRouteServers
	ec2：DescribeRouteTables
	ec2:DescribeScheduledInstanceAvailability
	ec2:DescribeScheduledInstances
	ec2：DescribeSecondaryInterfaces
	ec2：DescribeSecondaryNetworks
	ec2：DescribeSecondarySubnets
	ec2:DescribeSecurityGroupReferences
	ec2:DescribeSecurityGroupRules
	ec2:DescribeSecurityGroupVpcAssociations
	ec2：DescribeSecurityGroups
	ec2:DescribeServiceLinkVirtualInterfaces
	ec2:DescribeSnapshotAttribute
	ec2:DescribeSnapshotTierStatus
	ec2:DescribeSpotDatafeedSubscription
	ec2:DescribeSpotFleetInstances
	ec2:DescribeSpotFleetRequestHistory
	ec2:DescribeSpotFleetRequests
	ec2:DescribeSpotInstanceRequests
	ec2:DescribeSpotPriceHistory
	ec2:DescribeStaleSecurityGroups

服務前綴	動作
	ec2:DescribeStoreImageTasks
	ec2:DescribeTrafficMirrorFilterRules
	ec2:DescribeTrafficMirrorFilters
	ec2:DescribeTrafficMirrorSessions
	ec2:DescribeTrafficMirrorTargets
	ec2:DescribeTransitGatewayAttachments
	ec2:DescribeTransitGatewayConnectPeers
	ec2:DescribeTransitGatewayConnects
	ec2：DescribeTransitGatewayMeteringPolicies
	ec2:DescribeTransitGatewayMulticastDomains
	ec2:DescribeTransitGatewayPeeringAttachments
	ec2:DescribeTransitGatewayPolicyTables
	ec2:DescribeTransitGatewayRouteTableAnnouncements
	ec2:DescribeTransitGatewayRouteTables
	ec2:DescribeTransitGatewayVpcAttachments
	ec2:DescribeTransitGateways
	ec2:DescribeTrunkInterfaceAssociations
	ec2:DescribeVerifiedAccessEndpoints
	ec2:DescribeVerifiedAccessGroups
	ec2:DescribeVerifiedAccessInstanceLoggingConfigurations
	ec2:DescribeVerifiedAccessInstances

服務前綴	動作
	ec2:DescribeVerifiedAccessTrustProviders
	ec2:DescribeVolumeAttribute
	ec2:DescribeVolumeStatus
	ec2:DescribeVolumes
	ec2:DescribeVolumesModifications
	ec2:DescribeVpcAttribute
	ec2:DescribeVpcBlockPublicAccessExclusions
	ec2:DescribeVpcBlockPublicAccessOptions
	ec2:DescribeVpcClassicLink
	ec2:DescribeVpcClassicLinkDnsSupport
	ec2：DescribeVpcEncryptionControls
	ec2:DescribeVpcEndpointAssociations
	ec2:DescribeVpcEndpointConnectionNotifications
	ec2:DescribeVpcEndpointConnections
	ec2:DescribeVpcEndpointServiceConfigurations
	ec2:DescribeVpcEndpointServicePermissions
	ec2:DescribeVpcEndpointServices
	ec2:DescribeVpcEndpoints
	ec2:DescribeVpcPeeringConnections
	ec2：DescribeVpcs
	ec2：DescribeVpnConcentrators

服務前綴	動作
	ec2:DescribeVpnConnections
	ec2:DescribeVpnGateways
	ec2:DetachClassicLinkVpc
	ec2：DetachImageWatermark
	ec2:DetachInternetGateway
	ec2:DetachNetworkInterface
	ec2:DetachVerifiedAccessTrustProvider
	ec2:DetachVolume
	ec2:DetachVpnGateway
	ec2:DisableAddressTransfer
	ec2:DisableAllowedImagesSettings
	ec2:DisableAwsNetworkPerformanceMetricSubscription
	ec2：DisableCapacityManager
	ec2:DisableEbsEncryptionByDefault
	ec2:DisableFastLaunch
	ec2:DisableFastSnapshotRestores
	ec2:DisableImage
	ec2:DisableImageBlockPublicAccess
	ec2:DisableImageDeprecation
	ec2:DisableImageDeregistrationProtection
	ec2：DisableInstanceSqlHaStandbyDetections

服務前綴	動作
	ec2:DisableIpamOrganizationAdminAccount
	ec2：DisableIpamPolicy
	ec2:DisableRouteServerPropagation
	ec2:DisableSerialConsoleAccess
	ec2:DisableSnapshotBlockPublicAccess
	ec2:DisableTransitGatewayRouteTablePropagation
	ec2:DisableVgwRoutePropagation
	ec2:DisableVpcClassicLink
	ec2:DisableVpcClassicLinkDnsSupport
	ec2:DisassociateAddress
	ec2:DisassociateCapacityReservationBillingOwner
	ec2:DisassociateClientVpnTargetNetwork
	ec2:DisassociateEnclaveCertificateIamRole
	ec2:DisassociateIamInstanceProfile
	ec2:DisassociateInstanceEventWindow
	ec2:DisassociateIpamByoasn
	ec2:DisassociateIpamResourceDiscovery
	ec2:DisassociateNatGatewayAddress
	ec2:DisassociateRouteServer
	ec2:DisassociateRouteTable
	ec2:DisassociateSecurityGroupVpc

服務前綴	動作
	ec2:DisassociateSubnetCidrBlock
	ec2:DisassociateTransitGatewayMulticastDomain
	ec2:DisassociateTransitGatewayPolicyTable
	ec2:DisassociateTransitGatewayRouteTable
	ec2:DisassociateTrunkInterface
	ec2:DisassociateVpcCidrBlock
	ec2:EnableAddressTransfer
	ec2:EnableAllowedImagesSettings
	ec2:EnableAwsNetworkPerformanceMetricSubscription
	ec2：EnableCapacityManager
	ec2:EnableEbsEncryptionByDefault
	ec2:EnableFastLaunch
	ec2:EnableFastSnapshotRestores
	ec2:EnableImage
	ec2:EnableImageBlockPublicAccess
	ec2:EnableImageDeprecation
	ec2:EnableImageDeregistrationProtection
	ec2：EnableInstanceSqlHaStandbyDetections
	ec2:EnableIpamOrganizationAdminAccount
	ec2：EnableIpamPolicy
	ec2:EnableReachabilityAnalyzerOrganizationSharing

服務前綴	動作
	ec2:EnableRouteServerPropagation
	ec2:EnableSerialConsoleAccess
	ec2:EnableSnapshotBlockPublicAccess
	ec2:EnableTransitGatewayRouteTablePropagation
	ec2:EnableVgwRoutePropagation
	ec2:EnableVolumeIO
	ec2:EnableVpcClassicLink
	ec2:EnableVpcClassicLinkDnsSupport
	ec2:ExportClientVpnClientCertificateRevocationList
	ec2:ExportClientVpnClientConfiguration
	ec2:ExportImage
	ec2:ExportTransitGatewayRoutes
	ec2:ExportVerifiedAccessInstanceClientConfiguration
	ec2:GetActiveVpnTunnelStatus
	ec2:GetAllowedImagesSettings
	ec2:GetAssociatedEnclaveCertificateIamRoles
	ec2:GetAssociatedIpv6PoolCidrs
	ec2:GetAwsNetworkPerformanceData
	ec2：GetCapacityManagerAttributes
	ec2：GetCapacityManagerMetricData
	ec2：GetCapacityManagerMetricDimensions

服務前綴	動作
	ec2:GetCapacityReservationUsage
	ec2:GetCoipPoolUsage
	ec2:GetConsoleOutput
	ec2:GetConsoleScreenshot
	ec2:GetDeclarativePoliciesReportSummary
	ec2:GetDefaultCreditSpecification
	ec2:GetEbsDefaultKmsKeyId
	ec2:GetEbsEncryptionByDefault
	ec2：GetEnabledIpamPolicy
	ec2:GetFlowLogsIntegrationTemplate
	ec2:GetGroupsForCapacityReservation
	ec2:GetHostReservationPurchasePreview
	ec2：GetImageAncestry
	ec2:GetImageBlockPublicAccessState
	ec2:GetInstanceMetadataDefaults
	ec2:GetInstanceTpmEkPub
	ec2:GetInstanceTypesFromInstanceRequirements
	ec2:GetInstanceUefiData
	ec2:GetIpamAddressHistory
	ec2:GetIpamDiscoveredAccounts
	ec2:GetIpamDiscoveredPublicAddresses

服務前綴	動作
	ec2:GetIpamDiscoveredResourceCidrs
	ec2：GetIpamPolicyAllocationRules
	ec2：GetIpamPolicyOrganizationTargets
	ec2:GetIpamPoolAllocations
	ec2:GetIpamPoolCidrs
	ec2：GetIpamPrefixListResolverRules
	ec2：GetIpamPrefixListResolverVersionEntries
	ec2：GetIpamPrefixListResolverVersions
	ec2:GetIpamResourceCidrs
	ec2:GetLaunchTemplateData
	ec2:GetManagedPrefixListAssociations
	ec2:GetManagedPrefixListEntries
	ec2：GetManagedResourceVisibility
	ec2:GetNetworkInsightsAccessScopeAnalysisFindings
	ec2:GetNetworkInsightsAccessScopeContent
	ec2:GetPasswordData
	ec2:GetReservedInstancesExchangeQuote
	ec2:GetRouteServerAssociations
	ec2:GetRouteServerPropagations
	ec2:GetRouteServerRoutingDatabase
	ec2:GetSecurityGroupsForVpc

服務前綴	動作
	ec2:GetSerialConsoleAccessStatus
	ec2:GetSnapshotBlockPublicAccessState
	ec2:GetSpotPlacementScores
	ec2:GetSubnetCidrReservations
	ec2:GetTransitGatewayAttachmentPropagations
	ec2：GetTransitGatewayMeteringPolicyEntries
	ec2:GetTransitGatewayMulticastDomainAssociations
	ec2:GetTransitGatewayPolicyTableAssociations
	ec2:GetTransitGatewayPolicyTableEntries
	ec2:GetTransitGatewayPrefixListReferences
	ec2:GetTransitGatewayRouteTableAssociations
	ec2:GetTransitGatewayRouteTablePropagations
	ec2:GetVerifiedAccessEndpointPolicy
	ec2:GetVerifiedAccessEndpointTargets
	ec2:GetVerifiedAccessGroupPolicy
	ec2：GetVpcResourcesBlockingEncryptionEnforcement
	ec2:GetVpnConnectionDeviceSampleConfiguration
	ec2:GetVpnConnectionDeviceTypes
	ec2:GetVpnTunnelReplacementStatus
	ec2:ImportClientVpnClientCertificateRevocationList
	ec2:ImportImage

服務前綴	動作
	ec2:ImportInstance
	ec2：ImportKeyPair
	ec2:ImportSnapshot
	ec2:ImportVolume
	ec2:ListImagesInRecycleBin
	ec2:ListSnapshotsInRecycleBin
	ec2：ListVolumesInRecycleBin
	ec2:LockSnapshot
	ec2:ModifyAddressAttribute
	ec2:ModifyAvailabilityZoneGroup
	ec2:ModifyCapacityReservation
	ec2:ModifyCapacityReservationFleet
	ec2:ModifyClientVpnEndpoint
	ec2:ModifyDefaultCreditSpecification
	ec2:ModifyEbsDefaultKmsKeyId
	ec2:ModifyFleet
	ec2:ModifyFpgaImageAttribute
	ec2:ModifyHosts
	ec2:ModifyIdFormat
	ec2:ModifyIdentityIdFormat
	ec2:ModifyImageAttribute

服務前綴	動作
	ec2:ModifyInstanceAttribute
	ec2:ModifyInstanceCapacityReservationAttributes
	ec2：ModifyInstanceConnectEndpoint
	ec2:ModifyInstanceCpuOptions
	ec2:ModifyInstanceCreditSpecification
	ec2:ModifyInstanceEventStartTime
	ec2:ModifyInstanceEventWindow
	ec2:ModifyInstanceMaintenanceOptions
	ec2:ModifyInstanceMetadataDefaults
	ec2:ModifyInstanceMetadataOptions
	ec2:ModifyInstanceNetworkPerformanceOptions
	ec2:ModifyInstancePlacement
	ec2:ModifyIpam
	ec2：ModifyIpamPolicyAllocationRules
	ec2:ModifyIpamPool
	ec2：ModifyIpamPoolAllocation
	ec2：ModifyIpamPrefixListResolver
	ec2：ModifyIpamPrefixListResolverTarget
	ec2:ModifyIpamResourceCidr
	ec2:ModifyIpamResourceDiscovery
	ec2:ModifyIpamScope

服務前綴	動作
	ec2:ModifyLaunchTemplate
	ec2:ModifyLocalGatewayRoute
	ec2:ModifyManagedPrefixList
	ec2：ModifyManagedResourceVisibility
	ec2:ModifyNetworkInterfaceAttribute
	ec2:ModifyPrivateDnsNameOptions
	ec2:ModifyPublicIpDnsNameOptions
	ec2:ModifyReservedInstances
	ec2:ModifyRouteServer
	ec2:ModifySecurityGroupRules
	ec2:ModifySnapshotAttribute
	ec2:ModifySnapshotTier
	ec2:ModifySpotFleetRequest
	ec2：ModifySubnetAttribute
	ec2:ModifyTrafficMirrorFilterNetworkServices
	ec2:ModifyTrafficMirrorFilterRule
	ec2:ModifyTrafficMirrorSession
	ec2:ModifyTransitGateway
	ec2：ModifyTransitGatewayMeteringPolicy
	ec2:ModifyTransitGatewayPrefixListReference
	ec2:ModifyTransitGatewayVpcAttachment

服務前綴	動作
	ec2:ModifyVerifiedAccessEndpoint
	ec2:ModifyVerifiedAccessEndpointPolicy
	ec2:ModifyVerifiedAccessGroup
	ec2:ModifyVerifiedAccessGroupPolicy
	ec2:ModifyVerifiedAccessInstance
	ec2:ModifyVerifiedAccessInstanceLoggingConfiguration
	ec2:ModifyVerifiedAccessTrustProvider
	ec2:ModifyVolume
	ec2:ModifyVolumeAttribute
	ec2：ModifyVpcAttribute
	ec2:ModifyVpcBlockPublicAccessExclusion
	ec2:ModifyVpcBlockPublicAccessOptions
	ec2：ModifyVpcEncryptionControl
	ec2:ModifyVpcEndpoint
	ec2:ModifyVpcEndpointConnectionNotification
	ec2:ModifyVpcEndpointServiceConfiguration
	ec2:ModifyVpcEndpointServicePayerResponsibility
	ec2:ModifyVpcEndpointServicePermissions
	ec2:ModifyVpcPeeringConnectionOptions
	ec2:ModifyVpcTenancy
	ec2:ModifyVpnConnection

服務前綴	動作
	ec2:ModifyVpnConnectionOptions
	ec2:ModifyVpnTunnelCertificate
	ec2:ModifyVpnTunnelOptions
	ec2:MonitorInstances
	ec2:MoveAddressToVpc
	ec2:MoveByoipCidrToIpam
	ec2:MoveCapacityReservationInstances
	ec2:ProvisionByoipCidr
	ec2:ProvisionIpamByoasn
	ec2:ProvisionIpamPoolCidr
	ec2:ProvisionPublicIpv4PoolCidr
	ec2:PurchaseCapacityBlockExtension
	ec2:PurchaseHostReservation
	ec2:PurchaseReservedInstancesOffering
	ec2:PurchaseScheduledInstances
	ec2:RebootInstances
	ec2:RegisterImage
	ec2:RegisterInstanceEventNotificationAttributes
	ec2:RegisterTransitGatewayMulticastGroupMembers
	ec2:RegisterTransitGatewayMulticastGroupSources
	ec2:RejectCapacityReservationBillingOwnership

服務前綴	動作
	ec2：RejectTransitGatewayClientVpnAttachment
	ec2:RejectTransitGatewayMulticastDomainAssociations
	ec2:RejectTransitGatewayPeeringAttachment
	ec2:RejectTransitGatewayVpcAttachment
	ec2:RejectVpcEndpointConnections
	ec2:RejectVpcPeeringConnection
	ec2:ReleaseAddress
	ec2:ReleaseHosts
	ec2:ReleaseIpamPoolAllocation
	ec2:ReplaceIamInstanceProfileAssociation
	ec2:ReplaceImageCriteriaInAllowedImagesSettings
	ec2:ReplaceNetworkAclAssociation
	ec2:ReplaceNetworkAclEntry
	ec2:ReplaceRoute
	ec2:ReplaceRouteTableAssociation
	ec2:ReplaceTransitGatewayRoute
	ec2:ReplaceVpnTunnel
	ec2:ReportInstanceStatus
	ec2:RequestSpotFleet
	ec2:RequestSpotInstances
	ec2:ResetAddressAttribute

服務前綴	動作
	ec2:ResetEbsDefaultKmsKeyId
	ec2:ResetFpgaImageAttribute
	ec2:ResetImageAttribute
	ec2:ResetInstanceAttribute
	ec2:ResetNetworkInterfaceAttribute
	ec2:ResetSnapshotAttribute
	ec2:RestoreAddressToClassic
	ec2:RestoreImageFromRecycleBin
	ec2:RestoreManagedPrefixListVersion
	ec2:RestoreSnapshotFromRecycleBin
	ec2:RestoreSnapshotTier
	ec2：RestoreVolumeFromRecycleBin
	ec2:RevokeClientVpnIngress
	ec2:RevokeSecurityGroupEgress
	ec2:RevokeSecurityGroupIngress
	ec2:RunInstances
	ec2:RunScheduledInstances
	ec2:SearchLocalGatewayRoutes
	ec2:SearchTransitGatewayMulticastGroups
	ec2:SearchTransitGatewayRoutes
	ec2:SendDiagnosticInterrupt

服務前綴	動作
	ec2:StartDeclarativePoliciesReport
	ec2:StartInstances
	ec2:StartNetworkInsightsAccessScopeAnalysis
	ec2:StartNetworkInsightsAnalysis
	ec2:StartVpcEndpointServicePrivateDnsVerification
	ec2:TerminateClientVpnConnections
	ec2:UnassignIpv6Addresses
	ec2:UnassignPrivateIpAddresses
	ec2:UnassignPrivateNatGatewayAddress
	ec2:UnlockSnapshot
	ec2:UnmonitorInstances
	ec2：UpdateCapacityManagerOrganizationsAccess
	ec2：UpdateInterruptibleCapacityReservationAllocation
	ec2:UpdateSecurityGroupRuleDescriptionsEgress
	ec2:UpdateSecurityGroupRuleDescriptionsIngress
	ec2:WithdrawByoipCidr

服務前綴	動作
ecr	ecr:BatchCheckLayerAvailability ecr:BatchDeleteImage ecr:BatchGetImage ecr:BatchGetRepositoryScanningConfiguration ecr:CompleteLayerUpload ecr:CreatePullThroughCacheRule ecr:CreateRepositoryCreationTemplate ecr:DeleteLifecyclePolicy ecr:DeletePullThroughCacheRule ecr:DeleteRegistryPolicy ecr:DeleteRepository ecr:DeleteRepositoryCreationTemplate ecr:DeleteRepositoryPolicy ecr：DeleteSigningConfiguration ecr:DescribeImageReplicationStatus ecr:DescribeImageScanFindings ecr:DescribeImages ecr:DescribePullThroughCacheRules ecr:DescribeRegistry ecr:DescribeRepositories ecr:DescribeRepositoryCreationTemplates

服務前綴	動作
	ecr:GetAccountSetting ecr:GetAuthorizationToken ecr:GetDownloadUrlForLayer ecr:GetLifecyclePolicy ecr:GetLifecyclePolicyPreview ecr:GetRegistryPolicy ecr:GetRegistryScanningConfiguration ecr:GetRepositoryPolicy ecr：GetSigningConfiguration ecr:InitiateLayerUpload ecr:ListImages ecr：ListPullTimeUpdateExclusions ecr:PutAccountSetting ecr:PutImage ecr:PutImageScanningConfiguration ecr:PutRegistryPolicy ecr:PutRegistryScanningConfiguration ecr:PutReplicationConfiguration ecr:StartImageScan ecr:StartLifecyclePolicyPreview ecr:UpdatePullThroughCacheRule

服務前綴	動作
	ecr:UpdateRepositoryCreationTemplate ecr:UploadLayerPart ecr:ValidatePullThroughCacheRule

服務前綴	動作
ecr-public	ecr-public:BatchCheckLayerAvailability
	ecr-public:BatchDeleteImage
	ecr-public:CompleteLayerUpload
	ecr-public:CreateRepository
	ecr-public:DeleteRepository
	ecr-public:DeleteRepositoryPolicy
	ecr-public:DescribeImages
	ecr-public:DescribeRegistries
	ecr-public:DescribeRepositories
	ecr-public:GetAuthorizationToken
	ecr-public:GetRegistryCatalogData
	ecr-public:GetRepositoryCatalogData
	ecr-public:GetRepositoryPolicy
	ecr-public:InitiateLayerUpload
	ecr-public:PutImage
	ecr-public:PutRegistryCatalogData
	ecr-public:PutRepositoryCatalogData
	ecr-public:SetRepositoryPolicy
	ecr-public:UploadLayerPart

服務前綴	動作
ecs	ecs：ContinueServiceDeployment ecs:CreateCapacityProvider ecs:CreateCluster ecs：CreateDaemon ecs:CreateService ecs:CreateTaskSet ecs:DeleteAccountSetting ecs:DeleteAttributes ecs:DeleteCapacityProvider ecs:DeleteCluster ecs：DeleteDaemon ecs：DeleteDaemonTaskDefinition ecs：DeleteExpressGatewayService ecs:DeleteService ecs:DeleteTaskDefinitions ecs:DeleteTaskSet ecs:DeregisterContainerInstance ecs:DeregisterTaskDefinition ecs:DescribeCapacityProviders ecs:DescribeClusters ecs:DescribeContainerInstances

服務前綴	動作
	ecs：DescribeDaemon ecs：DescribeDaemonDeployments ecs：DescribeDaemonRevisions ecs：DescribeDaemonTaskDefinition ecs：DescribeExpressGatewayService ecs:DescribeServiceDeployments ecs:DescribeServiceRevisions ecs:DescribeServices ecs:DescribeTaskDefinition ecs:DescribeTaskSets ecs:DescribeTasks ecs:DiscoverPollEndpoint ecs:ExecuteCommand ecs:GetTaskProtection ecs:ListAccountSettings ecs:ListAttributes ecs:ListClusters ecs:ListContainerInstances ecs：ListDaemonDeployments ecs：ListDaemonTaskDefinitions ecs：ListDaemons

服務前綴	動作
	ecs:ListServiceDeployments ecs:ListServices ecs:ListServicesByNamespace ecs:ListTaskDefinitionFamilies ecs:ListTaskDefinitions ecs:ListTasks ecs:PutAccountSetting ecs:PutAccountSettingDefault ecs:PutAttributes ecs:PutClusterCapacityProviders ecs:RegisterContainerInstance ecs：RegisterDaemonTaskDefinition ecs:RunTask ecs:StartTask ecs:StopServiceDeployment ecs:StopTask ecs:SubmitAttachmentStateChanges ecs:SubmitContainerStateChange ecs:SubmitTaskStateChange ecs:UpdateCapacityProvider ecs:UpdateCluster

服務前綴	動作
	ecs:UpdateClusterSettings ecs:UpdateContainerAgent ecs:UpdateContainerInstancesState ecs：UpdateDaemon ecs：UpdateExpressGatewayService ecs:UpdateService ecs:UpdateServicePrimaryTaskSet ecs:UpdateTaskProtection ecs:UpdateTaskSet

服務前綴	動作
eks	eks:AssociateAccessPolicy
	eks:AssociateEncryptionConfig
	eks:AssociateIdentityProviderConfig
	eks：CancelUpdate
	eks:CreateAccessEntry
	eks:CreateAddon
	eks:CreateCluster
	eks:CreateEksAnywhereSubscription
	eks:CreateFargateProfile
	eks:CreateNodegroup
	eks:DeleteAccessEntry
	eks:DeleteAddon
	eks：DeleteCapability
	eks:DeleteCluster
	eks:DeleteEksAnywhereSubscription
	eks:DeleteFargateProfile
	eks:DeleteNodegroup
	eks:DeletePodIdentityAssociation
	eks:DeregisterCluster
	eks:DescribeAccessEntry
	eks:DescribeAddon

服務前綴	動作
	eks:DescribeAddonConfiguration
	eks:DescribeAddonVersions
	eks：DescribeCapability
	eks:DescribeCluster
	eks:DescribeClusterVersions
	eks:DescribeEksAnywhereSubscription
	eks:DescribeFargateProfile
	eks:DescribeIdentityProviderConfig
	eks:DescribeInsight
	eks：DescribeInsightsRefresh
	eks:DescribeNodegroup
	eks:DescribePodIdentityAssociation
	eks:DescribeUpdate
	eks:DisassociateAccessPolicy
	eks:DisassociateIdentityProviderConfig
	eks:ListAccessEntries
	eks:ListAccessPolicies
	eks:ListAddons
	eks:ListAssociatedAccessPolicies
	eks：ListCapabilities
	eks:ListClusters

服務前綴	動作
	eks:ListEksAnywhereSubscriptions
	eks:ListFargateProfiles
	eks:ListIdentityProviderConfigs
	eks:ListInsights
	eks:ListNodegroups
	eks:ListPodIdentityAssociations
	eks:ListUpdates
	eks:RegisterCluster
	eks：StartInsightsRefresh
	eks:UpdateAccessEntry
	eks:UpdateAddon
	eks：UpdateCapability
	eks:UpdateClusterConfig
	eks:UpdateClusterVersion
	eks:UpdateEksAnywhereSubscription
	eks:UpdateNodegroupConfig
	eks:UpdateNodegroupVersion
	eks:UpdatePodIdentityAssociation

服務前綴	動作
elasticache	elasticache:AuthorizeCacheSecurityGroupIngress elasticache:BatchApplyUpdateAction elasticache:BatchStopUpdateAction elasticache:CompleteMigration elasticache:CopyServerlessCacheSnapshot elasticache:CopySnapshot elasticache:CreateCacheCluster elasticache:CreateCacheParameterGroup elasticache:CreateCacheSecurityGroup elasticache:CreateCacheSubnetGroup elasticache:CreateGlobalReplicationGroup elasticache:CreateReplicationGroup elasticache:CreateServerlessCache elasticache:CreateServerlessCacheSnapshot elasticache:CreateSnapshot elasticache:CreateUser elasticache:CreateUserGroup elasticache:DecreaseNodeGroupsInGlobalReplicationGroup elasticache:DecreaseReplicaCount elasticache:DeleteCacheCluster elasticache:DeleteCacheParameterGroup

服務前綴	動作
	elasticache:DeleteCacheSecurityGroup elasticache:DeleteCacheSubnetGroup elasticache:DeleteGlobalReplicationGroup elasticache:DeleteReplicationGroup elasticache:DeleteServerlessCache elasticache:DeleteServerlessCacheSnapshot elasticache:DeleteSnapshot elasticache:DeleteUser elasticache:DeleteUserGroup elasticache:DescribeCacheClusters elasticache:DescribeCacheEngineVersions elasticache:DescribeCacheParameterGroups elasticache:DescribeCacheParameters elasticache:DescribeCacheSecurityGroups elasticache:DescribeCacheSubnetGroups elasticache:DescribeEngineDefaultParameters elasticache:DescribeEvents elasticache:DescribeGlobalReplicationGroups elasticache:DescribeReplicationGroups elasticache:DescribeReservedCacheNodes elasticache:DescribeReservedCacheNodesOfferings

服務前綴	動作
	elasticache:DescribeServerlessCacheSnapshots elasticache:DescribeServerlessCaches elasticache:DescribeServiceUpdates elasticache:DescribeSnapshots elasticache:DescribeUpdateActions elasticache:DescribeUserGroups elasticache:DescribeUsers elasticache:DisassociateGlobalReplicationGroup elasticache:ExportServerlessCacheSnapshot elasticache:FailoverGlobalReplicationGroup elasticache:IncreaseNodeGroupsInGlobalReplicationGroup elasticache:IncreaseReplicaCount elasticache:ListAllowedNodeTypeModifications elasticache:ModifyCacheCluster elasticache:ModifyCacheParameterGroup elasticache:ModifyCacheSubnetGroup elasticache:ModifyGlobalReplicationGroup elasticache:ModifyReplicationGroup elasticache:ModifyReplicationGroupShardConfiguration elasticache:ModifyServerlessCache elasticache:ModifyUser

服務前綴	動作
	elasticache:ModifyUserGroup elasticache:PurchaseReservedCacheNodesOffering elasticache:RebalanceSlotsInGlobalReplicationGroup elasticache:RebootCacheCluster elasticache:ResetCacheParameterGroup elasticache:RevokeCacheSecurityGroupIngress elasticache:StartMigration elasticache:TestFailover elasticache:TestMigration

服務前綴	動作
elasticbeanstalk	elasticbeanstalk:AbortEnvironmentUpdate
	elasticbeanstalk:ApplyEnvironmentManagedAction
	elasticbeanstalk:AssociateEnvironmentOperationsRole
	elasticbeanstalk:CheckDNSAvailability
	elasticbeanstalk:ComposeEnvironments
	elasticbeanstalk:CreateApplication
	elasticbeanstalk:CreateApplicationVersion
	elasticbeanstalk:CreateConfigurationTemplate
	elasticbeanstalk:CreateEnvironment
	elasticbeanstalk:CreatePlatformVersion
	elasticbeanstalk:CreateStorageLocation
	elasticbeanstalk:DeleteApplication
	elasticbeanstalk:DeleteApplicationVersion
	elasticbeanstalk:DeleteConfigurationTemplate
	elasticbeanstalk:DeleteEnvironmentConfiguration
	elasticbeanstalk:DeletePlatformVersion
	elasticbeanstalk:DescribeAccountAttributes
	elasticbeanstalk:DescribeApplicationVersions
	elasticbeanstalk:DescribeApplications
	elasticbeanstalk:DescribeConfigurationOptions
	elasticbeanstalk:DescribeConfigurationSettings

服務前綴	動作
	elasticbeanstalk:DescribeEnvironmentHealth
	elasticbeanstalk:DescribeEnvironmentManagedActionHistory
	elasticbeanstalk:DescribeEnvironmentManagedActions
	elasticbeanstalk:DescribeEnvironmentResources
	elasticbeanstalk:DescribeEnvironments
	elasticbeanstalk:DescribeEvents
	elasticbeanstalk:DescribeInstancesHealth
	elasticbeanstalk:DescribePlatformVersion
	elasticbeanstalk:DisassociateEnvironmentOperationsRole
	elasticbeanstalk:ListAvailableSolutionStacks
	elasticbeanstalk:ListPlatformBranches
	elasticbeanstalk:ListPlatformVersions
	elasticbeanstalk:RebuildEnvironment
	elasticbeanstalk:RequestEnvironmentInfo
	elasticbeanstalk:RestartAppServer
	elasticbeanstalk:RetrieveEnvironmentInfo
	elasticbeanstalk:SwapEnvironmentCNAMEs
	elasticbeanstalk:TerminateEnvironment
	elasticbeanstalk:UpdateApplication
	elasticbeanstalk:UpdateApplicationResourceLifecycle
	elasticbeanstalk:UpdateApplicationVersion

服務前綴	動作
	elasticbeanstalk:UpdateConfigurationTemplate elasticbeanstalk:UpdateEnvironment elasticbeanstalk:ValidateConfigurationSettings

服務前綴	動作
elasticfilesystem	elasticfilesystem:CreateAccessPoint elasticfilesystem:CreateFileSystem elasticfilesystem:CreateMountTarget elasticfilesystem:CreateReplicationConfiguration elasticfilesystem:DeleteAccessPoint elasticfilesystem:DeleteFileSystem elasticfilesystem:DeleteFileSystemPolicy elasticfilesystem:DeleteMountTarget elasticfilesystem:DeleteReplicationConfiguration elasticfilesystem:DescribeAccessPoints elasticfilesystem:DescribeAccountPreferences elasticfilesystem:DescribeBackupPolicy elasticfilesystem:DescribeFileSystemPolicy elasticfilesystem:DescribeFileSystems elasticfilesystem:DescribeLifecycleConfiguration elasticfilesystem:DescribeMountTargetSecurityGroups elasticfilesystem:DescribeMountTargets elasticfilesystem:DescribeReplicationConfigurations elasticfilesystem:ModifyMountTargetSecurityGroups elasticfilesystem:PutAccountPreferences elasticfilesystem:PutBackupPolicy

服務前綴	動作
	elasticfilesystem:PutFileSystemPolicy elasticfilesystem:PutLifecycleConfiguration elasticfilesystem:UpdateFileSystem elasticfilesystem:UpdateFileSystemProtection

服務前綴	動作
elasticloadbalancing	elasticloadbalancing:AddListenerCertificates
	elasticloadbalancing:AddTrustStoreRevocations
	elasticloadbalancing:ApplySecurityGroupsToLoadBalancer
	elasticloadbalancing:AttachLoadBalancerToSubnets
	elasticloadbalancing:ConfigureHealthCheck
	elasticloadbalancing:CreateAppCookieStickinessPolicy
	elasticloadbalancing:CreateLBCookieStickinessPolicy
	elasticloadbalancing:CreateListener
	elasticloadbalancing:CreateLoadBalancer
	elasticloadbalancing:CreateLoadBalancerListeners
	elasticloadbalancing:CreateLoadBalancerPolicy
	elasticloadbalancing:CreateRule
	elasticloadbalancing:CreateTargetGroup
	elasticloadbalancing:CreateTrustStore
	elasticloadbalancing:DeleteListener
	elasticloadbalancing:DeleteLoadBalancer
	elasticloadbalancing:DeleteLoadBalancerListeners
	elasticloadbalancing:DeleteLoadBalancerPolicy
	elasticloadbalancing:DeleteRule
	elasticloadbalancing:DeleteSharedTrustStoreAssociation
	elasticloadbalancing:DeleteTargetGroup

服務前綴	動作
	elasticloadbalancing:DeleteTrustStore
	elasticloadbalancing:DeregisterInstancesFromLoadBalancer
	elasticloadbalancing:DeregisterTargets
	elasticloadbalancing:DescribeAccountLimits
	elasticloadbalancing:DescribeCapacityReservation
	elasticloadbalancing:DescribeInstanceHealth
	elasticloadbalancing:DescribeListenerAttributes
	elasticloadbalancing:DescribeListenerCertificates
	elasticloadbalancing:DescribeListeners
	elasticloadbalancing:DescribeLoadBalancerAttributes
	elasticloadbalancing:DescribeLoadBalancerPolicies
	elasticloadbalancing:DescribeLoadBalancerPolicyTypes
	elasticloadbalancing:DescribeLoadBalancers
	elasticloadbalancing:DescribeRules
	elasticloadbalancing:DescribeSSLPolicies
	elasticloadbalancing:DescribeTargetGroupAttributes
	elasticloadbalancing:DescribeTargetGroups
	elasticloadbalancing:DescribeTargetHealth
	elasticloadbalancing:DescribeTrustStoreAssociations
	elasticloadbalancing:DescribeTrustStoreRevocations
	elasticloadbalancing:DescribeTrustStores

服務前綴	動作
	elasticloadbalancing:DetachLoadBalancerFromSubnets
	elasticloadbalancing:DisableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:EnableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:GetResourcePolicy
	elasticloadbalancing:GetTrustStoreCaCertificatesBundle
	elasticloadbalancing:GetTrustStoreRevocationContent
	elasticloadbalancing:ModifyCapacityReservation
	elasticloadbalancing:ModifyIpPools
	elasticloadbalancing:ModifyListener
	elasticloadbalancing:ModifyLoadBalancerAttributes
	elasticloadbalancing:ModifyRule
	elasticloadbalancing:ModifyTargetGroup
	elasticloadbalancing:ModifyTargetGroupAttributes
	elasticloadbalancing:ModifyTrustStore
	elasticloadbalancing:RegisterInstancesWithLoadBalancer
	elasticloadbalancing:RegisterTargets
	elasticloadbalancing:RemoveListenerCertificates
	elasticloadbalancing:RemoveTrustStoreRevocations
	elasticloadbalancing:SetIpAddressType
	elasticloadbalancing:SetLoadBalancerListenerSSLCertificate
	elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer

服務前綴	動作
	elasticloadbalancing:SetLoadBalancerPoliciesOfListener elasticloadbalancing:SetRulePriorities elasticloadbalancing:SetSecurityGroups elasticloadbalancing:SetSubnets

服務前綴	動作
emr-containers	emr-containers:CancelJobRun emr-containers:CreateJobTemplate emr-containers:CreateManagedEndpoint emr-containers:CreateSecurityConfiguration emr-containers:CreateVirtualCluster emr-containers:DeleteJobTemplate emr-containers:DeleteManagedEndpoint emr-containers：DeleteSecurityConfiguration emr-containers:DeleteVirtualCluster emr-containers:DescribeJobRun emr-containers:DescribeJobTemplate emr-containers:DescribeManagedEndpoint emr-containers:DescribeSecurityConfiguration emr-containers:DescribeVirtualCluster emr-containers:GetManagedEndpointSessionCredentials emr-containers:ListJobRuns emr-containers:ListJobTemplates emr-containers:ListManagedEndpoints emr-containers:ListSecurityConfigurations emr-containers:ListVirtualClusters emr-containers:StartJobRun

服務前綴	動作
emr-serverless	emr-serverless:CancelJobRun emr-serverless:CreateApplication emr-serverless:DeleteApplication emr-serverless:GetApplication emr-serverless:GetDashboardForJobRun emr-serverless:GetJobRun emr-serverless：GetResourceDashboard emr-serverless：GetSession emr-serverless：GetSessionEndpoint emr-serverless:ListApplications emr-serverless:ListJobRunAttempts emr-serverless:ListJobRuns emr-serverless：ListSessions emr-serverless:StartApplication emr-serverless:StartJobRun emr-serverless:StopApplication emr-serverless：TerminateSession emr-serverless:UpdateApplication

服務前綴	動作
es	es:AcceptInboundConnection es:AcceptInboundCrossClusterSearchConnection es:AssociatePackage es:AuthorizeVpcEndpointAccess es:CancelElasticsearchServiceSoftwareUpdate es:CancelServiceSoftwareUpdate es:CreateDomain es:CreateElasticsearchDomain es：CreateIndex es:CreateOutboundConnection es:CreateOutboundCrossClusterSearchConnection es:CreatePackage es:CreateVpcEndpoint es:DeleteDomain es:DeleteElasticsearchDomain es:DeleteElasticsearchServiceRole es:DeleteInboundConnection es:DeleteInboundCrossClusterSearchConnection es：DeleteIndex es:DeleteOutboundConnection es:DeleteOutboundCrossClusterSearchConnection

服務前綴	動作
	es:DeletePackage es:DeleteVpcEndpoint es:DescribeDomain es:DescribeDomainAutoTunes es:DescribeDomainChangeProgress es:DescribeDomainConfig es:DescribeDomainHealth es:DescribeDomainNodes es:DescribeDomains es:DescribeDryRunProgress es:DescribeElasticsearchDomain es:DescribeElasticsearchDomainConfig es:DescribeElasticsearchDomains es:DescribeElasticsearchInstanceTypeLimits es:DescribeInboundConnections es:DescribeInboundCrossClusterSearchConnections es:DescribeInstanceTypeLimits es:DescribeOutboundConnections es:DescribeOutboundCrossClusterSearchConnections es:DescribePackages es:DescribeReservedElasticsearchInstanceOfferings

服務前綴	動作
	es:DescribeReservedElasticsearchInstances es:DescribeReservedInstanceOfferings es:DescribeReservedInstances es:DescribeVpcEndpoints es:DissociatePackage es:DissociatePackages es:GetCompatibleElasticsearchVersions es:GetCompatibleVersions es:GetDataSource es:GetDomainMaintenanceStatus es:GetPackageVersionHistory es:GetUpgradeHistory es:GetUpgradeStatus es:ListDataSources es:ListDomainNames es:ListDomainsForPackage es:ListElasticsearchInstanceTypes es:ListElasticsearchVersions es：ListInsights es:ListInstanceTypeDetails es:ListPackagesForDomain

服務前綴	動作
	es:ListScheduledActions es:ListVersions es:ListVpcEndpointAccess es:ListVpcEndpoints es:ListVpcEndpointsForDomain es:PurchaseReservedElasticsearchInstanceOffering es:PurchaseReservedInstanceOffering es:RejectInboundConnection es:RejectInboundCrossClusterSearchConnection es:RevokeVpcEndpointAccess es：RollbackServiceSoftwareUpdate es:StartDomainMaintenance es:StartElasticsearchServiceSoftwareUpdate es:StartServiceSoftwareUpdate es:UpdateDataSource es:UpdateDomainConfig es:UpdateElasticsearchDomainConfig es：UpdateIndex es:UpdatePackage es:UpdatePackageScope es:UpdateScheduledAction

服務前綴	動作
	es:UpdateVpcEndpoint es:UpgradeDomain es:UpgradeElasticsearchDomain

服務前綴	動作
events	events:ActivateEventSource
	events:CancelReplay
	events:CreateApiDestination
	events:CreateArchive
	events:CreateConnection
	events:CreateEndpoint
	events:CreateEventBus
	events:CreatePartnerEventSource
	events:DeactivateEventSource
	events:DeauthorizeConnection
	events:DeleteApiDestination
	events:DeleteArchive
	events:DeleteConnection
	events:DeleteEndpoint
	events:DeleteEventBus
	events:DeletePartnerEventSource
	events:DeleteRule
	events:DescribeApiDestination
	events:DescribeArchive
	events:DescribeConnection
	events:DescribeEndpoint

服務前綴	動作
	events:DescribeEventBus
	events:DescribeEventSource
	events:DescribePartnerEventSource
	events:DescribeReplay
	events:DescribeRule
	events:DisableRule
	events:EnableRule
	events:ListApiDestinations
	events:ListArchives
	events:ListConnections
	events:ListEndpoints
	events:ListEventBuses
	events:ListEventSources
	events:ListPartnerEventSourceAccounts
	events:ListPartnerEventSources
	events:ListReplays
	events:ListRuleNamesByTarget
	events:ListRules
	events:ListTargetsByRule
	events:PutPermission
	events:PutRule

服務前綴	動作
	events:PutTargets
	events:RemovePermission
	events:RemoveTargets
	events:StartReplay
	events:TestEventPattern
	events:UpdateApiDestination
	events:UpdateArchive
	events:UpdateConnection
	events:UpdateEndpoint
	events:UpdateEventBus

服務前綴	動作
finspace	finspace:CreateEnvironment finspace:CreateKxChangeset finspace:CreateKxCluster finspace:CreateKxDatabase finspace:CreateKxDataview finspace:CreateKxEnvironment finspace:CreateKxScalingGroup finspace:CreateKxUser finspace:CreateKxVolume finspace:CreateUser finspace:DeleteEnvironment finspace:DeleteKxCluster finspace:DeleteKxClusterNode finspace:DeleteKxDatabase finspace:DeleteKxDataview finspace:DeleteKxEnvironment finspace:DeleteKxScalingGroup finspace:DeleteKxUser finspace:DeleteKxVolume finspace:GetEnvironment finspace:GetKxChangeset

服務前綴	動作
	finspace:GetKxCluster finspace:GetKxConnectionString finspace:GetKxDatabase finspace:GetKxDataview finspace:GetKxEnvironment finspace:GetKxScalingGroup finspace:GetKxUser finspace:GetKxVolume finspace:GetLoadSampleDataSetGroupIntoEnvironmentStatus finspace:GetUser finspace:ListEnvironments finspace:ListKxChangesets finspace:ListKxClusterNodes finspace:ListKxClusters finspace:ListKxDatabases finspace:ListKxDataviews finspace:ListKxEnvironments finspace:ListKxScalingGroups finspace:ListKxUsers finspace:ListKxVolumes finspace:ListUsers

服務前綴	動作
	finspace:LoadSampleDataSetGroupIntoEnvironment finspace:ResetUserPassword finspace:UpdateEnvironment finspace:UpdateKxClusterCodeConfiguration finspace:UpdateKxClusterDatabases finspace:UpdateKxDatabase finspace:UpdateKxDataview finspace:UpdateKxEnvironment finspace:UpdateKxEnvironmentNetwork finspace:UpdateKxUser finspace:UpdateKxVolume finspace:UpdateUser
firehose	firehose:CreateDeliveryStream firehose:DeleteDeliveryStream firehose:DescribeDeliveryStream firehose:ListDeliveryStreams firehose:StartDeliveryStreamEncryption firehose:StopDeliveryStreamEncryption firehose:UpdateDestination

服務前綴	動作
fis	fis:CreateExperimentTemplate fis:CreateTargetAccountConfiguration fis:DeleteExperimentTemplate fis:DeleteTargetAccountConfiguration fis:GetAction fis:GetExperiment fis:GetExperimentTargetAccountConfiguration fis:GetExperimentTemplate fis:GetSafetyLever fis:GetTargetAccountConfiguration fis:GetTargetResourceType fis:ListActions fis:ListExperimentResolvedTargets fis:ListExperimentTargetAccountConfigurations fis:ListExperimentTemplates fis:ListExperiments fis:ListTargetAccountConfigurations fis:ListTargetResourceTypes fis:StartExperiment fis:StopExperiment fis:UpdateExperimentTemplate

服務前綴	動作
	fis:UpdateSafetyLeverState fis:UpdateTargetAccountConfiguration

服務前綴	動作
fms	fms:AssociateAdminAccount
	fms:AssociateThirdPartyFirewall
	fms:BatchAssociateResource
	fms:BatchDisassociateResource
	fms:DeleteAppsList
	fms:DeleteNotificationChannel
	fms:DeletePolicy
	fms:DeleteProtocolsList
	fms:DeleteResourceSet
	fms:DisassociateAdminAccount
	fms:DisassociateThirdPartyFirewall
	fms:GetAdminAccount
	fms:GetAdminScope
	fms:GetAppsList
	fms:GetComplianceDetail
	fms:GetNotificationChannel
	fms:GetPolicy
	fms:GetProtectionStatus
	fms:GetProtocolsList
	fms:GetResourceSet
	fms:GetThirdPartyFirewallAssociationStatus

服務前綴	動作
	fms:GetViolationDetails
	fms:ListAdminAccountsForOrganization
	fms:ListAdminsManagingAccount
	fms:ListAppsLists
	fms:ListComplianceStatus
	fms:ListDiscoveredResources
	fms:ListMemberAccounts
	fms:ListPolicies
	fms:ListProtocolsLists
	fms:ListResourceSetResources
	fms:ListResourceSets
	fms:ListThirdPartyFirewallFirewallPolicies
	fms:PutAdminAccount
	fms:PutAppsList
	fms:PutNotificationChannel
	fms:PutPolicy
	fms:PutProtocolsList
	fms:PutResourceSet

服務前綴	動作
frauddetector	frauddetector:BatchCreateVariable frauddetector:BatchGetVariable frauddetector:CancelBatchImportJob frauddetector:CancelBatchPredictionJob frauddetector:CreateBatchImportJob frauddetector:CreateBatchPredictionJob frauddetector:CreateDetectorVersion frauddetector:CreateList frauddetector:CreateModel frauddetector:CreateModelVersion frauddetector:CreateRule frauddetector:CreateVariable frauddetector:DeleteBatchImportJob frauddetector:DeleteBatchPredictionJob frauddetector:DeleteDetector frauddetector:DeleteDetectorVersion frauddetector:DeleteEntityType frauddetector:DeleteEvent frauddetector:DeleteEventType frauddetector:DeleteEventsByEventType frauddetector:DeleteExternalModel

服務前綴	動作
	frauddetector:DeleteLabel frauddetector:DeleteList frauddetector:DeleteModel frauddetector:DeleteModelVersion frauddetector:DeleteOutcome frauddetector:DeleteRule frauddetector:DeleteVariable frauddetector:DescribeDetector frauddetector:DescribeModelVersions frauddetector:GetBatchImportJobs frauddetector:GetBatchPredictionJobs frauddetector:GetDeleteEventsByEventTypeStatus frauddetector:GetDetectorVersion frauddetector:GetDetectors frauddetector:GetEntityTypes frauddetector:GetEvent frauddetector:GetEventPrediction frauddetector:GetEventPredictionMetadata frauddetector:GetEventTypes frauddetector:GetExternalModels frauddetector:GetKMSEncryptionKey

服務前綴	動作
	frauddetector:GetLabels frauddetector:GetListElements frauddetector:GetListsMetadata frauddetector:GetModelVersion frauddetector:GetModels frauddetector:GetOutcomes frauddetector:GetRules frauddetector:GetVariables frauddetector:ListEventPredictions frauddetector:PutDetector frauddetector:PutEntityType frauddetector:PutEventType frauddetector:PutExternalModel frauddetector:PutKMSEncryptionKey frauddetector:PutLabel frauddetector:PutOutcome frauddetector:SendEvent frauddetector:UpdateDetectorVersion frauddetector:UpdateDetectorVersionMetadata frauddetector:UpdateDetectorVersionStatus frauddetector:UpdateEventLabel

服務前綴	動作
	frauddetector:UpdateList frauddetector:UpdateModel frauddetector:UpdateModelVersion frauddetector:UpdateModelVersionStatus frauddetector:UpdateRuleMetadata frauddetector:UpdateRuleVersion frauddetector:UpdateVariable

服務前綴	動作
fsx	fsx:AssociateFileSystemAliases
	fsx:CancelDataRepositoryTask
	fsx:CopyBackup
	fsx:CreateDataRepositoryTask
	fsx:CreateFileCache
	fsx:CreateFileSystem
	fsx:CreateFileSystemFromBackup
	fsx:CreateSnapshot
	fsx:CreateStorageVirtualMachine
	fsx:CreateVolume
	fsx:CreateVolumeFromBackup
	fsx:DeleteBackup
	fsx:DeleteFileCache
	fsx:DeleteFileSystem
	fsx:DeleteSnapshot
	fsx:DeleteStorageVirtualMachine
	fsx:DeleteVolume
	fsx:DescribeBackups
	fsx:DescribeDataRepositoryAssociations
	fsx:DescribeDataRepositoryTasks
	fsx:DescribeFileCaches

服務前綴	動作
	fsx:DescribeFileSystemAliases
	fsx:DescribeFileSystems
	fsx:DescribeS3AccessPointAttachments
	fsx:DescribeSharedVpcConfiguration
	fsx:DescribeSnapshots
	fsx:DescribeStorageVirtualMachines
	fsx:DescribeVolumes
	fsx:DetachAndDeleteS3AccessPoint
	fsx:DisassociateFileSystemAliases
	fsx:ReleaseFileSystemNfsV3Locks
	fsx:RestoreVolumeFromSnapshot
	fsx:StartMisconfiguredStateRecovery
	fsx:UpdateDataRepositoryAssociation
	fsx:UpdateFileCache
	fsx:UpdateFileSystem
	fsx:UpdateSharedVpcConfiguration
	fsx:UpdateSnapshot
	fsx:UpdateStorageVirtualMachine
	fsx:UpdateVolume

服務前綴	動作
gamelift	gamelift:AcceptMatch gamelift:ClaimGameServer gamelift:CreateAlias gamelift:CreateBuild gamelift:CreateContainerGroupDefinition gamelift:CreateFleet gamelift:CreateFleetLocations gamelift:CreateGameServerGroup gamelift:CreateGameSession gamelift:CreateGameSessionQueue gamelift:CreateLocation gamelift:CreateMatchmakingConfiguration gamelift:CreateMatchmakingRuleSet gamelift:CreatePlayerSession gamelift:CreatePlayerSessions gamelift:CreateScript gamelift:CreateVpcPeeringAuthorization gamelift:CreateVpcPeeringConnection gamelift:DeleteAlias gamelift:DeleteBuild gamelift:DeleteContainerGroupDefinition

服務前綴	動作
	gamelift:DeleteFleet gamelift:DeleteFleetLocations gamelift:DeleteGameServerGroup gamelift:DeleteGameSessionQueue gamelift:DeleteLocation gamelift:DeleteMatchmakingConfiguration gamelift:DeleteMatchmakingRuleSet gamelift:DeleteScalingPolicy gamelift:DeleteScript gamelift:DeleteVpcPeeringAuthorization gamelift:DeleteVpcPeeringConnection gamelift:DeregisterCompute gamelift:DeregisterGameServer gamelift:DescribeAlias gamelift:DescribeBuild gamelift:DescribeCompute gamelift:DescribeContainerFleet gamelift:DescribeContainerGroupDefinition gamelift：DescribeContainerGroupPortMappings gamelift:DescribeEC2InstanceLimits gamelift:DescribeFleetAttributes

服務前綴	動作
	gamelift:DescribeFleetCapacity gamelift:DescribeFleetEvents gamelift:DescribeFleetLocationAttributes gamelift:DescribeFleetLocationCapacity gamelift:DescribeFleetLocationUtilization gamelift:DescribeFleetPortSettings gamelift:DescribeFleetUtilization gamelift:DescribeGameServer gamelift:DescribeGameServerGroup gamelift:DescribeGameServerInstances gamelift:DescribeGameSessionDetails gamelift:DescribeGameSessionPlacement gamelift:DescribeGameSessionQueues gamelift:DescribeGameSessions gamelift:DescribeInstances gamelift:DescribeMatchmaking gamelift:DescribeMatchmakingConfigurations gamelift:DescribeMatchmakingRuleSets gamelift:DescribePlayerSessions gamelift:DescribeRuntimeConfiguration gamelift:DescribeScalingPolicies

服務前綴	動作
	gamelift:DescribeScript gamelift:DescribeVpcPeeringAuthorizations gamelift:DescribeVpcPeeringConnections gamelift:GetComputeAccess gamelift:GetComputeAuthToken gamelift:GetGameSessionLogUrl gamelift:GetInstanceAccess gamelift：GetPlayerConnectionDetails gamelift:ListAliases gamelift:ListBuilds gamelift:ListCompute gamelift:ListContainerFleets gamelift:ListContainerGroupDefinitionVersions gamelift:ListContainerGroupDefinitions gamelift:ListFleetDeployments gamelift:ListFleets gamelift:ListGameServerGroups gamelift:ListGameServers gamelift:ListLocations gamelift:ListScripts gamelift:PutScalingPolicy

服務前綴	動作
	gamelift:RegisterCompute gamelift:RegisterGameServer gamelift:RequestUploadCredentials gamelift:ResolveAlias gamelift:ResumeGameServerGroup gamelift:SearchGameSessions gamelift:StartFleetActions gamelift:StartGameSessionPlacement gamelift:StartMatchBackfill gamelift:StartMatchmaking gamelift:StopFleetActions gamelift:StopGameSessionPlacement gamelift:StopMatchmaking gamelift:SuspendGameServerGroup gamelift:TerminateGameSession gamelift:UpdateAlias gamelift:UpdateBuild gamelift:UpdateContainerGroupDefinition gamelift:UpdateFleetAttributes gamelift:UpdateFleetCapacity gamelift:UpdateFleetPortSettings

服務前綴	動作
	gamelift:UpdateGameServer gamelift:UpdateGameServerGroup gamelift:UpdateGameSession gamelift:UpdateGameSessionQueue gamelift:UpdateMatchmakingConfiguration gamelift:UpdateRuntimeConfiguration gamelift:UpdateScript gamelift:ValidateMatchmakingRuleSet

服務前綴	動作
geo	geo:AssociateTrackerConsumer
	geo:BatchDeleteDevicePositionHistory
	geo:BatchDeleteGeofence
	geo:BatchEvaluateGeofences
	geo:BatchGetDevicePosition
	geo:BatchPutGeofence
	geo:BatchUpdateDevicePosition
	geo:CalculateRoute
	geo:CalculateRouteMatrix
	geo：CancelJob
	geo:CreateGeofenceCollection
	geo:CreateMap
	geo:CreatePlaceIndex
	geo:CreateRouteCalculator
	geo:CreateTracker
	geo:DeleteGeofenceCollection
	geo:DeleteKey
	geo:DeleteMap
	geo:DeletePlaceIndex
	geo:DeleteRouteCalculator
	geo:DeleteTracker

服務前綴	動作
	geo:DescribeGeofenceCollection
	geo:DescribeKey
	geo:DescribeMap
	geo:DescribePlaceIndex
	geo:DescribeRouteCalculator
	geo:DescribeTracker
	geo:DisassociateTrackerConsumer
	geo:ForecastGeofenceEvents
	geo:GetDevicePosition
	geo:GetDevicePositionHistory
	geo:GetGeofence
	geo：GetJob
	geo:GetMapGlyphs
	geo:GetMapSprites
	geo:GetMapStyleDescriptor
	geo:GetMapTile
	geo:GetPlace
	geo:ListDevicePositions
	geo:ListGeofenceCollections
	geo:ListGeofences
	geo：ListJobs

服務前綴	動作
	geo:ListKeys
	geo:ListMaps
	geo:ListPlaceIndexes
	geo:ListRouteCalculators
	geo:ListTrackerConsumers
	geo:ListTrackers
	geo:PutGeofence
	geo:SearchPlaceIndexForPosition
	geo:SearchPlaceIndexForSuggestions
	geo:SearchPlaceIndexForText
	geo:UpdateGeofenceCollection
	geo:UpdateKey
	geo:UpdateMap
	geo:UpdatePlaceIndex
	geo:UpdateRouteCalculator
	geo:UpdateTracker
	geo:VerifyDevicePosition

服務前綴	動作
glacier	glacier:AbortMultipartUpload glacier:AbortVaultLock glacier:CompleteMultipartUpload glacier:CompleteVaultLock glacier:CreateVault glacier:DeleteArchive glacier:DeleteVault glacier:DeleteVaultAccessPolicy glacier:DeleteVaultNotifications glacier:DescribeJob glacier:DescribeVault glacier:GetDataRetrievalPolicy glacier:GetJobOutput glacier:GetVaultAccessPolicy glacier:GetVaultLock glacier:GetVaultNotifications glacier:InitiateJob glacier:InitiateMultipartUpload glacier:InitiateVaultLock glacier:ListJobs glacier:ListMultipartUploads

服務前綴	動作
	glacier:ListParts glacier:ListProvisionedCapacity glacier:ListVaults glacier:PurchaseProvisionedCapacity glacier:SetDataRetrievalPolicy glacier:SetVaultAccessPolicy glacier:SetVaultNotifications glacier:UploadArchive glacier:UploadMultipartPart

服務前綴	動作
grafana	grafana:AssociateLicense
	grafana:CreateWorkspace
	grafana:CreateWorkspaceApiKey
	grafana:CreateWorkspaceServiceAccount
	grafana:CreateWorkspaceServiceAccountToken
	grafana:DeleteWorkspace
	grafana:DeleteWorkspaceApiKey
	grafana:DeleteWorkspaceServiceAccount
	grafana:DeleteWorkspaceServiceAccountToken
	grafana:DescribeWorkspace
	grafana:DescribeWorkspaceAuthentication
	grafana:DescribeWorkspaceConfiguration
	grafana:DisassociateLicense
	grafana:ListPermissions
	grafana:ListVersions
	grafana:ListWorkspaceServiceAccountTokens
	grafana:ListWorkspaceServiceAccounts
	grafana:ListWorkspaces
	grafana:UpdatePermissions
	grafana:UpdateWorkspace
	grafana:UpdateWorkspaceAuthentication

服務前綴	動作
	grafana:UpdateWorkspaceConfiguration

服務前綴	動作
greengrass	greengrass:AssociateRoleToGroup greengrass:AssociateServiceRoleToAccount greengrass:BatchAssociateClientDeviceWithCoreDevice greengrass:BatchDisassociateClientDeviceFromCoreDevice greengrass:CancelDeployment greengrass:CreateComponentVersion greengrass:CreateConnectorDefinition greengrass:CreateConnectorDefinitionVersion greengrass:CreateCoreDefinition greengrass:CreateCoreDefinitionVersion greengrass:CreateDeployment greengrass:CreateDeviceDefinition greengrass:CreateDeviceDefinitionVersion greengrass:CreateFunctionDefinition greengrass:CreateFunctionDefinitionVersion greengrass:CreateGroup greengrass:CreateGroupCertificateAuthority greengrass:CreateGroupVersion greengrass:CreateLoggerDefinition greengrass:CreateLoggerDefinitionVersion greengrass:CreateResourceDefinition

服務前綴	動作
	greengrass:CreateResourceDefinitionVersion greengrass:CreateSoftwareUpdateJob greengrass:CreateSubscriptionDefinition greengrass:CreateSubscriptionDefinitionVersion greengrass:DeleteComponent greengrass:DeleteConnectorDefinition greengrass:DeleteCoreDefinition greengrass:DeleteCoreDevice greengrass:DeleteDeployment greengrass:DeleteDeviceDefinition greengrass:DeleteFunctionDefinition greengrass:DeleteGroup greengrass:DeleteLoggerDefinition greengrass:DeleteResourceDefinition greengrass:DeleteSubscriptionDefinition greengrass:DescribeComponent greengrass:DisassociateRoleFromGroup greengrass:DisassociateServiceRoleFromAccount greengrass:GetAssociatedRole greengrass:GetBulkDeploymentStatus greengrass:GetComponent

服務前綴	動作
	greengrass:GetComponentVersionArtifact greengrass:GetConnectivityInfo greengrass:GetConnectorDefinition greengrass:GetConnectorDefinitionVersion greengrass:GetCoreDefinition greengrass:GetCoreDefinitionVersion greengrass:GetCoreDevice greengrass:GetDeployment greengrass:GetDeploymentStatus greengrass:GetDeviceDefinition greengrass:GetDeviceDefinitionVersion greengrass:GetFunctionDefinition greengrass:GetFunctionDefinitionVersion greengrass:GetGroup greengrass:GetGroupCertificateAuthority greengrass:GetGroupCertificateConfiguration greengrass:GetGroupVersion greengrass:GetLoggerDefinition greengrass:GetLoggerDefinitionVersion greengrass:GetResourceDefinition greengrass:GetResourceDefinitionVersion

服務前綴	動作
	greengrass:GetServiceRoleForAccount greengrass:GetSubscriptionDefinition greengrass:GetSubscriptionDefinitionVersion greengrass:GetThingRuntimeConfiguration greengrass:ListBulkDeploymentDetailedReports greengrass:ListBulkDeployments greengrass:ListClientDevicesAssociatedWithCoreDevice greengrass:ListComponentVersions greengrass:ListComponents greengrass:ListConnectorDefinitionVersions greengrass:ListConnectorDefinitions greengrass:ListCoreDefinitionVersions greengrass:ListCoreDefinitions greengrass:ListCoreDevices greengrass:ListDeployments greengrass:ListDeviceDefinitionVersions greengrass:ListDeviceDefinitions greengrass:ListEffectiveDeployments greengrass:ListFunctionDefinitionVersions greengrass:ListFunctionDefinitions greengrass:ListGroupCertificateAuthorities

服務前綴	動作
	greengrass:ListGroupVersions greengrass:ListGroups greengrass:ListInstalledComponents greengrass:ListLoggerDefinitionVersions greengrass:ListLoggerDefinitions greengrass:ListResourceDefinitionVersions greengrass:ListResourceDefinitions greengrass:ListSubscriptionDefinitionVersions greengrass:ListSubscriptionDefinitions greengrass:ResetDeployments greengrass:StartBulkDeployment greengrass:StopBulkDeployment greengrass:UpdateConnectivityInfo greengrass:UpdateConnectorDefinition greengrass:UpdateCoreDefinition greengrass:UpdateDeviceDefinition greengrass:UpdateFunctionDefinition greengrass:UpdateGroup greengrass:UpdateGroupCertificateConfiguration greengrass:UpdateLoggerDefinition greengrass:UpdateResourceDefinition

服務前綴	動作
	greengrass:UpdateSubscriptionDefinition greengrass:UpdateThingRuntimeConfiguration

服務前綴	動作
groundstation	groundstation:CancelContact
	groundstation:CreateConfig
	groundstation:CreateDataflowEndpointGroup
	groundstation：CreateDataflowEndpointGroupV2
	groundstation:CreateEphemeris
	groundstation:CreateMissionProfile
	groundstation:DeleteConfig
	groundstation:DeleteDataflowEndpointGroup
	groundstation:DeleteEphemeris
	groundstation:DeleteMissionProfile
	groundstation:DescribeContact
	groundstation：DescribeContactVersion
	groundstation:DescribeEphemeris
	groundstation:GetConfig
	groundstation:GetDataflowEndpointGroup
	groundstation:GetMinuteUsage
	groundstation:GetMissionProfile
	groundstation:GetSatellite
	groundstation：ListAntennas
	groundstation:ListConfigs
	groundstation:ListContacts

服務前綴	動作
	groundstation:ListDataflowEndpointGroups
	groundstation:ListEphemerides
	Groundstation：ListGroundStationReservations
	groundstation:ListGroundStations
	groundstation:ListMissionProfiles
	groundstation:ListSatellites
	groundstation:RegisterAgent
	groundstation:ReserveContact
	groundstation:UpdateAgentStatus
	groundstation:UpdateConfig
	groundstation:UpdateEphemeris
	groundstation:UpdateMissionProfile

服務前綴	動作
guardduty	guardduty:AcceptAdministratorInvitation guardduty:AcceptInvitation guardduty:ArchiveFindings guardduty:CreateDetector guardduty:CreateFilter guardduty:CreateIPSet guardduty：CreateInvestigation guardduty:CreateMalwareProtectionPlan guardduty:CreateMembers guardduty:CreatePublishingDestination guardduty:CreateSampleFindings guardduty:CreateThreatEntitySet guardduty:CreateThreatIntelSet guardduty:CreateTrustedEntitySet guardduty:DeclineInvitations guardduty:DeleteDetector guardduty:DeleteFilter guardduty:DeleteIPSet guardduty:DeleteInvitations guardduty:DeleteMalwareProtectionPlan guardduty:DeleteMembers

服務前綴	動作
	guardduty:DeletePublishingDestination guardduty:DeleteThreatEntitySet guardduty:DeleteThreatIntelSet guardduty:DeleteTrustedEntitySet guardduty:DescribeMalwareScans guardduty:DescribeOrganizationConfiguration guardduty:DescribePublishingDestination guardduty:DisableOrganizationAdminAccount guardduty:DisassociateFromAdministratorAccount guardduty:DisassociateFromMasterAccount guardduty:DisassociateMembers guardduty:EnableOrganizationAdminAccount guardduty:GetAdministratorAccount guardduty:GetCoverageStatistics guardduty:GetDetector guardduty:GetFilter guardduty:GetFindings guardduty:GetFindingsStatistics guardduty:GetIPSet guardduty：GetInvestigation guardduty:GetInvitationsCount

服務前綴	動作
	guardduty:GetMalwareProtectionPlan guardduty：GetMalwareScan guardduty:GetMalwareScanSettings guardduty:GetMasterAccount guardduty:GetMemberDetectors guardduty:GetMembers guardduty:GetOrganizationStatistics guardduty:GetRemainingFreeTrialDays guardduty:GetThreatEntitySet guardduty:GetThreatIntelSet guardduty:GetTrustedEntitySet guardduty:GetUsageStatistics guardduty:InviteMembers guardduty:ListCoverage guardduty:ListDetectors guardduty:ListFilters guardduty:ListFindings guardduty:ListIPSets guardduty：ListInvestigations guardduty:ListInvitations guardduty:ListMalwareProtectionPlans

服務前綴	動作
	guardduty：ListMalwareScans guardduty:ListMembers guardduty:ListOrganizationAdminAccounts guardduty:ListPublishingDestinations guardduty:ListThreatEntitySets guardduty:ListThreatIntelSets guardduty:ListTrustedEntitySets guardduty:StartMalwareScan guardduty:StartMonitoringMembers guardduty:StopMonitoringMembers guardduty:UnarchiveFindings guardduty:UpdateDetector guardduty:UpdateFilter guardduty:UpdateFindingsFeedback guardduty:UpdateIPSet guardduty:UpdateMalwareProtectionPlan guardduty:UpdateMalwareScanSettings guardduty:UpdateMemberDetectors guardduty:UpdateOrganizationConfiguration guardduty:UpdatePublishingDestination guardduty:UpdateThreatEntitySet

服務前綴	動作
	guardduty:UpdateThreatIntelSet guardduty:UpdateTrustedEntitySet

服務前綴	動作
healthlake	healthlake:CancelFHIRExportJobWithDelete healthlake：CreateDataTransformationProfile healthlake:CreateFHIRDatastore healthlake:CreateResource healthlake：DeleteDataTransformationProfile healthlake:DeleteFHIRDatastore healthlake:DeleteResource healthlake：DescribeDataTransformationJob healthlake:DescribeFHIRDatastore healthlake:DescribeFHIRExportJob healthlake:DescribeFHIRExportJobWithGet healthlake:DescribeFHIRImportJob healthlake:GetCapabilities healthlake：GetDataTransformationProfile healthlake：ListDataTransformationJobs healthlake：ListDataTransformationProfileVersions healthlake：ListDataTransformationProfiles healthlake:ListFHIRDatastores healthlake:ListFHIRExportJobs healthlake:ListFHIRImportJobs healthlake：PublishDataTransformationProfile

服務前綴	動作
	healthlake:ReadResource
	healthlake:SearchEverything
	healthlake:SearchWithGet
	healthlake:SearchWithPost
	healthlake:StartFHIRExportJob
	healthlake:StartFHIRExportJobWithPost
	healthlake:StartFHIRImportJob
	healthlake：UpdateDataTransformationProfile
	healthlake：UpdateFHIRDatastore
	healthlake：UpdateProfileWithAgent
	healthlake:UpdateResource

服務前綴	動作
honeycode	honeycode:BatchCreateTableRows honeycode:BatchDeleteTableRows honeycode:BatchUpdateTableRows honeycode:BatchUpsertTableRows honeycode:DescribeTableDataImportJob honeycode:GetScreenData honeycode:InvokeScreenAutomation honeycode:ListTableColumns honeycode:ListTableRows honeycode:ListTables honeycode:QueryTableRows honeycode:StartTableDataImportJob

服務前綴	動作
iam	iam:AddClientIDToOpenIDConnectProvider
	iam:AddRoleToInstanceProfile
	iam:AddUserToGroup
	iam:AttachGroupPolicy
	iam:AttachRolePolicy
	iam:AttachUserPolicy
	iam:ChangePassword
	iam:CreateAccessKey
	iam:CreateAccountAlias
	iam:CreateGroup
	iam:CreateInstanceProfile
	iam:CreateLoginProfile
	iam:CreateOpenIDConnectProvider
	iam:CreatePolicy
	iam:CreatePolicyVersion
	iam:CreateRole
	iam:CreateSAMLProvider
	iam：CreateServiceLinkedRole
	iam:CreateServiceSpecificCredential
	iam:CreateUser
	iam:CreateVirtualMFADevice

服務前綴	動作
	iam:DeactivateMFADevice
	iam:DeleteAccessKey
	iam:DeleteAccountAlias
	iam:DeleteAccountPasswordPolicy
	iam:DeleteCloudFrontPublicKey
	iam:DeleteGroup
	iam:DeleteGroupPolicy
	iam:DeleteInstanceProfile
	iam:DeleteLoginProfile
	iam:DeleteOpenIDConnectProvider
	iam:DeletePolicy
	iam:DeletePolicyVersion
	iam:DeleteRole
	iam:DeleteRolePermissionsBoundary
	iam:DeleteRolePolicy
	iam:DeleteSAMLProvider
	iam:DeleteSSHPublicKey
	iam:DeleteServerCertificate
	iam:DeleteServiceLinkedRole
	iam:DeleteServiceSpecificCredential
	iam:DeleteSigningCertificate

服務前綴	動作
	iam:DeleteUser
	iam:DeleteUserPermissionsBoundary
	iam:DeleteUserPolicy
	iam:DeleteVirtualMFADevice
	iam:DetachGroupPolicy
	iam:DetachRolePolicy
	iam:DetachUserPolicy
	iam:DisableOrganizationsRootCredentialsManagement
	iam:DisableOrganizationsRootSessions
	iam：DisableOutboundWebIdentityFederation
	iam:EnableMFADevice
	iam:EnableOrganizationsRootCredentialsManagement
	iam:EnableOrganizationsRootSessions
	iam：EnableOutboundWebIdentityFederation
	iam:GenerateCredentialReport
	iam:GenerateOrganizationsAccessReport
	iam:GenerateServiceLastAccessedDetails
	iam:GetAccessKeyLastUsed
	iam:GetAccountAuthorizationDetails
	iam:GetAccountEmailAddress
	iam:GetAccountName

服務前綴	動作
	iam:GetAccountPasswordPolicy
	iam:GetAccountSummary
	iam:GetCloudFrontPublicKey
	iam:GetContextKeysForCustomPolicy
	iam:GetContextKeysForPrincipalPolicy
	iam:GetCredentialReport
	iam:GetGroup
	iam:GetGroupPolicy
	iam:GetInstanceProfile
	iam:GetLoginProfile
	iam:GetMFADevice
	iam:GetOpenIDConnectProvider
	iam:GetOrganizationsAccessReport
	iam：GetOutboundWebIdentityFederationInfo
	iam:GetPolicy
	iam:GetPolicyVersion
	iam:GetRole
	iam:GetRolePolicy
	iam:GetSAMLProvider
	iam:GetSSHPublicKey
	iam:GetServerCertificate

服務前綴	動作
	iam:GetServiceLastAccessedDetails
	iam:GetServiceLastAccessedDetailsWithEntities
	iam:GetServiceLinkedRoleDeletionStatus
	iam:GetUser
	iam:GetUserPolicy
	iam:ListAccessKeys
	iam:ListAccountAliases
	iam:ListAttachedGroupPolicies
	iam:ListAttachedRolePolicies
	iam:ListAttachedUserPolicies
	iam:ListCloudFrontPublicKeys
	iam：ListDelegationRequests
	iam:ListEntitiesForPolicy
	iam:ListGroupPolicies
	iam:ListGroups
	iam:ListGroupsForUser
	iam:ListInstanceProfiles
	iam:ListInstanceProfilesForRole
	iam:ListMFADevices
	iam:ListOpenIDConnectProviders
	iam:ListOrganizationsFeatures

服務前綴	動作
	iam:ListPolicies
	iam:ListPoliciesGrantingServiceAccess
	iam:ListPolicyVersions
	iam:ListRolePolicies
	iam:ListRoles
	iam:ListSAMLProviders
	iam:ListSSHPublicKeys
	iam:ListSTSRegionalEndpointsStatus
	iam:ListServerCertificates
	iam:ListServiceSpecificCredentials
	iam:ListSigningCertificates
	iam:ListUserPolicies
	iam:ListUsers
	iam:ListVirtualMFADevices
	iam:PutGroupPolicy
	iam:PutRolePermissionsBoundary
	iam：PutRolePolicy
	iam:PutUserPermissionsBoundary
	iam:PutUserPolicy
	iam:RemoveClientIDFromOpenIDConnectProvider
	iam:RemoveRoleFromInstanceProfile

服務前綴	動作
	iam:RemoveUserFromGroup
	iam:ResetServiceSpecificCredential
	iam:ResyncMFADevice
	iam:SetDefaultPolicyVersion
	iam:SetSTSRegionalEndpointStatus
	iam:SetSecurityTokenServicePreferences
	iam:SimulateCustomPolicy
	iam:SimulatePrincipalPolicy
	iam:UpdateAccessKey
	iam:UpdateAccountEmailAddress
	iam:UpdateAccountName
	iam:UpdateAccountPasswordPolicy
	iam:UpdateAssumeRolePolicy
	iam:UpdateCloudFrontPublicKey
	iam:UpdateGroup
	iam:UpdateLoginProfile
	iam:UpdateOpenIDConnectProviderThumbprint
	iam:UpdateRole
	iam:UpdateRoleDescription
	iam:UpdateSAMLProvider
	iam:UpdateSSHPublicKey

服務前綴	動作
	iam:UpdateServerCertificate
	iam:UpdateServiceSpecificCredential
	iam:UpdateSigningCertificate
	iam:UpdateUser
	iam:UploadCloudFrontPublicKey
	iam:UploadSSHPublicKey
	iam:UploadServerCertificate
	iam:UploadSigningCertificate

服務前綴	動作
identitystore	identitystore:CreateGroup identitystore:CreateGroupMembership identitystore:CreateUser identitystore:DeleteGroup identitystore:DeleteGroupMembership identitystore:DeleteUser identitystore:DescribeGroup identitystore:DescribeGroupMembership identitystore:DescribeUser identitystore:GetGroupId identitystore:GetGroupMembershipId identitystore:GetUserId identitystore:IsMemberInGroups identitystore:ListGroupMemberships identitystore:ListGroupMembershipsForMember identitystore:ListGroups identitystore:ListUsers identitystore:UpdateGroup identitystore:UpdateUser

服務前綴	動作
imagebuilder	imagebuilder:CancelImageCreation imagebuilder:CancelLifecycleExecution imagebuilder:CreateComponent imagebuilder:CreateContainerRecipe imagebuilder:CreateDistributionConfiguration imagebuilder:CreateImage imagebuilder:CreateImagePipeline imagebuilder:CreateImageRecipe imagebuilder:CreateInfrastructureConfiguration imagebuilder:CreateLifecyclePolicy imagebuilder:CreateWorkflow imagebuilder:DeleteComponent imagebuilder:DeleteContainerRecipe imagebuilder:DeleteDistributionConfiguration imagebuilder:DeleteImage imagebuilder:DeleteImagePipeline imagebuilder:DeleteImageRecipe imagebuilder:DeleteInfrastructureConfiguration imagebuilder:DeleteLifecyclePolicy imagebuilder:DeleteWorkflow imagebuilder：DistributeImage

服務前綴	動作
	imagebuilder:GetComponentPolicy
	imagebuilder:GetContainerRecipePolicy
	imagebuilder:GetImagePolicy
	imagebuilder:GetImageRecipePolicy
	imagebuilder:GetLifecycleExecution
	imagebuilder:GetLifecyclePolicy
	imagebuilder:GetMarketplaceResource
	imagebuilder:GetWorkflowExecution
	imagebuilder:GetWorkflowStepExecution
	imagebuilder:ImportComponent
	imagebuilder:ImportDiskImage
	imagebuilder:ImportVmImage
	imagebuilder:ListComponentBuildVersions
	imagebuilder:ListComponents
	imagebuilder:ListContainerRecipes
	imagebuilder:ListDistributionConfigurations
	imagebuilder:ListImageBuildVersions
	imagebuilder:ListImagePackages
	imagebuilder:ListImagePipelineImages
	imagebuilder:ListImagePipelines
	imagebuilder:ListImageRecipes

服務前綴	動作
	imagebuilder:ListImageScanFindingAggregations
	imagebuilder:ListImageScanFindings
	imagebuilder:ListImages
	imagebuilder:ListInfrastructureConfigurations
	imagebuilder:ListLifecycleExecutionResources
	imagebuilder:ListLifecycleExecutions
	imagebuilder:ListLifecyclePolicies
	imagebuilder:ListWaitingWorkflowSteps
	imagebuilder:ListWorkflowExecutions
	imagebuilder:ListWorkflowStepExecutions
	imagebuilder:ListWorkflows
	imagebuilder:PutComponentPolicy
	imagebuilder:PutContainerRecipePolicy
	imagebuilder:PutImagePolicy
	imagebuilder:PutImageRecipePolicy
	imagebuilder：RetryImage
	imagebuilder:SendWorkflowStepAction
	imagebuilder:StartImagePipelineExecution
	imagebuilder:StartResourceStateUpdate
	imagebuilder:UpdateDistributionConfiguration
	imagebuilder:UpdateImagePipeline

服務前綴	動作
	imagebuilder:UpdateInfrastructureConfiguration

服務前綴	動作
inspector	inspector:AddAttributesToFindings inspector:CreateAssessmentTarget inspector:CreateAssessmentTemplate inspector:CreateExclusionsPreview inspector:CreateResourceGroup inspector:DeleteAssessmentRun inspector:DeleteAssessmentTarget inspector:DeleteAssessmentTemplate inspector:DescribeAssessmentRuns inspector:DescribeAssessmentTargets inspector:DescribeAssessmentTemplates inspector:DescribeCrossAccountAccessRole inspector:DescribeExclusions inspector:DescribeFindings inspector:DescribeResourceGroups inspector:DescribeRulesPackages inspector:GetAssessmentReport inspector:GetExclusionsPreview inspector:GetTelemetryMetadata inspector:ListAssessmentRunAgents inspector:ListAssessmentRuns

服務前綴	動作
	inspector:ListAssessmentTargets inspector:ListAssessmentTemplates inspector:ListEventSubscriptions inspector:ListExclusions inspector:ListFindings inspector:ListRulesPackages inspector:PreviewAgents inspector:RegisterCrossAccountAccessRole inspector:RemoveAttributesFromFindings inspector:StartAssessmentRun inspector:StopAssessmentRun inspector:SubscribeToEvent inspector:UnsubscribeFromEvent inspector:UpdateAssessmentTarget

服務前綴	動作
inspector2	inspector2:AssociateMember
	inspector2:BatchGetAccountStatus
	inspector2:BatchGetCodeSnippet
	inspector2:BatchGetFindingDetails
	inspector2:BatchGetFreeTrialInfo
	inspector2:BatchGetMemberEc2DeepInspectionStatus
	inspector2:BatchUpdateMemberEc2DeepInspectionStatus
	inspector2:CancelFindingsReport
	inspector2:CancelSbomExport
	inspector2:CreateCisScanConfiguration
	inspector2:CreateCodeSecurityIntegration
	inspector2：CreateConnector
	inspector2:CreateFilter
	inspector2:CreateFindingsReport
	inspector2:CreateSbomExport
	inspector2:DeleteCisScanConfiguration
	inspector2:DeleteCodeSecurityIntegration
	inspector2：DeleteConnector
	inspector2:DeleteFilter
	inspector2:DescribeOrganizationConfiguration
	inspector2:Disable

服務前綴	動作
	inspector2:DisableDelegatedAdminAccount
	inspector2:DisassociateMember
	inspector2:Enable
	inspector2:EnableDelegatedAdminAccount
	inspector2:GetCisScanReport
	inspector2:GetCisScanResultDetails
	inspector2:GetClustersForImage
	inspector2:GetCodeSecurityIntegration
	inspector2:GetCodeSecurityScan
	inspector2:GetConfiguration
	inspector2:GetDelegatedAdminAccount
	inspector2:GetEc2DeepInspectionConfiguration
	inspector2:GetEncryptionKey
	inspector2:GetFindingsReportStatus
	inspector2:GetMember
	inspector2:GetSbomExport
	inspector2:ListAccountPermissions
	inspector2:ListCisScanConfigurations
	inspector2:ListCisScanResultsAggregatedByChecks
	inspector2:ListCisScanResultsAggregatedByTargetResource
	inspector2:ListCisScans

服務前綴	動作
	inspector2:ListCodeSecurityIntegrations
	inspector2:ListCodeSecurityScanConfigurations
	inspector2：ListConnectorScanConfigurations
	inspector2：ListConnectors
	inspector2:ListCoverage
	inspector2:ListCoverageStatistics
	inspector2:ListDelegatedAdminAccounts
	inspector2:ListFilters
	inspector2:ListFindingAggregations
	inspector2:ListFindings
	inspector2:ListMembers
	inspector2:ListUsageTotals
	inspector2:ResetEncryptionKey
	inspector2:SearchVulnerabilities
	inspector2:SendCisSessionHealth
	inspector2:SendCisSessionTelemetry
	inspector2:StartCisSession
	inspector2:StartCodeSecurityScan
	inspector2:StopCisSession
	inspector2:UpdateCisScanConfiguration
	inspector2:UpdateCodeSecurityIntegration

服務前綴	動作
	inspector2:UpdateConfiguration inspector2：UpdateConnector inspector2:UpdateEc2DeepInspectionConfiguration inspector2:UpdateEncryptionKey inspector2:UpdateFilter inspector2:UpdateOrgEc2DeepInspectionConfiguration inspector2:UpdateOrganizationConfiguration

服務前綴	動作
iot	iot:AcceptCertificateTransfer iot:AddThingToBillingGroup iot:AddThingToThingGroup iot:AssociateSbomWithPackageVersion iot:AssociateTargetsWithJob iot:AttachPolicy iot:AttachPrincipalPolicy iot:AttachSecurityProfile iot:AttachThingPrincipal iot:CancelAuditMitigationActionsTask iot:CancelAuditTask iot:CancelCertificateTransfer iot:CancelDetectMitigationActionsTask iot:CancelJob iot:CancelJobExecution iot:ClearDefaultAuthorizer iot:ConfirmTopicRuleDestination iot:CreateAuditSuppression iot:CreateAuthorizer iot:CreateBillingGroup iot:CreateCertificateFromCsr

服務前綴	動作
	iot:CreateCertificateProvider iot:CreateCommand iot:CreateCustomMetric iot:CreateDimension iot:CreateDomainConfiguration iot:CreateDynamicThingGroup iot:CreateFleetMetric iot:CreateJob iot:CreateJobTemplate iot:CreateKeysAndCertificate iot:CreateMitigationAction iot:CreateOTAUpdate iot:CreatePackage iot:CreatePackageVersion iot:CreatePolicy iot:CreatePolicyVersion iot:CreateProvisioningClaim iot:CreateProvisioningTemplate iot:CreateProvisioningTemplateVersion iot:CreateRoleAlias iot:CreateScheduledAudit

服務前綴	動作
	iot:CreateSecurityProfile iot:CreateStream iot:CreateThing iot:CreateThingGroup iot:CreateThingType iot:CreateTopicRule iot:CreateTopicRuleDestination iot:DeleteAccountAuditConfiguration iot:DeleteAuditSuppression iot:DeleteAuthorizer iot:DeleteBillingGroup iot:DeleteCACertificate iot:DeleteCertificate iot:DeleteCertificateProvider iot:DeleteCommand iot:DeleteCustomMetric iot:DeleteDimension iot:DeleteDomainConfiguration iot:DeleteDynamicThingGroup iot:DeleteFleetMetric iot:DeleteJob

服務前綴	動作
	iot:DeleteJobExecution iot:DeleteJobTemplate iot:DeleteMitigationAction iot:DeleteOTAUpdate iot:DeletePackage iot:DeletePackageVersion iot:DeletePolicy iot:DeletePolicyVersion iot:DeleteProvisioningTemplate iot:DeleteProvisioningTemplateVersion iot:DeleteRegistrationCode iot:DeleteRoleAlias iot:DeleteScheduledAudit iot:DeleteSecurityProfile iot:DeleteStream iot:DeleteThing iot:DeleteThingGroup iot:DeleteThingType iot:DeleteTopicRule iot:DeleteTopicRuleDestination iot:DeleteV2LoggingLevel

服務前綴	動作
	iot:DeprecateThingType iot:DescribeAccountAuditConfiguration iot:DescribeAuditFinding iot:DescribeAuditMitigationActionsTask iot:DescribeAuditSuppression iot:DescribeAuditTask iot:DescribeAuthorizer iot:DescribeBillingGroup iot:DescribeCACertificate iot:DescribeCertificate iot:DescribeCertificateProvider iot:DescribeCustomMetric iot:DescribeDefaultAuthorizer iot:DescribeDetectMitigationActionsTask iot:DescribeDimension iot:DescribeDomainConfiguration iot:DescribeEncryptionConfiguration iot:DescribeEndpoint iot:DescribeEventConfigurations iot:DescribeFleetMetric iot:DescribeIndex

服務前綴	動作
	iot:DescribeJob iot:DescribeJobExecution iot:DescribeJobTemplate iot:DescribeManagedJobTemplate iot:DescribeMitigationAction iot:DescribeProvisioningTemplate iot:DescribeProvisioningTemplateVersion iot:DescribeRoleAlias iot:DescribeScheduledAudit iot:DescribeSecurityProfile iot:DescribeStream iot:DescribeThing iot:DescribeThingGroup iot:DescribeThingRegistrationTask iot:DescribeThingType iot:DetachPolicy iot:DetachPrincipalPolicy iot:DetachSecurityProfile iot:DetachThingPrincipal iot:DisableTopicRule iot:DisassociateSbomFromPackageVersion

服務前綴	動作
	iot:EnableTopicRule iot:GetBehaviorModelTrainingSummaries iot:GetBucketsAggregation iot:GetCardinality iot:GetCommand iot:GetEffectivePolicies iot:GetJobDocument iot:GetLoggingOptions iot:GetOTAUpdate iot:GetPackage iot:GetPackageConfiguration iot:GetPackageVersion iot:GetPercentiles iot:GetPolicy iot:GetPolicyVersion iot:GetRegistrationCode iot:GetStatistics iot:GetThingConnectivityData iot:GetTopicRule iot:GetTopicRuleDestination iot:GetV2LoggingOptions

服務前綴	動作
	iot:ListActiveViolations iot:ListAttachedPolicies iot:ListAuditFindings iot:ListAuditMitigationActionsExecutions iot:ListAuditMitigationActionsTasks iot:ListAuditSuppressions iot:ListAuditTasks iot:ListAuthorizers iot:ListBillingGroups iot:ListCACertificates iot:ListCertificateProviders iot:ListCertificates iot:ListCertificatesByCA iot:ListCommands iot:ListCustomMetrics iot:ListDetectMitigationActionsExecutions iot:ListDetectMitigationActionsTasks iot:ListDimensions iot:ListDomainConfigurations iot:ListFleetMetrics iot:ListIndices

服務前綴	動作
	iot:ListJobExecutionsForJob iot:ListJobExecutionsForThing iot:ListJobTemplates iot:ListJobs iot:ListManagedJobTemplates iot:ListMetricValues iot:ListMitigationActions iot:ListOTAUpdates iot:ListOutgoingCertificates iot:ListPackageVersions iot:ListPackages iot:ListPolicies iot:ListPolicyPrincipals iot:ListPolicyVersions iot:ListPrincipalPolicies iot:ListPrincipalThings iot:ListPrincipalThingsV2 iot:ListProvisioningTemplateVersions iot:ListProvisioningTemplates iot:ListRelatedResourcesForAuditFinding iot:ListRoleAliases

服務前綴	動作
	iot:ListSbomValidationResults iot:ListScheduledAudits iot:ListSecurityProfiles iot:ListSecurityProfilesForTarget iot:ListStreams iot:ListTargetsForPolicy iot:ListTargetsForSecurityProfile iot:ListThingGroups iot:ListThingGroupsForThing iot:ListThingPrincipals iot:ListThingPrincipalsV2 iot:ListThingRegistrationTaskReports iot:ListThingRegistrationTasks iot:ListThingTypes iot:ListThings iot:ListThingsInBillingGroup iot:ListThingsInThingGroup iot:ListTopicRuleDestinations iot:ListTopicRules iot:ListV2LoggingLevels iot:ListViolationEvents

服務前綴	動作
	iot:PutVerificationStateOnViolation iot:RegisterCACertificate iot:RegisterCertificate iot:RegisterCertificateWithoutCA iot:RegisterThing iot:RejectCertificateTransfer iot:RemoveThingFromBillingGroup iot:RemoveThingFromThingGroup iot:ReplaceTopicRule iot:SearchIndex iot:SetDefaultAuthorizer iot:SetDefaultPolicyVersion iot:SetLoggingOptions iot:SetV2LoggingLevel iot:SetV2LoggingOptions iot:StartAuditMitigationActionsTask iot:StartDetectMitigationActionsTask iot:StartOnDemandAuditTask iot:StartThingRegistrationTask iot:StopThingRegistrationTask iot:TestAuthorization

服務前綴	動作
	iot:TestInvokeAuthorizer iot:TransferCertificate iot:UpdateAccountAuditConfiguration iot:UpdateAuditSuppression iot:UpdateAuthorizer iot:UpdateBillingGroup iot:UpdateCACertificate iot:UpdateCertificate iot:UpdateCertificateProvider iot:UpdateCommand iot:UpdateCustomMetric iot:UpdateDimension iot:UpdateDomainConfiguration iot:UpdateDynamicThingGroup iot:UpdateEncryptionConfiguration iot:UpdateEventConfigurations iot:UpdateFleetMetric iot:UpdateIndexingConfiguration iot:UpdateJob iot:UpdateMitigationAction iot:UpdatePackage

服務前綴	動作
	iot:UpdatePackageConfiguration iot:UpdatePackageVersion iot:UpdateProvisioningTemplate iot:UpdateRoleAlias iot:UpdateScheduledAudit iot:UpdateSecurityProfile iot:UpdateStream iot:UpdateThing iot:UpdateThingGroup iot:UpdateThingGroupsForThing iot:UpdateThingType iot:UpdateTopicRuleDestination iot:ValidateSecurityProfileBehaviors

服務前綴	動作
iotanalytics	iotanalytics:CancelPipelineReprocessing
	iotanalytics:CreateChannel
	iotanalytics:CreateDataset
	iotanalytics:CreateDatasetContent
	iotanalytics:CreateDatastore
	iotanalytics:CreatePipeline
	iotanalytics:DeleteChannel
	iotanalytics:DeleteDataset
	iotanalytics:DeleteDatasetContent
	iotanalytics:DeleteDatastore
	iotanalytics:DeletePipeline
	iotanalytics:DescribeChannel
	iotanalytics:DescribeDataset
	iotanalytics:DescribeDatastore
	iotanalytics:DescribeLoggingOptions
	iotanalytics:DescribePipeline
	iotanalytics:GetDatasetContent
	iotanalytics:ListChannels
	iotanalytics:ListDatasetContents
	iotanalytics:ListDatasets
	iotanalytics:ListDatastores

服務前綴	動作
	iotanalytics:ListPipelines iotanalytics:PutLoggingOptions iotanalytics:RunPipelineActivity iotanalytics:SampleChannelData iotanalytics:StartPipelineReprocessing iotanalytics:UpdateChannel iotanalytics:UpdateDataset iotanalytics:UpdateDatastore iotanalytics:UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor:CreateSuiteDefinition iotdeviceadvisor:DeleteSuiteDefinition iotdeviceadvisor:GetEndpoint iotdeviceadvisor:GetSuiteDefinition iotdeviceadvisor:GetSuiteRun iotdeviceadvisor:GetSuiteRunReport iotdeviceadvisor:ListSuiteDefinitions iotdeviceadvisor:ListSuiteRuns iotdeviceadvisor:StartSuiteRun iotdeviceadvisor:StopSuiteRun iotdeviceadvisor:UpdateSuiteDefinition

服務前綴	動作
iotevents	iotevents:BatchAcknowledgeAlarm
	iotevents:BatchDeleteDetector
	iotevents:BatchDisableAlarm
	iotevents:BatchEnableAlarm
	iotevents:BatchResetAlarm
	iotevents:BatchSnoozeAlarm
	iotevents:BatchUpdateDetector
	iotevents:CreateAlarmModel
	iotevents:CreateDetectorModel
	iotevents:CreateInput
	iotevents:DeleteAlarmModel
	iotevents:DeleteDetectorModel
	iotevents:DeleteInput
	iotevents:DescribeAlarm
	iotevents:DescribeAlarmModel
	iotevents:DescribeDetector
	iotevents:DescribeDetectorModel
	iotevents:DescribeDetectorModelAnalysis
	iotevents:DescribeInput
	iotevents:DescribeLoggingOptions
	iotevents:GetDetectorModelAnalysisResults

服務前綴	動作
	iotevents:ListAlarmModelVersions iotevents:ListAlarmModels iotevents:ListAlarms iotevents:ListDetectorModelVersions iotevents:ListDetectorModels iotevents:ListDetectors iotevents:ListInputRoutings iotevents:ListInputs iotevents:PutLoggingOptions iotevents:StartDetectorModelAnalysis iotevents:UpdateAlarmModel iotevents:UpdateDetectorModel iotevents:UpdateInput
iotfleethub	iotfleethub:CreateApplication iotfleethub:DeleteApplication iotfleethub:DescribeApplication iotfleethub:ListApplications iotfleethub:UpdateApplication

服務前綴	動作
iotsitewise	iotsitewise:AssociateAssets
	iotsitewise:AssociateTimeSeriesToAssetProperty
	iotsitewise:BatchAssociateProjectAssets
	iotsitewise:BatchDisassociateProjectAssets
	iotsitewise:CreateAccessPolicy
	iotsitewise:CreateAsset
	iotsitewise:CreateAssetModel
	iotsitewise:CreateAssetModelCompositeModel
	iotsitewise:CreateBulkImportJob
	iotsitewise:CreateComputationModel
	iotsitewise:CreateDashboard
	iotsitewise:CreateDataset
	iotsitewise:CreateGateway
	iotsitewise:CreatePortal
	iotsitewise:CreateProject
	iotsitewise:DeleteAccessPolicy
	iotsitewise:DeleteAsset
	iotsitewise:DeleteAssetModel
	iotsitewise:DeleteAssetModelCompositeModel
	iotsitewise:DeleteComputationModel
	iotsitewise:DeleteDashboard

服務前綴	動作
	iotsitewise:DeleteDataset
	iotsitewise:DeleteGateway
	iotsitewise:DeletePortal
	iotsitewise:DeleteProject
	iotsitewise:DeleteTimeSeries
	iotsitewise:DescribeAccessPolicy
	iotsitewise:DescribeAsset
	iotsitewise:DescribeAssetCompositeModel
	iotsitewise:DescribeAssetModel
	iotsitewise:DescribeAssetModelCompositeModel
	iotsitewise:DescribeAssetModelInterfaceRelationship
	iotsitewise:DescribeAssetProperty
	iotsitewise:DescribeBulkImportJob
	iotsitewise:DescribeComputationModel
	iotsitewise:DescribeComputationModelExecutionSummary
	iotsitewise:DescribeDashboard
	iotsitewise:DescribeDataset
	iotsitewise:DescribeDefaultEncryptionConfiguration
	iotsitewise:DescribeExecution
	iotsitewise:DescribeGateway
	iotsitewise:DescribeGatewayCapabilityConfiguration

服務前綴	動作
	iotsitewise:DescribeLoggingOptions
	iotsitewise:DescribePortal
	iotsitewise:DescribeProject
	iotsitewise:DescribeStorageConfiguration
	iotsitewise:DescribeTimeSeries
	iotsitewise:DisassociateAssets
	iotsitewise:DisassociateTimeSeriesFromAssetProperty
	iotsitewise:ExecuteAction
	iotsitewise:ExecuteQuery
	iotsitewise:ListAccessPolicies
	iotsitewise:ListActions
	iotsitewise:ListAssetModelCompositeModels
	iotsitewise:ListAssetModelProperties
	iotsitewise:ListAssetModels
	iotsitewise:ListAssetProperties
	iotsitewise:ListAssetRelationships
	iotsitewise:ListAssets
	iotsitewise:ListAssociatedAssets
	iotsitewise:ListBulkImportJobs
	iotsitewise:ListCompositionRelationships
	iotsitewise:ListComputationModelDataBindingUsages

服務前綴	動作
	iotsitewise:ListComputationModelResolveToResources
	iotsitewise:ListComputationModels
	iotsitewise:ListDashboards
	iotsitewise:ListDatasets
	iotsitewise:ListExecutions
	iotsitewise:ListGateways
	iotsitewise:ListInterfaceRelationships
	iotsitewise:ListPortals
	iotsitewise:ListProjectAssets
	iotsitewise:ListProjects
	iotsitewise:ListTimeSeries
	iotsitewise:PutDefaultEncryptionConfiguration
	iotsitewise:PutLoggingOptions
	iotsitewise:PutStorageConfiguration
	iotsitewise:UpdateAccessPolicy
	iotsitewise:UpdateAsset
	iotsitewise:UpdateAssetModel
	iotsitewise:UpdateAssetModelCompositeModel
	iotsitewise:UpdateAssetProperty
	iotsitewise:UpdateComputationModel
	iotsitewise:UpdateDashboard

服務前綴	動作
	iotsitewise:UpdateDataset iotsitewise:UpdateGateway iotsitewise:UpdateGatewayCapabilityConfiguration iotsitewise:UpdatePortal iotsitewise:UpdateProject

服務前綴	動作
iottwinmaker	iottwinmaker:CancelMetadataTransferJob iottwinmaker:CreateComponentType iottwinmaker:CreateEntity iottwinmaker:CreateMetadataTransferJob iottwinmaker:CreateScene iottwinmaker:CreateSyncJob iottwinmaker:CreateWorkspace iottwinmaker:DeleteComponentType iottwinmaker:DeleteEntity iottwinmaker:DeleteScene iottwinmaker:DeleteSyncJob iottwinmaker:DeleteWorkspace iottwinmaker:ExecuteQuery iottwinmaker:GetMetadataTransferJob iottwinmaker:GetPricingPlan iottwinmaker:GetScene iottwinmaker:GetSyncJob iottwinmaker:ListComponentTypes iottwinmaker:ListComponents iottwinmaker:ListEntities iottwinmaker:ListMetadataTransferJobs

服務前綴	動作
	iottwinmaker:ListProperties iottwinmaker:ListScenes iottwinmaker:ListSyncJobs iottwinmaker:ListSyncResources iottwinmaker:ListWorkspaces iottwinmaker:UpdateComponentType iottwinmaker:UpdateEntity iottwinmaker:UpdatePricingPlan iottwinmaker:UpdateScene iottwinmaker:UpdateWorkspace

服務前綴	動作
iotwireless	iotwireless:AssociateAwsAccountWithPartnerAccount
	iotwireless:AssociateMulticastGroupWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithMulticastGroup
	iotwireless:AssociateWirelessDeviceWithThing
	iotwireless:AssociateWirelessGatewayWithCertificate
	iotwireless:AssociateWirelessGatewayWithThing
	iotwireless:CancelMulticastGroupSession
	iotwireless:CreateDestination
	iotwireless:CreateDeviceProfile
	iotwireless:CreateFuotaTask
	iotwireless:CreateMulticastGroup
	iotwireless:CreateNetworkAnalyzerConfiguration
	iotwireless:CreateServiceProfile
	iotwireless:CreateWirelessDevice
	iotwireless:CreateWirelessGateway
	iotwireless:CreateWirelessGatewayTask
	iotwireless:CreateWirelessGatewayTaskDefinition
	iotwireless:DeleteDestination
	iotwireless:DeleteDeviceProfile
	iotwireless:DeleteFuotaTask

服務前綴	動作
	iotwireless:DeleteMulticastGroup
	iotwireless:DeleteNetworkAnalyzerConfiguration
	iotwireless:DeleteQueuedMessages
	iotwireless:DeleteServiceProfile
	iotwireless:DeleteWirelessDevice
	iotwireless:DeleteWirelessDeviceImportTask
	iotwireless:DeleteWirelessGateway
	iotwireless:DeleteWirelessGatewayTask
	iotwireless:DeleteWirelessGatewayTaskDefinition
	iotwireless:DeregisterWirelessDevice
	iotwireless:DisassociateAwsAccountFromPartnerAccount
	iotwireless:DisassociateMulticastGroupFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromMulticastGroup
	iotwireless:DisassociateWirelessDeviceFromThing
	iotwireless:DisassociateWirelessGatewayFromCertificate
	iotwireless:DisassociateWirelessGatewayFromThing
	iotwireless:GetDestination
	iotwireless:GetDeviceProfile
	iotwireless:GetEventConfigurationByResourceTypes
	iotwireless:GetFuotaTask

服務前綴	動作
	iotwireless:GetLogLevelsByResourceTypes
	iotwireless:GetMetricConfiguration
	iotwireless:GetMetrics
	iotwireless:GetMulticastGroup
	iotwireless:GetMulticastGroupSession
	iotwireless:GetNetworkAnalyzerConfiguration
	iotwireless:GetPartnerAccount
	iotwireless:GetPosition
	iotwireless:GetPositionConfiguration
	iotwireless:GetPositionEstimate
	iotwireless:GetResourceEventConfiguration
	iotwireless:GetResourceLogLevel
	iotwireless:GetResourcePosition
	iotwireless:GetServiceEndpoint
	iotwireless:GetServiceProfile
	iotwireless:GetWirelessDevice
	iotwireless:GetWirelessDeviceImportTask
	iotwireless:GetWirelessDeviceStatistics
	iotwireless:GetWirelessGateway
	iotwireless:GetWirelessGatewayCertificate
	iotwireless:GetWirelessGatewayFirmwareInformation

服務前綴	動作
	iotwireless:GetWirelessGatewayStatistics
	iotwireless:GetWirelessGatewayTask
	iotwireless:GetWirelessGatewayTaskDefinition
	iotwireless:ListDestinations
	iotwireless:ListDeviceProfiles
	iotwireless:ListDevicesForWirelessDeviceImportTask
	iotwireless:ListEventConfigurations
	iotwireless:ListFuotaTasks
	iotwireless:ListMulticastGroups
	iotwireless:ListMulticastGroupsByFuotaTask
	iotwireless:ListNetworkAnalyzerConfigurations
	iotwireless:ListPartnerAccounts
	iotwireless:ListPositionConfigurations
	iotwireless:ListQueuedMessages
	iotwireless:ListServiceProfiles
	iotwireless:ListWirelessDeviceImportTasks
	iotwireless:ListWirelessDevices
	iotwireless:ListWirelessGatewayTaskDefinitions
	iotwireless:ListWirelessGateways
	iotwireless:PutPositionConfiguration
	iotwireless:PutResourceLogLevel

服務前綴	動作
	iotwireless:ResetAllResourceLogLevels
	iotwireless:ResetResourceLogLevel
	iotwireless:SendDataToMulticastGroup
	iotwireless:SendDataToWirelessDevice
	iotwireless:StartBulkAssociateWirelessDeviceWithMulticastGroup
	iotwireless:StartBulkDisassociateWirelessDeviceFromMulticastGroup
	iotwireless:StartFuotaTask
	iotwireless:StartMulticastGroupSession
	iotwireless:StartNetworkAnalyzerStream
	iotwireless:StartSingleWirelessDeviceImportTask
	iotwireless:StartWirelessDeviceImportTask
	iotwireless:TestWirelessDevice
	iotwireless:UpdateDestination
	iotwireless:UpdateEventConfigurationByResourceTypes
	iotwireless:UpdateFuotaTask
	iotwireless:UpdateLogLevelsByResourceTypes
	iotwireless:UpdateMetricConfiguration
	iotwireless:UpdateMulticastGroup
	iotwireless:UpdateNetworkAnalyzerConfiguration
	iotwireless:UpdatePartnerAccount

服務前綴	動作
	iotwireless:UpdatePosition iotwireless:UpdateResourceEventConfiguration iotwireless:UpdateResourcePosition iotwireless:UpdateWirelessDevice iotwireless:UpdateWirelessDeviceImportTask iotwireless:UpdateWirelessGateway

服務前綴	動作
ivs	ivs:BatchGetChannel ivs:BatchGetStreamKey ivs:BatchStartViewerSessionRevocation ivs：CreateAdConfiguration ivs:CreateChannel ivs:CreateEncoderConfiguration ivs:CreateIngestConfiguration ivs:CreateParticipantToken ivs:CreatePlaybackRestrictionPolicy ivs:CreateRecordingConfiguration ivs:CreateStorageConfiguration ivs:CreateStreamKey ivs：DeleteAdConfiguration ivs:DeleteChannel ivs:DeleteEncoderConfiguration ivs:DeleteIngestConfiguration ivs:DeletePlaybackKeyPair ivs:DeletePlaybackRestrictionPolicy ivs:DeletePublicKey ivs:DeleteRecordingConfiguration ivs:DeleteStorageConfiguration

服務前綴	動作
	ivs:DeleteStreamKey ivs:DisconnectParticipant ivs：GetAdConfiguration ivs:GetChannel ivs:GetComposition ivs:GetEncoderConfiguration ivs:GetIngestConfiguration ivs:GetParticipant ivs:GetPlaybackKeyPair ivs:GetPlaybackRestrictionPolicy ivs:GetPublicKey ivs:GetRecordingConfiguration ivs:GetStorageConfiguration ivs:GetStream ivs:GetStreamKey ivs:GetStreamSession ivs:ImportPlaybackKeyPair ivs:ImportPublicKey ivs：InsertAdBreak ivs：ListAdConfigurations ivs:ListChannels

服務前綴	動作
	ivs:ListCompositions ivs:ListEncoderConfigurations ivs:ListIngestConfigurations ivs:ListParticipantEvents ivs:ListParticipantReplicas ivs:ListParticipants ivs:ListPlaybackKeyPairs ivs:ListPlaybackRestrictionPolicies ivs:ListPublicKeys ivs:ListRecordingConfigurations ivs:ListStorageConfigurations ivs:ListStreamKeys ivs:ListStreamSessions ivs:ListStreams ivs:PutMetadata ivs:StartComposition ivs:StartViewerSessionRevocation ivs:StopComposition ivs:StopStream ivs：UpdateAdConfiguration ivs:UpdateChannel

服務前綴	動作
	ivs:UpdateIngestConfiguration ivs:UpdatePlaybackRestrictionPolicy
ivschat	ivschat:CreateChatToken ivschat:CreateLoggingConfiguration ivschat:CreateRoom ivschat:DeleteLoggingConfiguration ivschat:DeleteMessage ivschat:DeleteRoom ivschat:DisconnectUser ivschat:GetLoggingConfiguration ivschat:GetRoom ivschat:ListLoggingConfigurations ivschat:ListRooms ivschat:SendEvent ivschat:UpdateLoggingConfiguration ivschat:UpdateRoom

服務前綴	動作
kafka	kafka:BatchAssociateScramSecret kafka:BatchDisassociateScramSecret kafka:CreateCluster kafka:CreateClusterV2 kafka:CreateConfiguration kafka:CreateReplicator kafka:CreateVpcConnection kafka:DeleteCluster kafka:DeleteClusterPolicy kafka:DeleteConfiguration kafka:DeleteReplicator kafka:DeleteVpcConnection kafka:DescribeCluster kafka:DescribeClusterOperation kafka:DescribeClusterOperationV2 kafka:DescribeClusterV2 kafka:DescribeConfiguration kafka:DescribeConfigurationRevision kafka:DescribeVpcConnection kafka:GetBootstrapBrokers kafka:GetClusterPolicy

服務前綴	動作
	kafka:GetCompatibleKafkaVersions kafka:ListClientVpcConnections kafka:ListClusterOperations kafka:ListClusterOperationsV2 kafka:ListClusters kafka:ListClustersV2 kafka:ListConfigurationRevisions kafka:ListConfigurations kafka:ListKafkaVersions kafka:ListNodes kafka:ListReplicators kafka:ListScramSecrets kafka:ListVpcConnections kafka:PutClusterPolicy kafka:RebootBroker kafka:RejectClientVpcConnection kafka:UpdateBrokerCount kafka:UpdateBrokerStorage kafka:UpdateBrokerType kafka:UpdateClusterConfiguration kafka:UpdateClusterKafkaVersion

服務前綴	動作
	kafka:UpdateConfiguration kafka:UpdateConnectivity kafka:UpdateMonitoring kafka：UpdateRebalancing kafka:UpdateReplicationInfo kafka:UpdateSecurity kafka:UpdateStorage
kafkaconnect	kafkaconnect:CreateConnector kafkaconnect:CreateCustomPlugin kafkaconnect:CreateWorkerConfiguration kafkaconnect:DeleteConnector kafkaconnect:DeleteCustomPlugin kafkaconnect:DeleteWorkerConfiguration kafkaconnect:DescribeConnector kafkaconnect:DescribeCustomPlugin kafkaconnect:DescribeWorkerConfiguration kafkaconnect:ListConnectorOperations kafkaconnect:ListConnectors kafkaconnect:ListCustomPlugins kafkaconnect:ListWorkerConfigurations kafkaconnect:UpdateConnector

服務前綴	動作
kendra	kendra:AssociateEntitiesToExperience kendra:AssociatePersonasToEntities kendra:BatchDeleteDocument kendra:BatchDeleteFeaturedResultsSet kendra:BatchGetDocumentStatus kendra:BatchPutDocument kendra:ClearQuerySuggestions kendra:CreateAccessControlConfiguration kendra:CreateDataSource kendra:CreateExperience kendra:CreateFaq kendra:CreateFeaturedResultsSet kendra:CreateIndex kendra:CreateQuerySuggestionsBlockList kendra:CreateThesaurus kendra:DeleteDataSource kendra:DeleteExperience kendra:DeleteFaq kendra:DeleteIndex kendra:DeletePrincipalMapping kendra:DeleteQuerySuggestionsBlockList

服務前綴	動作
	kendra:DeleteThesaurus kendra:DescribeAccessControlConfiguration kendra:DescribeDataSource kendra:DescribeExperience kendra:DescribeFaq kendra:DescribeFeaturedResultsSet kendra:DescribeIndex kendra:DescribePrincipalMapping kendra:DescribeQuerySuggestionsBlockList kendra:DescribeQuerySuggestionsConfig kendra:DescribeThesaurus kendra:DisassociateEntitiesFromExperience kendra:DisassociatePersonasFromEntities kendra:GetQuerySuggestions kendra:GetSnapshots kendra:ListAccessControlConfigurations kendra:ListDataSourceSyncJobs kendra:ListDataSources kendra:ListEntityPersonas kendra:ListExperienceEntities kendra:ListExperiences

服務前綴	動作
	kendra:ListFaqs kendra:ListFeaturedResultsSets kendra:ListGroupsOlderThanOrderingId kendra:ListIndices kendra:ListQuerySuggestionsBlockLists kendra:ListThesauri kendra:PutPrincipalMapping kendra:Query kendra:Retrieve kendra:StartDataSourceSyncJob kendra:StopDataSourceSyncJob kendra:SubmitFeedback kendra:UpdateDataSource kendra:UpdateExperience kendra:UpdateFeaturedResultsSet kendra:UpdateIndex kendra:UpdateQuerySuggestionsBlockList kendra:UpdateQuerySuggestionsConfig kendra:UpdateThesaurus

服務前綴	動作
kinesis	kinesis:CreateStream kinesis:DecreaseStreamRetentionPeriod kinesis:DeleteStream kinesis:DeregisterStreamConsumer kinesis：DescribeAccountSettings kinesis:DescribeLimits kinesis:DescribeStream kinesis:DescribeStreamConsumer kinesis:DescribeStreamSummary kinesis:DisableEnhancedMonitoring kinesis:EnableEnhancedMonitoring kinesis:IncreaseStreamRetentionPeriod kinesis:ListShards kinesis:ListStreamConsumers kinesis:ListStreams kinesis:MergeShards kinesis:RegisterStreamConsumer kinesis:SplitShard kinesis:StartStreamEncryption kinesis:StopStreamEncryption kinesis：UpdateAccountSettings

服務前綴	動作
	kinesis:UpdateShardCount kinesis:UpdateStreamMode

服務前綴	動作
kinesisanalytics	kinesisanalytics:AddApplicationCloudWatchLoggingOption kinesisanalytics:AddApplicationInput kinesisanalytics:AddApplicationInputProcessingConfiguration kinesisanalytics:AddApplicationOutput kinesisanalytics:AddApplicationReferenceDataSource kinesisanalytics:AddApplicationVpcConfiguration kinesisanalytics:CreateApplication kinesisanalytics:CreateApplicationPresignedUrl kinesisanalytics:CreateApplicationSnapshot kinesisanalytics:DeleteApplication kinesisanalytics:DeleteApplicationCloudWatchLoggingOption kinesisanalytics:DeleteApplicationInputProcessingConfiguration kinesisanalytics:DeleteApplicationOutput kinesisanalytics:DeleteApplicationReferenceDataSource kinesisanalytics:DeleteApplicationSnapshot kinesisanalytics:DeleteApplicationVpcConfiguration kinesisanalytics:DescribeApplication kinesisanalytics:DescribeApplicationOperation kinesisanalytics:DescribeApplicationSnapshot kinesisanalytics:DescribeApplicationVersion kinesisanalytics:DiscoverInputSchema

服務前綴	動作
	kinesisanalytics:ListApplicationOperations kinesisanalytics:ListApplicationSnapshots kinesisanalytics:ListApplicationVersions kinesisanalytics:ListApplications kinesisanalytics:RollbackApplication kinesisanalytics:StartApplication kinesisanalytics:StopApplication kinesisanalytics:UpdateApplication kinesisanalytics:UpdateApplicationMaintenanceConfiguration

服務前綴	動作
kms	kms:CancelKeyDeletion
	kms:ConnectCustomKeyStore
	kms:CreateAlias
	kms:CreateCustomKeyStore
	kms:創建權限
	kms:CreateKey
	kms:解密
	kms:DeleteAlias
	kms:DeleteCustomKeyStore
	kms:DeleteImportedKeyMaterial
	kms:DeriveSharedSecret
	kms:DescribeCustomKeyStores
	kms:描述金鑰
	kms:DisableKey
	kms:DisableKeyRotation
	kms:DisconnectCustomKeyStore
	kms:EnableKey
	kms:EnableKeyRotation
	kms:Encrypt
	kms:產生資料金鑰
	kms:GenerateDataKeyPair

服務前綴	動作
	kms:GenerateDataKeyPairWithoutPlaintext
	kms:GenerateDataKeyWithoutPlaintext
	kms:GenerateMac
	kms:GenerateRandom
	kms:GetKeyLastUsage
	kms:GetKeyPolicy
	kms:GetKeyRotationStatus
	kms:GetParametersForImport
	kms:GetPublicKey
	kms:ImportKeyMaterial
	kms:ListAliases
	kms:ListGrants
	kms:ListKeyPolicies
	kms:ListKeyRotations
	kms:ListKeys
	kms:ListRetirableGrants
	kms:ReplicateKey
	kms:RetireGrant
	kms:RevokeGrant
	kms:RotateKeyOnDemand
	kms:ScheduleKeyDeletion

服務前綴	動作
	kms:Sign kms:UpdateAlias kms:UpdateCustomKeyStore kms:UpdateKeyDescription kms:UpdatePrimaryRegion kms:Verify kms:VerifyMac

服務前綴	動作
lambda	lambda:AddLayerVersionPermission lambda:AddPermission lambda:CreateAlias lambda:CreateCodeSigningConfig lambda:CreateEventSourceMapping lambda:CreateFunction lambda:CreateFunctionUrlConfig lambda:DeleteAlias lambda：DeleteCapacityProvider lambda:DeleteCodeSigningConfig lambda:DeleteEventSourceMapping lambda:DeleteFunction lambda:DeleteFunctionCodeSigningConfig lambda:DeleteFunctionConcurrency lambda:DeleteFunctionEventInvokeConfig lambda:DeleteFunctionUrlConfig lambda:DeleteLayerVersion lambda:DeleteProvisionedConcurrencyConfig lambda:GetAccountSettings lambda:GetAlias lambda：GetCapacityProvider

服務前綴	動作
	lambda:GetCodeSigningConfig lambda:GetEventSourceMapping lambda:GetFunction lambda:GetFunctionCodeSigningConfig lambda:GetFunctionConcurrency lambda:GetFunctionConfiguration lambda:GetFunctionEventInvokeConfig lambda:GetFunctionRecursionConfig lambda：GetFunctionScalingConfig lambda:GetFunctionUrlConfig lambda:GetLayerVersion lambda:GetLayerVersionPolicy lambda:GetPolicy lambda:GetProvisionedConcurrencyConfig lambda:GetRuntimeManagementConfig lambda:ListAliases lambda：ListCapacityProviders lambda:ListCodeSigningConfigs lambda：ListDurableExecutionsByFunction lambda:ListEventSourceMappings lambda:ListFunctionEventInvokeConfigs

服務前綴	動作
	lambda:ListFunctionUrlConfigs lambda:ListFunctions lambda:ListFunctionsByCodeSigningConfig lambda:ListLayerVersions lambda:ListLayers lambda:ListProvisionedConcurrencyConfigs lambda:ListVersionsByFunction lambda:PublishLayerVersion lambda:PublishVersion lambda:PutFunctionCodeSigningConfig lambda:PutFunctionConcurrency lambda:PutFunctionEventInvokeConfig lambda:PutFunctionRecursionConfig lambda：PutFunctionScalingConfig lambda:PutProvisionedConcurrencyConfig lambda:PutRuntimeManagementConfig lambda:RemoveLayerVersionPermission lambda:RemovePermission lambda:UpdateAlias lambda：UpdateCapacityProvider lambda:UpdateCodeSigningConfig

服務前綴	動作
	lambda:UpdateEventSourceMapping lambda:UpdateFunctionCode lambda:UpdateFunctionConfiguration lambda:UpdateFunctionEventInvokeConfig lambda:UpdateFunctionUrlConfig

服務前綴	動作
lex	lex:BatchCreateCustomVocabularyItem
	lex:BatchDeleteCustomVocabularyItem
	lex:BatchUpdateCustomVocabularyItem
	lex:BuildBotLocale
	lex:CreateBotAlias
	lex:CreateBotReplica
	lex:CreateBotVersion
	lex:CreateExport
	lex:CreateIntentVersion
	lex:CreateResourcePolicy
	lex:CreateSlotTypeVersion
	lex:CreateTestSetDiscrepancyReport
	lex:CreateUploadUrl
	lex:DeleteBot
	lex：DeleteBotAnalyzerRecommendation
	lex:DeleteBotChannelAssociation
	lex:DeleteBotReplica
	lex:DeleteExport
	lex:DeleteImport
	lex:DeleteIntentVersion
	lex:DeleteResourcePolicy

服務前綴	動作
	lex:DeleteSlotTypeVersion
	lex:DeleteTestSet
	lex:DeleteUtterances
	lex:DescribeBotAlias
	lex：DescribeBotAnalyzerRecommendation
	lex:DescribeBotRecommendation
	lex:DescribeBotReplica
	lex:DescribeBotResourceGeneration
	lex:DescribeBotVersion
	lex:DescribeCustomVocabularyMetadata
	lex:DescribeExport
	lex:DescribeImport
	lex:DescribeResourcePolicy
	lex:DescribeTestExecution
	lex:DescribeTestSet
	lex:DescribeTestSetDiscrepancyReport
	lex:DescribeTestSetGeneration
	lex:GenerateBotElement
	lex:GetBot
	lex:GetBotAlias
	lex:GetBotAliases

服務前綴	動作
	lex:GetBotChannelAssociation
	lex:GetBotChannelAssociations
	lex:GetBotVersions
	lex:GetBots
	lex:GetBuiltinIntent
	lex:GetBuiltinIntents
	lex:GetBuiltinSlotTypes
	lex:GetExport
	lex:GetImport
	lex:GetIntent
	lex:GetIntentVersions
	lex:GetIntents
	lex:GetMigration
	lex:GetMigrations
	lex:GetSlotType
	lex:GetSlotTypeVersions
	lex:GetSlotTypes
	lex:GetTestExecutionArtifactsUrl
	lex:GetUtterancesView
	lex:ListBotAliasReplicas
	lex:ListBotAliases

服務前綴	動作
	lex:ListBotRecommendations
	lex:ListBotReplicas
	lex:ListBotResourceGenerations
	lex:ListBotVersionReplicas
	lex:ListBotVersions
	lex:ListBots
	lex:ListBuiltInIntents
	lex:ListBuiltInSlotTypes
	lex:ListCustomVocabularyItems
	lex:ListExports
	lex:ListImports
	lex:ListIntentMetrics
	lex:ListIntentPaths
	lex:ListRecommendedIntents
	lex:ListSessionAnalyticsData
	lex:ListSessionMetrics
	lex:ListTestExecutionResultItems
	lex:ListTestExecutions
	lex:ListTestSets
	lex:PutBot
	lex:PutBotAlias

服務前綴	動作
	lex:PutIntent
	lex:PutSlotType
	lex:SearchAssociatedTranscripts
	lex：StartBotAnalyzer
	lex:StartBotRecommendation
	lex:StartImport
	lex:StartMigration
	lex:StartTestExecution
	lex:StartTestSetGeneration
	lex：StopBotAnalyzer
	lex:StopBotRecommendation
	lex:UpdateBotAlias
	lex:UpdateBotRecommendation
	lex:UpdateExport
	lex:UpdateResourcePolicy

服務前綴	動作
license-manager-linux-subscriptions	license-manager-linux-subscriptions:DeregisterSubscriptionProvider license-manager-linux-subscriptions:GetRegisteredSubscriptionProvider license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions:ListRegisteredSubscriptionProviders license-manager-linux-subscriptions:RegisterSubscriptionProvider license-manager-linux-subscriptions:UpdateServiceSettings

服務前綴	動作
lightsail	lightsail:AllocateStaticIp
	lightsail:AttachCertificateToDistribution
	lightsail:AttachDisk
	lightsail:AttachInstancesToLoadBalancer
	lightsail:AttachLoadBalancerTlsCertificate
	lightsail:AttachStaticIp
	lightsail:CloseInstancePublicPorts
	lightsail:CopySnapshot
	lightsail:CreateBucket
	lightsail:CreateBucketAccessKey
	lightsail:CreateCertificate
	lightsail:CreateCloudFormationStack
	lightsail:CreateContactMethod
	lightsail:CreateContainerService
	lightsail:CreateContainerServiceDeployment
	lightsail:CreateContainerServiceRegistryLogin
	lightsail:CreateDisk
	lightsail:CreateDiskFromSnapshot
	lightsail:CreateDiskSnapshot
	lightsail:CreateDistribution
	lightsail:CreateDomain

服務前綴	動作
	lightsail:CreateGUISessionAccessDetails
	lightsail:CreateInstanceSnapshot
	lightsail:CreateInstances
	lightsail:CreateInstancesFromSnapshot
	lightsail:CreateKeyPair
	lightsail:CreateLoadBalancer
	lightsail:CreateLoadBalancerTlsCertificate
	lightsail:CreateRelationalDatabase
	lightsail:CreateRelationalDatabaseFromSnapshot
	lightsail:CreateRelationalDatabaseSnapshot
	lightsail:DeleteAlarm
	lightsail:DeleteAutoSnapshot
	lightsail:DeleteBucket
	lightsail:DeleteBucketAccessKey
	lightsail:DeleteCertificate
	lightsail:DeleteContactMethod
	lightsail:DeleteContainerImage
	lightsail:DeleteContainerService
	lightsail:DeleteDisk
	lightsail:DeleteDiskSnapshot
	lightsail:DeleteDistribution

服務前綴	動作
	lightsail:DeleteDomain
	lightsail:DeleteDomainEntry
	lightsail:DeleteInstance
	lightsail:DeleteInstanceSnapshot
	lightsail:DeleteKeyPair
	lightsail:DeleteKnownHostKeys
	lightsail:DeleteLoadBalancer
	lightsail:DeleteLoadBalancerTlsCertificate
	lightsail:DeleteRelationalDatabase
	lightsail:DeleteRelationalDatabaseSnapshot
	lightsail:DetachCertificateFromDistribution
	lightsail:DetachDisk
	lightsail:DetachInstancesFromLoadBalancer
	lightsail:DetachStaticIp
	lightsail:DisableAddOn
	lightsail:DownloadDefaultKeyPair
	lightsail:EnableAddOn
	lightsail:ExportSnapshot
	lightsail:GetActiveNames
	lightsail:GetAlarms
	lightsail:GetAutoSnapshots

服務前綴	動作
	lightsail:GetBlueprints
	lightsail:GetBucketAccessKeys
	lightsail:GetBucketBundles
	lightsail:GetBucketMetricData
	lightsail:GetBuckets
	lightsail:GetBundles
	lightsail:GetCertificates
	lightsail:GetCloudFormationStackRecords
	lightsail:GetContactMethods
	lightsail:GetContainerAPIMetadata
	lightsail:GetContainerImages
	lightsail:GetContainerLog
	lightsail:GetContainerServiceDeployments
	lightsail:GetContainerServiceMetricData
	lightsail:GetContainerServicePowers
	lightsail:GetContainerServices
	lightsail:GetCostEstimate
	lightsail:GetDisk
	lightsail:GetDiskSnapshot
	lightsail:GetDiskSnapshots
	lightsail:GetDisks

服務前綴	動作
	lightsail:GetDistributionBundles
	lightsail:GetDistributionLatestCacheReset
	lightsail:GetDistributionMetricData
	lightsail:GetDistributions
	lightsail:GetDomain
	lightsail:GetExportSnapshotRecords
	lightsail:GetInstance
	lightsail:GetInstanceMetricData
	lightsail:GetInstancePortStates
	lightsail:GetInstanceSnapshot
	lightsail:GetInstanceSnapshots
	lightsail:GetInstanceState
	lightsail:GetInstances
	lightsail:GetKeyPair
	lightsail:GetKeyPairs
	lightsail:GetLoadBalancer
	lightsail:GetLoadBalancerMetricData
	lightsail:GetLoadBalancerTlsCertificates
	lightsail:GetLoadBalancerTlsPolicies
	lightsail:GetLoadBalancers
	lightsail:GetOperation

服務前綴	動作
	lightsail:GetOperations
	lightsail:GetOperationsForResource
	lightsail:GetRegions
	lightsail:GetRelationalDatabase
	lightsail:GetRelationalDatabaseBlueprints
	lightsail:GetRelationalDatabaseBundles
	lightsail:GetRelationalDatabaseEvents
	lightsail:GetRelationalDatabaseLogEvents
	lightsail:GetRelationalDatabaseLogStreams
	lightsail:GetRelationalDatabaseMasterUserPassword
	lightsail:GetRelationalDatabaseMetricData
	lightsail:GetRelationalDatabaseParameters
	lightsail:GetRelationalDatabaseSnapshot
	lightsail:GetRelationalDatabaseSnapshots
	lightsail:GetRelationalDatabases
	lightsail:GetSetupHistory
	lightsail:GetStaticIp
	lightsail:GetStaticIps
	lightsail:ImportKeyPair
	lightsail:IsVpcPeered
	lightsail:OpenInstancePublicPorts

服務前綴	動作
	lightsail:PeerVpc
	lightsail:PutAlarm
	lightsail:PutInstancePublicPorts
	lightsail:RebootInstance
	lightsail:RebootRelationalDatabase
	lightsail:RegisterContainerImage
	lightsail:ReleaseStaticIp
	lightsail:ResetDistributionCache
	lightsail:SendContactMethodVerification
	lightsail:SetIpAddressType
	lightsail:SetResourceAccessForBucket
	lightsail:SetupInstanceHttps
	lightsail:StartGUISession
	lightsail:StartInstance
	lightsail:StartRelationalDatabase
	lightsail:StopGUISession
	lightsail:StopInstance
	lightsail:StopRelationalDatabase
	lightsail:TestAlarm
	lightsail:UnpeerVpc
	lightsail:UpdateBucket

服務前綴	動作
	lightsail:UpdateBucketBundle lightsail:UpdateContainerService lightsail:UpdateDistribution lightsail:UpdateDistributionBundle lightsail:UpdateDomainEntry lightsail:UpdateInstanceMetadataOptions lightsail:UpdateLoadBalancerAttribute lightsail:UpdateRelationalDatabase lightsail:UpdateRelationalDatabaseParameters

服務前綴	動作
日誌	logs:AssociateKmsKey log：AssociateSourceToS3TableIntegration logs:CancelExportTask log：CancelImportTask logs:CreateDelivery logs:CreateExportTask logs:CreateLogAnomalyDetector logs:CreateLogGroup logs:CreateLogStream log：CreateLookupTable logs:DeleteDataProtectionPolicy logs:DeleteDelivery logs:DeleteDeliveryDestination logs:DeleteDeliveryDestinationPolicy logs:DeleteDeliverySource logs:DeleteDestination logs:DeleteIndexPolicy logs:DeleteIntegration logs:DeleteLogAnomalyDetector logs:DeleteLogGroup logs:DeleteLogStream

服務前綴	動作
	log：DeleteLookupTable logs:DeleteMetricFilter logs:DeleteQueryDefinition logs:DeleteResourcePolicy logs:DeleteRetentionPolicy log：DeleteScheduledQuery logs:DeleteSubscriptionFilter log：DeleteSyslogConfiguration logs:DeleteTransformer logs:DescribeAccountPolicies logs:DescribeConfigurationTemplates logs:DescribeDeliveries logs:DescribeDeliveryDestinations logs:DescribeDeliverySources logs:DescribeDestinations logs:DescribeExportTasks logs:DescribeFieldIndexes log：DescribeImportTaskBatches log：DescribeImportTasks logs:DescribeIndexPolicies logs:DescribeLogGroups

服務前綴	動作
	logs:DescribeLogStreams log：DescribeLookupTables logs:DescribeMetricFilters logs:DescribeQueries logs:DescribeQueryDefinitions logs:DescribeResourcePolicies logs:DescribeSubscriptionFilters logs:DisassociateKmsKey log：DisassociateSourceFromS3TableIntegration logs:GetDataProtectionPolicy logs:GetDelivery logs:GetDeliveryDestination logs:GetDeliveryDestinationPolicy logs:GetDeliverySource logs:GetIntegration logs:GetLogAnomalyDetector log：GetLogFields logs:GetLogGroupFields logs:GetLogRecord log：GetLookupTable logs:GetQueryResults

服務前綴	動作
	log：GetScheduledQuery log：GetScheduledQueryHistory log：GetStorageTierPolicy logs:GetTransformer logs:ListAnomalies logs:ListIntegrations logs:ListLogAnomalyDetectors logs:ListLogGroupsForQuery log：ListScheduledQueries log：ListSourcesForS3TableIntegration log：ListSyslogConfigurations log：PutBearerTokenAuthentication logs:PutDataProtectionPolicy logs:PutDeliveryDestination logs:PutDeliveryDestinationPolicy logs:PutDeliverySource logs:PutDestination logs:PutDestinationPolicy logs:PutIndexPolicy logs:PutIntegration log：PutLogGroupDeletionProtection

服務前綴	動作
	logs:PutMetricFilter logs:PutQueryDefinition logs:PutResourcePolicy logs:PutRetentionPolicy log：PutStorageTierPolicy logs:PutSubscriptionFilter log：PutSyslogConfiguration logs:PutTransformer logs:StartLiveTail logs:StartQuery logs:StopQuery logs:TestMetricFilter logs:TestTransformer logs:UpdateAnomaly logs:UpdateDeliveryConfiguration logs:UpdateLogAnomalyDetector log：UpdateLookupTable

服務前綴	動作
lookoutequipment	lookoutequipment:CreateDataset lookoutequipment:CreateInferenceScheduler lookoutequipment:CreateLabel lookoutequipment:CreateLabelGroup lookoutequipment:CreateModel lookoutequipment:DeleteDataset lookoutequipment:DeleteInferenceScheduler lookoutequipment:DeleteLabel lookoutequipment:DeleteLabelGroup lookoutequipment:DeleteModel lookoutequipment:DeleteResourcePolicy lookoutequipment:DeleteRetrainingScheduler lookoutequipment:DescribeDataIngestionJob lookoutequipment:DescribeDataset lookoutequipment:DescribeInferenceScheduler lookoutequipment:DescribeLabelGroup lookoutequipment:DescribeModel lookoutequipment:DescribeModelVersion lookoutequipment:DescribeResourcePolicy lookoutequipment:DescribeRetrainingScheduler lookoutequipment:Describelabel

服務前綴	動作
	lookoutequipment:ImportDataset
	lookoutequipment:ImportModelVersion
	lookoutequipment:ListDataIngestionJobs
	lookoutequipment:ListDatasets
	lookoutequipment:ListInferenceEvents
	lookoutequipment:ListInferenceExecutions
	lookoutequipment:ListInferenceSchedulers
	lookoutequipment:ListLabelGroups
	lookoutequipment:ListLabels
	lookoutequipment:ListModelVersions
	lookoutequipment:ListModels
	lookoutequipment:ListRetrainingSchedulers
	lookoutequipment:ListSensorStatistics
	lookoutequipment:PutResourcePolicy
	lookoutequipment:StartDataIngestionJob
	lookoutequipment:StartInferenceScheduler
	lookoutequipment:StartRetrainingScheduler
	lookoutequipment:StopInferenceScheduler
	lookoutequipment:StopRetrainingScheduler
	lookoutequipment:UpdateActiveModelVersion
	lookoutequipment:UpdateInferenceScheduler

服務前綴	動作
	lookoutequipment:UpdateLabelGroup lookoutequipment:UpdateModel lookoutequipment:UpdateRetrainingScheduler

服務前綴	動作
lookoutmetrics	lookoutmetrics:ActivateAnomalyDetector lookoutmetrics:BackTestAnomalyDetector lookoutmetrics:CreateAlert lookoutmetrics:CreateAnomalyDetector lookoutmetrics:CreateMetricSet lookoutmetrics:DeactivateAnomalyDetector lookoutmetrics:DeleteAlert lookoutmetrics:DeleteAnomalyDetector lookoutmetrics:DescribeAlert lookoutmetrics:DescribeAnomalyDetectionExecutions lookoutmetrics:DescribeAnomalyDetector lookoutmetrics:DescribeMetricSet lookoutmetrics:DetectMetricSetConfig lookoutmetrics:GetAnomalyGroup lookoutmetrics:GetDataQualityMetrics lookoutmetrics:GetFeedback lookoutmetrics:GetSampleData lookoutmetrics:ListAlerts lookoutmetrics:ListAnomalyDetectors lookoutmetrics:ListAnomalyGroupRelatedMetrics lookoutmetrics:ListAnomalyGroupSummaries

服務前綴	動作
	lookoutmetrics:ListAnomalyGroupTimeSeries lookoutmetrics:ListMetricSets lookoutmetrics:PutFeedback lookoutmetrics:UpdateAlert lookoutmetrics:UpdateAnomalyDetector lookoutmetrics:UpdateMetricSet

服務前綴	動作
m2	m2:CancelBatchJobExecution
	m2:CreateApplication
	m2:CreateDataSetExportTask
	m2:CreateDataSetImportTask
	m2:CreateDeployment
	m2:CreateEnvironment
	m2:DeleteApplication
	m2:DeleteApplicationFromEnvironment
	m2:DeleteEnvironment
	m2:GetApplication
	m2:GetApplicationVersion
	m2:GetBatchJobExecution
	m2:GetDataSetDetails
	m2:GetDataSetExportTask
	m2:GetDataSetImportTask
	m2:GetDeployment
	m2:GetEnvironment
	m2:GetSignedBluinsightsUrl
	m2:ListApplicationVersions
	m2:ListApplications
	m2:ListBatchJobDefinitions

服務前綴	動作
	m2:ListBatchJobExecutions
	m2:ListBatchJobRestartPoints
	m2:ListDataSetExportHistory
	m2:ListDataSetImportHistory
	m2:ListDataSets
	m2:ListDeployments
	m2:ListEngineVersions
	m2:ListEnvironments
	m2:StartApplication
	m2:StartBatchJob
	m2:StopApplication
	m2:UpdateApplication
	m2:UpdateEnvironment

服務前綴	動作
managedblockchain	managedblockchain:CreateAccessor managedblockchain:CreateMember managedblockchain:CreateNetwork managedblockchain:CreateNode managedblockchain:CreateProposal managedblockchain:DeleteAccessor managedblockchain:DeleteMember managedblockchain:DeleteNode managedblockchain:GetAccessor managedblockchain:GetMember managedblockchain:GetNetwork managedblockchain:GetNode managedblockchain:GetProposal managedblockchain:InvokeRpcPolygonMainnet managedblockchain:InvokeRpcPolygonMumbaiTestnet managedblockchain:ListAccessors managedblockchain:ListInvitations managedblockchain:ListMembers managedblockchain:ListNetworks managedblockchain:ListNodes managedblockchain:ListProposalVotes

服務前綴	動作
	managedblockchain:ListProposals managedblockchain:RejectInvitation managedblockchain:UpdateMember managedblockchain:UpdateNode managedblockchain:VoteOnProposal

服務前綴	動作
mediaconnect	mediaconnect:AddBridgeOutputs
	mediaconnect:AddBridgeSources
	mediaconnect:AddFlowMediaStreams
	mediaconnect:AddFlowOutputs
	mediaconnect:AddFlowSources
	mediaconnect:AddFlowVpcInterfaces
	mediaconnect:CreateBridge
	mediaconnect:CreateFlow
	mediaconnect:CreateGateway
	mediaconnect:DeleteBridge
	mediaconnect:DeleteFlow
	mediaconnect:DeleteGateway
	mediaconnect：DeleteRouterInput
	mediaconnect：DeleteRouterNetworkInterface
	mediaconnect：DeleteRouterOutput
	mediaconnect:DeregisterGatewayInstance
	mediaconnect:DescribeBridge
	mediaconnect:DescribeFlow
	mediaconnect:DescribeFlowSourceMetadata
	mediaconnect:DescribeFlowSourceThumbnail
	mediaconnect:DescribeGateway

服務前綴	動作
	mediaconnect:DescribeGatewayInstance
	mediaconnect:DescribeOffering
	mediaconnect:DescribeReservation
	mediaconnect：GetRouterInput
	mediaconnect：GetRouterInputSourceMetadata
	mediaconnect：GetRouterInputThumbnail
	mediaconnect：GetRouterNetworkInterface
	mediaconnect：GetRouterOutput
	mediaconnect:GrantFlowEntitlements
	mediaconnect:ListBridges
	mediaconnect:ListEntitlements
	mediaconnect:ListFlows
	mediaconnect:ListGatewayInstances
	mediaconnect:ListGateways
	mediaconnect:ListOfferings
	mediaconnect:ListReservations
	mediaconnect：ListRouterInputs
	mediaconnect：ListRouterNetworkInterfaces
	mediaconnect：ListRouterOutputs
	mediaconnect:PurchaseOffering
	mediaconnect:RemoveBridgeOutput

服務前綴	動作
	mediaconnect:RemoveBridgeSource
	mediaconnect:RemoveFlowMediaStream
	mediaconnect:RemoveFlowOutput
	mediaconnect:RemoveFlowSource
	mediaconnect:RemoveFlowVpcInterface
	mediaconnect：RestartRouterInput
	mediaconnect：RestartRouterOutput
	mediaconnect:RevokeFlowEntitlement
	mediaconnect:StartFlow
	mediaconnect：StartRouterInput
	mediaconnect：StartRouterOutput
	mediaconnect:StopFlow
	mediaconnect：StopRouterInput
	mediaconnect：StopRouterOutput
	mediaconnect：TakeRouterInput
	mediaconnect:UpdateBridge
	mediaconnect:UpdateBridgeOutput
	mediaconnect:UpdateBridgeSource
	mediaconnect:UpdateBridgeState
	mediaconnect:UpdateFlow
	mediaconnect:UpdateFlowEntitlement

服務前綴	動作
	mediaconnect:UpdateFlowMediaStream mediaconnect:UpdateGatewayInstance

服務前綴	動作
mediaconvert	mediaconvert:AssociateCertificate mediaconvert:CancelJob mediaconvert:CreateJob mediaconvert:CreateJobTemplate mediaconvert:CreatePreset mediaconvert:CreateQueue mediaconvert：CreateResourceShare mediaconvert:DeleteJobTemplate mediaconvert:DeletePolicy mediaconvert:DeletePreset mediaconvert:DeleteQueue mediaconvert:DescribeEndpoints mediaconvert:DisassociateCertificate mediaconvert:GetJob mediaconvert:GetJobTemplate mediaconvert:GetPolicy mediaconvert:GetPreset mediaconvert:GetQueue mediaconvert:ListJobTemplates mediaconvert:ListJobs mediaconvert:ListPresets

服務前綴	動作
	mediaconvert:ListQueues mediaconvert:ListVersions mediaconvert:Probe mediaconvert:PutPolicy mediaconvert:SearchJobs mediaconvert:UpdateJobTemplate mediaconvert:UpdatePreset mediaconvert:UpdateQueue

服務前綴	動作
medialive	medialive:AcceptInputDeviceTransfer
	medialive:BatchDelete
	medialive:BatchStart
	medialive:BatchStop
	medialive:BatchUpdateSchedule
	medialive:CancelInputDeviceTransfer
	medialive:ClaimDevice
	medialive:CreateChannel
	medialive:CreateChannelPlacementGroup
	medialive:CreateCloudWatchAlarmTemplate
	medialive:CreateCloudWatchAlarmTemplateGroup
	medialive:CreateCluster
	medialive:CreateEventBridgeRuleTemplate
	medialive:CreateEventBridgeRuleTemplateGroup
	medialive:CreateInput
	medialive:CreateInputSecurityGroup
	medialive:CreateMultiplex
	medialive:CreateMultiplexProgram
	medialive:CreateNetwork
	medialive:CreateNode
	medialive:CreateNodeRegistrationScript

服務前綴	動作
	medialive:CreatePartnerInput
	medialive:CreateSdiSource
	medialive:CreateSignalMap
	medialive:DeleteChannel
	medialive:DeleteChannelPlacementGroup
	medialive:DeleteCloudWatchAlarmTemplate
	medialive:DeleteCloudWatchAlarmTemplateGroup
	medialive:DeleteCluster
	medialive:DeleteEventBridgeRuleTemplate
	medialive:DeleteEventBridgeRuleTemplateGroup
	medialive:DeleteInput
	medialive:DeleteInputSecurityGroup
	medialive:DeleteMultiplex
	medialive:DeleteMultiplexProgram
	medialive:DeleteNetwork
	medialive:DeleteNode
	medialive:DeleteReservation
	medialive:DeleteSchedule
	medialive:DeleteSdiSource
	medialive:DeleteSignalMap
	medialive:DescribeAccountConfiguration

服務前綴	動作
	medialive:DescribeChannel
	medialive:DescribeChannelPlacementGroup
	medialive:DescribeCluster
	medialive:DescribeInput
	medialive:DescribeInputDevice
	medialive:DescribeInputDeviceThumbnail
	medialive:DescribeInputSecurityGroup
	medialive:DescribeMultiplex
	medialive:DescribeMultiplexProgram
	medialive:DescribeNetwork
	medialive:DescribeNode
	medialive:DescribeOffering
	medialive:DescribeReservation
	medialive:DescribeSchedule
	medialive:DescribeSdiSource
	medialive:DescribeThumbnails
	medialive:GetCloudWatchAlarmTemplate
	medialive:GetCloudWatchAlarmTemplateGroup
	medialive:GetEventBridgeRuleTemplate
	medialive:GetEventBridgeRuleTemplateGroup
	medialive:GetSignalMap

服務前綴	動作
	medialive：ListAlerts
	medialive:ListChannelPlacementGroups
	medialive:ListChannels
	medialive:ListCloudWatchAlarmTemplateGroups
	medialive:ListCloudWatchAlarmTemplates
	medialive：ListClusterAlerts
	medialive:ListClusters
	medialive:ListEventBridgeRuleTemplateGroups
	medialive:ListEventBridgeRuleTemplates
	medialive:ListInputDeviceTransfers
	medialive:ListInputDevices
	medialive:ListInputSecurityGroups
	medialive:ListInputs
	medialive：ListMultiplexAlerts
	medialive:ListMultiplexPrograms
	medialive:ListMultiplexes
	medialive:ListNetworks
	medialive:ListNodes
	medialive:ListOfferings
	medialive:ListReservations
	medialive:ListSdiSources

服務前綴	動作
	medialive:ListSignalMaps
	medialive:ListVersions
	medialive:PurchaseOffering
	medialive:RebootInputDevice
	medialive:RejectInputDeviceTransfer
	medialive:RestartChannelPipelines
	medialive:StartChannel
	medialive:StartDeleteMonitorDeployment
	medialive:StartInputDevice
	medialive:StartInputDeviceMaintenanceWindow
	medialive:StartMonitorDeployment
	medialive:StartMultiplex
	medialive:StartUpdateSignalMap
	medialive:StopChannel
	medialive:StopInputDevice
	medialive:StopMultiplex
	medialive:TransferInputDevice
	medialive:UpdateAccountConfiguration
	medialive:UpdateChannel
	medialive:UpdateChannelClass
	medialive:UpdateChannelPlacementGroup

服務前綴	動作
	medialive:UpdateCloudWatchAlarmTemplate
	medialive:UpdateCloudWatchAlarmTemplateGroup
	medialive:UpdateCluster
	medialive:UpdateEventBridgeRuleTemplate
	medialive:UpdateEventBridgeRuleTemplateGroup
	medialive:UpdateInput
	medialive:UpdateInputDevice
	medialive:UpdateInputSecurityGroup
	medialive:UpdateMultiplex
	medialive:UpdateMultiplexProgram
	medialive:UpdateNetwork
	medialive:UpdateNode
	medialive:UpdateNodeState
	medialive:UpdateReservation
	medialive:UpdateSdiSource

服務前綴	動作
mediatailor	mediatailor:ConfigureLogsForPlaybackConfiguration mediatailor:CreateChannel mediatailor:CreateLiveSource mediatailor:CreatePrefetchSchedule mediatailor:CreateProgram mediatailor:CreateSourceLocation mediatailor:CreateVodSource mediatailor:DeleteChannel mediatailor:DeleteChannelPolicy mediatailor:DeleteLiveSource mediatailor:DeletePlaybackConfiguration mediatailor:DeletePrefetchSchedule mediatailor:DeleteProgram mediatailor:DeleteSourceLocation mediatailor:DeleteVodSource mediatailor:DescribeChannel mediatailor:DescribeLiveSource mediatailor:DescribeProgram mediatailor:DescribeSourceLocation mediatailor:DescribeVodSource mediatailor:GetChannelPolicy

服務前綴	動作
	mediatailor:GetChannelSchedule mediatailor:GetPlaybackConfiguration mediatailor:GetPrefetchSchedule mediatailor:ListAlerts mediatailor:ListChannels mediatailor:ListLiveSources mediatailor:ListPlaybackConfigurations mediatailor:ListPrefetchSchedules mediatailor:ListSourceLocations mediatailor:ListVodSources mediatailor:PutChannelPolicy mediatailor:PutPlaybackConfiguration mediatailor:StartChannel mediatailor:StopChannel mediatailor:UpdateChannel mediatailor:UpdateLiveSource mediatailor:UpdateProgram mediatailor:UpdateSourceLocation mediatailor:UpdateVodSource

服務前綴	動作
memorydb	memorydb:BatchUpdateCluster
	memorydb:CopySnapshot
	memorydb:CreateAcl
	memorydb:CreateCluster
	memorydb:CreateMultiRegionCluster
	memorydb:CreateParameterGroup
	memorydb:CreateSnapshot
	memorydb:CreateSubnetGroup
	memorydb:CreateUser
	memorydb:DeleteAcl
	memorydb:DeleteCluster
	memorydb:DeleteMultiRegionCluster
	memorydb:DeleteParameterGroup
	memorydb:DeleteSnapshot
	memorydb:DeleteSubnetGroup
	memorydb:DeleteUser
	memorydb:DescribeAcls
	memorydb:DescribeClusters
	memorydb:DescribeEngineVersions
	memorydb:DescribeEvents
	memorydb:DescribeMultiRegionClusters

服務前綴	動作
	memorydb：DescribeMultiRegionParameterGroups
	memorydb：DescribeMultiRegionParameters
	memorydb:DescribeParameterGroups
	memorydb:DescribeParameters
	memorydb:DescribeReservedNodes
	memorydb:DescribeReservedNodesOfferings
	memorydb:DescribeServiceUpdates
	memorydb:DescribeSnapshots
	memorydb:DescribeSubnetGroups
	memorydb:DescribeUsers
	memorydb:FailoverShard
	memorydb:ListAllowedMultiRegionClusterUpdates
	memorydb:ListAllowedNodeTypeUpdates
	memorydb:PurchaseReservedNodesOffering
	memorydb:ResetParameterGroup
	memorydb:UpdateAcl
	memorydb:UpdateCluster
	memorydb:UpdateMultiRegionCluster
	memorydb:UpdateParameterGroup
	memorydb:UpdateSubnetGroup
	memorydb:UpdateUser

服務前綴	動作
mgh	mgh:AssociateCreatedArtifact mgh:AssociateDiscoveredResource mgh:AssociateSourceResource mgh:CreateHomeRegionControl mgh:CreateProgressUpdateStream mgh:DeleteHomeRegionControl mgh:DeleteProgressUpdateStream mgh:DescribeApplicationState mgh:DescribeHomeRegionControls mgh:DescribeMigrationTask mgh:DisassociateCreatedArtifact mgh:DisassociateDiscoveredResource mgh:DisassociateSourceResource mgh:GetHomeRegion mgh:ImportMigrationTask mgh:ListApplicationStates mgh:ListCreatedArtifacts mgh:ListDiscoveredResources mgh:ListMigrationTaskUpdates mgh:ListMigrationTasks mgh:ListProgressUpdateStreams

服務前綴	動作
	mgh:ListSourceResources mgh:NotifyApplicationState mgh:NotifyMigrationTaskState mgh:PutResourceAttributes

服務前綴	動作
mgn	mgn:ArchiveApplication
	mgn:ArchiveWave
	mgn:AssociateApplications
	mgn:AssociateSourceServers
	mgn:ChangeServerLifeCycleState
	mgn:CreateApplication
	mgn:CreateConnector
	mgn:CreateLaunchConfigurationTemplate
	mgn：CreateNetworkMigrationDefinition
	mgn:CreateReplicationConfigurationTemplate
	mgn:CreateWave
	mgn:DeleteApplication
	mgn:DeleteConnector
	mgn:DeleteJob
	mgn:DeleteLaunchConfigurationTemplate
	mgn：DeleteNetworkMigrationDefinition
	mgn:DeleteReplicationConfigurationTemplate
	mgn:DeleteSourceServer
	mgn:DeleteVcenterClient
	mgn:DeleteWave
	mgn:DescribeJobLogItems

服務前綴	動作
	mgn:DescribeJobs
	mgn:DescribeLaunchConfigurationTemplates
	mgn:DescribeReplicationConfigurationTemplates
	mgn:DescribeVcenterClients
	mgn:DisassociateApplications
	mgn:DisassociateSourceServers
	mgn:DisconnectFromService
	mgn:FinalizeCutover
	mgn：GetNetworkMigrationDefinition
	mgn：GetNetworkMigrationMapperSegmentConstruct
	mgn:GetReplicationConfiguration
	mgn:InitializeService
	mgn:ListConnectors
	mgn:ListExportErrors
	mgn:ListExports
	mgn:ListImportErrors
	mgn：ListImportFileEnrichments
	mgn:ListImports
	mgn:ListManagedAccounts
	mgn：ListNetworkMigrationAnalyses
	mgn：ListNetworkMigrationAnalysisResults

服務前綴	動作
	mgn：ListNetworkMigrationCodeGenerationSegments
	mgn：ListNetworkMigrationCodeGenerations
	mgn：ListNetworkMigrationDefinitions
	mgn：ListNetworkMigrationDeployedStacks
	mgn：ListNetworkMigrationDeployments
	mgn：ListNetworkMigrationExecutions
	mgn：ListNetworkMigrationMapperSegmentConstructs
	mgn：ListNetworkMigrationMapperSegments
	mgn：ListNetworkMigrationMappingUpdates
	mgn：ListNetworkMigrationMappings
	mgn:ListSourceServerActions
	mgn:ListTemplateActions
	mgn:MarkAsArchived
	mgn:PauseReplication
	mgn:PutSourceServerAction
	mgn:PutTemplateAction
	mgn:RemoveSourceServerAction
	mgn:RemoveTemplateAction
	mgn:ResumeReplication
	mgn:RetryDataReplication
	mgn:StartCutover

服務前綴	動作
	mgn:StartExport
	mgn:StartImport
	mgn：StartImportFileEnrichment
	mgn：StartNetworkMigrationCodeGeneration
	mgn：StartNetworkMigrationMapping
	mgn：StartNetworkMigrationMappingUpdate
	mgn:StartReplication
	mgn:StartTest
	mgn:StopReplication
	mgn:TerminateTargetInstances
	mgn:UnarchiveApplication
	mgn:UnarchiveWave
	mgn:UpdateApplication
	mgn:UpdateConnector
	mgn:UpdateLaunchConfigurationTemplate
	mgn：UpdateNetworkMigrationDefinition
	mgn：UpdateNetworkMigrationMapperSegment
	mgn:UpdateReplicationConfiguration
	mgn:UpdateReplicationConfigurationTemplate
	mgn:UpdateSourceServer
	mgn:UpdateSourceServerReplicationType

服務前綴	動作
	mgn:UpdateWave

服務前綴	動作
migrationhub-strategy	migrationhub-strategy:GetAntiPattern migrationhub-strategy:GetApplicationComponentDetails migrationhub-strategy:GetApplicationComponentStrategies migrationhub-strategy:GetAssessment migrationhub-strategy:GetImportFileTask migrationhub-strategy:GetLatestAssessmentId migrationhub-strategy:GetMessage migrationhub-strategy:GetPortfolioPreferences migrationhub-strategy:GetPortfolioSummary migrationhub-strategy:GetRecommendationReportDetails migrationhub-strategy:GetServerDetails migrationhub-strategy:GetServerStrategies migrationhub-strategy:ListAnalyzableServers migrationhub-strategy:ListAntiPatterns migrationhub-strategy:ListApplicationComponents migrationhub-strategy:ListCollectors migrationhub-strategy:ListImportFileTask migrationhub-strategy:ListJarArtifacts migrationhub-strategy:ListServers migrationhub-strategy:PutLogData migrationhub-strategy:PutMetricData

服務前綴	動作
	migrationhub-strategy:PutPortfolioPreferences migrationhub-strategy:RegisterCollector migrationhub-strategy:SendMessage migrationhub-strategy:StartAssessment migrationhub-strategy:StartImportFileTask migrationhub-strategy:StartRecommendationReportGeneration migrationhub-strategy:StopAssessment migrationhub-strategy:UpdateApplicationComponentConfig migrationhub-strategy:UpdateCollectorConfiguration migrationhub-strategy:UpdateServerConfig

服務前綴	動作
mobiletargeting	mobiletargeting:CreateApp
	mobiletargeting:CreateCampaign
	mobiletargeting:CreateEmailTemplate
	mobiletargeting:CreateExportJob
	mobiletargeting:CreateImportJob
	mobiletargeting:CreateInAppTemplate
	mobiletargeting:CreateJourney
	mobiletargeting:CreatePushTemplate
	mobiletargeting:CreateRecommenderConfiguration
	mobiletargeting:CreateSegment
	mobiletargeting:CreateSmsTemplate
	mobiletargeting:CreateVoiceTemplate
	mobiletargeting:DeleteAdmChannel
	mobiletargeting:DeleteApnsChannel
	mobiletargeting:DeleteApnsSandboxChannel
	mobiletargeting:DeleteApnsVoipChannel
	mobiletargeting:DeleteApnsVoipSandboxChannel
	mobiletargeting:DeleteApp
	mobiletargeting:DeleteBaiduChannel
	mobiletargeting:DeleteCampaign
	mobiletargeting:DeleteEmailChannel

服務前綴	動作
	mobiletargeting:DeleteEmailTemplate
	mobiletargeting:DeleteEndpoint
	mobiletargeting:DeleteEventStream
	mobiletargeting:DeleteGcmChannel
	mobiletargeting:DeleteInAppTemplate
	mobiletargeting:DeleteJourney
	mobiletargeting:DeletePushTemplate
	mobiletargeting:DeleteRecommenderConfiguration
	mobiletargeting:DeleteSegment
	mobiletargeting:DeleteSmsChannel
	mobiletargeting:DeleteSmsTemplate
	mobiletargeting:DeleteUserEndpoints
	mobiletargeting:DeleteVoiceChannel
	mobiletargeting:DeleteVoiceTemplate
	mobiletargeting:GetAdmChannel
	mobiletargeting:GetApnsChannel
	mobiletargeting:GetApnsSandboxChannel
	mobiletargeting:GetApnsVoipChannel
	mobiletargeting:GetApnsVoipSandboxChannel
	mobiletargeting:GetApp
	mobiletargeting:GetApplicationDateRangeKpi

服務前綴	動作
	mobiletargeting:GetApplicationSettings
	mobiletargeting:GetApps
	mobiletargeting:GetBaiduChannel
	mobiletargeting:GetCampaign
	mobiletargeting:GetCampaignActivities
	mobiletargeting:GetCampaignDateRangeKpi
	mobiletargeting:GetCampaignVersion
	mobiletargeting:GetCampaignVersions
	mobiletargeting:GetCampaigns
	mobiletargeting:GetChannels
	mobiletargeting:GetEmailChannel
	mobiletargeting:GetEmailTemplate
	mobiletargeting:GetEndpoint
	mobiletargeting:GetEventStream
	mobiletargeting:GetExportJob
	mobiletargeting:GetExportJobs
	mobiletargeting:GetGcmChannel
	mobiletargeting:GetImportJob
	mobiletargeting:GetImportJobs
	mobiletargeting:GetInAppMessages
	mobiletargeting:GetInAppTemplate

服務前綴	動作
	mobiletargeting:GetJourney
	mobiletargeting:GetJourneyDateRangeKpi
	mobiletargeting:GetJourneyExecutionActivityMetrics
	mobiletargeting:GetJourneyExecutionMetrics
	mobiletargeting:GetJourneyRunExecutionActivityMetrics
	mobiletargeting:GetJourneyRunExecutionMetrics
	mobiletargeting:GetJourneyRuns
	mobiletargeting:GetPushTemplate
	mobiletargeting:GetRecommenderConfiguration
	mobiletargeting:GetRecommenderConfigurations
	mobiletargeting:GetSegment
	mobiletargeting:GetSegmentExportJobs
	mobiletargeting:GetSegmentImportJobs
	mobiletargeting:GetSegmentVersion
	mobiletargeting:GetSegmentVersions
	mobiletargeting:GetSegments
	mobiletargeting:GetSmsChannel
	mobiletargeting:GetSmsTemplate
	mobiletargeting:GetUserEndpoints
	mobiletargeting:GetVoiceChannel
	mobiletargeting:GetVoiceTemplate

服務前綴	動作
	mobiletargeting:ListJourneys
	mobiletargeting:ListTemplateVersions
	mobiletargeting:ListTemplates
	mobiletargeting:PhoneNumberValidate
	mobiletargeting:PutEventStream
	mobiletargeting:RemoveAttributes
	mobiletargeting:UpdateAdmChannel
	mobiletargeting:UpdateApnsChannel
	mobiletargeting:UpdateApnsSandboxChannel
	mobiletargeting:UpdateApnsVoipChannel
	mobiletargeting:UpdateApnsVoipSandboxChannel
	mobiletargeting:UpdateApplicationSettings
	mobiletargeting:UpdateBaiduChannel
	mobiletargeting:UpdateCampaign
	mobiletargeting:UpdateEmailChannel
	mobiletargeting:UpdateEmailTemplate
	mobiletargeting:UpdateEndpoint
	mobiletargeting:UpdateEndpointsBatch
	mobiletargeting:UpdateGcmChannel
	mobiletargeting:UpdateInAppTemplate
	mobiletargeting:UpdateJourney

服務前綴	動作
	mobiletargeting:UpdateJourneyState
	mobiletargeting:UpdatePushTemplate
	mobiletargeting:UpdateRecommenderConfiguration
	mobiletargeting:UpdateSegment
	mobiletargeting:UpdateSmsChannel
	mobiletargeting:UpdateSmsTemplate
	mobiletargeting:UpdateTemplateActiveVersion
	mobiletargeting:UpdateVoiceChannel
	mobiletargeting:UpdateVoiceTemplate
	mobiletargeting:VerifyOTPMessage

服務前綴	動作
mq	mq:CreateBroker mq:CreateConfiguration mq:CreateUser mq:DeleteBroker mq:DeleteConfiguration mq:DeleteUser mq:DescribeBroker mq:DescribeBrokerEngineTypes mq:DescribeBrokerInstanceOptions mq:DescribeConfiguration mq:DescribeConfigurationRevision mq：DescribeSharedResources mq:DescribeUser mq:ListBrokers mq:ListConfigurationRevisions mq:ListConfigurations mq:ListUsers mq:Promote mq:RebootBroker mq:UpdateBroker mq:UpdateConfiguration

服務前綴	動作
	mq:UpdateUser

服務前綴	動作
networkmanager	networkmanager:AcceptAttachment
	networkmanager:AssociateConnectPeer
	networkmanager:AssociateCustomerGateway
	networkmanager:AssociateLink
	networkmanager:AssociateTransitGatewayConnectPeer
	networkmanager:CreateConnectAttachment
	networkmanager:CreateConnectPeer
	networkmanager:CreateConnection
	networkmanager:CreateCoreNetwork
	networkmanager:CreateDevice
	networkmanager:CreateDirectConnectGatewayAttachment
	networkmanager:CreateGlobalNetwork
	networkmanager:CreateLink
	networkmanager:CreateSite
	networkmanager:CreateSiteToSiteVpnAttachment
	networkmanager:CreateTransitGatewayPeering
	networkmanager:CreateTransitGatewayRouteTableAttachment
	networkmanager:CreateVpcAttachment
	networkmanager:DeleteAttachment
	networkmanager:DeleteConnectPeer
	networkmanager:DeleteConnection

服務前綴	動作
	networkmanager:DeleteCoreNetwork
	networkmanager:DeleteCoreNetworkPolicyVersion
	networkmanager:DeleteDevice
	networkmanager:DeleteGlobalNetwork
	networkmanager:DeleteLink
	networkmanager:DeletePeering
	networkmanager:DeleteResourcePolicy
	networkmanager:DeleteSite
	networkmanager:DeregisterTransitGateway
	networkmanager:DescribeGlobalNetworks
	networkmanager:DisassociateConnectPeer
	networkmanager:DisassociateCustomerGateway
	networkmanager:DisassociateLink
	networkmanager:DisassociateTransitGatewayConnectPeer
	networkmanager:ExecuteCoreNetworkChangeSet
	networkmanager:GetConnectAttachment
	networkmanager:GetConnectPeer
	networkmanager:GetConnectPeerAssociations
	networkmanager:GetConnections
	networkmanager:GetCoreNetwork
	networkmanager:GetCoreNetworkChangeEvents

服務前綴	動作
	networkmanager:GetCoreNetworkChangeSet
	networkmanager:GetCoreNetworkPolicy
	networkmanager:GetCustomerGatewayAssociations
	networkmanager:GetDevices
	networkmanager:GetLinkAssociations
	networkmanager:GetLinks
	networkmanager:GetNetworkResourceCounts
	networkmanager:GetNetworkResourceRelationships
	networkmanager:GetNetworkResources
	networkmanager:GetNetworkRoutes
	networkmanager:GetNetworkTelemetry
	networkmanager:GetResourcePolicy
	networkmanager:GetRouteAnalysis
	networkmanager:GetSiteToSiteVpnAttachment
	networkmanager:GetSites
	networkmanager:GetTransitGatewayConnectPeerAssociations
	networkmanager:GetTransitGatewayPeering
	networkmanager:GetTransitGatewayRegistrations
	networkmanager:GetTransitGatewayRouteTableAttachment
	networkmanager:GetVpcAttachment
	networkmanager：ListAttachmentRoutingPolicyAssociations

服務前綴	動作
	networkmanager:ListAttachments
	networkmanager:ListConnectPeers
	networkmanager:ListCoreNetworkPolicyVersions
	networkmanager：ListCoreNetworkPrefixListAssociations
	networkmanager：ListCoreNetworkRoutingInformation
	networkmanager:ListCoreNetworks
	networkmanager:ListOrganizationServiceAccessStatus
	networkmanager:ListPeerings
	networkmanager：PutAttachmentRoutingPolicyLabel
	networkmanager:PutCoreNetworkPolicy
	networkmanager:PutResourcePolicy
	networkmanager:RegisterTransitGateway
	networkmanager:RejectAttachment
	networkmanager：RemoveAttachmentRoutingPolicyLabel
	networkmanager:RestoreCoreNetworkPolicyVersion
	networkmanager:StartOrganizationServiceAccessUpdate
	networkmanager:StartRouteAnalysis
	networkmanager:UpdateConnection
	networkmanager:UpdateCoreNetwork
	networkmanager:UpdateDevice
	networkmanager:UpdateDirectConnectGatewayAttachment

服務前綴	動作
	networkmanager:UpdateGlobalNetwork networkmanager:UpdateLink networkmanager:UpdateNetworkResourceMetadata networkmanager:UpdateSite networkmanager:UpdateVpcAttachment

服務前綴	動作
nimble	nimble:AcceptEulas nimble:CreateLaunchProfile nimble:CreateStreamingImage nimble:CreateStreamingSession nimble:CreateStreamingSessionStream nimble:CreateStudio nimble:CreateStudioComponent nimble:DeleteLaunchProfile nimble:DeleteLaunchProfileMember nimble:DeleteStreamingImage nimble:DeleteStreamingSession nimble:DeleteStudio nimble:DeleteStudioComponent nimble:DeleteStudioMember nimble:GetEula nimble:GetLaunchProfileDetails nimble:GetStreamingImage nimble:GetStreamingSession nimble:GetStreamingSessionBackup nimble:GetStreamingSessionStream nimble:GetStudio

服務前綴	動作
	nimble:GetStudioComponent nimble:GetStudioMember nimble:ListEulas nimble:ListLaunchProfileMembers nimble:ListLaunchProfiles nimble:ListStreamingImages nimble:ListStreamingSessionBackups nimble:ListStreamingSessions nimble:ListStudioComponents nimble:ListStudioMembers nimble:ListStudios nimble:PutLaunchProfileMembers nimble:PutStudioMembers nimble:StartStreamingSession nimble:StartStudioSSOConfigurationRepair nimble:StopStreamingSession nimble:UpdateLaunchProfile nimble:UpdateLaunchProfileMember nimble:UpdateStreamingImage nimble:UpdateStudio nimble:UpdateStudioComponent

服務前綴	動作
omics	omics:AbortMultipartReadSetUpload
	omics:AcceptShare
	omics:BatchDeleteReadSet
	omics:CancelAnnotationImportJob
	omics:CancelRun
	omics：CancelRunBatch
	omics:CancelVariantImportJob
	omics:CompleteMultipartReadSetUpload
	omics:CreateAnnotationStore
	omics:CreateAnnotationStoreVersion
	omics：CreateConfiguration
	omics:CreateMultipartReadSetUpload
	omics:CreateReferenceStore
	omics:CreateRunGroup
	omics:CreateSequenceStore
	omics:CreateShare
	omics:CreateVariantStore
	omics:CreateWorkflow
	omics:CreateWorkflowVersion
	omics:DeleteAnnotationStore
	omics:DeleteAnnotationStoreVersions

服務前綴	動作
	omics：DeleteBatch
	omics：DeleteConfiguration
	omics:DeleteReference
	omics:DeleteReferenceStore
	omics:DeleteRun
	omics：DeleteRunBatch
	omics:DeleteRunGroup
	omics:DeleteSequenceStore
	omics:DeleteShare
	omics:DeleteVariantStore
	omics:DeleteWorkflow
	omics:DeleteWorkflowVersion
	omics:GetAnnotationImportJob
	omics:GetAnnotationStore
	omics:GetAnnotationStoreVersion
	omics：GetBatch
	omics：GetConfiguration
	omics:GetReadSet
	omics:GetReadSetActivationJob
	omics:GetReadSetExportJob
	omics:GetReadSetImportJob

服務前綴	動作
	omics:GetReadSetMetadata
	omics:GetReference
	omics:GetReferenceImportJob
	omics:GetReferenceMetadata
	omics:GetReferenceStore
	omics:GetRun
	omics:GetRunGroup
	omics:GetRunTask
	omics:GetSequenceStore
	omics:GetShare
	omics:GetVariantImportJob
	omics:GetVariantStore
	omics:GetWorkflow
	omics:GetWorkflowVersion
	omics:ListAnnotationImportJobs
	omics:ListAnnotationStoreVersions
	omics:ListAnnotationStores
	omics：ListBatch
	omics：ListConfigurations
	omics:ListMultipartReadSetUploads
	omics:ListReadSetActivationJobs

服務前綴	動作
	omics:ListReadSetExportJobs
	omics:ListReadSetImportJobs
	omics:ListReadSetUploadParts
	omics:ListReadSets
	omics:ListReferenceImportJobs
	omics:ListReferenceStores
	omics:ListReferences
	omics:ListRunGroups
	omics:ListRunTasks
	omics:ListRuns
	omics：ListRunsInBatch
	omics:ListSequenceStores
	omics:ListShares
	omics:ListVariantImportJobs
	omics:ListVariantStores
	omics:ListWorkflowVersions
	omics:ListWorkflows
	omics:StartAnnotationImportJob
	omics:StartReadSetActivationJob
	omics:StartReadSetExportJob
	omics:StartReadSetImportJob

服務前綴	動作
	omics:StartReferenceImportJob
	omics:StartRun
	omics：StartRunBatch
	omics:StartVariantImportJob
	omics:UpdateAnnotationStore
	omics:UpdateAnnotationStoreVersion
	omics:UpdateRunGroup
	omics:UpdateVariantStore
	omics:UpdateWorkflow
	omics:UpdateWorkflowVersion
	omics:UploadReadSetPart

服務前綴	動作
opsworks	opsworks:AssignInstance opsworks:AssignVolume opsworks:AssociateElasticIp opsworks:AttachElasticLoadBalancer opsworks:CloneStack opsworks:CreateApp opsworks:CreateDeployment opsworks:CreateInstance opsworks:CreateLayer opsworks:CreateStack opsworks:CreateUserProfile opsworks:DeleteApp opsworks:DeleteInstance opsworks:DeleteLayer opsworks:DeleteStack opsworks:DeleteUserProfile opsworks:DeregisterEcsCluster opsworks:DeregisterElasticIp opsworks:DeregisterInstance opsworks:DeregisterRdsDbInstance opsworks:DeregisterVolume

服務前綴	動作
	opsworks:DescribeAgentVersions opsworks:DescribeApps opsworks:DescribeCommands opsworks:DescribeDeployments opsworks:DescribeEcsClusters opsworks:DescribeElasticIps opsworks:DescribeElasticLoadBalancers opsworks:DescribeInstances opsworks:DescribeLayers opsworks:DescribeLoadBasedAutoScaling opsworks:DescribeMyUserProfile opsworks:DescribeOperatingSystems opsworks:DescribePermissions opsworks:DescribeRaidArrays opsworks:DescribeRdsDbInstances opsworks:DescribeServiceErrors opsworks:DescribeStackProvisioningParameters opsworks:DescribeStackSummary opsworks:DescribeStacks opsworks:DescribeTimeBasedAutoScaling opsworks:DescribeUserProfiles

服務前綴	動作
	opsworks:DescribeVolumes opsworks:DetachElasticLoadBalancer opsworks:DisassociateElasticIp opsworks:GetHostnameSuggestion opsworks:GrantAccess opsworks:RebootInstance opsworks:RegisterEcsCluster opsworks:RegisterElasticIp opsworks:RegisterInstance opsworks:RegisterRdsDbInstance opsworks:RegisterVolume opsworks:SetLoadBasedAutoScaling opsworks:SetPermission opsworks:SetTimeBasedAutoScaling opsworks:StartInstance opsworks:StartStack opsworks:StopInstance opsworks:StopStack opsworks:UnassignInstance opsworks:UnassignVolume opsworks:UpdateApp

服務前綴	動作
	opsworks:UpdateElasticIp opsworks:UpdateInstance opsworks:UpdateLayer opsworks:UpdateMyUserProfile opsworks:UpdateRdsDbInstance opsworks:UpdateStack opsworks:UpdateUserProfile opsworks:UpdateVolume

服務前綴	動作
opsworks-cm	opsworks-cm:AssociateNode
	opsworks-cm:CreateBackup
	opsworks-cm:CreateServer
	opsworks-cm:DeleteBackup
	opsworks-cm:DeleteServer
	opsworks-cm:DescribeAccountAttributes
	opsworks-cm:DescribeBackups
	opsworks-cm:DescribeEvents
	opsworks-cm:DescribeNodeAssociationStatus
	opsworks-cm:DescribeServers
	opsworks-cm:DisassociateNode
	opsworks-cm:ExportServerEngineAttribute
	opsworks-cm:RestoreServer
	opsworks-cm:StartMaintenance
	opsworks-cm:UpdateServer
	opsworks-cm:UpdateServerEngineAttributes

服務前綴	動作
組織	organizations:AcceptHandshake organizations:AttachPolicy organizations:CancelHandshake organizations:CloseAccount organizations:CreateAccount organizations:CreateGovCloudAccount organizations:CreateOrganization organizations:CreateOrganizationalUnit organizations:CreatePolicy organizations:DeclineHandshake organizations:DeleteOrganization organizations:DeleteOrganizationalUnit organizations:DeletePolicy organizations:DeleteResourcePolicy organizations:DeregisterDelegatedAdministrator organizations:DescribeAccount organizations:DescribeCreateAccountStatus organizations:DescribeEffectivePolicy organizations:DescribeHandshake organizations：DescribeOrganization organizations:DescribeOrganizationalUnit

服務前綴	動作
	organizations:DescribePolicy organizations:DescribeResourcePolicy organizations：DescribeResponsibilityTransfer organizations:DetachPolicy organizations:DisableAWSServiceAccess organizations:DisablePolicyType organizations:EnableAWSServiceAccess organizations:EnableAllFeatures organizations:EnablePolicyType organizations:InviteAccountToOrganization organizations:LeaveOrganization organizations：ListAWSServiceAccessForOrganization organizations:ListAccounts organizations:ListAccountsForParent organizations：ListAccountsWithInvalidEffectivePolicy organizations:ListChildren organizations:ListCreateAccountStatus organizations:ListDelegatedAdministrators organizations:ListDelegatedServicesForAccount organizations:ListHandshakesForAccount organizations:ListHandshakesForOrganization

服務前綴	動作
	organizations：ListInboundResponsibilityTransfers organizations:ListOrganizationalUnitsForParent organizations：ListOutboundResponsibilityTransfers organizations:ListParents organizations:ListPolicies organizations:ListPoliciesForTarget organizations:ListRoots organizations:ListTargetsForPolicy organizations:MoveAccount organizations:PutResourcePolicy organizations:RegisterDelegatedAdministrator organizations:RemoveAccountFromOrganization organizations：TerminateResponsibilityTransfer organizations:UpdateOrganizationalUnit organizations:UpdatePolicy organizations：UpdateResponsibilityTransfer

服務前綴	動作
outposts	outposts:CancelCapacityTask
	outposts:CancelOrder
	outposts:CreateOrder
	outposts:CreateOutpost
	outposts:CreatePrivateConnectivityConfig
	outposts：CreateQuote
	outposts：CreateRenewal
	outposts:CreateSite
	outposts:DeleteOutpost
	outposts：DeleteQuote
	outposts:DeleteSite
	outposts:GetCapacityTask
	outposts:GetCatalogItem
	outposts:GetConnection
	outposts:GetOrder
	outposts:GetOutpost
	outposts:GetOutpostBillingInformation
	outposts:GetOutpostInstanceTypes
	outposts:GetOutpostSupportedInstanceTypes
	outposts:GetPrivateConnectivityConfig
	outposts：GetQuote

服務前綴	動作
	outposts：GetRenewalPricing
	outposts:GetSite
	outposts:GetSiteAddress
	outposts:ListAssetInstances
	outposts:ListAssets
	outposts:ListBlockingInstancesForCapacityTask
	outposts:ListCapacityTasks
	outposts:ListCatalogItems
	outposts：ListOrderableInstanceTypes
	outposts:ListOrders
	outposts:ListOutposts
	outposts：ListQuotes
	outposts:ListSites
	outposts:StartCapacityTask
	outposts:StartConnection
	outposts:UpdateOutpost
	outposts：UpdateQuote
	outposts:UpdateSite
	outposts:UpdateSiteAddress
	outposts:UpdateSiteRackPhysicalProperties

服務前綴	動作
pi	pi:CreatePerformanceAnalysisReport pi:DeletePerformanceAnalysisReport pi:DescribeDimensionKeys pi:GetDimensionKeyDetails pi:GetPerformanceAnalysisReport pi:GetResourceMetadata pi:GetResourceMetrics pi:ListAvailableResourceDimensions pi:ListAvailableResourceMetrics pi:ListPerformanceAnalysisReports
pipes	pipes:CreatePipe pipes:DeletePipe pipes:DescribePipe pipes:ListPipes pipes:StartPipe pipes:StopPipe pipes:UpdatePipe

服務前綴	動作
polly	polly:DeleteLexicon polly:DescribeVoices polly:GetLexicon polly:GetSpeechSynthesisTask polly:ListLexicons polly:ListSpeechSynthesisTasks polly:PutLexicon polly:StartSpeechSynthesisTask polly:SynthesizeSpeech

服務前綴	動作
profile	profile:AddProfileKey profile:BatchGetCalculatedAttributeForProfile profile:BatchGetProfile profile:CreateCalculatedAttributeDefinition profile:CreateDomain profile:CreateEventStream profile:CreateProfile profile：CreateRecommender profile：CreateRecommenderFilter profile：CreateRecommenderSchema profile:CreateSegmentDefinition profile:CreateSegmentEstimate profile:CreateSegmentSnapshot profile:CreateUploadJob profile:DeleteCalculatedAttributeDefinition profile:DeleteDomain profile：DeleteDomainObjectType profile:DeleteEventStream profile:DeleteIntegration profile:DeleteProfile profile:DeleteProfileKey

服務前綴	動作
	profile:DeleteProfileObject
	profile:DeleteProfileObjectType
	profile：DeleteRecommender
	profile：DeleteRecommenderFilter
	profile：DeleteRecommenderSchema
	profile:DeleteSegmentDefinition
	profile:DeleteWorkflow
	profile:DetectProfileObjectType
	profile:GetAutoMergingPreview
	profile:GetCalculatedAttributeDefinition
	profile:GetCalculatedAttributeForProfile
	profile:GetDomain
	profile：GetDomainObjectType
	profile:GetEventStream
	profile:GetIdentityResolutionJob
	profile:GetIntegration
	profile:GetMatches
	profile：GetObjectTypeAttributeStatistics
	profile:GetProfileObjectType
	profile:GetProfileObjectTypeTemplate
	profile：GetProfileRecommendations

服務前綴	動作
	profile：GetRecommender
	profile：GetRecommenderFilter
	profile：GetRecommenderSchema
	profile:GetSegmentDefinition
	profile:GetSegmentEstimate
	profile:GetSegmentMembership
	profile:GetSegmentSnapshot
	profile:GetSimilarProfiles
	profile:GetUploadJob
	profile:GetUploadJobPath
	profile:GetWorkflow
	profile:GetWorkflowSteps
	profile:ListAccountIntegrations
	profile:ListCalculatedAttributeDefinitions
	profile:ListCalculatedAttributesForProfile
	profile:ListDomainLayouts
	profile：ListDomainObjectTypes
	profile:ListDomains
	profile:ListEventStreams
	profile:ListIdentityResolutionJobs
	profile:ListIntegrations

服務前綴	動作
	profile：ListObjectTypeAttributeValues
	profile:ListObjectTypeAttributes
	profile:ListProfileAttributeValues
	profile:ListProfileObjectTypeTemplates
	profile:ListProfileObjectTypes
	profile:ListProfileObjects
	profile：ListRecommenderFilters
	profile：ListRecommenderRecipes
	profile：ListRecommenderSchemas
	profile：ListRecommenders
	profile:ListRuleBasedMatches
	profile:ListSegmentDefinitions
	profile:ListUploadJobs
	profile:ListWorkflows
	profile:MergeProfiles
	profile：PutDomainObjectType
	profile:PutIntegration
	profile:PutProfileObject
	profile:PutProfileObjectType
	profile:SearchProfiles
	profile：StartRecommender

服務前綴	動作
	profile:StartUploadJob profile：StopRecommender profile:StopUploadJob profile:UpdateCalculatedAttributeDefinition profile:UpdateDomain profile:UpdateProfile profile：UpdateRecommender

服務前綴	動作
qldb	qldb:CancelJournalKinesisStream qldb:CreateLedger qldb:DeleteLedger qldb:DescribeJournalKinesisStream qldb:DescribeJournalS3Export qldb:DescribeLedger qldb:ExportJournalToS3 qldb:GetBlock qldb:GetDigest qldb:GetRevision qldb:ListJournalKinesisStreamsForLedger qldb:ListJournalS3Exports qldb:ListJournalS3ExportsForLedger qldb:ListLedgers qldb:StreamJournalToKinesis qldb:UpdateLedger qldb:UpdateLedgerPermissionsMode

服務前綴	動作
ram	ram:AcceptResourceShareInvitation
	ram:AssociateResourceShare
	ram:AssociateResourceSharePermission
	ram:CreatePermission
	ram:CreatePermissionVersion
	ram:CreateResourceShare
	ram:DeletePermission
	ram:DeletePermissionVersion
	ram:DeleteResourceShare
	ram:DisassociateResourceShare
	ram:DisassociateResourceSharePermission
	ram:EnableSharingWithAwsOrganization
	ram:GetPermission
	ram:GetResourcePolicies
	ram:GetResourceShareAssociations
	ram:GetResourceShareInvitations
	ram:GetResourceShares
	ram:ListPendingInvitationResources
	ram:ListPermissionAssociations
	ram:ListPermissionVersions
	ram:ListPermissions

服務前綴	動作
	ram:ListPrincipals ram:ListReplacePermissionAssociationsWork ram:ListResourceSharePermissions ram:ListResourceTypes ram:ListResources ram：ListSourceAssociations ram:PromotePermissionCreatedFromPolicy ram:PromoteResourceShareCreatedFromPolicy ram:RejectResourceShareInvitation ram:ReplacePermissionAssociations ram:SetDefaultPermissionVersion ram:UpdateResourceShare
rbin	rbin:CreateRule rbin:DeleteRule rbin:GetRule rbin:ListRules rbin:LockRule rbin:UnlockRule rbin:UpdateRule

服務前綴	動作
rds	rds:AddRoleToDBCluster
	rds:AddRoleToDBInstance
	rds:AddSourceIdentifierToSubscription
	rds:ApplyPendingMaintenanceAction
	rds:AuthorizeDBSecurityGroupIngress
	rds:BacktrackDBCluster
	rds:CancelExportTask
	rds:CopyDBClusterParameterGroup
	rds:CopyDBClusterSnapshot
	rds:CopyDBParameterGroup
	rds:CopyDBSnapshot
	rds:CopyOptionGroup
	rds:CreateCustomDBEngineVersion
	rds:CreateDBClusterParameterGroup
	rds:CreateDBParameterGroup
	rds:CreateDBProxy
	rds:CreateDBProxyEndpoint
	rds:CreateDBSecurityGroup
	rds:CreateDBSubnetGroup
	rds:CreateEventSubscription
	rds:CreateGlobalCluster

服務前綴	動作
	rds:CreateOptionGroup
	rds:DeleteBlueGreenDeployment
	rds:DeleteDBClusterAutomatedBackup
	rds:DeleteDBClusterParameterGroup
	rds:DeleteDBClusterSnapshot
	rds:DeleteDBInstanceAutomatedBackup
	rds:DeleteDBParameterGroup
	rds:DeleteDBProxy
	rds:DeleteDBProxyEndpoint
	rds:DeleteDBSecurityGroup
	rds:DeleteDBSnapshot
	rds:DeleteDBSubnetGroup
	rds:DeleteEventSubscription
	rds:DeleteGlobalCluster
	rds:DeleteOptionGroup
	rds:DeregisterDBProxyTargets
	rds:DescribeAccountAttributes
	rds:DescribeBlueGreenDeployments
	rds:DescribeCertificates
	rds:DescribeDBClusterAutomatedBackups
	rds:DescribeDBClusterBacktracks

服務前綴	動作
	rds:DescribeDBClusterEndpoints
	rds:DescribeDBClusterParameterGroups
	rds:DescribeDBClusterParameters
	rds:DescribeDBClusterSnapshotAttributes
	rds:DescribeDBClusterSnapshots
	rds:DescribeDBClusters
	rds:DescribeDBEngineVersions
	rds:DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rds:DescribeDBLogFiles
	rds:DescribeDBMajorEngineVersions
	rds:DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rds:DescribeDBProxyEndpoints
	rds:DescribeDBProxyTargetGroups
	rds:DescribeDBProxyTargets
	rds:DescribeDBRecommendations
	rds:DescribeDBSecurityGroups
	rds:DescribeDBSnapshotAttributes
	rds:DescribeDBSnapshotTenantDatabases

服務前綴	動作
	rds:DescribeDBSnapshots
	rds:DescribeDBSubnetGroups
	rds:DescribeEngineDefaultClusterParameters
	rds:DescribeEngineDefaultParameters
	rds:DescribeEventCategories
	rds:DescribeEventSubscriptions
	rds:DescribeEvents
	rds:DescribeExportTasks
	rds:DescribeGlobalClusters
	rds:DescribeIntegrations
	rds:DescribeOptionGroupOptions
	rds:DescribeOptionGroups
	rds:DescribeOrderableDBInstanceOptions
	rds:DescribePendingMaintenanceActions
	rds:DescribeReservedDBInstances
	rds:DescribeReservedDBInstancesOfferings
	rds:DescribeSourceRegions
	rds:DescribeTenantDatabases
	rds:DescribeValidDBInstanceModifications
	rds:DownloadCompleteDBLogFile
	rds:DownloadDBLogFilePortion

服務前綴	動作
	rds:FailoverDBCluster
	rds:FailoverGlobalCluster
	rds:ModifyActivityStream
	rds:ModifyCertificates
	rds:ModifyCurrentDBClusterCapacity
	rds:ModifyDBClusterEndpoint
	rds:ModifyDBClusterParameterGroup
	rds:ModifyDBClusterSnapshotAttribute
	rds:ModifyDBParameterGroup
	rds:ModifyDBProxy
	rds:ModifyDBProxyEndpoint
	rds:ModifyDBProxyTargetGroup
	rds:ModifyDBRecommendation
	rds:ModifyDBSnapshot
	rds:ModifyDBSnapshotAttribute
	rds:ModifyDBSubnetGroup
	rds:ModifyEventSubscription
	rds:ModifyGlobalCluster
	rds:ModifyOptionGroup
	rds:ModifyTenantDatabase
	rds:PurchaseReservedDBInstancesOffering

服務前綴	動作
	rds:RebootDBCluster
	rds:RegisterDBProxyTargets
	rds:RemoveFromGlobalCluster
	rds:RemoveRoleFromDBCluster
	rds:RemoveRoleFromDBInstance
	rds:RemoveSourceIdentifierFromSubscription
	rds:ResetDBClusterParameterGroup
	rds:ResetDBParameterGroup
	rds:RestoreDBClusterFromS3
	rds:RestoreDBClusterFromSnapshot
	rds:RestoreDBClusterToPointInTime
	rds:RestoreDBInstanceFromDBSnapshot
	rds:RestoreDBInstanceFromS3
	rds:RestoreDBInstanceToPointInTime
	rds:RevokeDBSecurityGroupIngress
	rds:StartActivityStream
	rds:StartDBCluster
	rds:StartDBInstance
	rds:StartDBInstanceAutomatedBackupsReplication
	rds:StartExportTask
	rds:StopActivityStream

服務前綴	動作
	rds:StopDBCluster
	rds:StopDBInstance
	rds:StopDBInstanceAutomatedBackupsReplication
	rds:SwitchoverBlueGreenDeployment
	rds:SwitchoverGlobalCluster
	rds:SwitchoverReadReplica

服務前綴	動作
redshift	redshift:AcceptReservedNodeExchange redshift:AddPartner redshift:AssociateDataShareConsumer redshift:AuthorizeClusterSecurityGroupIngress redshift:AuthorizeDataShare redshift:AuthorizeEndpointAccess redshift:AuthorizeSnapshotAccess redshift:BatchDeleteClusterSnapshots redshift:BatchModifyClusterSnapshots redshift:CancelResize redshift:CopyClusterSnapshot redshift:CreateAuthenticationProfile redshift:CreateCluster redshift:CreateClusterParameterGroup redshift:CreateClusterSecurityGroup redshift:CreateClusterSnapshot redshift:CreateClusterSubnetGroup redshift:CreateCustomDomainAssociation redshift:CreateEndpointAccess redshift:CreateEventSubscription redshift:CreateHsmClientCertificate

服務前綴	動作
	redshift:CreateHsmConfiguration redshift:CreateIntegration redshift:CreateRedshiftIdcApplication redshift:CreateScheduledAction redshift:CreateSnapshotCopyGrant redshift:CreateSnapshotSchedule redshift:CreateUsageLimit redshift:DeauthorizeDataShare redshift:DeleteAuthenticationProfile redshift:DeleteCluster redshift:DeleteClusterParameterGroup redshift:DeleteClusterSecurityGroup redshift:DeleteClusterSnapshot redshift:DeleteClusterSubnetGroup redshift:DeleteCustomDomainAssociation redshift:DeleteEndpointAccess redshift:DeleteEventSubscription redshift:DeleteHsmClientCertificate redshift:DeleteHsmConfiguration redshift:DeletePartner redshift:DeleteRedshiftIdcApplication

服務前綴	動作
	redshift:DeleteResourcePolicy redshift:DeleteScheduledAction redshift:DeleteSnapshotCopyGrant redshift:DeleteSnapshotSchedule redshift:DeleteUsageLimit redshift:DeregisterNamespace redshift:DescribeAccountAttributes redshift:DescribeAuthenticationProfiles redshift:DescribeClusterDbRevisions redshift:DescribeClusterParameterGroups redshift:DescribeClusterParameters redshift:DescribeClusterSecurityGroups redshift:DescribeClusterSnapshots redshift:DescribeClusterSubnetGroups redshift:DescribeClusterTracks redshift:DescribeClusterVersions redshift:DescribeClusters redshift:DescribeCustomDomainAssociations redshift:DescribeDataShares redshift:DescribeDataSharesForConsumer redshift:DescribeDataSharesForProducer

服務前綴	動作
	redshift:DescribeDefaultClusterParameters redshift:DescribeEndpointAccess redshift:DescribeEndpointAuthorization redshift:DescribeEventCategories redshift:DescribeEventSubscriptions redshift:DescribeEvents redshift:DescribeHsmClientCertificates redshift:DescribeHsmConfigurations redshift:DescribeInboundIntegrations redshift:DescribeIntegrations redshift:DescribeLoggingStatus redshift:DescribeNodeConfigurationOptions redshift:DescribeOrderableClusterOptions redshift:DescribePartners redshift:DescribeRedshiftIdcApplications redshift:DescribeReservedNodeExchangeStatus redshift:DescribeReservedNodeOfferings redshift:DescribeReservedNodes redshift:DescribeResize redshift:DescribeScheduledActions redshift:DescribeSnapshotCopyGrants

服務前綴	動作
	redshift:DescribeSnapshotSchedules redshift:DescribeStorage redshift:DescribeTableRestoreStatus redshift:DescribeUsageLimits redshift:DisableLogging redshift:DisableSnapshotCopy redshift:DisassociateDataShareConsumer redshift:EnableLogging redshift:EnableSnapshotCopy redshift:FailoverPrimaryCompute redshift:GetClusterCredentials redshift:GetClusterCredentialsWithIAM redshift：GetIdentityCenterAuthToken redshift:GetReservedNodeExchangeConfigurationOptions redshift:GetReservedNodeExchangeOfferings redshift:GetResourcePolicy redshift:ListRecommendations redshift:ModifyAquaConfiguration redshift:ModifyAuthenticationProfile redshift:ModifyCluster redshift:ModifyClusterDbRevision

服務前綴	動作
	redshift:ModifyClusterIamRoles redshift:ModifyClusterMaintenance redshift:ModifyClusterParameterGroup redshift:ModifyClusterSnapshot redshift:ModifyClusterSnapshotSchedule redshift:ModifyClusterSubnetGroup redshift:ModifyCustomDomainAssociation redshift:ModifyEndpointAccess redshift:ModifyEventSubscription redshift:ModifyRedshiftIdcApplication redshift:ModifyScheduledAction redshift:ModifySnapshotCopyRetentionPeriod redshift:ModifySnapshotSchedule redshift:ModifyUsageLimit redshift:PauseCluster redshift:PurchaseReservedNodeOffering redshift:PutResourcePolicy redshift:RebootCluster redshift:RegisterNamespace redshift:RejectDataShare redshift:ResetClusterParameterGroup

服務前綴	動作
	redshift:ResizeCluster redshift:RestoreFromClusterSnapshot redshift:RestoreTableFromClusterSnapshot redshift:ResumeCluster redshift:RevokeClusterSecurityGroupIngress redshift:RevokeEndpointAccess redshift:RevokeSnapshotAccess redshift:RotateEncryptionKey redshift:UpdatePartnerStatus
redshift-data	redshift-data:BatchExecuteStatement redshift-data:CancelStatement redshift-data:DescribeStatement redshift-data:DescribeTable redshift-data:ExecuteStatement redshift-data:GetStatementResult redshift-data:ListDatabases redshift-data:ListSchemas redshift-data：ListSessions redshift-data:ListStatements redshift-data:ListTables

服務前綴	動作
refactor-spaces	refactor-spaces:CreateApplication refactor-spaces:CreateEnvironment refactor-spaces:CreateRoute refactor-spaces:CreateService refactor-spaces:DeleteApplication refactor-spaces:DeleteEnvironment refactor-spaces:DeleteResourcePolicy refactor-spaces:DeleteRoute refactor-spaces:DeleteService refactor-spaces:GetApplication refactor-spaces:GetEnvironment refactor-spaces:GetResourcePolicy refactor-spaces:GetRoute refactor-spaces:GetService refactor-spaces:ListApplications refactor-spaces:ListEnvironmentVpcs refactor-spaces:ListEnvironments refactor-spaces:ListRoutes refactor-spaces:ListServices refactor-spaces:PutResourcePolicy refactor-spaces:UpdateRoute

服務前綴	動作
rekognition	rekognition:AssociateFaces
	rekognition:CompareFaces
	rekognition:CopyProjectVersion
	rekognition:CreateCollection
	rekognition:CreateDataset
	rekognition:CreateFaceLivenessSession
	rekognition:CreateProject
	rekognition:CreateProjectVersion
	rekognition:CreateStreamProcessor
	rekognition:CreateUser
	rekognition:DeleteCollection
	rekognition:DeleteDataset
	rekognition:DeleteFaces
	rekognition:DeleteProject
	rekognition:DeleteProjectPolicy
	rekognition:DeleteProjectVersion
	rekognition:DeleteStreamProcessor
	rekognition:DeleteUser
	rekognition:DescribeCollection
	rekognition:DescribeDataset
	rekognition:DescribeProjectVersions

服務前綴	動作
	rekognition:DescribeProjects
	rekognition:DescribeStreamProcessor
	rekognition:DetectCustomLabels
	rekognition:DetectFaces
	rekognition:DetectLabels
	rekognition:DetectModerationLabels
	rekognition:DetectProtectiveEquipment
	rekognition:DetectText
	rekognition:DisassociateFaces
	rekognition:DistributeDatasetEntries
	rekognition:GetCelebrityInfo
	rekognition:GetCelebrityRecognition
	rekognition:GetContentModeration
	rekognition:GetFaceDetection
	rekognition:GetFaceLivenessSessionResults
	rekognition:GetFaceSearch
	rekognition:GetLabelDetection
	rekognition:GetMediaAnalysisJob
	rekognition:GetPersonTracking
	rekognition:GetSegmentDetection
	rekognition:GetTextDetection

服務前綴	動作
	rekognition:IndexFaces
	rekognition:ListCollections
	rekognition:ListDatasetEntries
	rekognition:ListDatasetLabels
	rekognition:ListFaces
	rekognition:ListMediaAnalysisJobs
	rekognition:ListProjectPolicies
	rekognition:ListStreamProcessors
	rekognition:ListUsers
	rekognition:PutProjectPolicy
	rekognition:RecognizeCelebrities
	rekognition:SearchFaces
	rekognition:SearchFacesByImage
	rekognition:SearchUsers
	rekognition:SearchUsersByImage
	rekognition:StartCelebrityRecognition
	rekognition:StartContentModeration
	rekognition:StartFaceDetection
	rekognition:StartFaceLivenessSession
	rekognition:StartFaceSearch
	rekognition:StartLabelDetection

服務前綴	動作
	rekognition:StartMediaAnalysisJob
	rekognition:StartPersonTracking
	rekognition:StartProjectVersion
	rekognition:StartSegmentDetection
	rekognition:StartStreamProcessor
	rekognition:StartTextDetection
	rekognition:StopProjectVersion
	rekognition:StopStreamProcessor
	rekognition:UpdateDatasetEntries
	rekognition:UpdateStreamProcessor

服務前綴	動作
resiliencehub	resiliencehub:AcceptResourceGroupingRecommendations resiliencehub:AddDraftAppVersionResourceMappings resiliencehub:BatchUpdateRecommendationStatus resiliencehub:CreateApp resiliencehub:CreateAppVersionAppComponent resiliencehub:CreateAppVersionResource resiliencehub:CreateRecommendationTemplate resiliencehub:CreateResiliencyPolicy resiliencehub:DeleteApp resiliencehub:DeleteAppAssessment resiliencehub:DeleteAppInputSource resiliencehub:DeleteAppVersionAppComponent resiliencehub:DeleteAppVersionResource resiliencehub:DeleteRecommendationTemplate resiliencehub:DeleteResiliencyPolicy resiliencehub:DescribeApp resiliencehub:DescribeAppAssessment resiliencehub:DescribeAppVersion resiliencehub:DescribeAppVersionAppComponent resiliencehub:DescribeAppVersionResource resiliencehub:DescribeAppVersionResourcesResolutionStatus

服務前綴	動作
	resiliencehub:DescribeAppVersionTemplate resiliencehub:DescribeDraftAppVersionResourcesImportStatus resiliencehub:DescribeMetricsExport resiliencehub:DescribeResiliencyPolicy resiliencehub:DescribeResourceGroupingRecommendationTask resiliencehub:ImportResourcesToDraftAppVersion resiliencehub:ListAlarmRecommendations resiliencehub:ListAppAssessmentComplianceDrifts resiliencehub:ListAppAssessmentResourceDrifts resiliencehub:ListAppAssessments resiliencehub:ListAppComponentCompliances resiliencehub:ListAppComponentRecommendations resiliencehub:ListAppInputSources resiliencehub:ListAppVersionAppComponents resiliencehub:ListAppVersionResourceMappings resiliencehub:ListAppVersionResources resiliencehub:ListAppVersions resiliencehub:ListApps resiliencehub:ListMetrics resiliencehub:ListRecommendationTemplates resiliencehub:ListResiliencyPolicies

服務前綴	動作
	resiliencehub:ListResourceGroupingRecommendations resiliencehub:ListSopRecommendations resiliencehub:ListSuggestedResiliencyPolicies resiliencehub:ListTestRecommendations resiliencehub:ListUnsupportedAppVersionResources resiliencehub:PublishAppVersion resiliencehub:PutDraftAppVersionTemplate resiliencehub:RejectResourceGroupingRecommendations resiliencehub:RemoveDraftAppVersionResourceMappings resiliencehub:ResolveAppVersionResources resiliencehub:StartAppAssessment resiliencehub:StartResourceGroupingRecommendationTask resiliencehub:UpdateApp resiliencehub:UpdateAppVersion resiliencehub:UpdateAppVersionAppComponent resiliencehub:UpdateAppVersionResource resiliencehub:UpdateResiliencyPolicy

服務前綴	動作
resource-explorer-2	resource-explorer-2:AssociateDefaultView
	resource-explorer-2:BatchGetView
	resource-explorer-2:CreateIndex
	resource-explorer-2：CreateResourceExplorerSetup
	resource-explorer-2:CreateView
	resource-explorer-2:DeleteIndex
	resource-explorer-2：DeleteResourceExplorerSetup
	resource-explorer-2:DeleteView
	resource-explorer-2:DisassociateDefaultView
	resource-explorer-2:GetAccountLevelServiceConfiguration
	resource-explorer-2:GetDefaultView
	resource-explorer-2:GetIndex
	resource-explorer-2:GetManagedView
	resource-explorer-2：GetResourceExplorerSetup
	resource-explorer-2：GetServiceIndex
	resource-explorer-2：GetServiceView
	resource-explorer-2:ListIndexes
	resource-explorer-2:ListIndexesForMembers
	resource-explorer-2:ListManagedViews
	resource-explorer-2：ListServiceIndexes
	resource-explorer-2：ListServiceViews

服務前綴	動作
	resource-explorer-2：ListStreamingAccessForServices
	resource-explorer-2:ListSupportedResourceTypes
	resource-explorer-2:ListViews
	resource-explorer-2:Search
	resource-explorer-2:UpdateIndexType
	resource-explorer-2:UpdateView

服務前綴	動作
resource-groups	resource-groups:CancelTagSyncTask resource-groups:GetAccountSettings resource-groups:GetGroup resource-groups:GetGroupConfiguration resource-groups:GetGroupQuery resource-groups:GetTagSyncTask resource-groups:GroupResources resource-groups:ListGroupResources resource-groups:ListGroupingStatuses resource-groups:ListGroups resource-groups:ListTagSyncTasks resource-groups:PutGroupConfiguration resource-groups:SearchResources resource-groups:StartTagSyncTask resource-groups:UngroupResources resource-groups:UpdateAccountSettings resource-groups:UpdateGroup resource-groups:UpdateGroupQuery

服務前綴	動作
robomaker	robomaker:BatchDeleteWorlds
	robomaker:BatchDescribeSimulationJob
	robomaker:CancelDeploymentJob
	robomaker:CancelSimulationJob
	robomaker:CancelSimulationJobBatch
	robomaker:CancelWorldExportJob
	robomaker:CancelWorldGenerationJob
	robomaker:CreateDeploymentJob
	robomaker:CreateFleet
	robomaker:CreateRobot
	robomaker:CreateRobotApplication
	robomaker:CreateRobotApplicationVersion
	robomaker:CreateSimulationApplication
	robomaker:CreateSimulationApplicationVersion
	robomaker:CreateSimulationJob
	robomaker:CreateWorldExportJob
	robomaker:CreateWorldGenerationJob
	robomaker:CreateWorldTemplate
	robomaker:DeleteFleet
	robomaker:DeleteRobot
	robomaker:DeleteRobotApplication

服務前綴	動作
	robomaker:DeleteSimulationApplication
	robomaker:DeleteWorldTemplate
	robomaker:DeregisterRobot
	robomaker:DescribeDeploymentJob
	robomaker:DescribeFleet
	robomaker:DescribeRobot
	robomaker:DescribeRobotApplication
	robomaker:DescribeSimulationApplication
	robomaker:DescribeSimulationJob
	robomaker:DescribeSimulationJobBatch
	robomaker:DescribeWorld
	robomaker:DescribeWorldExportJob
	robomaker:DescribeWorldGenerationJob
	robomaker:DescribeWorldTemplate
	robomaker:GetWorldTemplateBody
	robomaker:ListDeploymentJobs
	robomaker:ListFleets
	robomaker:ListRobotApplications
	robomaker:ListRobots
	robomaker:ListSimulationApplications
	robomaker:ListSimulationJobBatches

服務前綴	動作
	robomaker:ListSimulationJobs
	robomaker:ListWorldExportJobs
	robomaker:ListWorldGenerationJobs
	robomaker:ListWorldTemplates
	robomaker:ListWorlds
	robomaker:RegisterRobot
	robomaker:RestartSimulationJob
	robomaker:StartSimulationJobBatch
	robomaker:SyncDeploymentJob
	robomaker:UpdateRobotApplication
	robomaker:UpdateSimulationApplication
	robomaker:UpdateWorldTemplate

服務前綴	動作
rolesanywhere	rolesanywhere:CreateProfile rolesanywhere:CreateTrustAnchor rolesanywhere:DeleteAttributeMapping rolesanywhere:DeleteCrl rolesanywhere:DeleteProfile rolesanywhere:DeleteTrustAnchor rolesanywhere:DisableCrl rolesanywhere:DisableProfile rolesanywhere:DisableTrustAnchor rolesanywhere:EnableCrl rolesanywhere:EnableProfile rolesanywhere:EnableTrustAnchor rolesanywhere:GetCrl rolesanywhere:GetProfile rolesanywhere:GetSubject rolesanywhere:GetTrustAnchor rolesanywhere:ImportCrl rolesanywhere:ListCrls rolesanywhere:ListProfiles rolesanywhere:ListSubjects rolesanywhere:ListTrustAnchors

服務前綴	動作
	rolesanywhere:PutAttributeMapping rolesanywhere:PutNotificationSettings rolesanywhere:ResetNotificationSettings rolesanywhere:UpdateCrl rolesanywhere:UpdateProfile rolesanywhere:UpdateTrustAnchor

服務前綴	動作
route53	route53:ActivateKeySigningKey
	route53:AssociateVPCWithHostedZone
	route53:ChangeCidrCollection
	route53:ChangeResourceRecordSets
	route53:CreateCidrCollection
	route53:CreateHealthCheck
	route53:CreateHostedZone
	route53:CreateKeySigningKey
	route53:CreateQueryLoggingConfig
	route53:CreateReusableDelegationSet
	route53:CreateTrafficPolicy
	route53:CreateTrafficPolicyInstance
	route53:CreateTrafficPolicyVersion
	route53:CreateVPCAssociationAuthorization
	route53:DeactivateKeySigningKey
	route53:DeleteCidrCollection
	route53:DeleteHealthCheck
	route53:DeleteHostedZone
	route53:DeleteKeySigningKey
	route53:DeleteQueryLoggingConfig
	route53:DeleteReusableDelegationSet

服務前綴	動作
	route53:DeleteTrafficPolicy
	route53:DeleteTrafficPolicyInstance
	route53:DeleteVPCAssociationAuthorization
	route53:DisableHostedZoneDNSSEC
	route53:DisassociateVPCFromHostedZone
	route53:EnableHostedZoneDNSSEC
	route53:GetAccountLimit
	route53:GetChange
	route53:GetCheckerIpRanges
	route53:GetDNSSEC
	route53:GetGeoLocation
	route53:GetHealthCheck
	route53:GetHealthCheckCount
	route53:GetHealthCheckLastFailureReason
	route53:GetHealthCheckStatus
	route53:GetHostedZone
	route53:GetHostedZoneCount
	route53:GetHostedZoneLimit
	route53:GetQueryLoggingConfig
	route53:GetReusableDelegationSet
	route53:GetReusableDelegationSetLimit

服務前綴	動作
	route53:GetTrafficPolicy
	route53:GetTrafficPolicyInstance
	route53:GetTrafficPolicyInstanceCount
	route53:ListCidrBlocks
	route53:ListCidrCollections
	route53:ListCidrLocations
	route53:ListGeoLocations
	route53:ListHealthChecks
	route53:ListHostedZones
	route53:ListHostedZonesByName
	route53:ListHostedZonesByVPC
	route53:ListQueryLoggingConfigs
	route53:ListResourceRecordSets
	route53:ListReusableDelegationSets
	route53:ListTrafficPolicies
	route53:ListTrafficPolicyInstances
	route53:ListTrafficPolicyInstancesByHostedZone
	route53:ListTrafficPolicyInstancesByPolicy
	route53:ListTrafficPolicyVersions
	route53:ListVPCAssociationAuthorizations
	route53:TestDNSAnswer

服務前綴	動作
	route53:UpdateHealthCheck route53:UpdateHostedZoneComment route53:UpdateTrafficPolicyComment route53:UpdateTrafficPolicyInstance

服務前綴	動作
route53-recovery-control-config	route53-recovery-control-config:CreateCluster route53-recovery-control-config:CreateControlPanel route53-recovery-control-config:CreateRoutingControl route53-recovery-control-config:CreateSafetyRule route53-recovery-control-config:DeleteCluster route53-recovery-control-config:DeleteControlPanel route53-recovery-control-config:DeleteRoutingControl route53-recovery-control-config:DeleteSafetyRule route53-recovery-control-config:DescribeCluster route53-recovery-control-config:DescribeControlPanel route53-recovery-control-config:DescribeRoutingControl route53-recovery-control-config:DescribeSafetyRule route53-recovery-control-config:GetResourcePolicy route53-recovery-control-config:ListAssociatedRoute53HealthChecks route53-recovery-control-config:ListClusters route53-recovery-control-config:ListControlPanels route53-recovery-control-config:ListRoutingControls route53-recovery-control-config:ListSafetyRules route53-recovery-control-config:UpdateCluster route53-recovery-control-config:UpdateControlPanel

服務前綴	動作
	route53-recovery-control-config:UpdateRoutingControl route53-recovery-control-config:UpdateSafetyRule

服務前綴	動作
route53-recovery-readiness	route53-recovery-readiness:CreateCell route53-recovery-readiness:CreateCrossAccountAuthorization route53-recovery-readiness:CreateReadinessCheck route53-recovery-readiness:CreateRecoveryGroup route53-recovery-readiness:CreateResourceSet route53-recovery-readiness:DeleteCell route53-recovery-readiness:DeleteCrossAccountAuthorization route53-recovery-readiness:DeleteReadinessCheck route53-recovery-readiness:DeleteRecoveryGroup route53-recovery-readiness:DeleteResourceSet route53-recovery-readiness:GetArchitectureRecommendations route53-recovery-readiness:GetCell route53-recovery-readiness:GetCellReadinessSummary route53-recovery-readiness:GetReadinessCheck route53-recovery-readiness:GetReadinessCheckResourceStatus route53-recovery-readiness:GetReadinessCheckStatus route53-recovery-readiness:GetRecoveryGroup route53-recovery-readiness:GetRecoveryGroupReadinessSummary route53-recovery-readiness:GetResourceSet route53-recovery-readiness:ListCells route53-recovery-readiness:ListCrossAccountAuthorizations

服務前綴	動作
	route53-recovery-readiness:ListReadinessChecks
	route53-recovery-readiness:ListRecoveryGroups
	route53-recovery-readiness:ListResourceSets
	route53-recovery-readiness:ListRules
	route53-recovery-readiness:UpdateCell
	route53-recovery-readiness:UpdateReadinessCheck
	route53-recovery-readiness:UpdateRecoveryGroup
	route53-recovery-readiness:UpdateResourceSet

服務前綴	動作
route53resolver	route53resolver:AssociateFirewallRuleGroup route53resolver:AssociateResolverEndpointIpAddress route53resolver:AssociateResolverQueryLogConfig route53resolver:AssociateResolverRule route53resolver:CreateFirewallDomainList route53resolver:CreateFirewallRule route53resolver:CreateFirewallRuleGroup route53resolver:CreateResolverEndpoint route53resolver:CreateResolverQueryLogConfig route53resolver:CreateResolverRule route53resolver:DeleteFirewallDomainList route53resolver:DeleteFirewallRule route53resolver:DeleteFirewallRuleGroup route53resolver:DeleteOutpostResolver route53resolver:DeleteResolverEndpoint route53resolver:DeleteResolverQueryLogConfig route53resolver:DeleteResolverRule route53resolver:DisassociateFirewallRuleGroup route53resolver:DisassociateResolverEndpointIpAddress route53resolver:DisassociateResolverQueryLogConfig route53resolver:DisassociateResolverRule

服務前綴	動作
	route53resolver:GetFirewallConfig route53resolver:GetFirewallDomainList route53resolver:GetFirewallRuleGroup route53resolver:GetFirewallRuleGroupAssociation route53resolver:GetFirewallRuleGroupPolicy route53resolver:GetOutpostResolver route53resolver:GetResolverConfig route53resolver:GetResolverDnssecConfig route53resolver:GetResolverEndpoint route53resolver:GetResolverQueryLogConfig route53resolver:GetResolverQueryLogConfigAssociation route53resolver:GetResolverQueryLogConfigPolicy route53resolver:GetResolverRule route53resolver:GetResolverRuleAssociation route53resolver:GetResolverRulePolicy route53resolver:ImportFirewallDomains route53resolver:ListFirewallConfigs route53resolver:ListFirewallDomainLists route53resolver:ListFirewallDomains route53resolver:ListFirewallRuleGroupAssociations route53resolver:ListFirewallRuleGroups

服務前綴	動作
	route53resolver：ListFirewallRuleTypes route53resolver:ListFirewallRules route53resolver:ListOutpostResolvers route53resolver:ListResolverConfigs route53resolver:ListResolverDnssecConfigs route53resolver:ListResolverEndpointIpAddresses route53resolver:ListResolverEndpoints route53resolver:ListResolverQueryLogConfigAssociations route53resolver:ListResolverQueryLogConfigs route53resolver:ListResolverRuleAssociations route53resolver:ListResolverRules route53resolver:PutFirewallRuleGroupPolicy route53resolver:PutResolverQueryLogConfigPolicy route53resolver:UpdateFirewallConfig route53resolver:UpdateFirewallDomains route53resolver:UpdateFirewallRule route53resolver:UpdateFirewallRuleGroupAssociation route53resolver:UpdateOutpostResolver route53resolver:UpdateResolverConfig route53resolver:UpdateResolverDnssecConfig route53resolver:UpdateResolverEndpoint

服務前綴	動作
	route53resolver:UpdateResolverRule
rum	rum:BatchCreateRumMetricDefinitions
	rum:BatchDeleteRumMetricDefinitions
	rum:BatchGetRumMetricDefinitions
	rum:CreateAppMonitor
	rum:DeleteAppMonitor
	rum:DeleteResourcePolicy
	rum:DeleteRumMetricsDestination
	rum:GetAppMonitor
	rum:GetAppMonitorData
	rum:GetResourcePolicy
	rum:ListAppMonitors
	rum:ListRumMetricsDestinations
	rum:PutResourcePolicy
	rum:PutRumMetricsDestination
	rum:UpdateAppMonitor
	rum:UpdateRumMetricDefinition

服務前綴	動作
s3	s3:AssociateAccessGrantsIdentityCenter s3:CreateAccessGrant s3:CreateAccessGrantsInstance s3:CreateAccessGrantsLocation s3:CreateAccessPoint s3:CreateAccessPointForObjectLambda s3:CreateBucket s3:CreateBucketMetadataTableConfiguration s3:CreateJob s3:CreateMultiRegionAccessPoint s3:DeleteAccessGrant s3:DeleteAccessGrantsInstance s3:DeleteAccessGrantsInstanceResourcePolicy s3:DeleteAccessGrantsLocation s3:DeleteAccessPoint s3:DeleteAccessPointForObjectLambda s3:DeleteAccessPointPolicy s3:DeleteAccessPointPolicyForObjectLambda s3:DeleteBucket s3:DeleteBucketMetadataTableConfiguration s3:DeleteBucketPolicy

服務前綴	動作
	s3:DeleteBucketWebsite s3:DeleteMultiRegionAccessPoint s3:DeleteStorageLensConfiguration s3:DescribeJob s3:DescribeMultiRegionAccessPointOperation s3:DissociateAccessGrantsIdentityCenter s3:GetAccelerateConfiguration s3:GetAccessGrant s3:GetAccessGrantsInstance s3:GetAccessGrantsInstanceForPrefix s3:GetAccessGrantsInstanceResourcePolicy s3:GetAccessGrantsLocation s3:GetAccessPoint s3:GetAccessPointConfigurationForObjectLambda s3:GetAccessPointForObjectLambda s3:GetAccessPointPolicy s3:GetAccessPointPolicyForObjectLambda s3:GetAccessPointPolicyStatus s3:GetAccessPointPolicyStatusForObjectLambda s3:GetAccountPublicAccessBlock s3:GetAnalyticsConfiguration

服務前綴	動作
	s3：GetBucketAbac s3:GetBucketAcl s3:GetBucketCORS s3:GetBucketLocation s3:GetBucketLogging s3:GetBucketNotification s3:GetBucketObjectLockConfiguration s3:GetBucketOwnershipControls s3:GetBucketPolicy s3:GetBucketPolicyStatus s3:GetBucketPublicAccessBlock s3:GetBucketRequestPayment s3:GetBucketVersioning s3:GetBucketWebsite s3:GetDataAccess s3:GetEncryptionConfiguration s3:GetIntelligentTieringConfiguration s3:GetInventoryConfiguration s3:GetLifecycleConfiguration s3:GetMetricsConfiguration s3:GetMultiRegionAccessPoint

服務前綴	動作
	s3:GetMultiRegionAccessPointPolicy s3:GetMultiRegionAccessPointPolicyStatus s3:GetMultiRegionAccessPointRoutes s3:GetReplicationConfiguration s3:GetStorageLensConfiguration s3:GetStorageLensDashboard s3:ListAccessGrants s3:ListAccessGrantsInstances s3:ListAccessGrantsLocations s3:ListAccessPoints s3:ListAccessPointsForObjectLambda s3:ListAllMyBuckets s3:ListBucketMultipartUploads s3:ListCallerAccessGrants s3:ListJobs s3:ListMultiRegionAccessPoints s3:ListStorageLensConfigurations s3:PutAccelerateConfiguration s3:PutAccessGrantsInstanceResourcePolicy s3:PutAccessPointConfigurationForObjectLambda s3:PutAccessPointPolicy

服務前綴	動作
	s3:PutAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:PutAnalyticsConfiguration s3：PutBucketAbac s3:PutBucketAcl s3:PutBucketCORS s3:PutBucketLogging s3:PutBucketNotification s3:PutBucketObjectLockConfiguration s3:PutBucketOwnershipControls s3:PutBucketPolicy s3:PutBucketPublicAccessBlock s3:PutBucketRequestPayment s3:PutBucketVersioning s3:PutBucketWebsite s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutMetricsConfiguration s3:PutMultiRegionAccessPointPolicy

服務前綴	動作
	s3:PutReplicationConfiguration s3:PutStorageLensConfiguration s3:SubmitMultiRegionAccessPointRoutes s3:UpdateAccessGrantsLocation s3：UpdateBucketMetadataAnnotationTableConfiguration s3:UpdateBucketMetadataJournalTableConfiguration s3:UpdateJobPriority s3:UpdateJobStatus
s3-outposts	s3-outposts:CreateEndpoint s3-outposts:DeleteEndpoint s3-outposts:ListEndpoints s3-outposts:ListOutpostsWithS3 s3-outposts:ListSharedEndpoints

服務前綴	動作
sagemaker-geospatial	sagemaker-geospatial:DeleteEarthObservationJob sagemaker-geospatial:DeleteVectorEnrichmentJob sagemaker-geospatial:ExportEarthObservationJob sagemaker-geospatial:ExportVectorEnrichmentJob sagemaker-geospatial:GetEarthObservationJob sagemaker-geospatial:GetRasterDataCollection sagemaker-geospatial:GetTile sagemaker-geospatial:GetVectorEnrichmentJob sagemaker-geospatial:ListEarthObservationJobs sagemaker-geospatial:ListRasterDataCollections sagemaker-geospatial:ListVectorEnrichmentJobs sagemaker-geospatial:SearchRasterDataCollection sagemaker-geospatial:StartEarthObservationJob sagemaker-geospatial:StartVectorEnrichmentJob sagemaker-geospatial:StopEarthObservationJob sagemaker-geospatial:StopVectorEnrichmentJob

服務前綴	動作
savingsplans	savingsplans:CreateSavingsPlan savingsplans:DeleteQueuedSavingsPlan savingsplans:DescribeSavingsPlanRates savingsplans:DescribeSavingsPlans savingsplans:DescribeSavingsPlansOfferingRates savingsplans:DescribeSavingsPlansOfferings savingsplans:ReturnSavingsPlan

服務前綴	動作
schemas	schemas:CreateDiscoverer schemas:CreateRegistry schemas:CreateSchema schemas:DeleteDiscoverer schemas:DeleteRegistry schemas:DeleteResourcePolicy schemas:DeleteSchema schemas:DeleteSchemaVersion schemas:DescribeCodeBinding schemas:DescribeDiscoverer schemas:DescribeRegistry schemas:DescribeSchema schemas:ExportSchema schemas:GetCodeBindingSource schemas:GetDiscoveredSchema schemas:GetResourcePolicy schemas:ListDiscoverers schemas:ListRegistries schemas:ListSchemaVersions schemas:ListSchemas schemas:PutCodeBinding

服務前綴	動作
	schemas:PutResourcePolicy schemas:SearchSchemas schemas:StartDiscoverer schemas:StopDiscoverer schemas:UpdateDiscoverer schemas:UpdateRegistry schemas:UpdateSchema
sdb	sdb:CreateDomain sdb:DeleteDomain sdb:DomainMetadata sdb:ListDomains

服務前綴	動作
secretsmanager	secretsmanager:CancelRotateSecret secretsmanager:CreateSecret secretsmanager:DeleteResourcePolicy secretsmanager:DeleteSecret secretsmanager:DescribeSecret secretsmanager:GetRandomPassword secretsmanager:GetResourcePolicy secretsmanager:GetSecretValue secretsmanager:ListSecretVersionIds secretsmanager:ListSecrets secretsmanager:PutResourcePolicy secretsmanager:PutSecretValue secretsmanager:RemoveRegionsFromReplication secretsmanager:ReplicateSecretToRegions secretsmanager:RestoreSecret secretsmanager:RotateSecret secretsmanager:StopReplicationToReplica secretsmanager:UpdateSecret secretsmanager:ValidateResourcePolicy

服務前綴	動作
securityhub	securityhub:AcceptAdministratorInvitation
	securityhub:AcceptInvitation
	securityhub:BatchDeleteAutomationRules
	securityhub:BatchDisableStandards
	securityhub:BatchEnableStandards
	securityhub:BatchGetAutomationRules
	securityhub:BatchGetConfigurationPolicyAssociations
	securityhub：BatchGetEnabledRegionsV2
	securityhub:BatchGetSecurityControls
	securityhub:BatchGetStandardsControlAssociations
	securityhub:BatchImportFindings
	securityhub:BatchUpdateAutomationRules
	securityhub:BatchUpdateFindings
	securityhub:BatchUpdateStandardsControlAssociations
	securityhub:ConnectorRegistrationsV2
	securityhub:CreateActionTarget
	securityhub:CreateAggregatorV2
	securityhub:CreateAutomationRule
	securityhub:CreateAutomationRuleV2
	securityhub:CreateConfigurationPolicy
	securityhub：CreateConnector

服務前綴	動作
	securityhub:CreateConnectorV2
	securityhub:CreateFindingAggregator
	securityhub:CreateInsight
	securityhub:CreateMembers
	securityhub:CreateTicketV2
	securityhub:DeclineInvitations
	securityhub:DeleteActionTarget
	securityhub:DeleteAggregatorV2
	securityhub:DeleteAutomationRuleV2
	securityhub:DeleteConfigurationPolicy
	securityhub：DeleteConnector
	securityhub:DeleteConnectorV2
	securityhub:DeleteFindingAggregator
	securityhub:DeleteInsight
	securityhub:DeleteInvitations
	securityhub:DeleteMembers
	securityhub:DescribeActionTargets
	securityhub:DescribeHub
	securityhub:DescribeOrganizationConfiguration
	securityhub:DescribeProducts
	securityhub:DescribeSecurityHubV2

服務前綴	動作
	securityhub:DescribeStandards
	securityhub:DisableImportFindingsForProduct
	securityhub:DisableOrganizationAdminAccount
	securityhub:DisableSecurityHub
	securityhub：DisableSecurityHubFeatureV2
	securityhub:DisableSecurityHubV2
	securityhub:DisassociateFromAdministratorAccount
	securityhub:DisassociateFromMasterAccount
	securityhub:DisassociateMembers
	securityhub:EnableImportFindingsForProduct
	securityhub:EnableOrganizationAdminAccount
	securityhub:EnableSecurityHub
	securityhub：EnableSecurityHubFeatureV2
	securityhub:GetAdministratorAccount
	securityhub:GetAggregatorV2
	securityhub:GetAutomationRuleV2
	securityhub:GetConfigurationPolicy
	securityhub:GetConfigurationPolicyAssociation
	securityhub：GetConnector
	securityhub:GetConnectorV2
	securityhub：GetCoverageStatisticsV2

服務前綴	動作
	securityhub:GetEnabledStandards
	securityhub:GetFindingAggregator
	securityhub:GetFindingHistory
	securityhub:GetFindings
	securityhub:GetInsightResults
	securityhub:GetInsights
	securityhub:GetInvitationsCount
	securityhub:GetMasterAccount
	securityhub:GetMembers
	securityhub:GetSecurityControlDefinition
	securityhub:InviteMembers
	securityhub:ListAggregatorsV2
	securityhub:ListAutomationRules
	securityhub:ListAutomationRulesV2
	securityhub:ListConfigurationPolicies
	securityhub:ListConfigurationPolicyAssociations
	securityhub：ListConnectors
	securityhub:ListConnectorsV2
	securityhub:ListEnabledProductsForImport
	securityhub:ListFindingAggregators
	securityhub:ListInvitations

服務前綴	動作
	securityhub:ListMembers
	securityhub:ListOrganizationAdminAccounts
	securityhub:ListSecurityControlDefinitions
	securityhub:ListStandardsControlAssociations
	securityhub:StartConfigurationPolicyAssociation
	securityhub:StartConfigurationPolicyDisassociation
	securityhub:UpdateActionTarget
	securityhub:UpdateAggregatorV2
	securityhub:UpdateAutomationRuleV2
	securityhub:UpdateConfigurationPolicy
	securityhub：UpdateConnector
	securityhub:UpdateConnectorV2
	securityhub:UpdateFindingAggregator
	securityhub:UpdateFindings
	securityhub:UpdateInsight
	securityhub:UpdateOrganizationConfiguration
	securityhub:UpdateSecurityControl
	securityhub:UpdateSecurityHubConfiguration

服務前綴	動作
securitylake	securitylake:CreateAwsLogSource securitylake:CreateCustomLogSource securitylake:CreateDataLakeExceptionSubscription securitylake:CreateDataLakeOrganizationConfiguration securitylake:CreateSubscriber securitylake:CreateSubscriberNotification securitylake:DeleteAwsLogSource securitylake:DeleteCustomLogSource securitylake:DeleteDataLakeExceptionSubscription securitylake:DeleteDataLakeOrganizationConfiguration securitylake:DeleteSubscriber securitylake:DeleteSubscriberNotification securitylake:DeregisterDataLakeDelegatedAdministrator securitylake:GetDataLakeExceptionSubscription securitylake:GetDataLakeOrganizationConfiguration securitylake:GetDataLakeSources securitylake:GetSubscriber securitylake:ListDataLakes securitylake:ListLogSources securitylake:ListSubscribers securitylake:RegisterDataLakeDelegatedAdministrator

服務前綴	動作
	securitylake:UpdateDataLakeExceptionSubscription securitylake:UpdateSubscriber securitylake:UpdateSubscriberNotification
serverlessrepo	serverlessrepo:CreateApplication serverlessrepo:CreateApplicationVersion serverlessrepo:CreateCloudFormationChangeSet serverlessrepo:CreateCloudFormationTemplate serverlessrepo:DeleteApplication serverlessrepo:GetApplication serverlessrepo:GetApplicationPolicy serverlessrepo:GetCloudFormationTemplate serverlessrepo:ListApplicationDependencies serverlessrepo:ListApplicationVersions serverlessrepo:ListApplications serverlessrepo:PutApplicationPolicy serverlessrepo:UnshareApplication serverlessrepo:UpdateApplication

服務前綴	動作
servicecatalog	servicecatalog:AcceptPortfolioShare servicecatalog:AssociateBudgetWithResource servicecatalog:AssociatePrincipalWithPortfolio servicecatalog:AssociateProductWithPortfolio servicecatalog:AssociateServiceActionWithProvisioningArtifact servicecatalog:BatchAssociateServiceActionWithProvisioningArtifact servicecatalog:BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog:CopyProduct servicecatalog:CreateAttributeGroup servicecatalog:CreateConstraint servicecatalog:CreatePortfolio servicecatalog:CreatePortfolioShare servicecatalog:CreateProduct servicecatalog:CreateProvisionedProductPlan servicecatalog:CreateProvisioningArtifact servicecatalog:CreateServiceAction servicecatalog:DeleteAttributeGroup servicecatalog:DeleteConstraint servicecatalog:DeletePortfolio servicecatalog:DeletePortfolioShare

服務前綴	動作
	servicecatalog:DeleteProduct servicecatalog:DeleteProvisionedProductPlan servicecatalog:DeleteProvisioningArtifact servicecatalog:DeleteServiceAction servicecatalog:DescribeConstraint servicecatalog:DescribeCopyProductStatus servicecatalog:DescribePortfolio servicecatalog:DescribePortfolioShareStatus servicecatalog:DescribePortfolioShares servicecatalog:DescribeProduct servicecatalog:DescribeProductAsAdmin servicecatalog:DescribeProductView servicecatalog:DescribeProvisionedProduct servicecatalog:DescribeProvisionedProductPlan servicecatalog:DescribeProvisioningArtifact servicecatalog:DescribeProvisioningParameters servicecatalog:DescribeRecord servicecatalog:DescribeServiceAction servicecatalog:DescribeServiceActionExecutionParameters servicecatalog:DisableAWSOrganizationsAccess servicecatalog:DisassociateBudgetFromResource

服務前綴	動作
	servicecatalog:DisassociatePrincipalFromPortfolio servicecatalog:DisassociateProductFromPortfolio servicecatalog:DisassociateServiceActionFromProvisioningArtifact servicecatalog:EnableAWSOrganizationsAccess servicecatalog:ExecuteProvisionedProductPlan servicecatalog:ExecuteProvisionedProductServiceAction servicecatalog:GetAWSOrganizationsAccessStatus servicecatalog:GetProvisionedProductOutputs servicecatalog:ImportAsProvisionedProduct servicecatalog:ListAcceptedPortfolioShares servicecatalog:ListAttributeGroups servicecatalog:ListBudgetsForResource servicecatalog:ListConstraintsForPortfolio servicecatalog:ListLaunchPaths servicecatalog:ListOrganizationPortfolioAccess servicecatalog:ListPortfolioAccess servicecatalog:ListPortfolios servicecatalog:ListPortfoliosForProduct servicecatalog:ListPrincipalsForPortfolio servicecatalog:ListProvisionedProductPlans servicecatalog:ListProvisioningArtifacts

服務前綴	動作
	servicecatalog:ListProvisioningArtifactsForServiceAction servicecatalog:ListRecordHistory servicecatalog:ListServiceActions servicecatalog:ListServiceActionsForProvisioningArtifact servicecatalog:ListStackInstancesForProvisionedProduct servicecatalog:NotifyProvisionProductEngineWorkflowResult servicecatalog:NotifyTerminateProvisionedProductEngine WorkflowResult servicecatalog:NotifyUpdateProvisionedProductEngineWor kflowResult servicecatalog:ProvisionProduct servicecatalog:RejectPortfolioShare servicecatalog:ScanProvisionedProducts servicecatalog:SearchProducts servicecatalog:SearchProductsAsAdmin servicecatalog:SearchProvisionedProducts servicecatalog:TerminateProvisionedProduct servicecatalog:UpdateConstraint servicecatalog:UpdatePortfolio servicecatalog:UpdatePortfolioShare servicecatalog:UpdateProduct servicecatalog:UpdateProvisionedProduct

服務前綴	動作
	servicecatalog:UpdateProvisionedProductProperties servicecatalog:UpdateProvisioningArtifact servicecatalog:UpdateServiceAction

服務前綴	動作
servicediscovery	servicediscovery:CreateHttpNamespace
	servicediscovery:CreatePrivateDnsNamespace
	servicediscovery:CreatePublicDnsNamespace
	servicediscovery:CreateService
	servicediscovery:DeleteNamespace
	servicediscovery:DeleteService
	servicediscovery:DeleteServiceAttributes
	servicediscovery:DeregisterInstance
	servicediscovery:GetInstance
	servicediscovery:GetInstancesHealthStatus
	servicediscovery:GetNamespace
	servicediscovery:GetOperation
	servicediscovery:GetService
	servicediscovery:ListInstances
	servicediscovery:ListNamespaces
	servicediscovery:ListOperations
	servicediscovery:ListServices
	servicediscovery:RegisterInstance
	servicediscovery:UpdateHttpNamespace
	servicediscovery:UpdateInstanceCustomHealthStatus
	servicediscovery:UpdatePrivateDnsNamespace

服務前綴	動作
	servicediscovery:UpdatePublicDnsNamespace servicediscovery:UpdateService servicediscovery:UpdateServiceAttributes

服務前綴	動作
servicequotas	servicequotas:AssociateServiceQuotaTemplate servicequotas:CreateSupportCase servicequotas:DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas:DisassociateServiceQuotaTemplate servicequotas:GetAWSDefaultServiceQuota servicequotas:GetAssociationForServiceQuotaTemplate servicequotas：GetAutoManagementConfiguration servicequotas：GetQuotaUtilizationReport servicequotas:GetRequestedServiceQuotaChange servicequotas:GetServiceQuota servicequotas:GetServiceQuotaIncreaseRequestFromTemplate servicequotas:ListAWSDefaultServiceQuotas servicequotas:ListRequestedServiceQuotaChangeHistory servicequotas:ListRequestedServiceQuotaChangeHistoryByQuota servicequotas:ListServiceQuotaIncreaseRequestsInTemplate servicequotas:ListServiceQuotas servicequotas:ListServices servicequotas:PutServiceQuotaIncreaseRequestIntoTemplate servicequotas:RequestServiceQuotaIncrease servicequotas：StartAutoManagement servicequotas：StartQuotaUtilizationReport

服務前綴	動作
	servicequotas：StopAutoManagement servicequotas：UpdateAutoManagement

服務前綴	動作
ses	ses:BatchGetMetricData
	ses:CloneReceiptRuleSet
	ses:CreateAddonInstance
	ses:CreateAddonSubscription
	ses:CreateAddressList
	ses:CreateAddressListImportJob
	ses:CreateArchive
	ses:CreateConfigurationSet
	ses:CreateConfigurationSetEventDestination
	ses:CreateConfigurationSetTrackingOptions
	ses:CreateContact
	ses:CreateContactList
	ses:CreateCustomVerificationEmailTemplate
	ses:CreateDedicatedIpPool
	ses:CreateDeliverabilityTestReport
	ses:CreateEmailIdentity
	ses:CreateEmailIdentityPolicy
	ses:CreateEmailTemplate
	ses:CreateImportJob
	ses:CreateIngressPoint
	ses:CreateMultiRegionEndpoint

服務前綴	動作
	ses:CreateReceiptFilter
	ses:CreateReceiptRule
	ses:CreateReceiptRuleSet
	ses:CreateRelay
	ses:CreateRuleSet
	ses:CreateTemplate
	ses:CreateTenant
	ses:CreateTenantResourceAssociation
	ses:CreateTrafficPolicy
	ses:DeleteAddonInstance
	ses:DeleteAddonSubscription
	ses:DeleteAddressList
	ses:DeleteArchive
	ses:DeleteConfigurationSet
	ses:DeleteConfigurationSetEventDestination
	ses:DeleteConfigurationSetTrackingOptions
	ses:DeleteContact
	ses:DeleteContactList
	ses:DeleteCustomVerificationEmailTemplate
	ses:DeleteDedicatedIpPool
	ses:DeleteEmailIdentity

服務前綴	動作
	ses:DeleteEmailIdentityPolicy
	ses:DeleteEmailTemplate
	ses:DeleteIdentity
	ses:DeleteIdentityPolicy
	ses:DeleteIngressPoint
	ses:DeleteMultiRegionEndpoint
	ses:DeleteReceiptFilter
	ses:DeleteReceiptRule
	ses:DeleteReceiptRuleSet
	ses:DeleteRelay
	ses:DeleteRuleSet
	ses:DeleteSuppressedDestination
	ses:DeleteTemplate
	ses:DeleteTenant
	ses:DeleteTenantResourceAssociation
	ses:DeleteTrafficPolicy
	ses:DeleteVerifiedEmailAddress
	ses:DeregisterMemberFromAddressList
	ses:DescribeActiveReceiptRuleSet
	ses:DescribeConfigurationSet
	ses:DescribeReceiptRule

服務前綴	動作
	ses:DescribeReceiptRuleSet
	ses:GetAccount
	ses:GetAccountSendingEnabled
	ses:GetAddonInstance
	ses:GetAddonSubscription
	ses:GetAddressList
	ses:GetArchive
	ses:GetArchiveExport
	ses:GetArchiveMessage
	ses:GetArchiveMessageContent
	ses:GetArchiveSearch
	ses:GetArchiveSearchResults
	ses:GetBlacklistReports
	ses:GetConfigurationSet
	ses:GetConfigurationSetEventDestinations
	ses:GetContact
	ses:GetContactList
	ses:GetCustomVerificationEmailTemplate
	ses:GetDedicatedIp
	ses:GetDedicatedIpPool
	ses:GetDedicatedIps

服務前綴	動作
	ses:GetDeliverabilityDashboardOptions
	ses:GetDeliverabilityTestReport
	ses:GetDomainDeliverabilityCampaign
	ses:GetDomainStatisticsReport
	ses：GetEmailAddressInsights
	ses:GetEmailIdentity
	ses:GetEmailIdentityPolicies
	ses:GetEmailTemplate
	ses:GetIdentityDkimAttributes
	ses:GetIdentityMailFromDomainAttributes
	ses:GetIdentityNotificationAttributes
	ses:GetIdentityPolicies
	ses:GetIdentityVerificationAttributes
	ses:GetImportJob
	ses:GetIngressPoint
	ses:GetMemberOfAddressList
	ses:GetMessageInsights
	ses:GetMultiRegionEndpoint
	ses:GetRelay
	ses:GetRuleSet
	ses:GetSendQuota

服務前綴	動作
	ses:GetSendStatistics
	ses:GetSuppressedDestination
	ses:GetTemplate
	ses:GetTenant
	ses:GetTrafficPolicy
	ses:ListAddonInstances
	ses:ListAddonSubscriptions
	ses:ListAddressListImportJobs
	ses:ListAddressLists
	ses:ListArchiveExports
	ses:ListArchiveSearches
	ses:ListArchives
	ses:ListConfigurationSets
	ses:ListContactLists
	ses:ListContacts
	ses:ListCustomVerificationEmailTemplates
	ses:ListDedicatedIpPools
	ses:ListDeliverabilityTestReports
	ses:ListDomainDeliverabilityCampaigns
	ses:ListEmailIdentities
	ses:ListEmailTemplates

服務前綴	動作
	ses:ListExportJobs
	ses:ListIdentities
	ses:ListIdentityPolicies
	ses:ListImportJobs
	ses:ListIngressPoints
	ses:ListMembersOfAddressList
	ses:ListMultiRegionEndpoints
	ses:ListReceiptFilters
	ses:ListReceiptRuleSets
	ses:ListRecommendations
	ses:ListRelays
	ses:ListReputationEntities
	ses:ListResourceTenants
	ses:ListRuleSets
	ses:ListSuppressedDestinations
	ses:ListTemplates
	ses:ListTenantResources
	ses:ListTenants
	ses:ListTrafficPolicies
	ses:ListVerifiedEmailAddresses
	ses:PutAccountDedicatedIpWarmupAttributes

服務前綴	動作
	ses:PutAccountDetails
	ses:PutAccountSendingAttributes
	ses:PutAccountSuppressionAttributes
	ses:PutAccountVdmAttributes
	ses:PutConfigurationSetArchivingOptions
	ses:PutConfigurationSetDeliveryOptions
	ses:PutConfigurationSetReputationOptions
	ses:PutConfigurationSetSendingOptions
	ses:PutConfigurationSetSuppressionOptions
	ses:PutConfigurationSetTrackingOptions
	ses:PutConfigurationSetVdmOptions
	ses:PutDedicatedIpInPool
	ses:PutDedicatedIpPoolScalingAttributes
	ses:PutDedicatedIpWarmupAttributes
	ses:PutDeliverabilityDashboardOption
	ses:PutEmailIdentityConfigurationSetAttributes
	ses:PutEmailIdentityDkimAttributes
	ses:PutEmailIdentityDkimSigningAttributes
	ses:PutEmailIdentityFeedbackAttributes
	ses:PutEmailIdentityMailFromAttributes
	ses:PutIdentityPolicy

服務前綴	動作
	ses:PutSuppressedDestination
	ses：PutTenantSuppressionAttributes
	ses:RegisterMemberToAddressList
	ses:ReorderReceiptRuleSet
	ses:SendBounce
	ses:SendCustomVerificationEmail
	ses:SetActiveReceiptRuleSet
	ses:SetIdentityDkimEnabled
	ses:SetIdentityFeedbackForwardingEnabled
	ses:SetIdentityHeadersInNotificationsEnabled
	ses:SetIdentityMailFromDomain
	ses:SetIdentityNotificationTopic
	ses:SetReceiptRulePosition
	ses:StartArchiveExport
	ses:StartArchiveSearch
	ses:StopArchiveExport
	ses:StopArchiveSearch
	ses:TestRenderEmailTemplate
	ses:TestRenderTemplate
	ses:UpdateAccountSendingEnabled
	ses:UpdateArchive

服務前綴	動作
	ses:UpdateConfigurationSetEventDestination
	ses:UpdateConfigurationSetReputationMetricsEnabled
	ses:UpdateConfigurationSetSendingEnabled
	ses:UpdateConfigurationSetTrackingOptions
	ses:UpdateContact
	ses:UpdateContactList
	ses:UpdateCustomVerificationEmailTemplate
	ses:UpdateEmailIdentityPolicy
	ses:UpdateEmailTemplate
	ses:UpdateIngressPoint
	ses:UpdateReceiptRule
	ses:UpdateRelay
	ses:UpdateRuleSet
	ses:UpdateTemplate
	ses:UpdateTrafficPolicy
	ses:VerifyDomainDkim
	ses:VerifyDomainIdentity
	ses:VerifyEmailAddress
	ses:VerifyEmailIdentity

服務前綴	動作
shield	shield:AssociateDRTLogBucket shield:AssociateHealthCheck shield:AssociateProactiveEngagementDetails shield:CreateProtection shield:CreateProtectionGroup shield:CreateSubscription shield:DeleteProtection shield:DeleteProtectionGroup shield:DeleteSubscription shield:DescribeAttack shield:DescribeAttackStatistics shield:DescribeDRTAccess shield:DescribeEmergencyContactSettings shield:DescribeProtection shield:DescribeProtectionGroup shield:DescribeSubscription shield:DisableApplicationLayerAutomaticResponse shield:DisableProactiveEngagement shield:DisassociateDRTLogBucket shield:DisassociateDRTRole shield:DisassociateHealthCheck

服務前綴	動作
	shield:EnableApplicationLayerAutomaticResponse shield:EnableProactiveEngagement shield:GetSubscriptionState shield:ListAttacks shield:ListProtectionGroups shield:ListProtections shield:ListResourcesInProtectionGroup shield:UpdateApplicationLayerAutomaticResponse shield:UpdateEmergencyContactSettings shield:UpdateProtectionGroup shield:UpdateSubscription

服務前綴	動作
signer	signer:AddProfilePermission signer:CancelSigningProfile signer:DescribeSigningJob signer:GetSigningPlatform signer:GetSigningProfile signer:ListProfilePermissions signer:ListSigningJobs signer:ListSigningPlatforms signer:ListSigningProfiles signer:PutSigningProfile signer:RemoveProfilePermission signer:RevokeSignature signer:RevokeSigningProfile signer:SignPayload signer:StartSigningJob

服務前綴	動作
simspaceweaver	simspaceweaver:CreateSnapshot simspaceweaver:DeleteApp simspaceweaver:DeleteSimulation simspaceweaver:DescribeApp simspaceweaver:DescribeSimulation simspaceweaver:ListApps simspaceweaver:ListSimulations simspaceweaver:StartApp simspaceweaver:StartClock simspaceweaver:StartSimulation simspaceweaver:StopApp simspaceweaver:StopClock simspaceweaver:StopSimulation

服務前綴	動作
sms	sms:CreateApp
	sms:CreateReplicationJob
	sms:DeleteApp
	sms:DeleteAppLaunchConfiguration
	sms:DeleteAppReplicationConfiguration
	sms:DeleteAppValidationConfiguration
	sms:DeleteReplicationJob
	sms:DeleteServerCatalog
	sms:DisassociateConnector
	sms:GenerateChangeSet
	sms:GenerateTemplate
	sms:GetApp
	sms:GetAppLaunchConfiguration
	sms:GetAppReplicationConfiguration
	sms:GetAppValidationConfiguration
	sms:GetAppValidationOutput
	sms:GetConnectors
	sms:GetReplicationJobs
	sms:GetReplicationRuns
	sms:GetServers
	sms:ImportAppCatalog

服務前綴	動作
	sms:ImportServerCatalog
	sms:LaunchApp
	sms:ListApps
	sms:NotifyAppValidationOutput
	sms:PutAppLaunchConfiguration
	sms:PutAppReplicationConfiguration
	sms:PutAppValidationConfiguration
	sms:StartAppReplication
	sms:StartOnDemandAppReplication
	sms:StartOnDemandReplicationRun
	sms:StopAppReplication
	sms:TerminateApp
	sms:UpdateApp
	sms:UpdateReplicationJob

服務前綴	動作
sms-voice	sms-voice:AssociateProtectConfiguration sms-voice:CreateConfigurationSet sms-voice:CreateConfigurationSetEventDestination sms-voice:CreateEventDestination sms-voice:CreateOptOutList sms-voice:CreatePool sms-voice:CreateProtectConfiguration sms-voice:CreateRegistration sms-voice:CreateRegistrationAssociation sms-voice:CreateRegistrationAttachment sms-voice:CreateRegistrationVersion sms-voice:CreateVerifiedDestinationNumber sms-voice:DeleteAccountDefaultProtectConfiguration sms-voice:DeleteConfigurationSet sms-voice:DeleteConfigurationSetEventDestination sms-voice:DeleteDefaultMessageType sms-voice:DeleteDefaultSenderId sms-voice:DeleteEventDestination sms-voice:DeleteKeyword sms-voice:DeleteMediaMessageSpendLimitOverride sms-voice:DeleteOptOutList

服務前綴	動作
	sms-voice:DeleteOptedOutNumber sms-voice:DeletePool sms-voice:DeleteProtectConfiguration sms-voice:DeleteProtectConfigurationRuleSetNumberOverride sms-voice:DeleteRegistration sms-voice:DeleteRegistrationAttachment sms-voice:DeleteResourcePolicy sms-voice:DeleteTextMessageSpendLimitOverride sms-voice:DeleteVerifiedDestinationNumber sms-voice:DeleteVoiceMessageSpendLimitOverride sms-voice:DescribeAccountAttributes sms-voice:DescribeAccountLimits sms-voice:DescribeConfigurationSets sms-voice:DescribeKeywords sms-voice:DescribeOptOutLists sms-voice:DescribeOptedOutNumbers sms-voice:DescribePhoneNumbers sms-voice:DescribePools sms-voice:DescribeProtectConfigurations sms-voice:DescribeRegistrationAttachments sms-voice:DescribeRegistrationFieldDefinitions

服務前綴	動作
	sms-voice:DescribeRegistrationFieldValues sms-voice:DescribeRegistrationSectionDefinitions sms-voice:DescribeRegistrationTypeDefinitions sms-voice:DescribeRegistrationVersions sms-voice:DescribeRegistrations sms-voice:DescribeSenderIds sms-voice:DescribeSpendLimits sms-voice:DescribeVerifiedDestinationNumbers sms-voice:DisassociateOriginationIdentity sms-voice:DisassociateProtectConfiguration sms-voice:DiscardRegistrationVersion sms-voice:GetConfigurationSetEventDestinations sms-voice:GetProtectConfigurationCountryRuleSet sms-voice:GetResourcePolicy sms-voice:ListConfigurationSets sms-voice:ListPoolOriginationIdentities sms-voice:ListProtectConfigurationRuleSetNumberOverrides sms-voice:ListRegistrationAssociations sms-voice:PutKeyword sms-voice:PutOptedOutNumber sms-voice:PutProtectConfigurationRuleSetNumberOverride

服務前綴	動作
	sms-voice:PutResourcePolicy sms-voice:ReleasePhoneNumber sms-voice:ReleaseSenderId sms-voice:RequestPhoneNumber sms-voice:RequestSenderId sms-voice:SendDestinationNumberVerificationCode sms-voice:SetAccountDefaultProtectConfiguration sms-voice:SetDefaultMessageFeedbackEnabled sms-voice:SetDefaultMessageType sms-voice:SetDefaultSenderId sms-voice:SetMediaMessageSpendLimitOverride sms-voice:SetTextMessageSpendLimitOverride sms-voice:SetVoiceMessageSpendLimitOverride sms-voice:SubmitRegistrationVersion sms-voice:UpdateConfigurationSetEventDestination sms-voice:UpdateEventDestination sms-voice:UpdatePhoneNumber sms-voice:UpdatePool sms-voice:UpdateProtectConfiguration sms-voice:UpdateProtectConfigurationCountryRuleSet sms-voice:UpdateSenderId

服務前綴	動作
snowball	snowball:CancelCluster
	snowball:CancelJob
	snowball:CreateAddress
	snowball:CreateCluster
	snowball:CreateJob
	snowball:CreateLongTermPricing
	snowball:CreateReturnShippingLabel
	snowball:DescribeAddress
	snowball:DescribeAddresses
	snowball:DescribeCluster
	snowball:DescribeJob
	snowball:DescribeReturnShippingLabel
	snowball:GetJobManifest
	snowball:GetJobUnlockCode
	snowball:GetSnowballUsage
	snowball:GetSoftwareUpdates
	snowball:ListClusterJobs
	snowball:ListClusters
	snowball:ListCompatibleImages
	snowball:ListJobs
	snowball:ListLongTermPricing

服務前綴	動作
	snowball:ListPickupLocations snowball:ListServiceVersions snowball:UpdateCluster snowball:UpdateJob snowball:UpdateJobShipmentState snowball:UpdateLongTermPricing
sqs	sqs:AddPermission sqs:CancelMessageMoveTask sqs:CreateQueue sqs:DeleteQueue sqs:PurgeQueue sqs:RemovePermission sqs:SetQueueAttributes

服務前綴	動作
ssm	ssm:AssociateOpsItemRelatedItem
	ssm:CancelCommand
	ssm:CancelMaintenanceWindowExecution
	ssm:CreateActivation
	ssm:CreateAssociation
	ssm:CreateAssociationBatch
	ssm:CreateDocument
	ssm:CreateMaintenanceWindow
	ssm:CreateOpsItem
	ssm:CreateOpsMetadata
	ssm:CreatePatchBaseline
	ssm:CreateResourceDataSync
	ssm:DeleteActivation
	ssm:DeleteAssociation
	ssm：DeleteCloudConnector
	ssm:DeleteDocument
	ssm:DeleteInventory
	ssm:DeleteMaintenanceWindow
	ssm:DeleteOpsItem
	ssm:DeleteOpsMetadata
	ssm:DeleteParameter

服務前綴	動作
	ssm:DeleteParameters
	ssm:DeletePatchBaseline
	ssm:DeleteResourceDataSync
	ssm:DeleteResourcePolicy
	ssm:DeregisterManagedInstance
	ssm:DeregisterPatchBaselineForPatchGroup
	ssm:DeregisterTargetFromMaintenanceWindow
	ssm:DeregisterTaskFromMaintenanceWindow
	ssm:DescribeActivations
	ssm:DescribeAssociation
	ssm:DescribeAssociationExecutionTargets
	ssm:DescribeAssociationExecutions
	ssm:DescribeAutomationExecutions
	ssm:DescribeAutomationStepExecutions
	ssm:DescribeAvailablePatches
	ssm:DescribeDocument
	ssm:DescribeDocumentParameters
	ssm:DescribeDocumentPermission
	ssm:DescribeEffectiveInstanceAssociations
	ssm:DescribeEffectivePatchesForPatchBaseline
	ssm:DescribeInstanceAssociationsStatus

服務前綴	動作
	ssm:DescribeInstanceInformation
	ssm:DescribeInstancePatchStates
	ssm:DescribeInstancePatchStatesForPatchGroup
	ssm:DescribeInstancePatches
	ssm:DescribeInstanceProperties
	ssm:DescribeInventoryDeletions
	ssm:DescribeMaintenanceWindowExecutionTaskInvocations
	ssm:DescribeMaintenanceWindowExecutionTasks
	ssm:DescribeMaintenanceWindowExecutions
	ssm:DescribeMaintenanceWindowSchedule
	ssm:DescribeMaintenanceWindowTargets
	ssm:DescribeMaintenanceWindowTasks
	ssm:DescribeMaintenanceWindows
	ssm:DescribeMaintenanceWindowsForTarget
	ssm:DescribeOpsItems
	ssm:DescribeParameters
	ssm:DescribePatchBaselines
	ssm:DescribePatchGroupState
	ssm:DescribePatchGroups
	ssm:DescribePatchProperties
	ssm:DescribeSessions

服務前綴	動作
	ssm:DisassociateOpsItemRelatedItem
	ssm:GetAccessToken
	ssm:GetAutomationExecution
	ssm:GetCalendarState
	ssm：GetCloudConnector
	ssm:GetCommandInvocation
	ssm:GetConnectionStatus
	ssm:GetDefaultPatchBaseline
	ssm:GetDeployablePatchSnapshotForInstance
	ssm:GetDocument
	ssm:GetExecutionPreview
	ssm:GetInventory
	ssm:GetInventorySchema
	ssm:GetMaintenanceWindow
	ssm:GetMaintenanceWindowExecution
	ssm:GetMaintenanceWindowExecutionTask
	ssm:GetMaintenanceWindowExecutionTaskInvocation
	ssm:GetMaintenanceWindowTask
	ssm:GetOpsItem
	ssm:GetOpsMetadata
	ssm:GetOpsSummary

服務前綴	動作
	ssm:GetParameter
	ssm:GetParameterHistory
	ssm:GetParameters
	ssm:GetParametersByPath
	ssm:GetPatchBaseline
	ssm:GetPatchBaselineForPatchGroup
	ssm:GetResourcePolicies
	ssm:GetServiceSetting
	ssm:LabelParameterVersion
	ssm:ListAssociationVersions
	ssm:ListAssociations
	ssm：ListCloudConnectors
	ssm:ListCommandInvocations
	ssm:ListCommands
	ssm:ListComplianceItems
	ssm:ListComplianceSummaries
	ssm:ListDocumentMetadataHistory
	ssm:ListDocumentVersions
	ssm:ListDocuments
	ssm:ListInstanceAssociations
	ssm:ListInventoryEntries

服務前綴	動作
	ssm:ListNodes
	ssm:ListNodesSummary
	ssm:ListOpsItemEvents
	ssm:ListOpsItemRelatedItems
	ssm:ListOpsMetadata
	ssm:ListResourceComplianceSummaries
	ssm:ListResourceDataSync
	ssm:ModifyDocumentPermission
	ssm:PutComplianceItems
	ssm:PutInventory
	ssm:PutParameter
	ssm:PutResourcePolicy
	ssm:RegisterDefaultPatchBaseline
	ssm:RegisterManagedInstance
	ssm:RegisterPatchBaselineForPatchGroup
	ssm:RegisterTargetWithMaintenanceWindow
	ssm:RegisterTaskWithMaintenanceWindow
	ssm:ResetServiceSetting
	ssm:ResumeSession
	ssm:SendAutomationSignal
	ssm:SendCommand

服務前綴	動作
	ssm:StartAssociationsOnce
	ssm:StartAutomationExecution
	ssm:StartChangeRequestExecution
	ssm:StartSession
	ssm:StopAutomationExecution
	ssm:TerminateSession
	ssm:UnlabelParameterVersion
	ssm:UpdateAssociation
	ssm:UpdateAssociationStatus
	ssm：UpdateCloudConnector
	ssm:UpdateDocument
	ssm:UpdateDocumentDefaultVersion
	ssm:UpdateDocumentMetadata
	ssm:UpdateInstanceInformation
	ssm:UpdateMaintenanceWindow
	ssm:UpdateMaintenanceWindowTarget
	ssm:UpdateMaintenanceWindowTask
	ssm:UpdateManagedInstanceRole
	ssm:UpdateOpsItem
	ssm:UpdateOpsMetadata
	ssm:UpdatePatchBaseline

服務前綴	動作
	ssm:UpdateResourceDataSync ssm:UpdateServiceSetting ssm：ValidateCloudConnector

服務前綴	動作
ssm-incidents	ssm-incidents:BatchGetIncidentFindings ssm-incidents:CreateReplicationSet ssm-incidents:CreateResponsePlan ssm-incidents:CreateTimelineEvent ssm-incidents:DeleteIncidentRecord ssm-incidents:DeleteReplicationSet ssm-incidents:DeleteResourcePolicy ssm-incidents:DeleteResponsePlan ssm-incidents:DeleteTimelineEvent ssm-incidents:GetIncidentRecord ssm-incidents:GetReplicationSet ssm-incidents:GetResourcePolicies ssm-incidents:GetResponsePlan ssm-incidents:GetTimelineEvent ssm-incidents:ListIncidentFindings ssm-incidents:ListIncidentRecords ssm-incidents:ListRelatedItems ssm-incidents:ListReplicationSets ssm-incidents:ListResponsePlans ssm-incidents:ListTimelineEvents ssm-incidents:PutResourcePolicy

服務前綴	動作
	ssm-incidents:StartIncident ssm-incidents:UpdateDeletionProtection ssm-incidents:UpdateIncidentRecord ssm-incidents:UpdateRelatedItems ssm-incidents:UpdateReplicationSet ssm-incidents:UpdateResponsePlan ssm-incidents:UpdateTimelineEvent

服務前綴	動作
ssm-sap	ssm-sap:BackupDatabase
	ssm-sap:DeleteResourcePermission
	ssm-sap:DeregisterApplication
	ssm-sap:GetApplication
	ssm-sap:GetComponent
	ssm-sap：GetConfigurationCheckOperation
	ssm-sap:GetDatabase
	ssm-sap:GetOperation
	ssm-sap:GetResourcePermission
	ssm-sap:ListApplications
	ssm-sap:ListComponents
	ssm-sap：ListConfigurationCheckDefinitions
	ssm-sap：ListConfigurationCheckOperations
	ssm-sap:ListDatabases
	ssm-sap:ListOperationEvents
	ssm-sap:ListOperations
	ssm-sap：ListSubCheckResults
	ssm-sap：ListSubCheckRuleResults
	ssm-sap:PutResourcePermission
	ssm-sap:RegisterApplication
	ssm-sap:RestoreDatabase

服務前綴	動作
	ssm-sap:StartApplication
	ssm-sap:StartApplicationRefresh
	ssm-sap：StartConfigurationChecks
	ssm-sap:StopApplication
	ssm-sap:UpdateApplicationSettings
	ssm-sap:UpdateHANABackupSettings

服務前綴	動作
states	states:CreateActivity states:CreateStateMachine states:CreateStateMachineAlias states:DeleteActivity states:DeleteStateMachine states:DeleteStateMachineAlias states:DeleteStateMachineVersion states:DescribeActivity states:DescribeExecution states:DescribeMapRun states:DescribeStateMachine states:DescribeStateMachineAlias states:DescribeStateMachineForExecution states:GetExecutionHistory states:ListActivities states:ListExecutions states:ListMapRuns states:ListStateMachineAliases states:ListStateMachineVersions states:ListStateMachines states:SendTaskFailure

服務前綴	動作
	states:SendTaskHeartbeat states:SendTaskSuccess states:StartExecution states:StopExecution states:UpdateMapRun states:UpdateStateMachine states:UpdateStateMachineAlias states:ValidateStateMachineDefinition
sts	sts:AssumeRole sts:AssumeRoleWithSAML sts:AssumeRoleWithWebIdentity sts:DecodeAuthorizationMessage sts:GetAccessKeyInfo sts:GetCallerIdentity sts:GetFederationToken sts:GetSessionToken sts：GetWebIdentityToken

服務前綴	動作
swf	swf:DeleteActivityType swf:DeleteWorkflowType swf:DeprecateActivityType swf:DeprecateDomain swf:DeprecateWorkflowType swf:DescribeActivityType swf:DescribeDomain swf:DescribeWorkflowType swf:ListActivityTypes swf:ListDomains swf:ListWorkflowTypes swf:RegisterActivityType swf:RegisterDomain swf:RegisterWorkflowType swf:UndeprecateActivityType swf:UndeprecateDomain swf:UndeprecateWorkflowType

服務前綴	動作
synthetics	synthetics:AssociateResource
	synthetics:CreateCanary
	synthetics:CreateGroup
	synthetics:DeleteCanary
	synthetics:DeleteGroup
	synthetics:DescribeCanaries
	synthetics:DescribeCanariesLastRun
	synthetics:DescribeRuntimeVersions
	synthetics:DisassociateResource
	synthetics:GetCanary
	synthetics:GetCanaryRuns
	synthetics:GetGroup
	synthetics:ListAssociatedGroups
	synthetics:ListGroupResources
	synthetics:ListGroups
	synthetics:StartCanary
	synthetics:StartCanaryDryRun
	synthetics:StopCanary
	synthetics:UpdateCanary

服務前綴	動作
標籤	tag:DescribeReportCreation tag:GetComplianceSummary tag:GetResources tag:StartReportCreation

服務前綴	動作
textract	textract:AnalyzeDocument
	textract:AnalyzeExpense
	textract:AnalyzeID
	textract:CreateAdapter
	textract:CreateAdapterVersion
	textract:DeleteAdapter
	textract:DeleteAdapterVersion
	textract:DetectDocumentText
	textract:GetAdapter
	textract:GetAdapterVersion
	textract:GetDocumentAnalysis
	textract:GetDocumentTextDetection
	textract:GetExpenseAnalysis
	textract:GetLendingAnalysis
	textract:GetLendingAnalysisSummary
	textract:ListAdapterVersions
	textract:ListAdapters
	textract:StartDocumentAnalysis
	textract:StartDocumentTextDetection
	textract:StartExpenseAnalysis
	textract:StartLendingAnalysis

服務前綴	動作
	textract:UpdateAdapter

服務前綴	動作
timestream	timestream:CancelQuery timestream:CreateDatabase timestream:CreateScheduledQuery timestream:CreateTable timestream:DeleteDatabase timestream:DeleteScheduledQuery timestream:DeleteTable timestream:DescribeAccountSettings timestream:DescribeDatabase timestream:DescribeScheduledQuery timestream:DescribeTable timestream:ExecuteScheduledQuery timestream:ListBatchLoadTasks timestream:ListDatabases timestream:ListScheduledQueries timestream:ListTables timestream:PrepareQuery timestream:UpdateAccountSettings timestream:UpdateDatabase timestream:UpdateScheduledQuery timestream:UpdateTable

服務前綴	動作
tnb	tnb:CancelSolNetworkOperation
	tnb:CreateSolFunctionPackage
	tnb:CreateSolNetworkInstance
	tnb:CreateSolNetworkPackage
	tnb:DeleteSolFunctionPackage
	tnb:DeleteSolNetworkInstance
	tnb:DeleteSolNetworkPackage
	tnb:GetSolFunctionInstance
	tnb:GetSolFunctionPackage
	tnb:GetSolFunctionPackageContent
	tnb:GetSolFunctionPackageDescriptor
	tnb:GetSolNetworkInstance
	tnb:GetSolNetworkOperation
	tnb:GetSolNetworkPackage
	tnb:GetSolNetworkPackageContent
	tnb:GetSolNetworkPackageDescriptor
	tnb:InstantiateSolNetworkInstance
	tnb:ListSolFunctionInstances
	tnb:ListSolFunctionPackages
	tnb:ListSolNetworkInstances
	tnb:ListSolNetworkOperations

服務前綴	動作
	tnb:ListSolNetworkPackages tnb:PutSolFunctionPackageContent tnb:PutSolNetworkPackageContent tnb:TerminateSolNetworkInstance tnb:UpdateSolFunctionPackage tnb:UpdateSolNetworkInstance tnb:UpdateSolNetworkPackage tnb:ValidateSolFunctionPackageContent tnb:ValidateSolNetworkPackageContent

服務前綴	動作
transcribe	transcribe:CreateCallAnalyticsCategory transcribe:CreateLanguageModel transcribe:CreateMedicalVocabulary transcribe:CreateVocabulary transcribe:CreateVocabularyFilter transcribe:DeleteCallAnalyticsCategory transcribe:DeleteCallAnalyticsJob transcribe:DeleteLanguageModel transcribe:DeleteMedicalScribeJob transcribe:DeleteMedicalTranscriptionJob transcribe:DeleteMedicalVocabulary transcribe:DeleteTranscriptionJob transcribe:DeleteVocabulary transcribe:DeleteVocabularyFilter transcribe:DescribeLanguageModel transcribe:GetCallAnalyticsCategory transcribe:GetCallAnalyticsJob transcribe:GetMedicalScribeJob transcribe:GetMedicalTranscriptionJob transcribe:GetMedicalVocabulary transcribe:GetTranscriptionJob

服務前綴	動作
	transcribe:GetVocabulary transcribe:GetVocabularyFilter transcribe:ListCallAnalyticsCategories transcribe:ListCallAnalyticsJobs transcribe:ListLanguageModels transcribe:ListMedicalScribeJobs transcribe:ListMedicalTranscriptionJobs transcribe:ListMedicalVocabularies transcribe:ListTranscriptionJobs transcribe:ListVocabularies transcribe:ListVocabularyFilters transcribe:StartCallAnalyticsJob transcribe:StartCallAnalyticsStreamTranscription transcribe:StartCallAnalyticsStreamTranscriptionWebSocket transcribe:StartMedicalScribeJob transcribe:StartMedicalStreamTranscription transcribe:StartMedicalStreamTranscriptionWebSocket transcribe:StartMedicalTranscriptionJob transcribe:StartStreamTranscription transcribe:StartStreamTranscriptionWebSocket transcribe:StartTranscriptionJob

服務前綴	動作
	transcribe:UpdateCallAnalyticsCategory transcribe:UpdateMedicalVocabulary transcribe:UpdateVocabulary transcribe:UpdateVocabularyFilter

服務前綴	動作
傳輸	transfer:CreateAccess
	transfer:CreateAgreement
	transfer:CreateConnector
	transfer:CreateProfile
	transfer:CreateServer
	transfer:CreateUser
	transfer:CreateWebApp
	transfer:CreateWorkflow
	transfer:DeleteAccess
	transfer:DeleteAgreement
	transfer:DeleteCertificate
	transfer:DeleteConnector
	transfer:DeleteHostKey
	transfer:DeleteProfile
	transfer:DeleteServer
	transfer:DeleteSshPublicKey
	transfer:DeleteUser
	transfer:DeleteWebApp
	transfer:DeleteWebAppCustomization
	transfer:DeleteWorkflow
	transfer:DescribeAccess

服務前綴	動作
	transfer:DescribeAgreement
	transfer:DescribeCertificate
	transfer:DescribeConnector
	transfer:DescribeExecution
	transfer:DescribeHostKey
	transfer:DescribeProfile
	transfer:DescribeSecurityPolicy
	transfer:DescribeServer
	transfer:DescribeUser
	transfer:DescribeWebApp
	transfer:DescribeWebAppCustomization
	transfer:DescribeWorkflow
	transfer:ImportCertificate
	transfer:ImportHostKey
	transfer:ImportSshPublicKey
	transfer:ListAccesses
	transfer:ListCertificates
	transfer:ListConnectors
	transfer:ListExecutions
	transfer:ListFileTransferResults
	transfer:ListHostKeys

服務前綴	動作
	transfer:ListProfiles
	transfer:ListSecurityPolicies
	transfer:ListServers
	transfer:ListUsers
	transfer:ListWebApps
	transfer:ListWorkflows
	transfer:SendWorkflowStepState
	transfer:StartDirectoryListing
	transfer:StartFileTransfer
	transfer:StartRemoteDelete
	transfer:StartRemoteMove
	transfer:StartServer
	transfer:StopServer
	transfer:TestConnection
	transfer:TestIdentityProvider
	transfer:UpdateAccess
	transfer:UpdateAgreement
	transfer:UpdateCertificate
	transfer:UpdateConnector
	transfer:UpdateHostKey
	transfer:UpdateProfile

服務前綴	動作
	transfer:UpdateServer transfer:UpdateUser transfer:UpdateWebApp transfer:UpdateWebAppCustomization
translate	translate:CreateParallelData translate:DeleteParallelData translate:DeleteTerminology translate:DescribeTextTranslationJob translate:GetParallelData translate:GetTerminology translate:ImportTerminology translate:ListLanguages translate:ListParallelData translate:ListTerminologies translate:ListTextTranslationJobs translate:StartTextTranslationJob translate:StopTextTranslationJob translate:TranslateDocument translate:TranslateText translate:UpdateParallelData

服務前綴	動作
voiceid	voiceid:AssociateFraudster
	voiceid:CreateDomain
	voiceid:CreateWatchlist
	voiceid:DeleteDomain
	voiceid:DeleteFraudster
	voiceid:DeleteSpeaker
	voiceid:DeleteWatchlist
	voiceid:DescribeDomain
	voiceid:DescribeFraudster
	voiceid:DescribeFraudsterRegistrationJob
	voiceid:DescribeSpeaker
	voiceid:DescribeSpeakerEnrollmentJob
	voiceid:DescribeWatchlist
	voiceid:DisassociateFraudster
	voiceid:EvaluateSession
	voiceid:ListDomains
	voiceid:ListFraudsterRegistrationJobs
	voiceid:ListFraudsters
	voiceid:ListSpeakerEnrollmentJobs
	voiceid:ListSpeakers
	voiceid:ListWatchlists

服務前綴	動作
	voiceid:OptOutSpeaker voiceid:StartFraudsterRegistrationJob voiceid:StartSpeakerEnrollmentJob voiceid:UpdateDomain voiceid:UpdateWatchlist

服務前綴	動作
vpc-lattice	vpc-lattice:CreateAccessLogSubscription vpc-lattice:CreateListener vpc-lattice:CreateResourceConfiguration vpc-lattice:CreateResourceGateway vpc-lattice:CreateRule vpc-lattice:CreateService vpc-lattice:CreateServiceNetwork vpc-lattice:CreateServiceNetworkResourceAssociation vpc-lattice:CreateServiceNetworkServiceAssociation vpc-lattice:CreateServiceNetworkVpcAssociation vpc-lattice:CreateTargetGroup vpc-lattice:DeleteAccessLogSubscription vpc-lattice:DeleteAuthPolicy vpc-lattice：DeleteDomainVerification vpc-lattice:DeleteListener vpc-lattice:DeleteResourceConfiguration vpc-lattice:DeleteResourceEndpointAssociation vpc-lattice:DeleteResourceGateway vpc-lattice:DeleteResourcePolicy vpc-lattice:DeleteRule vpc-lattice:DeleteService

服務前綴	動作
	vpc-lattice:DeleteServiceNetwork vpc-lattice:DeleteServiceNetworkResourceAssociation vpc-lattice:DeleteServiceNetworkServiceAssociation vpc-lattice:DeleteServiceNetworkVpcAssociation vpc-lattice:DeleteTargetGroup vpc-lattice:DeregisterTargets vpc-lattice:GetAccessLogSubscription vpc-lattice:GetAuthPolicy vpc-lattice：GetDomainVerification vpc-lattice:GetListener vpc-lattice:GetResourceConfiguration vpc-lattice:GetResourceGateway vpc-lattice:GetResourcePolicy vpc-lattice:GetRule vpc-lattice:GetService vpc-lattice:GetServiceNetwork vpc-lattice:GetServiceNetworkResourceAssociation vpc-lattice:GetServiceNetworkServiceAssociation vpc-lattice:GetServiceNetworkVpcAssociation vpc-lattice:GetTargetGroup vpc-lattice:ListAccessLogSubscriptions

服務前綴	動作
	vpc-lattice：ListDomainVerifications vpc-lattice:ListListeners vpc-lattice:ListResourceConfigurations vpc-lattice:ListResourceEndpointAssociations vpc-lattice:ListResourceGateways vpc-lattice:ListRules vpc-lattice:ListServiceNetworkResourceAssociations vpc-lattice:ListServiceNetworkServiceAssociations vpc-lattice:ListServiceNetworkVpcAssociations vpc-lattice:ListServiceNetworkVpcEndpointAssociations vpc-lattice:ListServiceNetworks vpc-lattice:ListServices vpc-lattice:ListTargetGroups vpc-lattice:ListTargets vpc-lattice:PutAuthPolicy vpc-lattice:PutResourcePolicy vpc-lattice:RegisterTargets vpc-lattice：StartDomainVerification vpc-lattice:UpdateAccessLogSubscription vpc-lattice:UpdateListener vpc-lattice:UpdateResourceConfiguration

服務前綴	動作
	vpc-lattice:UpdateResourceGateway vpc-lattice:UpdateRule vpc-lattice:UpdateService vpc-lattice:UpdateServiceNetwork vpc-lattice:UpdateServiceNetworkVpcAssociation vpc-lattice:UpdateTargetGroup

服務前綴	動作
wafv2	wafv2:AssociateWebACL
	wafv2:CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2:CreateRegexPatternSet
	wafv2:CreateRuleGroup
	wafv2:CreateWebACL
	wafv2:DeleteAPIKey
	wafv2:DeleteFirewallManagerRuleGroups
	wafv2:DeleteIPSet
	wafv2:DeleteLoggingConfiguration
	wafv2:DeletePermissionPolicy
	wafv2:DeleteRegexPatternSet
	wafv2:DeleteRuleGroup
	wafv2:DeleteWebACL
	wafv2:DescribeAllManagedProducts
	wafv2:DescribeManagedProductsByVendor
	wafv2:DescribeManagedRuleGroup
	wafv2:DisassociateWebACL
	wafv2:GenerateMobileSdkReleaseUrl
	wafv2:GetDecryptedAPIKey

服務前綴	動作
	wafv2:GetIPSet
	wafv2:GetLoggingConfiguration
	wafv2:GetManagedRuleSet
	wafv2:GetMobileSdkRelease
	wafv2:GetRateBasedStatementManagedKeys
	wafv2:GetRegexPatternSet
	wafv2：GetRevenueStatistics
	wafv2：GetRevenueStatisticsSummary
	wafv2：GetRevenueStatisticsTimeSeries
	wafv2:GetRuleGroup
	wafv2:GetSampledRequests
	wafv2:GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2:ListAvailableManagedRuleGroupVersions
	wafv2:ListAvailableManagedRuleGroups
	wafv2:ListIPSets
	wafv2:ListLoggingConfigurations
	wafv2:ListManagedRuleSets
	wafv2:ListMobileSdkReleases
	wafv2:ListRegexPatternSets
	wafv2:ListResourcesForWebACL

服務前綴	動作
	wafv2:ListRuleGroups
	wafv2：ListSettlementRecords
	wafv2:ListWebACLs
	wafv2:PutLoggingConfiguration
	wafv2:PutManagedRuleSetVersions
	wafv2:UpdateIPSet
	wafv2:UpdateManagedRuleSetVersionExpiryDate
	wafv2:UpdateRegexPatternSet
	wafv2:UpdateRuleGroup
	wafv2:UpdateWebACL

服務前綴	動作
wellarchitected	wellarchitected:AssociateLenses wellarchitected:AssociateProfiles wellarchitected:CreateLensShare wellarchitected:CreateLensVersion wellarchitected:CreateMilestone wellarchitected:CreateProfile wellarchitected:CreateProfileShare wellarchitected:CreateReviewTemplate wellarchitected:CreateWorkload wellarchitected:CreateWorkloadShare wellarchitected:DeleteLens wellarchitected:DeleteLensShare wellarchitected:DeleteProfile wellarchitected:DeleteProfileShare wellarchitected:DeleteReviewTemplate wellarchitected:DeleteTemplateShare wellarchitected:DeleteWorkload wellarchitected:DeleteWorkloadShare wellarchitected:DisassociateLenses wellarchitected:DisassociateProfiles wellarchitected:ExportLens

服務前綴	動作
	wellarchitected:GetAnswer wellarchitected:GetConsolidatedReport wellarchitected:GetGlobalSettings wellarchitected:GetLens wellarchitected:GetLensReview wellarchitected:GetLensReviewReport wellarchitected:GetLensVersionDifference wellarchitected:GetMilestone wellarchitected:GetProfile wellarchitected:GetProfileTemplate wellarchitected:GetReviewTemplate wellarchitected:GetReviewTemplateAnswer wellarchitected:GetReviewTemplateLensReview wellarchitected:GetWorkload wellarchitected:ImportLens wellarchitected:ListAnswers wellarchitected:ListCheckDetails wellarchitected:ListCheckSummaries wellarchitected:ListLensReviewImprovements wellarchitected:ListLensReviews wellarchitected:ListLensShares

服務前綴	動作
	wellarchitected:ListLenses wellarchitected:ListMilestones wellarchitected:ListNotifications wellarchitected:ListProfileNotifications wellarchitected:ListProfileShares wellarchitected:ListProfiles wellarchitected:ListReviewTemplateAnswers wellarchitected:ListReviewTemplates wellarchitected:ListShareInvitations wellarchitected:ListTemplateShares wellarchitected:ListWorkloadShares wellarchitected:ListWorkloads wellarchitected:UpdateAnswer wellarchitected:UpdateGlobalSettings wellarchitected:UpdateIntegration wellarchitected:UpdateLensReview wellarchitected:UpdateProfile wellarchitected:UpdateReviewTemplate wellarchitected:UpdateReviewTemplateLensReview wellarchitected:UpdateShareInvitation wellarchitected:UpdateWorkload

服務前綴	動作
	wellarchitected:UpdateWorkloadShare wellarchitected:UpgradeLensReview wellarchitected:UpgradeProfileVersion wellarchitected:UpgradeReviewTemplateLensReview

服務前綴	動作
wisdom	wisdom:CreateAssistant
	wisdom:CreateAssistantAssociation
	wisdom:CreateContent
	wisdom:CreateKnowledgeBase
	wisdom:CreateQuickResponse
	wisdom:CreateSession
	wisdom:DeleteAssistant
	wisdom:DeleteAssistantAssociation
	wisdom:DeleteContent
	wisdom:DeleteImportJob
	wisdom:DeleteKnowledgeBase
	wisdom:DeleteQuickResponse
	wisdom:GetAssistant
	wisdom:GetAssistantAssociation
	wisdom:GetContent
	wisdom:GetContentAssociation
	wisdom:GetContentSummary
	wisdom:GetImportJob
	wisdom:GetKnowledgeBase
	wisdom:GetRecommendations
	wisdom:GetSession

服務前綴	動作
	wisdom:ListAssistantAssociations
	wisdom:ListAssistants
	wisdom:ListContentAssociations
	wisdom:ListContents
	wisdom:ListImportJobs
	wisdom:ListKnowledgeBases
	wisdom:ListQuickResponses
	wisdom:NotifyRecommendationsReceived
	wisdom:QueryAssistant
	wisdom:RemoveKnowledgeBaseTemplateUri
	wisdom:SearchContent
	wisdom:SearchQuickResponses
	wisdom:SearchSessions
	wisdom:StartContentUpload
	wisdom:StartImportJob
	wisdom:UpdateContent
	wisdom:UpdateKnowledgeBaseTemplateUri
	wisdom:UpdateQuickResponse
	wisdom:UpdateSession

服務前綴	動作
worklink	worklink:AssociateDomain worklink:AssociateWebsiteAuthorizationProvider worklink:AssociateWebsiteCertificateAuthority worklink:CreateFleet worklink:DeleteFleet worklink:DescribeAuditStreamConfiguration worklink:DescribeCompanyNetworkConfiguration worklink:DescribeDevice worklink:DescribeDevicePolicyConfiguration worklink:DescribeDomain worklink:DescribeFleetMetadata worklink:DescribeIdentityProviderConfiguration worklink:DescribeWebsiteCertificateAuthority worklink:DisassociateDomain worklink:DisassociateWebsiteAuthorizationProvider worklink:DisassociateWebsiteCertificateAuthority worklink:ListDevices worklink:ListDomains worklink:ListFleets worklink:ListWebsiteAuthorizationProviders worklink:ListWebsiteCertificateAuthorities

服務前綴	動作
	worklink:RestoreDomainAccess worklink:RevokeDomainAccess worklink:SignOutUser worklink:UpdateAuditStreamConfiguration worklink:UpdateCompanyNetworkConfiguration worklink:UpdateDevicePolicyConfiguration worklink:UpdateDomainMetadata worklink:UpdateFleetMetadata worklink:UpdateIdentityProviderConfiguration

服務前綴	動作
工作區	workspaces:AcceptAccountLinkInvitation workspaces:AssociateConnectionAlias workspaces:AssociateIpGroups workspaces:AssociateWorkspaceApplication workspaces:CopyWorkspaceImage workspaces:CreateAccountLinkInvitation workspaces:CreateConnectClientAddIn workspaces:CreateConnectionAlias workspaces:CreateIpGroup workspaces:CreateStandbyWorkspaces workspaces:CreateUpdatedWorkspaceImage workspaces:CreateWorkspaceBundle workspaces:CreateWorkspaceImage workspaces:CreateWorkspaces workspaces:CreateWorkspacesPool workspaces:DeleteAccountLinkInvitation workspaces:DeleteClientBranding workspaces:DeleteConnectClientAddIn workspaces:DeleteConnectionAlias workspaces:DeleteIpGroup workspaces:DeleteWorkspaceBundle

服務前綴	動作
	workspaces:DeleteWorkspaceImage
	workspaces:DeployWorkspaceApplications
	workspaces:DeregisterWorkspaceDirectory
	workspaces:DescribeAccount
	workspaces:DescribeAccountModifications
	workspaces:DescribeApplicationAssociations
	workspaces:DescribeApplications
	workspaces:DescribeBundleAssociations
	workspaces:DescribeClientBranding
	workspaces:DescribeClientProperties
	workspaces:DescribeConnectClientAddIns
	workspaces:DescribeConnectionAliasPermissions
	workspaces:DescribeConnectionAliases
	workspaces:DescribeCustomWorkspaceImageImport
	workspaces:DescribeImageAssociations
	workspaces:DescribeIpGroups
	workspaces:DescribeWorkspaceAssociations
	workspaces:DescribeWorkspaceBundles
	workspaces:DescribeWorkspaceDirectories
	workspaces:DescribeWorkspaceImagePermissions
	workspaces:DescribeWorkspaceSnapshots

服務前綴	動作
	workspaces:DescribeWorkspaces
	workspaces:DescribeWorkspacesConnectionStatus
	workspaces:DescribeWorkspacesPoolSessions
	workspaces:DescribeWorkspacesPools
	workspaces:DisassociateConnectionAlias
	workspaces:DisassociateIpGroups
	workspaces:DisassociateWorkspaceApplication
	workspaces:GetAccountLink
	workspaces:ImportClientBranding
	workspaces:ImportWorkspaceImage
	workspaces:ListAccountLinks
	workspaces:ListAvailableManagementCidrRanges
	workspaces:MigrateWorkspace
	workspaces:ModifyAccount
	workspaces:ModifyCertificateBasedAuthProperties
	workspaces:ModifyClientProperties
	workspaces:ModifyEndpointEncryptionMode
	workspaces:ModifySamlProperties
	workspaces:ModifySelfservicePermissions
	workspaces:ModifyStreamingProperties
	workspaces:ModifyWorkspaceAccessProperties

服務前綴	動作
	workspaces:ModifyWorkspaceCreationProperties
	workspaces:ModifyWorkspaceProperties
	workspaces:ModifyWorkspaceState
	workspaces:RebootWorkspaces
	workspaces:RebuildWorkspaces
	workspaces:RegisterWorkspaceDirectory
	workspaces:RejectAccountLinkInvitation
	workspaces:RestoreWorkspace
	workspaces:StartWorkspaces
	workspaces:StartWorkspacesPool
	workspaces:StopWorkspaces
	workspaces:StopWorkspacesPool
	workspaces:TerminateWorkspaces
	workspaces:TerminateWorkspacesPool
	workspaces:TerminateWorkspacesPoolSession
	workspaces:UpdateConnectClientAddIn
	workspaces:UpdateConnectionAliasPermission
	workspaces:UpdateWorkspaceBundle
	workspaces:UpdateWorkspaceImagePermission
	workspaces:UpdateWorkspacesPool

服務前綴	動作
xray	xray:CreateGroup xray:CreateSamplingRule xray:DeleteGroup xray:DeleteResourcePolicy xray:DeleteSamplingRule xray:GetEncryptionConfig xray:GetGroup xray:GetGroups xray:GetInsight xray:GetInsightEvents xray:GetInsightImpactGraph xray:GetInsightSummaries xray:GetSamplingRules xray:ListResourcePolicies xray:PutEncryptionConfig xray:PutResourcePolicy xray:UpdateGroup xray:UpdateSamplingRule

IAM Access Analyzer 配額

IAM Access Analyzer 具有以下配額：

資源	預設配額	最大配額
每個區域每個 AWS 帳戶 每個分析器類型的帳戶層級分析器上限	1	1
AWS 帳戶 每個區域每個分析器類型的組織層級外部或未使用的存取分析器上限	5	20[1]
每個區域每個 AWS 組織的組織層級內部存取分析器數量上限	1	1
每個分析器的存檔規則數量上限	100 每個封存規則的每個條件最多可擁有 20 個值。	1,000[1]
每小時每個分析器的存取預覽數目上限	1,000	1,000
AWS CloudTrail 每個政策世代處理的日誌檔案	100,000	100,000
並行政策產生	1	1
政策產生 AWS CloudTrail 資料大小	25 GB	25 GB
政策產生 AWS CloudTrail 時間範圍	90 天	90 天
每天政策產生	非洲 (開普敦)：5 亞太區域 (香港)：5 亞太區域 (雅加達)：5 歐洲 (米蘭)：5	非洲 (開普敦)：5 亞太區域 (香港)：5 亞太區域 (雅加達)：5 歐洲 (米蘭)：5

資源	預設配額	最大配額
	中東 (巴林)：5 所有其他支援的區域：50 **Note** 取消的政策產生要求會套用至每日配額。	中東 (巴林)：5 所有其他支援的區域：50

[1]客戶可使用 Service Quotas 設定某些配額。

對 IAM 進行疑難排解

使用此處的資訊，來協助您針對使用 AWS Identity and Access Management (IAM) 時的常見問題進行診斷與修正。

問題

- 我無法登入我的 AWS 帳戶
- 如果遺失存取金鑰
- 政策變數無法運作
- 我所做的變更不一定都會立刻生效
- 我未獲得授權，不得執行：iam:DeleteVirtualMFADevice
- 如何安全地建立 IAM 使用者？
- 其他資源
- 對拒絕存取錯誤訊息進行疑難排解
- 使用授權 ID 對存取遭拒錯誤訊息進行故障診斷 （ 預覽 ）
- 使用根使用者來疑難排解問題
- IAM 政策疑難排解
- 對通行密鑰和 FIDO 安全金鑰進行疑難排解
- IAM 角色疑難排解
- 對 IAM 和 Amazon EC2 進行疑難排解
- 對 IAM 和 Amazon S3 進行疑難排解
- 使用 IAM 對 SAML 聯合進行疑難排解

我無法登入我的 AWS 帳戶

請驗證您有正確的憑證，並且正在使用正確的方法登入。如需詳細資訊，請參閱《AWS 登入 使用者指南》 中的 登入問題故障診斷。

如果遺失存取金鑰

存取金鑰包含兩個部分：

- 存取金鑰識別符。識別符是公開的，您可以在列出存取金鑰的任意 IAM 主控台中進行查看，例如使用者摘要頁面。
- 私密存取金鑰。該金鑰會在您最初建立存取金鑰對時提供。它與密碼一樣，之後無法再擷取。如果您遺失了私密存取金鑰，則必須建立新的存取金鑰對。如果您已擁有最大數量的存取金鑰，則必須先刪除現有的金鑰對，才能建立另一個。

如果您遺失了私密存取金鑰，則必須刪除該存取金鑰並新建一個。如需更多說明，請參閱 更新存取金鑰。

政策變數無法運作

如果政策變數不起作用，則已發生下列錯誤之一：

版本政策元素中的日期出錯。

請確認包含變數的所有政策是否在政策中包含以下版本編號：`"Version": "2012-10-17"`。若沒有正確的版本編號，在評估期間不會替換這些變數。反之，只會從字面上評估這些變數。如果包含最新的版本編號，則不包含變數的政策仍將具有效用。

`Version` 政策元素與政策版本不同。`Version` 政策元素是在政策內使用，並定義政策語言的版本。在 IAM 中修改客戶管理政策時會建立政策版本。變更的政策不會覆寫現有的政策。IAM 反而會建立新版本的受管政策。若要進一步了解 `Version` 政策元素，請參閱 IAM JSON 政策元素：Version。若要進一步了解政策版本，請參閱 the section called “版本控制 IAM 政策”。

變數字元存在字母大小寫錯誤。

確認您的政策變數為正確的大小寫。如需詳細資訊，請參閱 IAM 政策元素：變數與標籤。

我所做的變更不一定都會立刻生效

作為可透過全球各地資料中心的電腦存取的服務，IAM 採用了稱為最終一致性的分散式運算模式。您在 IAM （或其他 AWS 服務）中所做的任何變更，包括屬性型存取控制 (ABAC) 標籤，都需要時間才能從所有可能的端點中顯示。部分延遲是由於在伺服器之間、複寫區域之間和地區之間傳送資料所花費的時間所造成。IAM 也會使用快取以提升效能，但在某些情況下，這可能會增加時間。直到先前快取的資料逾時後，才能看到變更。

您設計的全域應用程式必須能夠處理這些可能的延遲問題。確保它們即使在某個位置所做的變更不會立即顯示在另一個位置時，仍能如預期般運作。此類變更包括建立或更新使用者、群組、角色或政策。在

應用程式的關鍵、高可用性代碼路徑中，不建議進行此類 IAM 變更。而應在不常運作的、單獨的初始化或設定常式中進行 IAM 變更。另外，在生產工作流程套用這些變更之前，請務必確認變更已傳播完畢。

如需其他一些 AWS 服務如何受到此影響的詳細資訊，請參閱下列資源：

- Amazon DynamoDB：《DynamoDB 開發人員指南》中的讀取一致性，以及《Amazon DynamoDB 開發人員指南》中的讀取一致性。
- Amazon EC2：Amazon EC2 API 參考中的 EC2 最終一致性。
- Amazon EMR： AWS 大數據部落格中的在使用 Amazon S3 和適用於 ETL 工作流程的 Amazon EMR 時確保一致性
- Amazon Redshift： Amazon Redshift 資料庫開發人員指南中的管理資料一致性
- Amazon S3：Amazon Simple Storage Service 使用者指南中的 Amazon S3 資料一致性模式

我未獲得授權，不得執行：iam:DeleteVirtualMFADevice

當您嘗試為您自己或其他人指派或移除虛擬 MFA 裝置時，您可能會接收到下列錯誤：

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to
 perform: iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego
 with an explicit deny
```

如果有人先前在 IAM 主控台中開始將虛擬 MFA 裝置指派給使用者的程序又將其取消，就可能發生此錯誤。這會在 IAM 中為該使用者建立虛擬 MFA 裝置，但從未將此裝置指派給該使用者。先刪除現有的虛擬 MFA 裝置，才能使用相同的裝置名稱建立新的虛擬 MFA 裝置。

若要修正此問題，管理員不應編輯政策許可。反之，管理員必須使用 AWS CLI 或 AWS API 來刪除現有但未指派的虛擬 MFA 裝置。

刪除現有但未指派的虛擬 MFA 裝置

1. 檢視您帳戶中的虛擬 MFA 裝置。
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - AWS API： `ListVirtualMFADevices`
2. 在回應中，找到您嘗試修正的使用者虛擬 MFA 裝置 ARN。
3. 刪除虛擬 MFA 裝置。

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API： `DeleteVirtualMFADevice`

如何安全地建立 IAM 使用者？

如果您有員工需要存取 AWS，您可以選擇建立 IAM 使用者或使用 IAM Identity Center 進行身分驗證。如果您使用 IAM， AWS 建議您建立 IAM 使用者，並安全地將登入資料傳達給員工。如果您不在員工旁邊，請使用安全的工作流程將憑證傳達給員工。

使用下列安全工作流程，在 IAM 中建立新使用者：

1. 使用 AWS 管理主控台建立新使用者。選擇以使用產生的密碼授予 AWS 管理主控台 存取權。如有必要，請選取 Users must create a new password at next sign-in (使用者必須在下次登入時建立新密碼) 核取方塊。在使用者變更密碼之前，請勿將許可政策新增至使用者。
2. 新增使用者之後，複製新使用者的登入 URL、使用者名稱和密碼。若要檢視密碼，請選擇 Show (顯示)。
3. 使用公司內的安全通訊方法，例如電子郵件、聊天或票證系統，將密碼傳送給您的員工。另外，為使用者提供 IAM 使用者主控台連結及其使用者名稱。請員工確認他們可以成功登入，然後再授與他們許可。
4. 員工確認之後，新增他們所需的許可。安全最佳實務是新增要求使用者使用 MFA 進行身分驗證以管理其憑證的政策。如需政策範例，請參閱 AWS：允許經過 MFA 驗證的 IAM 使用者在「安全憑證」頁面中管理其憑證。。

其他資源

下列資源可協助您在使用 時進行故障診斷 AWS。

- AWS CloudTrail 使用者指南 – 使用 AWS CloudTrail 追蹤對 發出的 API 呼叫歷史記錄， AWS 並將該資訊存放在日誌檔案中。這有助於您判斷哪些使用者和帳戶存取帳戶中的資源，何時進行呼叫，請求了哪些動作等等。如需詳細資訊，請參閱使用 記錄 IAM 和 AWS STS API 呼叫 AWS CloudTrail。
- AWS 知識中心 – 尋找FAQs和其他資源的連結，以協助您疑難排解問題。
- AWS 支援中心 – 取得技術支援。
- AWS 進階支援中心 – 取得進階技術支援。

對拒絕存取錯誤訊息進行疑難排解

以下資訊可協助您使用 識別、診斷和解決存取遭拒錯誤 AWS Identity and Access Management。當 AWS 明確或隱含拒絕授權請求時，會出現存取遭拒錯誤。

- 當政策包含特定 AWS 動作的Deny陳述式時，會發生明確拒絕。
- 如果沒有適用的 Deny 陳述式，也沒有適用的 Allow 陳述式，則會發生隱含拒絕。由於 IAM 政策預設會拒絕 IAM 主體，因此該政策必須明確允許主體執行動作。否則，政策會隱含拒絕存取。如需詳細資訊，請參閱明確和隱含拒絕之間的差異。

向服務或資源提出請求時，多個政策可能適用於該請求。除了錯誤訊息中指定的政策之外，請檢閱所有適用政策。

- 如果相同政策類型的多個政策拒絕請求，則拒絕存取錯誤訊息不會指定政策數目。
- 如果多個政策類型拒絕授權請求，則 只會在錯誤訊息中 AWS 包含其中一個政策類型。

> **Important**
>
> 無法登入 AWS？ 請確定您位在使用者類型的正確 AWS 登入頁面。如果您是 AWS 帳戶根使用者 （帳戶擁有者），您可以使用您在建立 時設定的 AWS 登入資料登入 AWS 帳戶。如果您是 IAM 使用者，帳戶管理員可為您提供 AWS 登入憑證。如果需要請求支援，請勿使用此頁面中的意見回饋連結。 AWS 文件團隊不會收到表單 支援。相反地，在聯絡我們頁面上，選擇仍然無法登入 AWS 您的帳戶，然後選擇其中一個可用的支援選項。

向 AWS 服務提出請求時，我收到「存取遭拒」

- 檢查錯誤訊息是否包含負責拒絕存取的政策類型和 Amazon Resource Name (ARN)。如果是這種情況，請檢查指定政策中動作的拒絕陳述式。如果提供政策類型，但沒有政策 ARN，則專注於該政策類型的疑難排解問題：檢查指定類型政策中動作的拒絕陳述式。如果錯誤訊息未提及發出拒絕存取的政策類型，請使用本節中的其餘指導方針進一步疑難排解。
- 確任您擁有呼叫所請求之動作和資源的以身分為基礎的政策許可。如果設定了任何條件，還必須在送出請求時滿足這些條件。有關查看或修改用於 IAM 使用者、群組或角色的政策的資訊，請參閱 管理 IAM 政策。
- 如果 AWS 管理主控台 傳回訊息，指出您無權執行 動作，則您必須聯絡管理員尋求協助。您的管理員提供您的登入憑證或登入連結。

以下範例錯誤的發生情境是 mateojackson IAM 使用者嘗試使用主控台檢視虛構 *my-example-widget* 資源的詳細資訊，但卻沒有虛構 widgets:*GetWidget* 許可。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 widgets:GetWidget on resource: my-example-widget
```

在此情況下，Mateo 必須請求管理員更新他的政策，允許他使用 widgets:*GetWidget* 動作存取 *my-example-widget* 資源。

- 您是否試著存取支援以資源為基礎的政策的服務，例如 Amazon S3，Amazon SNS 或 Amazon SQS？ 若是如此，確認政策將您指定為主體，並授與您存取權。如果您要對在您的帳戶中的服務發出請求，則您的以身分為基礎的政策或以資源為基礎的政策中之一可為您授與許可。如果您要對不同帳戶中的服務發出請求，則您的以身分為基礎的政策或以資源為基礎的政策都必須為您授與許可。若要檢視支援以資源為基礎的政策之服務，請參閱 AWS 使用 IAM 的 服務。
- 如果您的政策包含搭配金鑰值組的條件，請仔細地檢閱該條件。範例包括aws:RequestTag/*tag-key*全域條件金鑰、 AWS KMS kms:EncryptionContext:*encryption_context_key*和多個服務支援ResourceTag/*tag-key*的條件金鑰。確定金鑰名稱不符合多個結果。由於條件金鑰名稱不區分大小寫，條件會檢查符合 foo、Foo 或 F00 且名為 foo 的金鑰。如果您的請求包含多個鍵值組，且其中有鍵名稱只有大小寫不同，則您的存取可能會意外遭拒。如需更多詳細資訊，請參閱 IAM JSON 政策元素：Condition。
- 如果您具有許可界限，請確認允許您的請求時之許可界限所使用的政策。如果您的以身分為基礎的政策允許該請求，但您的許可界限不允許，則請求會遭拒絕。許可界限會控制 IAM 主體 (使用者或角色) 可以擁有的許可上限。以資源為基礎的政策不會受到許可界限所限制。許可界限不常見。如需 如何 AWS 評估政策的詳細資訊，請參閱 政策評估邏輯。
- 如果您是手動簽署請求 (而未使用 AWS 開發套件)，請確認您已正確地簽署請求。
- 如果您使用 Amazon VPC 端點政策，且收到未記錄在 AWS CloudTrail中的存取遭拒錯誤，這可能是因為 VPC 端點擁有者帳戶與呼叫帳戶或目標角色帳戶不同。

當我使用臨時安全憑證來發出請求時，出現「存取遭拒」

- 首先，確定您不是因為與臨時憑證無關的原因而遭拒絕存取。如需更多詳細資訊，請參閱 向 AWS 服務提出請求時，我收到「存取遭拒」。
- 要確認服務是否接受臨時安全憑證，請參閱 AWS 使用 IAM 的 服務。
- 確認您的請求正確簽署且請求的格式也正確。如需詳細資訊，請參閱您的工具組文件或 搭配 AWS 資源使用臨時登入資料。

- 確認您的臨時安全憑證並未過期。如需更多詳細資訊，請參閱 IAM 中的暫時安全憑證。
- 確認 IAM 使用者或角色擁有正確許可。臨時安全性憑證的許可衍生自 IAM 使用者或角色。因此，這些許可僅限於您所擔任臨時憑證之角色所獲得授與的許可。有關如何確定臨時安全憑證的許可的更多資訊，請參閱 臨時安全憑證的許可。
- 如果您擔任角色，您的角色工作階段可能受工作階段政策限制。當您使用 以程式設計方式請求臨時安全登入資料時 AWS STS，您可以選擇傳遞內嵌或受管工作階段政策。工作階段政策是一種進階政策，且您會在以程式設計方式建立角色的暫時憑證工作階段時，以參數方式傳遞。您可以使用 `Policy` 參數傳遞單一 JSON 內嵌工作階段政策文件。您可以使用 `PolicyArns` 參數，指定多達 10 個受管工作階段政策。所產生工作階段的許可會是角色的以身分為基礎的政策和工作階段政策的交集。或者，如果您的管理員或自訂程式為您提供暫時憑證，他們可能已包含限制您存取的工作階段政策。
- 如果您是 AWS STS 聯合身分使用者主體，您的工作階段可能會受到工作階段政策的限制。您以 IAM 使用者 AWS 身分登入 ，然後請求聯合字符，以建立聯合身分使用者工作階段。如需詳細資訊，請參閱透過自訂身分經紀人請求憑證。如果您或身分經紀人在請求聯合權杖時傳遞工作階段政策，則您的工作階段會受這些政策限制。所產生工作階段的許可會是 IAM 使用者的以身分為基礎的政策和工作階段政策的交集。如需有關工作階段政策的詳細資訊，請參閱 工作階段政策。
- 如果您使用角色存取具有以資源為基礎之政策的資源，則請確認政策已授與該角色許可。例如，以下政策允許 `MyRole` 從帳戶 `111122223333` 存取 `amzn-s3-demo-bucket`。

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
  }]
}
```

拒絕存取錯誤訊息範例

大多數拒絕存取錯誤訊息的格式為 `User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*。在此範例中，*###*是拒絕存取之主體的 ARN，*##*是政策拒絕的服務

動作，而*##*是政策對其執行動作之資源的 ARN。*##*欄位提供有關拒絕存取的政策類型的其他內容。在某些情況下，它還包含拒絕存取的政策的 ARN。

當政策因為政策包含Deny陳述式而明確拒絕存取時， 會在拒絕存取錯誤訊息`with an explicit deny in a` *`type`* `policy`中 AWS 包含該片語。此片語也可以指定政策的 ARN，如下所示：`with an explicit deny in a` *`type`* `policy:` *`policy ARN`*。

當政策隱含拒絕存取時， 會在拒絕存取錯誤訊息`because no` *`type`* `policy allows the` *`action`* `action`中 AWS 包含 片語。

> **Note**
>
> 有些 AWS 服務不支援此存取遭拒錯誤訊息格式。存取遭拒錯誤訊息的內容可能會因提出授權請求的服務而有所不同。

以下範例顯示不同類型的拒絕存取錯誤訊息的格式。

因服務控制政策而拒絕存取 – 隱含拒絕

1. 在服務控制政策 (SCP) 中檢查該動作是否有缺少的 `Allow` 陳述式。對於下列範例，動作是 `codecommit:ListRepositories`。
2. 透過新增 `Allow` 陳述式來更新您的 SCP。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的更新 SCP。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no service control policy allows the codecommit:ListRepositories action
```

因服務控制政策而拒絕存取 – 明確拒絕

1. 如果錯誤訊息中提供政策 ARN，請檢查指定服務控制政策 (SCP) 中動作的Deny陳述式。在下面的範例中，動作為 `codecommit:ListRepositories`。
2. 如果錯誤訊息中未提供任何政策 ARN，請檢查 SCPs 中動作的Deny陳述式。
3. 透過移除 Deny 陳述式來更新您的 SCP。如需詳細資訊，請參閱AWS Organizations 《 使用者指南》中的更新服務控制政策 (SCP)。

具有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
with an explicit deny in a service control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/service_control_policy/p-
examplepolicyid123
```

沒有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
with an explicit deny in a service control policy
```

因資源控制政策而拒絕存取 – 明確拒絕

1. 如果在錯誤訊息中提供政策 ARN，請檢查指定資源控制政策 (RCP) 中是否有 動作的Deny陳述式。在下面的範例中，動作為 `secretsmanager:GetSecretValue`。
2. 如果錯誤訊息中未提供任何政策 ARN，請檢查 RCPs 中動作的Deny陳述式。
3. 透過移除 Deny 陳述式來更新您的 RCP。如需詳細資訊，請參閱AWS Organizations 《 使用者指南》中的更新資源控制政策 (RCP)。

具有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/resource_control_policy/p-
examplepolicyid456
```

沒有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy
```

因 VPC 端點政策而拒絕存取 – 隱含拒絕

1. 在您的虛擬私有雲端 (VPC) 端點政策中檢查該動作是否有遺失的 `Allow` 陳述式。對於下列範例，動作是 `codecommit:ListRepositories`。
2. 透過新增 `Allow` 陳述式來更新 VPC 端點政策。如需詳細資訊，請參閱《AWS PrivateLink 指南》中的更新 VPC 端點政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no VPC endpoint policy allows the codecommit:ListRepositories action
```

因 VPC 端點政策而拒絕存取 – 明確拒絕

1. 在您的虛擬私有雲端 (VPC) 端點政策中檢查該動作是否有明確的 `Deny` 陳述式。對於下列範例，動作是 `codedeploy:ListDeployments`。
2. 透過移除 `Deny` 陳述式來更新 VPC 端點政策。如需詳細資訊，請參閱《AWS PrivateLink 指南》中的更新 VPC 端點政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a VPC endpoint policy
```

因許可界限而拒絕存取 – 隱含拒絕

1. 在許可界限中檢查該動作是否有遺失的 `Allow` 陳述式。對於下列範例，動作是 `codedeploy:ListDeployments`。
2. 透過將 `Allow` 陳述式新增至您的 IAM 政策來更新您的許可界限。如需詳細資訊，請參閱IAM 實體的許可界限及編輯 IAM 政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
because no permissions boundary allows the codedeploy:ListDeployments action
```

因許可界限而拒絕存取 – 明確拒絕

1. 如果錯誤訊息中提供政策 ARN，請檢查指定許可界限中動作的Deny陳述式。在下面的範例中，動作為 sagemaker:ListModels。
2. 如果錯誤訊息中未提供任何政策 ARN，請檢查連接到主體的許可界限中是否有動作的Deny陳述式。
3. 透過從您的 IAM 政策中移除 Deny 陳述式來更新您的許可界限。如需詳細資訊，請參閱IAM 實體的許可界限及編輯 IAM 政策。

具有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
with an explicit deny in a permissions boundary: arn:aws:iam::123456789012:policy/
DeveloperPermissionBoundary
```

沒有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
with an explicit deny in a permissions boundary
```

因工作階段政策而拒絕存取 – 隱含拒絕

1. 在工作階段政策中檢查該動作是否有遺失的 Allow 陳述式。對於下列範例，動作是 codecommit:ListRepositories。
2. 透過新增 Allow 陳述式來更新您的工作階段政策。如需詳細資訊，請參閱工作階段政策和 編輯 IAM 政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no session policy allows the codecommit:ListRepositories action
```

因工作階段政策而拒絕存取 – 明確拒絕

1. 如果錯誤訊息中提供政策 ARN，請檢查指定工作階段政策中動作的Deny陳述式。在下面的範例中，動作為 codedeploy:ListDeployments。

2. 如果錯誤訊息中未提供任何政策 ARN，請檢查工作階段政策中動作的Deny陳述式。
3. 透過移除 Deny 陳述式來更新您的工作階段政策。如需詳細資訊，請參閱工作階段政策和 編輯 IAM 政策。

具有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy: arn:aws:iam::123456789012:policy/
DeveloperSessionPolicy
```

沒有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy
```

因以資源為基礎的政策而拒絕存取 – 隱含拒絕

1. 在以資源為基礎的政策中檢查該動作是否有遺失的 Allow 陳述式。對於下列範例，動作是 `secretsmanager:GetSecretValue`。
2. 透過新增 Allow 陳述式來更新您的政策。如需詳細資訊，請參閱以資源為基礎的政策和 編輯 IAM 政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
because no resource-based policy allows the secretsmanager:GetSecretValue action
```

因以資源為基礎的政策而拒絕存取 – 明確拒絕

1. 在以資源為基礎的政策中檢查該動作是否有明確的 Deny 陳述式。對於下列範例，動作是 `secretsmanager:GetSecretValue`。
2. 透過移除 Deny 陳述式來更新您的政策。如需詳細資訊，請參閱以資源為基礎的政策和 編輯 IAM 政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource-based policy
```

因角色信任政策而拒絕存取 – 隱含拒絕

1. 在角色信任政策中檢查該動作是否有遺失的 Allow 陳述式。對於下列範例，動作是 sts:AssumeRole。
2. 透過新增 Allow 陳述式來更新您的政策。如需詳細資訊，請參閱以資源為基礎的政策和 編輯 IAM 政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
because no role trust policy allows the sts:AssumeRole action
```

因角色信任政策而拒絕存取 – 明確拒絕

1. 在角色信任政策中檢查該動作是否有明確的 Deny 陳述式。對於下列範例，動作是 sts:AssumeRole。
2. 透過移除 Deny 陳述式來更新您的政策。如需詳細資訊，請參閱以資源為基礎的政策和 編輯 IAM 政策。

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
with an explicit deny in the role trust policy
```

因以身分為基礎的政策而拒絕存取 – 隱含拒絕

1. 在連接至身分的以身分為基礎的政策中，檢查該動作是否有遺失的 Allow 陳述式。對於下列範例，動作是連接至角色 HR 的 codecommit:ListRepositories。
2. 透過新增 Allow 陳述式來更新您的政策。如需詳細資訊，請參閱以身分為基礎的政策和 編輯 IAM 政策。

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codecommit:ListRepositories
because no identity-based policy allows the codecommit:ListRepositories action
```

因以身分為基礎的政策而拒絕存取 – 明確拒絕

1. 如果錯誤訊息中提供政策 ARN，請檢查指定政策中動作的Deny陳述式。在下面的範例中，動作為 `codedeploy:ListDeployments`。
2. 如果錯誤訊息中未提供任何政策 ARN，請檢查連接到身分的身分型政策中是否有 動作的Deny陳述式。
3. 透過移除 Deny 陳述式來更新您的政策。如需詳細資訊，請參閱以身分為基礎的政策和 編輯 IAM 政策。

具有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
HRAccessPolicy
```

沒有政策 ARN 的錯誤訊息：

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy
```

使用授權 ID 對存取遭拒錯誤訊息進行故障診斷 （預覽）

> **Important**
>
> 存取故障診斷程式處於公開預覽狀態。

概觀

當 AWS 拒絕支援 API 的請求時，存取遭拒錯誤回應通常包含授權 ID。此 ID 參考拒絕請求的授權評估。若要擷取評估的詳細資訊並識別拒絕的原因，請向 `GetRequestAuthorizationDetails` API 提出請求並提供授權 ID，或開啟錯誤訊息中的連結。這可讓您在單一呼叫中識別拒絕請求的原因，而不是一次解決拒絕。

> **Note**
>
> AWS 不保證每個拒絕請求的授權 ID。請參閱本文件當您未收到授權 ID 時中的 。

下列範例顯示存取遭拒錯誤訊息，其中包含政策 ARN、 AWS 管理主控台的連結，以及授權 ID：

```
User: arn:aws:iam::123456789012:user/janedoe is not authorized to perform:
 iam:CreateInstanceProfile
on resource: arn:aws:iam::123456789012:instance-profile/my-instance-profile
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
ExampleDenyPolicy.
Go to https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/
authorization-details/EXAMPLE1a2b3c4d5e6f7g8h9i0j for complete details,
or call the GetRequestAuthorizationDetails API with the following authorization id:
 EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

概念和定義

- 授權詳細資訊。授權詳細資訊包括請求內容、執行的評估，以及評估的政策。
- 授權 ID。支援存取遭拒錯誤回應中包含的唯一識別符。它參考拒絕請求的授權評估，而 是您傳遞至 AWS 管理主控台連結或 `GetRequestAuthorizationDetails` API 的值。
- 請求內容。為請求評估的條件內容索引鍵值。這包括全域條件內容索引鍵和服務特定條件內容索引鍵。
- 評估。對於請求中的每個動作/資源對，套用的條件內容金鑰和評估結果：`ALLOW`、`EXPLICIT_DENY`（政策明確拒絕動作）或 `IMPLICIT_DENY`（政策不允許該動作），以及符合動作/資源對的每個政策和陳述式的參考。每個參考都是在整個回應中重複使用的政策 ARN（適用於受管政策）或 URI（適用於內嵌和其他非受管政策）。
- 授權期間評估的政策。 AWS 考慮請求的政策，包括不符合動作/資源對的政策。每個項目都會提供政策類型（例如，身分型、資源型、SCP、RCP、許可界限或工作階段政策）和評估中使用的相同 ARN（適用於受管政策）或 URI（適用於內嵌和其他非受管政策）。使用 ARN 或 URI 將政策比對回參考它的評估。「連接至」欄位會顯示評估政策所連接之委託人的委託人 ARN。

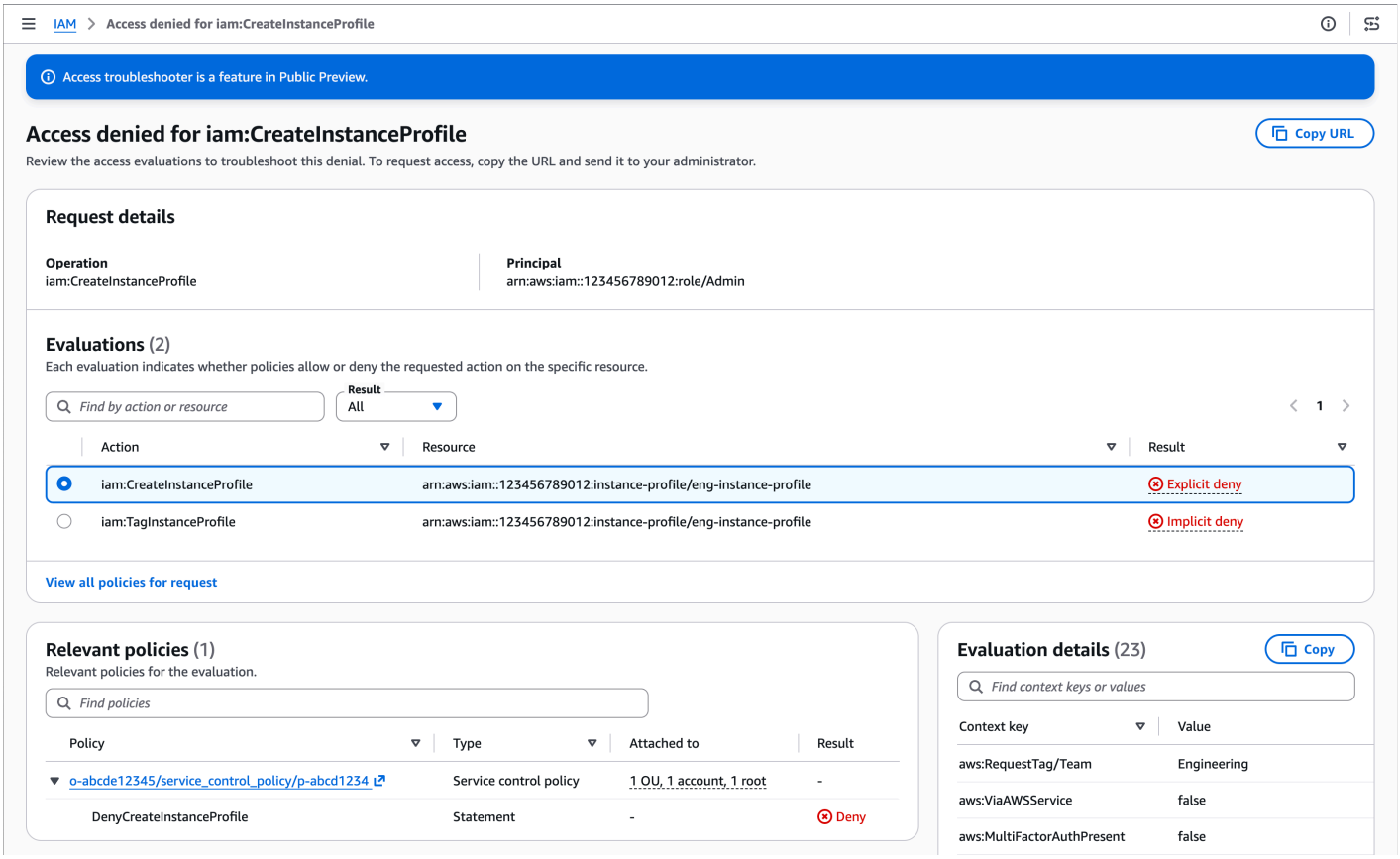
將這些結果一起使用，以識別哪些政策拒絕存取及其原因。評估會反映拒絕時的政策。自那時起，政策可能已變更。

AWS 會自動在支援 APIs 的存取遭拒錯誤回應中包含授權 ID，但無法保證授權 IDs。在某些情況下，AWS 不會記錄拒絕的授權詳細資訊，而且錯誤回應不包含授權 ID。記錄詳細資訊時，可能無法立即使用，因為 會以非同步方式 AWS 記錄詳細資訊。如果呼叫 GetRequestAuthorizationDetails傳回找不到的回應，請等待一會兒，然後再試一次。 會在拒絕後 AWS 保留授權詳細資訊 24 小時。

使用 AWS 管理主控台進行故障診斷

選擇存取遭拒錯誤訊息中的連結，以在存取疑難排解程式主控台中開啟授權詳細資訊。

- 要尋找的內容。存取故障診斷程式主控台會顯示請求中每個動作/資源對的評估、每個動作/資源對的結果 (Allow、 或 Implicit deny)Explicit deny，以及考慮的政策。尋找遭拒的配對，以及造成拒絕的政策。請參閱下列螢幕擷取範例：



- 判斷是否打算拒絕。變更政策之前，請檢閱評估。拒絕可能是正確的，原因有時是請求內容，而不是政策。例如，請求可能缺少預期的工作階段標籤，或來自非預期的網路路徑。
- 如何解決拒絕。找到對應的政策並加以更新，以解決存取問題。對於明確拒絕，請移除或縮小 動作的Deny陳述式。對於隱含拒絕，請為 動作新增 Allow陳述式。如需詳細資訊，請參閱編輯 IAM 政策。

誰可以存取授權詳細資訊。對於單一帳戶或單一組織中的請求，有權存取 的任何人`iam:GetRequestAuthorizationDetails`都可以存取請求的詳細資訊。對於跨越多個組織的請求，指定組織中的使用者只能檢視與其自身組織相關的詳細資訊。

使用 AWS API 或 進行故障診斷 AWS CLI

若要使用 AWS API 或 對拒絕進行疑難排解 AWS CLI：

1. 在存取遭拒錯誤回應中尋找授權 ID。
2. 在發生拒絕的相同 AWS 區域 中，`GetRequestAuthorizationDetails`使用授權 ID 呼叫 。下列範例顯示 AWS CLI 命令：

```
aws iam-toolbox get-request-authorization-details \
    --region us-east-1 \
    --authorization-id EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

3. 檢閱評估結果，以識別哪些政策拒絕存取，以及拒絕的原因。
4. 更新已識別的政策以解決存取問題。如需詳細資訊，請參閱編輯 IAM 政策。

當您未收到授權 ID 時

AWS 在下列情況下， 不會傳回授權 ID：

- 授權太大，無法記錄。當請求涉及大量政策時， AWS 可能會決定不記錄請求。
- 尚不支援此服務。請參閱 支援的服務。
- 不支援 API。受支援服務中的某些 APIs 不會傳回授權 ID。
- 不支援請求類型。這包括服務代表您提出的請求和匿名請求。

如果您沒有收到授權 ID，請參閱 對拒絕存取錯誤訊息進行疑難排解 以取得診斷拒絕的其他方法。

其他考量

必要的許可。若要呼叫 `GetRequestAuthorizationDetails`，您必須擁有`iam:GetRequestAuthorizationDetails`動作的許可。

區域 API。 GetRequestAuthorizationDetails 是區域 API。在產生存取遭拒錯誤和授權 ID 的相同 AWS 區域 中呼叫它。

跨組織請求。當拒絕的請求涉及多個組織時， 只會GetRequestAuthorizationDetails傳回屬於您自己的組織的詳細資訊。這適用於兩個方向：如果您在拒絕發起人的組織中，則只會看到與該組織相關的詳細資訊。如果您在拒絕資源的組織中，則只會看到與該組織相關的詳細資訊。不會傳回其他組織擁有的詳細資訊，例如該組織的政策或內容索引鍵值。

如果您從尚未包含授權 ID 的 API、服務或區域收到存取遭拒錯誤，請參閱 對拒絕存取錯誤訊息進行疑難排解。

支援的服務

的支援GetRequestAuthorizationDetails正在跨 AWS APIs和 逐漸推出 AWS 區域。下列 （服務） 支援 GetRequestAuthorizationDetails：

- IAM 中的大多數 APIs

使用根使用者來疑難排解問題

使用此處的資訊來幫助您疑難排解與 AWS 帳戶根使用者有關的問題。

> **Note**
>
> AWS 帳戶 使用 受管 AWS Organizations 可能已啟用成員帳戶的集中式根存取。這些成員帳戶沒有根使用者憑證，無法以根使用者身分登入，而且無法復原根使用者密碼。如果需要執行要求根使用者憑證的任務，請聯絡您的管理員。

在以帳戶根使用者身分登入時，我無法執行本該可以執行的任務

您的帳戶可能是 AWS Organizations中組織的成員。您的組織管理員可能擁有服務控制政策 (SCP)，以限制帳戶的許可。SCP 會影響所有使用者，包括根使用者。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的服務控制政策。

我忘記 的根使用者密碼 AWS 帳戶

如果您是根使用者且您遺失或忘記 AWS 帳戶的密碼，您可以重設密碼。您必須知道用來建立 的電子郵件地址 AWS 帳戶，而且必須能夠存取電子郵件帳戶。如需詳細資訊，請參閱重設遺失或忘記的根使用者密碼。

我無法存取我的 的電子郵件 AWS 帳戶

建立 時 AWS 帳戶，您會提供電子郵件地址和密碼。這些是 AWS 帳戶根使用者的登入資料。如果您不確定與您的 相關聯的電子郵件地址 AWS 帳戶，請搜尋可能已用於開啟 之組織中從 傳送@signin.aws或@verify.signin.aws傳送至任何電子郵件地址的訊息 AWS 帳戶。

如果您知道電子郵件地址，但無法存取電子郵件，請嘗試復原對電子郵件的存取。使用以下任一選項，重新獲得對電子郵件的存取權：

- 如果您擁有電子郵件地址的網域，您可以還原刪除的電子郵件地址。或者，您可以將您的電子郵件帳戶設為全部截獲。全部截獲會收集傳送到郵件伺服器中已不存在的電子郵件地址的所有郵件，並將它們重新導向到另一個電子郵件地址。
- 如果帳戶上的電子郵件地址屬於您的公司電子郵件系統，我們建議您與 IT 系統管理員聯絡。這也許有助您重新取得電子郵件的存取許可。

如果您仍然無法登入您的 AWS 帳戶，您可以在聯絡我們找到替代支援選項。

IAM 政策疑難排解

政策是 中的實體，當連接到身分或資源時， AWS 會定義其許可。當委託人如使用者提出請求時， 會 AWS 評估這些政策。政策中的許可決定是否允許或拒絕請求。

政策會以 JSON 文件 AWS 形式存放在 中，這些文件會以身分型政策的形式連接至主體，或以資源型政策的形式連接至資源。身分型政策包括 AWS 受管政策、客戶受管政策和內嵌政策。

您可以使用視覺化和 JSON 編輯器選項 AWS 管理主控台 ，在 中建立和編輯客戶受管政策。當您在 中檢視政策時 AWS 管理主控台，您可以看到該政策授予的許可摘要。您可以使用視覺化編輯器和政策摘要幫助診斷並修正在管理 IAM 政策時遇到的常見錯誤。

請記住，所有 IAM 政策是使用以 JavaScript Object Notation (JavaScript 物件註釋) (JSON) 規則開頭的語法所儲存的。您不需要了解該語法即可建立或管理您政策。您可以在 AWS 管理主控台中使用視覺化編輯器來建立和編輯政策。若要進一步了解 IAM 中 JSON 語法的詳細資訊，請參閱 IAM JSON 政策語言的文法。

針對 IAM 政策主題進行故障診斷

- 使用視覺化編輯器進行故障排除
  - 政策結構調整

- 在視覺化編輯器中選擇資源 ARN
- 在視覺化編輯器中拒絕許可
- 在視覺化編輯器中指定多個服務
- 在視覺化編輯器中縮減政策大小
- 在視覺化編輯器中修正無法識別的服務、動作或資源類型
- 使用政策摘要進行疑難排解
  - 缺少政策摘要
  - 政策摘要包含無法識別的服務、動作或資源類型
  - 服務不支援 IAM 政策摘要
  - 我的政策未授與預期的許可
- 故障排除政策管理
  - 在 IAM 帳戶中連接或分開政策
  - 根據 IAM 身分的活動來變更其政策
- JSON 政策文件故障排除
  - 驗證您的政策
  - 我在 JSON 編輯器中沒有政策驗證的許可
  - 多個 JSON 政策物件
  - 多個 JSON 陳述式元素
  - 在 JSON 陳述式元素中具有多個效果、動作或資源元素
  - 缺少 JSON 版本元素

使用視覺化編輯器進行故障排除

在建立或編輯客戶管理政策時，您可以使用視覺化編輯器中的資訊，來協助您針對政策中的錯誤進行故障排除。若要檢視使用視覺化編輯器建立政策的範例，請參閱 the section called “控制對身分的存取”。

政策結構調整

當您建立政策時， 會在儲存政策之前 AWS 驗證、處理和轉換政策。擷取政策時， 會將政策 AWS 轉換回人類可讀取的格式，而不會變更許可。這會導致您在政策視覺化編輯器或 JSON 索引標籤中看到的內容有所不同。

- 可以新增、移除或重新排序視覺化編輯器許可區塊，而且區塊內的內容可以最佳化。

- 在 JSON 標籤中，可以移除較不重要的空格，並且可以將 JSON 映射中的元素重新排序。此外，委託人元素中的 AWS 帳戶 IDs 可以由 的 Amazon Resource Name (ARN) 取代 AWS 帳戶根使用者。

由於可能發生這些更改，不應以字串形式來比較 JSON 政策文件。

當您在 中建立客戶受管政策時 AWS 管理主控台，您可以選擇在 JSON 編輯器中完全運作。如果從未在視覺化編輯器中變更政策，並在 JSON 編輯器中選擇下一步，則不太可能會調整政策結構。當您使用視覺化編輯器時，IAM 可能會調整政策結構，以最佳化其外觀。此結構調整僅在您的編輯工作階段中進行，不會自動儲存。

如果您的政策在編輯工作階段中進行結構調整，則 IAM 會根據以下情況確定是否儲存結構調整：

使用此編輯器選項	若您編輯政策	然後從該索引標籤選擇下一步	當您選擇 Save changes (儲存變更) 時
視覺化	已編輯	視覺化	對政策進行結構調整
視覺化	已編輯	JSON	對政策進行結構調整
視覺化	未編輯	視覺化	對政策進行結構調整
JSON	已編輯	視覺化	對政策進行結構調整
JSON	已編輯	JSON	此政策結構未變更
JSON	未編輯	JSON	此政策結構未變更

IAM 可能會對複雜政策或具有允許多個服務、資源類型或條件金鑰的許可區塊或陳述式之政策進行結構調整。

在視覺化編輯器中選擇資源 ARN

在使用視覺化編輯器建立或編輯政策時，您必須先選擇一個服務，然後從該服務中選擇動作。如果選取的服務和動作支援選擇的特定資源，視覺化編輯器將列出支援的資源類型。然後您可以選擇 Add ARN (新增 ARN) 來提供關於資源的詳細資訊。您可以從以下選項中進行選擇以新增資源類型的 ARN。

- 使用 ARN 產生器：根據資源類型，您可能會看到用於建置 ARN 的不同欄位。您也可以選擇 Any (任意) 來為指定設定的任何值提供許可。例如，若您選取了 Amazon EC2 Read (讀取) 存取等級群組，

那麼您政策中的動作便支援 `instance` 資源類型。提供區域、帳戶以及資源的 InstanceId 值。如果提供帳戶 ID 並為區域和執行個體 ID 選擇任意，則政策會為您帳戶中的任何執行個體授予許可。

- 輸入或貼上 ARN：您可以根據資源的 Amazon Resource Name (ARN) 來指定資源。您也可以在 ARN 的任何欄位中包含萬用字元 (*) (在每組冒號之間)。如需詳細資訊，請參閱IAM JSON 政策元素：Resource。

在視覺化編輯器中拒絕許可

預設情況下，您使用視覺化編輯器建立的政策允許執行您選擇的動作。若要拒絕選擇的動作，請選擇 Switch to deny permissions (切換為拒絕許可)。由於預設會拒絕請求，建議您僅允許使用者所需動作和資源的許可。只有在要覆蓋其他陳述式或政策單獨允許的許可時，才應建立拒絕陳述式。我們建議您將拒絕許可數限制為最低，因為它們可能會增加解決許可問題的難度。如需 IAM 評估政策邏輯之方法的詳細資訊，請參閱 政策評估邏輯。

> **Note**
>
> 根據預設，只有 AWS 帳戶根使用者 可以存取該帳戶中的所有資源。因此，如果未以根使用者登入，您必須具有政策授與的許可。

在視覺化編輯器中指定多個服務

在使用視覺化編輯器建構政策時，您每次只能選擇一個服務。這是一項最佳實務，因為視覺化編輯器允許您從該服務的動作中進行選擇。然後，您可以從該服務和選定的動作支援的資源中進行選擇。如此一來就可以更輕鬆地建立政策和進行故障排除。

還可以使用萬用字元 (*) 手動指定多個服務。例如，輸入 **`Code*`**，以便為以 `Code` 開頭的所有服務 (如 `CodeBuild` 與 `CodeCommit`) 提供許可。不過，您必須接著輸入動作和資源 ARN 以完成您的政策。此外，在儲存您的政策時，可能會進行結構調整，以在單獨的許可區塊中包含每個服務。

或者，若要在服務中使用 JSON 語法 (如萬用字元)，請使用 JSON 編輯器選項來建立、編輯和儲存政策。

在視覺化編輯器中縮減政策大小

在使用視覺化編輯器建立政策時，IAM 將建立一個 JSON 文件來存放您的政策。您可以切換到 JSON 編輯器選項來檢視該文件。如果該 JSON 文件超過政策的大小限制，視覺化編輯器將顯示錯誤訊息。您將無法檢閱和儲存政策。若要查看 IAM 對於受管政策的大小限制，請參閱 IAM 和 STS 字元限制。

若要在視覺化編輯器中縮減您的政策大小，請編輯您的政策或將許可區塊移到另一個政策。錯誤訊息包括政策文件所包含的字元數。可以使用此資訊來協助縮減政策的大小。

在視覺化編輯器中修正無法識別的服務、動作或資源類型

在視覺化編輯器可能會看到一條警告，指出您的政策包含無法識別的服務、動作或資源類型。

> **Note**
>
> IAM 對支援政策摘要的服務檢閱服務名稱、動作和資源類型。不過，您的政策摘要可能包含不存在資源值或條件。請一律使用政策模擬器來測試政策。

如果您的政策包含無法識別的服務、動作或資源類型，則存在以下錯誤之一：

- 預覽服務：處於預覽狀態的服務不支援視覺化編輯器。如果您參與預覽，則必須手動輸入動作和資源 ARN 以完成您的政策。可以忽略任何警告並繼續。或者，您可以選擇 JSON 編輯器選項，來輸入或貼上 JSON 政策文件。
- 自訂服務：自訂服務不支援視覺化編輯器。如果您使用自訂服務，則必須手動輸入動作和資源 ARN 以完成您的政策。可以忽略任何警告並繼續。或者，您可以選擇 JSON 編輯器選項，來輸入或貼上 JSON 政策文件。
- 服務不支援視覺化編輯器：如果您的政策包含不支援視覺化編輯器的公開提供 (GA) 服務，則必須手動輸入動作和資源 ARN 以完成您的政策。可以忽略任何警告並繼續。或者，您可以選擇 JSON 編輯器選項，來輸入或貼上 JSON 政策文件。

  公開提供服務是公開發佈的服務，不是預覽或自訂服務。如果無法識別的服務是公開提供的，並且名稱拼寫正確，則該服務不支援視覺化編輯器。若要了解如何請求 GA 服務的視覺化編輯器或政策摘要支援，請參閱 服務不支援 IAM 政策摘要。
- 動作不支援視覺化編輯器：如果您政策包含的受支援服務具有不支援的動作，則必須手動輸入動作和資源 ARN 以完成您的政策。可以忽略任何警告並繼續。或者，您可以選擇 JSON 編輯器選項，來輸入或貼上 JSON 政策文件。

  如果您的政策包含的受支援服務具有不支援的動作，則該服務不完全支援視覺化編輯器。若要了解如何請求 GA 服務的視覺化編輯器或政策摘要支援，請參閱 服務不支援 IAM 政策摘要。
- 資源類型不支援視覺化編輯器：如果您的政策包含的受支援動作具有不支援的資源類型，您可以忽略該警告並繼續。不過，IAM 無法確認是否包含所有選定動作的資源，且您可能會看到額外的警告。

- 拼寫錯誤：在視覺化編輯器中手動輸入服務、動作或資源時，您建立的政策可能會包含拼寫錯誤。建議您從服務和動作清單中進行選取，以使用視覺化編輯器。然後，根據提示完成資源區段。如果服務不完全支援視覺化編輯器，您可能需要手動輸入某些政策部分。

  如果您確定自己的政策不包含上述任何錯誤，那麼您的政策可能包含拼寫錯誤。檢查是否有下列問題：

  - 拼寫錯誤的服務、動作和資源類型名稱，例如 `s2` 而非 `s3` 或 `ListMyBuckets` 而非 `ListAllMyBuckets`
  - ARN 中不必要的文字，例如 `arn:aws:s3: : :*`
  - 動作中缺少冒號，例如 `iam.CreateUser`

  您可以透過選擇下一步來檢視政策摘要，從而對可能包含拼寫錯誤的政策進行評估。然後，確認政策是否提供預期的許可。

使用政策摘要進行疑難排解

您可以診斷並解決與政策摘要相關的問題。

缺少政策摘要

IAM 主控台中提供了政策摘要表，這些表總結政策中對每個服務允許或拒絕的存取級別、資源和條件。政策摘要列於三個表中：政策摘要、服務摘要以及動作摘要。

政策摘要表包括由所選政策定義的服務清單和許可摘要。政策摘要可在任何附加政策的政策詳細資訊頁面上取得，或在受管政策的政策頁面上取得。如果 AWS 無法轉譯政策的摘要，主控台會顯示 JSON 政策文件和下列錯誤：

A summary for this policy cannot be generated. (無法產生此政策的摘要。) You can still view or edit the JSON policy document. (您仍可以檢視或編輯 JSON 政策文件。)

如果您的政策不包含摘要，則已發生下列錯誤之一：

- 不支援的政策元素：IAM 不支援為包含以下其中一項政策元素的政策產生政策摘要：
  - `Principal`
  - `NotPrincipal`
  - `NotResource`

- 無政策許可：如果政策不提供任何有效許可，則無法產生政策摘要。例如，如果政策包含元素 `"NotAction": "*"` 的單一陳述式，則它將授與對於除「所有動作」(*) 之外的所有動作的存取權。這代表它未授與對任何內容的 `Deny` 或 `Allow` 的存取權。

> **Note**
>
> 請謹慎使用 `NotPrincipal`、`NotAction` 和 `NotResource` 等政策元素。如需使用政策元素的資訊，請參閱 IAM JSON 政策元素參考。

如果提供不相符的服務和資源，則可能會建立一個不提供有效許可的政策。指定一個服務中的動作和另一個服務中的資源時可能會發生這種情況。在這種情況下，仍然會出現政策摘要。唯一顯示出有問題的跡象是摘要中的資源列可能包含來自不同服務的資源。如果此欄位包含不相符的資源，則您應該檢閱您的政策錯誤。使用政策模擬器來測試政策，從而更好地了解政策。

政策摘要包含無法識別的服務、動作或資源類型

在 IAM 主控台中，若政策摘要包含警告符號
( )，
則政策可能包含無法識別的服務、動作或資源類型。若要了解政策摘要中的警示，請參閱 政策摘要 (服務清單)。

> **Note**
>
> IAM 會對支援政策摘要的服務檢閱服務名稱、動作和資源類型。不過，您的政策摘要可能包含不存在資源值或條件。請一律使用政策模擬器來測試政策。

如果您的政策包含無法識別的服務、動作或資源類型，則存在以下錯誤之一：

- 預覽服務：處於預覽狀態的服務不支援政策摘要。
- 自訂服務：自訂服務不支援政策摘要。
- 服務不支援摘要：如果您的政策包括不支援政策摘要的公開提供 (GA) 服務，則該服務將包含在政策摘要表的 Unrecognized services (無法識別的服務) 部分。公開提供服務是公開發佈的服務，不是預覽或自訂服務。如果無法識別的服務是公開提供的，並且名稱拼寫正確，則該服務不支援 IAM 政策摘要。若要了解如何請求 GA 服務的政策摘要支援，請參閱 服務不支援 IAM 政策摘要。

- 動作不支援摘要：如果您的政策包含的受支援服務具有不受支援的動作，則該動作將包含在服務摘要表的 Unrecognized actions (無法識別的動作) 部分。若要了解服務摘要中的警示，請參閱 服務摘要 (動作清單)。
- 資源類型不支援摘要：如果您政策包含的受支援動作具有不受支援的資源類型，則該資源將包含在服務摘要表的 Unrecognized resource types (無法識別的資源類型) 部分。若要了解服務摘要中的警示，請參閱 服務摘要 (動作清單)。
- Typo – AWS 檢查 JSON 在語法上是否正確，並且政策不包含錯別字或其他錯誤作為政策驗證的一部分。

> **Note**
>
> 作為最佳實務，我們會建議使用您 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作。我們建議您開啟現有政策並檢閱並解決任何政策驗證建議。

服務不支援 IAM 政策摘要

IAM 政策摘要或視覺化編輯器可能不支援一般可用 (GA) 的服務或動作。公開提供服務是公開發佈的服務而非預覽或自訂服務。如果無法識別的服務是公開提供的，並且名稱拼寫正確，則該服務不支援這些功能。如果您的政策包含的受支援服務具有不支援的動作，則該服務不完全支援 IAM 政策摘要。

請求服務新增 IAM 政策摘要或視覺化編輯器支援

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 找到包含不受支援服務的政策：
   - 若政策為受管政策，請在導覽窗格中選擇 Policies (政策)。在政策清單中，選擇您要檢視的政策名稱。
   - 如果該政策是連接到使用者的內嵌政策，請在導覽窗格中選擇 Users (使用者)。在使用者清單中，選擇要檢視其政策的使用者的名稱。在使用者的政策表中，展開您要查看政策摘要的標題。
3. 在 AWS 管理主控台 頁尾的左側，選擇意見回饋。在 IAM 的意見反應方塊中，輸入 **`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`**。若您希望一個以上的服務支援摘要，請輸入 **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**。

請求服務新增對缺漏動作的 IAM 政策摘要支援

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 找到包含不受支援服務的政策：
   - 若政策為受管政策，請在導覽窗格中選擇 Policies (政策)。在政策清單中，選擇您要檢視的政策名稱。
   - 如果該政策是連接到使用者的內嵌政策，請在導覽窗格中選擇 Users (使用者)。在使用者清單中，選擇要檢視其政策的使用者的名稱。在使用者的政策表中，選擇您要檢視的政策名稱來展開政策摘要。
3. 在政策摘要中，選擇包含不支援動作的服務的名稱。
4. 在 AWS 管理主控台 頁尾的左側，選擇意見回饋。在 IAM 的意見反應方塊中，輸入 **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`**。若您希望報告超過一個未受支援的動作，請輸入 **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions`**。

若要請求包含缺少之動作的不同服務，請重複最後三個步驟。

我的政策未授與預期的許可

若要指派許可給使用者、群組、角色或資源，您必須建立一個政策，其為一個定義許可的文件。政策文件包含下列元素：

- 效果：政策是否允許或拒絕存取
- 動作：政策允許或拒絕的動作清單
- 資源：動作可在其中發生的資源清單
- 條件：(選擇性) 政策授與許可的條件

若要了解這些內容與其他政策元素，請參閱 IAM JSON 政策元素參考。

若要授與存取，您的政策必須定義含有支援的資源之動作。如果您的政策也包含條件，則該條件必須包含一個全域條件鍵或者必須套用到動作。若要了解受動作支援的資源，請參閱您服務的 AWS 文件。若要了解 動作支援的條件，請參閱 AWS 服務的動作、資源和條件金鑰。

檢查您的政策是否定義不會授予許可的動作、資源或條件。使用 IAM 主控台 (網址為 https://console.aws.amazon.com/iam/) 檢視政策的政策摘要。您可以使用政策摘要來辨識並修正政策中的問題。

對於元素不依照 IAM 政策中的定義授與許可的情況，原因可能有：

- 定義了動作，但是沒有適用的資源
- 定義了資源，但是沒有適用的動作
- 定義了條件，但是沒有適用的動作

若要檢視包含警告的政策摘要的範例，請參閱 the section called “政策摘要 (服務清單)”。

已定義動作，但是沒有適用的資源

下方的政策定義了所有 `ec2:Describe*` 動作和一個特定資源。這些動作都不支援資源級許可，因而未授與任何 `ec2:Describe` 動作。資源級許可表示動作使用政策的 `Resource` 元素中的 ARN 來支援資源。如果動作不支援資源層級許可，則政策中的陳述式必須在 `Resource` 元素中使用萬用字元 (*)。若要了解哪些服務支援資源層級的許可，請參閱 AWS 使用 IAM 的 服務。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "arn:aws:ec2:us-east-2:111122223333:instance/*"
        }
    ]
}
```

此政策不提供任何許可，而政策摘要包含下列錯誤：

```
This policy does not grant any permissions. To grant access, policies must
have an action that has an applicable resource or condition.
```

若要修正此政策，您必須使用 `Resource` 元素中的 *。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
}
```

已定義資源，但是沒有適用的動作

下面的政策定義了 Amazon S3 儲存貯體資源，但不包含可在該資源上執行的 S3 動作。此政策也會授與完整存取權限給所有 Amazon CloudFront 動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ]
        }
    ]
}
```

此政策為所有 CloudFront 動作提供許可。但是，因為政策定義 S3 `amzn-s3-demo-bucket` 資源卻未定義任何 S3 動作，政策摘要包含下列警告：

```
This policy defines some actions, resources, or conditions that do not
provide permissions. To grant access, policies must have an action that has
an applicable resource or condition.
```

若要修正此政策以提供 S3 儲存貯體許可，您必須定義可在儲存貯體資源上執行的 S3 動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudfront:*",
                "s3:CreateBucket",
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:cloudfront:*",
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

或者，若要修正此政策僅提供 CloudFront 許可，請移除 S3 資源。

已定義條件，但是沒有適用的動作

如果 S3 字首等同於 `custom` 而版本 ID 等同於 `1234`，下列政策會為所有 S3 資源定義兩項 Amazon S3 動作。但是，`s3:VersionId` 條件金鑰用於物件版本標記，不受所定義的儲存貯體動作的支援。若要了解 動作支援的條件，請參閱 AWS 服務的動作、資源和條件金鑰，然後選擇 服務以檢視條件金鑰的服務文件。

JSON

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

```
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

如果儲存貯體名稱包含 s3:ListBucketVersions 字首，此政策為 s3:ListBucket 動作和 custom 動作提供許可。但是，由於 s3:VersionId 條件不受任何定義的動作支援，政策摘要包含下列錯誤：

```
This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.
```

若要修正此政策來使用 S3 物件版本標記，您必須定義支援 s3:VersionId 條件金鑰的 S3 動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

此政策為政策中的每個動作和條件提供許可。但由於不存在單一動作符合兩個條件的情況，政策仍不提供任何許可。為解決這一問題，您必須建立兩個單獨的陳述式，每個陳述式只包含具有套用條件的動作。

若要修正此政策，請建立兩個陳述式。第一個陳述式包含支援 `s3:prefix` 條件的動作，第二個陳述式包含支援 `s3:VersionId` 條件的動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

故障排除政策管理

您可以診斷並解決與政策管理相關的問題。

在 IAM 帳戶中連接或分開政策

有些 AWS 受管政策會連結至 服務。這些政策僅用於該服務的某個服務連結角色。在 IAM 主控台中，當您檢視政策詳細資訊頁面時，該頁面會包含一個橫幅，顯示該政策已與服務建立關聯。您不可將此政策連接到 IAM 中的使用者、群組或角色。當您為服務建立與服務相關的角色時，該政策會自動連接到您的新角色。由於政策是必要的您無法將政策與服務相關角色分開。

根據 IAM 身分的活動來變更其政策

您可以為您的 IAM 身分 (使用者、群組和角色) 根據其活動更新政策。若要這麼做，請在 CloudTrail 中檢視您帳戶的事件事件歷史記錄 (Event istory)。CloudTrail 事件日誌包含可供您用來變更政策之許可的詳細事件資訊。

使用者或角色正在嘗試在 中執行動作， AWS 而該請求遭拒。

考慮使用者或角色是否應該有執行該動作的許可。若是如此，您可以在其政策中新增動作，甚至其所嘗試存取資源的 ARN。

使用者或角色具有他們未使用的許可。

考慮從其政策中移除這些許可。請確定您的政策僅授與執行必要動作所需的最低權限。

如需使用 CloudTrail 的詳細資訊，請參閱 AWS CloudTrail 使用者指南中的在 CloudTrail 主控台中檢閱 CloudTrail 事件。

JSON 政策文件故障排除

您可以診斷並解決與 JSON 政策文件相關的問題。

驗證您的政策

當您建立或編輯 JSON 政策時，IAM 可以執行政策驗證以協助您建立有效的政策。IAM 會識別 JSON 語法錯誤，而 IAM Access Analyzer 會提供額外的政策檢查及建議，協助您進一步改良政策。若要進一步了解政策驗證的資訊，請參閱 IAM 政策驗證。若要進一步了解 IAM Access Analyzer 政策檢查和可動作的建議，請參閱 IAM Access Analyzer 政策驗證。

我在 JSON 編輯器中沒有政策驗證的許可

在 中 AWS 管理主控台，如果您沒有檢視 IAM Access Analyzer 政策驗證結果的許可，您可能會收到下列錯誤：

```
You need permissions. You do not have the permissions required to perform
this operation. Ask your administrator to add permissions.
```

若要修正此錯誤，請要求管理員為您新增 `access-analyzer:ValidatePolicy` 許可。

多個 JSON 政策物件

IAM 政策必須僅包含一個 JSON 物件。可在兩旁放置 { } 括弧來表示物件。透過在外層括號對中嵌入額外的 { } 括號，將其他物件嵌套到 JSON 物件中。政策只能包含一對最外層的 { } 括號。以下範例不正確，因為它在最上層包含兩個物件 (以##標示)：

JSON

```
{
      "Version":"2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
   }
   {
     "Statement": {
```

```
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

不過，您可以使用正確的政策語法來達成上面範例的意圖。可以將兩個區塊合併到單個 Statement 元素中，而非包含兩個各自擁有 Statement 元素的完整政策物件。Statement 元素將有兩個物件的陣列做為其值，如以下範例所示 (以粗體標示)：

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
      ]
    }
```

多個 JSON 陳述式元素

此錯誤乍看可能會像是前一章節中錯誤的變形。但是，它在語法上是不同類型的錯誤。在以下範例中，頂層只有一個政策物件，由單一 { } 括弧對表示。但是，該物件包含兩個 Statement 元素。

一個 IAM 政策只能包含一個 Statement 元素，包含在冒號左側的名稱 (Statement) 和跟隨其後在冒號右側的值。Statement 元素的值必須是物件，以 { } 括弧表示，其中包含一個 Effect 元素、一個 Action 元素和一個 Resource 元素。以下範例不正確，因為它在政策物件中包含兩個 Statement 元素 (以##標示)：

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

值物件可以是多個值物件組成的陣列。若要解決此問題，可使用物件陣列將兩個 Statement 元素合併為一個元素，如以下範例所示 (以粗體標示)：

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
     ]
    }
```

Statement 元素的值是一種物件陣列。此示例中的陣列包含兩個物件，每個物件本身是 Statement 元素的正確值。陣列中的每個物件之間用逗號隔開。

在 JSON 陳述式元素中具有多個效果、動作或資源元素

在 Statement 名稱/值對的值側，物件只能包含一個 Effect 元素、一個 Action 元素和一個 Resource 元素。以下政策不正確，因為在 Effect 中有兩個 Statement 元素：

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

> **Note**
>
> 政策引擎不允許新的或再編輯的政策中出現此類錯誤。但是，政策引擎會繼續允許在引擎更新之前儲存的政策。現有包含錯誤的政策之行為如下所示：
>
> - 多個 Effect 元素：僅遵循最後一個 Effect 元素。忽略其他元素。
> - 多個 Action 元素：所有 Action 元素進行內部合併，被視為單一列表。
> - 多個 Resource 元素：所有 Resource 元素進行內部合併，被視為單一列表。
>
> 政策引擎不允許您儲存任何有語法錯誤的政策。在儲存之前修正政策中的錯誤。檢閱並糾正政策的任何政策驗證建議。

在每種情況下，解決方案都是刪除不正確的多餘元素。針對 Effect 元素，這十分簡單：若您希望先前的範例拒絕針對 Amazon EC2 執行個體的許可，您必須從政策移除 "Effect": "Allow", 一行，如下所示：

JSON

```
{
```

```
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:*",
        "Resource": "*"
      }
    }
```

但是，如果重複元素是 Action 或 Resource，則解決方法可能會更加複雜。您可能需要提供多個動作允許 (或拒絕) 許可，或者需要控制對多個資源的存取。例如，以下範例不正確，因為它有多個 Resource 元素 (以##標示)：

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Statement 元素的值物件中的每個必要元素都只能出現一次。解決方案是將每個值置於陣列中。以下範例運用將兩個單獨資源元素合併為一個以陣列為值物件的 Resource 元素，來對此進行說明 (以粗體標示)：

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket",
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
```

```
    }
  }
```

缺少 JSON 版本元素

Version 政策元素與政策版本不同。Version 政策元素是在政策內使用，並定義政策語言的版本。相比之下，當您在 IAM 中變更客戶管理政策時會建立政策版本。變更的政策不會覆寫現有的政策。IAM 反而會建立新版本的受管政策。若要進一步了解 Version 政策元素，請參閱 IAM JSON 政策元素：Version。若要進一步了解政策版本，請參閱 the section called “版本控制 IAM 政策”。

隨著 AWS 功能演進，新的功能會新增至 IAM 政策以支援這些功能。有時，政策語法更新包括新版本編號。如果您在政策中使用政策語法的較新功能，則必須告知政策分析引擎所使用的版本。

預設政策版本是「2008-10-17」。如果要使用之後推出的任何政策功能，則必須指定支援所需功能的版本編號。我們建議您一律包含最新的政策語法版本編號 (目前為 "Version": "2012-10-17")。

例如，以下政策不正確，因為它在資源 ARN 中使用政策變數 ${...}。但是，它沒有指定支援政策變數的政策語法版本 (以##標示)：

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

透過在政策頂層新增值 2012-10-17 (支援政策變數的第一個 IAM API 版本) 的 Version 元素，可解決此問題 (以粗體標示)：

JSON

```
{
  "Version":"2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
```

```
  }
}
```

對通行密鑰和 FIDO 安全金鑰進行疑難排解

使用此處的資訊，來協助您針對在使用 FIDO2 安全性金鑰時可能遇到的常見問題進行診斷。

主題

- 我無法啟用 FIDO 安全性金鑰
- 我無法使用 FIDO 安全性金鑰登入
- 我的 FIDO 安全性金鑰遺失或損壞
- 其他問題

我無法啟用 FIDO 安全性金鑰

請根據您的狀態，以 IAM 使用者或系統管理員身分參閱下列解決方案

IAM 使用者

如果您無法啟用 FIDO 安全性金鑰，請檢查下列各項：

- 您使用的是支援的組態嗎？

  IAM 支援透過 USB、Bluetooth 或 NFC 連線到裝置的 FIDO2 安全裝置。IAM 也支援平台驗證器，例如 TouchID 或 FaceID。IAM 不支援 Windows Hello 的本機通行密鑰註冊。若要建立和使用通行密鑰，Windows 使用者應該使用跨裝置身分驗證，您在其中使用諸如行動裝置等裝置的通行密鑰或硬體安全金鑰，登入諸如筆記型電腦等其他裝置。

  如需有關可與 WebAuthn 搭配使用之裝置和瀏覽器的資訊 AWS，請參閱 使用通行密鑰和安全金鑰的支援組態。
- 您正在使用任何瀏覽器外掛程式嗎？

  AWS 不支援使用外掛程式來新增 WebAuthn 瀏覽器支援。反之，請使用可提供原生 WebAuthn 標準支援的瀏覽器。

  即使您使用支援的瀏覽器，還是可能有與 WebAuthn 不相容的外掛程式。不相容的外掛程式可能會讓您無法啟用和使用 FIDO 相容安全金鑰。停用任何可能不相容的外掛程式，並重新啟動瀏覽器。然後，重試啟用 FIDO 安全金鑰。

- 您具有適當的許可嗎？

  如果您沒有上述任何相容性問題，則可能沒有適當的許可。請聯絡系統管理員。

系統管理員

如果 IAM 使用者無法使用支援的組態來啟用其 FIDO 安全金鑰，請檢查其許可。如需詳細範例，請參閱 IAM 教學課程：允許使用者管理其憑證和 MFA 設定。

我無法使用 FIDO 安全性金鑰登入

如果您無法使用 AWS 管理主控台 FIDO 安全金鑰登入 ，請先參閱 使用通行密鑰和安全金鑰的支援組態。如果您使用支援的組態，但無法登入，請聯絡系統管理員以尋求協助。

我的 FIDO 安全性金鑰遺失或損壞

以目前受支援 MFA 類型任意組合的最多八台 MFA 裝置可被指派給一個使用者。對於多台 MFA 裝置，您只需要有一台 MFA 裝置登入 AWS 管理主控台。取代 FIDO 安全金鑰與取代硬體 TOTP 權杖類似。如果遺失或破壞任何類型的 MFA 裝置，請參閱 在 IAM 中復原受 MFA 保護的身分。

其他問題

如果發生這裡未涵蓋的 FIDO 安全性金鑰問題，請採取下列行動之一：

- IAM 使用者：請聯絡系統管理員。
- AWS 帳戶 根使用者：聯絡 AWS Support。

IAM 角色疑難排解

使用此處的資訊，來協助您針對在使用 IAM 角色時所可能遇到的常見問題，進行診斷與排除。

主題

- 我無法擔任角色
- 我的 AWS 帳戶中出現新角色
- 我無法在 中編輯或刪除角色 AWS 帳戶
- 我未獲授權執行： iam:PassRole
- 為何我無法擔任具 12 小時工作階段的角色？ (AWS CLI， AWS API)

- 當我嘗試在 IAM 主控台中切換角色時收到錯誤
- 我的角色具有允許我執行動作的政策，但我收到「存取遭拒」
- 服務未建立角色的預設政策版本
- 主控台中沒有服務角色的使用案例

我無法擔任角色

請檢查以下內容：

- 若要允許使用者在角色工作階段中再次擔任目前角色，請在角色信任政策中將角色 ARN 或 AWS 帳戶 ARN 指定為主體。 AWS 服務 會提供運算資源，例如 Amazon EC2、Amazon ECS、Amazon EKS 和 Lambda，可提供臨時登入資料並自動更新這些登入資料。此可確保您始終擁有一組有效的憑證。對於這些服務，不需要再次擔任目前的角色即可取得臨時憑證。但是，如果您想要傳遞工作階段標籤或一個工作階段政策，則需要再次擔任目前的角色。若要了解如何修改角色信任政策以新增主要角色 ARN 或 AWS 帳戶 ARN，請參閱 更新角色信任政策。
- 當您使用 擔任角色時 AWS 管理主控台，請務必使用角色的確切名稱。在您擔任角色時，角色名稱會區分大小寫。
- 當您使用 AWS STS API 或 擔任角色時 AWS CLI，請務必在 ARN 中使用角色的確切名稱。在您擔任角色時，角色名稱會區分大小寫。
- 當您使用 SAML 型聯合身分提供者擔任角色並啟用 SAML 加密時，請確定已上傳 SAML 身分提供者的有效私有解密金鑰。如需詳細資訊，請參閱管理 SAML 加密金鑰。
- 確認您的 IAM 政策對於您要擔任的角色呼叫 `sts:AssumeRole` 授與了許可。您的 IAM 政策的 `Action` 元素必須允許您呼叫 `AssumeRole` 動作。此外，您的 IAM 政策的 `Resource` 元素必須指定您要擔任的角色。例如，`Resource` 元素可以透過其 Amazon Resource Name (ARN) 或萬用字元 (*) 來指定角色。例如，您必須至少有一個適用政策授與類似如下的許可：

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- 請確認您的 IAM 身分已用 IAM 政策要求的任何標籤加上標籤。例如，在以下政策許可中，`Condition` 元素要求作為請求擔任該角色的主體，也就是您，必須有特定標籤。您必須以 `department = HR` 或 `department = CS` 加上標籤。否則，您無法擔任該角色。如需有關標記 IAM 使用者和角色的詳細資訊，請參閱 the section called “IAM 資源的標籤”。

```
    "Effect": "Allow",
```

```
    "Action": "sts:AssumeRole",
    "Resource": "*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/department": [
            "HR",
            "CS"
        ]}}
```

- 確定您符合角色的信任政策中指定的所有條件。Condition 可以指定過期日期、外部 ID、或請求必須只來自特定的 IP 地址。考慮下列範例，如果目前的日期為指定的日期之後的任何時間，則政策永遠不會相符，且無法授與您擔任該角色的許可。

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
    "Condition": {
        "DateLessThan" : {
            "aws:CurrentTime" : "2016-05-01T12:00:00Z"
        }
    }
```

- 確認您呼叫的 AWS 帳戶 AssumeRole是您所擔任角色的信任實體。受信任實體被定義為角色的信任政策中的 Principal。以下範例是連接到您要擔任之角色的信任政策。在此範例中，您登入所使用的帳戶 ID 和 IAM 使用者必須是 123456789012。如果您的帳戶號碼沒有列在角色的信任政策的 Principal 元素中，則您無法擔任該角色。無論存取政策中授與您什麼許可均是如此。請注意，範例政策會限制發生在 2017 年 7 月 1 日到 2017 年 12 月 31 日 (包含)(UTC) 間的動作許可。如果您在這些日期之前或之後登入，則政策不會相符，而且您無法擔任角色。

```
    "Effect": "Allow",
    "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
    }
```

- 來源身分 – 管理員可以設定角色來要求身分傳遞自訂字串，以識別正在其中執行動作的人員或應用程式 AWS，稱為來源身分。確認所擔任的角色是否要求設定來源身分。如需來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。

我的 AWS 帳戶中出現新角色

有些 AWS 服務要求您使用直接連結至服務的唯一服務角色類型。此服務連結角色是由服務預先定義的，並且包含服務所需要的所有許可。這會使設定服務更為簡單，因為您不必手動新增必要的許可。如需服務連結角色的一般資訊，請參閱 建立服務連結角色。

當服務開始支援服務連結的角色時，您可能已經在使用服務。若是如此，您可能會收到一封電子郵件，通知您關於您的帳戶中的新角色。此角色包含服務代表您執行動作所需的所有許可。您不需要執行任何動作來支援此角色。不過，您不應從您的帳戶刪除該角色。如此可能會移除服務存取 AWS 資源所需的許可。您可以前往 IAM 主控台的 IAM Roles (角色) 頁面，檢視您帳戶中的服務連結角色。服務連結角色會在表格的 Trusted entities (受信任實體) 欄中以 (Service-linked role) ((服務連結角色)) 顯示。

如需哪些服務支援服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找在 Service-Linked Role (服務連結角色) 一欄中顯示 Yes (是) 的服務。如需使用服務的服務連結角色相關資訊，請選擇 Yes (是) 連結。

我無法在 中編輯或刪除角色 AWS 帳戶

您無法在 IAM 中刪除或編輯服務連結角色 的許可。這些角色包括服務所需的預先定義的信任和許可，以代表您執行動作。您可以使用 IAM 主控台 AWS CLI或 API，僅編輯服務連結角色的描述。您可以前往主控台的 IAM Roles (角色) 頁面，檢視您帳戶中的服務連結角色。服務連結角色會在表格的 Trusted entities (受信任實體) 欄中以 (Service-linked role) ((服務連結角色)) 顯示。在 Summary (摘要) 頁面上的橫幅，也會指出角色是一個服務連結角色。您可以僅透過連結的服務來管理和刪除這些角色，如果該服務支援動作。修改或刪除服務連結的角色時請務必謹慎，因為這樣可能移除服務存取 AWS 資源所需的許可。

如需哪些服務支援服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找在 Service-Linked Role (服務連結角色) 一欄中顯示 Yes (是) 的服務。

我未獲授權執行： iam:PassRole

當您建立服務連結的角色時，您必須擁有傳遞該角色到服務的許可。有些服務會自動在您在該服務中執行動作時，在您的帳戶中建立服務連結角色。例如，Amazon EC2 Auto Scaling 會在您第一次建立 Auto Scaling 群組時，為您建立 AWSServiceRoleForAutoScaling 服務連結角色。如果您嘗試建立 Auto Scaling 群組而無 PassRole 許可，您會收到以下錯誤：

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
```

```
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

若要修正此錯誤，請要求管理員為您新增 `iam:PassRole` 許可。

若要了解哪些服務支援服務連結角色，請參閱 AWS 使用 IAM 的 服務。若要了解服務是否為您自動建立服務連結角色，請選擇 Yes (是) 連結以檢視該服務的服務連結角色文件。

為何我無法擔任具 12 小時工作階段的角色？ (AWS CLI， AWS API)

當您使用 AWS STS `AssumeRole*` API 或 `assume-role*` CLI 操作來擔任角色時，您可以指定 `DurationSeconds` 參數的值。您可以指定從 900 秒 (15 分鐘) 到角色的 Maximum session duration (最大工作階段持續時間) 設定的值。如果您指定高於此設定的值，操作會失敗。此設定的上限值為 12 小時。例如，如果您指定 12 小時的工作階段持續時間，但是您的管理員設定最大工作階段持續時間為 6 小時，則您的操作會失敗。若要了解如何檢視角色的最大值，請參閱 更新角色的最大工作階段持續時間。

如果您使用角色鏈結 (使用角色來擔任第二個角色)，則您的工作階段上限為一小時。如果您隨後使用 `DurationSeconds` 參數提供大於一小時的值，則操作會失敗。

當我嘗試在 IAM 主控台中切換角色時收到錯誤

您在 Switch Role (切換角色) 頁面上輸入的資訊必須符合角色的資訊。否則，操作會失敗，並且您會收到下列錯誤：

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

如果您收到此錯誤，請確認下列資訊正確：

- 帳戶 ID 或別名 – AWS 帳戶 ID 是 12 位數字。您的帳戶可能有別名，這是一個易記的識別符，例如您的公司名稱，可以用來代替您的 AWS 帳戶 ID。您可以在此欄位中使用帳戶 ID 或別名。
- 角色名稱：角色名稱會區分大小寫。帳戶 ID 和角色名稱必須符合為角色設定的帳戶 ID 和角色名稱。

如果您持續收到錯誤訊息，請連絡您的管理員以驗證先前的資訊。角色信任政策或 IAM 使用者政策可能會限制您的存取。您的管理員可以驗證這些政策的許可。

我的角色具有允許我執行動作的政策，但我收到「存取遭拒」

您的角色工作階段可能受工作階段政策限制。當您使用 以程式設計方式請求臨時安全登入資料時 AWS STS，您可以選擇傳遞內嵌或受管工作階段政策。工作階段政策是一種進階政策，且您會在以程式設計方式建立角色的暫時憑證工作階段時，以參數方式傳遞。您可以使用 Policy 參數傳遞單一 JSON 內嵌工作階段政策文件。您可以使用 PolicyArns 參數，指定多達 10 個受管工作階段政策。所產生工作階段的許可會是角色的以身分為基礎的政策和工作階段政策的交集。或者，如果您的管理員或自訂程式為您提供暫時憑證，他們可能已包含限制您存取的工作階段政策。

服務未建立角色的預設政策版本

服務角色是服務擔任的角色，以代表您在您的帳戶中執行動作。當您設定某些 AWS 服務環境時，您必須定義服務要擔任的角色。在某些情況下，服務會為您在 IAM 中建立服務角色及其政策。雖然您可以從 IAM 之內修改或刪除服務角色及其政策，但 AWS 不建議這樣做。該角色和政策本應僅供該服務使用。如果您編輯政策並設定另一個環境，當服務嘗試使用相同的角色和政策時，操作可能會失敗。

例如，當您 AWS CodeBuild 第一次使用 時，服務會建立名為 的角色codebuild-RWBCore-service-role。該服務角色使用名為 codebuild-RWBCore-managed-policy 的政策。如果您編輯政策，它會建立新版本，並將該版本儲存為預設版本。如果您在 中執行後續操作 AWS CodeBuild，服務可能會嘗試更新政策。如果這樣做，您會收到下列錯誤：

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
RWBCore-service-role role. To continue, detach the policy from any other
identities and then delete the policy and the role.
```

如果您收到這個錯誤，您必須在 IAM 中進行變更，才能繼續進行您的服務操作。首先，將預設政策版本設定為 V1，然後重試操作。如果先前已刪除 V1，或選擇 V1 無法運作，請清理並刪除現有的政策和角色。

如需編輯受管政策的詳細資訊，請參閱 編輯客戶受管政策 (主控台)。如需政策版本的詳細資訊，請參閱 版本控制 IAM 政策。

刪除服務角色及其政策

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格上選擇 Policies (政策)。
3. 在政策清單中，選擇您要刪除的政策名稱。

4. 選擇連接的實體索引標籤，可檢視哪些 IAM 使用者、群組或角色使用此政策。如果其中任一個身分使用政策，請完成下列任務：

   a. 建立具有必要許可的新受管政策。若要確保身分在您的動作之前和之後具有相同的許可，請從現有政策複製 JSON 政策文件。然後，建立新的受管政策並貼上 JSON 文件，如使用 JSON 編輯器建立政策所述。

   b. 對於每個受影響的身分，請連接新的政策，然後分開舊的政策。如需詳細資訊，請參閱新增和移除 IAM 身分許可。

5. 在導覽窗格中，選擇 Roles (角色)。

6. 在角色清單中，選擇您要刪除的角色名稱。

7. 選擇 Trust relationships (信任關係) 標籤，以檢視哪些實體可以擔任角色。如果列出服務以外的任何實體，請完成下列任務：

   a. 建立新角色，這些角色信任那些實體。

   b. 您在上一個步驟中建立的政策。如果您略過了該步驟，請立即建立新的受管政策。

   c. 通知任何擔任該角色的人，他們不能再這樣做。為他們提供如何擔任新角色和具有相同許可的相關資訊。

8. 刪除策略。

9. 刪除角色。

主控台中沒有服務角色的使用案例

某些服務需要您手動建立服務角色，才能授與服務許可，以代表您執行動作。如果服務未列在 IAM 主控台中，您必須手動將服務列為受信任主體。如果您使用的服務或功能的文件不包含將服務列為受信任主體的說明，請提供頁面的意見回饋。

若要手動建立服務角色，您必須知道將擔任該角色之服務的服務主體。服務主體是用來將許可授與給服務的識別符。服務主體是由服務定義。

您可以檢查下列項目，找到某些服務的服務主體：

1. 打開 AWS 使用 IAM 的 服務.
2. 請檢查服務是否在 Service-linked roles (服務連結角色) 欄位中為 Yes (是)。
3. 選擇是連結，以檢視該服務的服務連結角色文件。
4. 尋找該服務的 Service-Linked Role Permissions (服務連結角色許可) 區段，以檢視服務主體。

您可以使用 AWS CLI 命令或 AWS API 操作手動建立服務角色。若要使用 IAM 主控台手動建立服務角色，請完成下列任務：

1. 使用您的帳戶 ID 建立 IAM 角色。請勿連接政策或授與任何許可。如需詳細資訊，請參閱 建立角色以將許可授予 IAM 使用者。
2. 開啟角色並編輯信任關係。角色必須信任服務，而不是信任帳戶。例如，更新下列 `Principal` 元素：

   ```
   "Principal": { "AWS": "arn:aws:iam::123456789012:root" }
   ```

   將主體變更為服務的值，例如 IAM。

   ```
   "Principal": { "Service": "iam.amazonaws.com" }
   ```

3. 將許可政策連接至角色，以新增服務所需的許可。
4. 返回需要許可的服務，並使用文件中描述的方法來通知服務有關新的服務角色。

對 IAM 和 Amazon EC2 進行疑難排解

以下資訊有助於排解 Amazon EC2 的 IAM 疑難問題。

主題

- 嘗試啟動執行個體時，我在 Amazon EC2 主控台 IAM 角色清單中看不到角色
- 我的執行個體上的憑證針對錯誤的角色
- 當我嘗試呼叫 AddRoleToInstanceProfile 時，出現 AccessDenied 錯誤
- Amazon EC2：嘗試使用角色啟動執行個體時，出現 AccessDenied 錯誤。
- 我無法存取 EC2 執行個體上的暫時安全憑證
- IAM 子目錄中的 info 文件的錯誤是什麼意思？

嘗試啟動執行個體時，我在 Amazon EC2 主控台 IAM 角色清單中看不到角色

請檢查以下內容：

- 如果您是以 IAM 使用者身分登入，請確認您有呼叫 `ListInstanceProfiles` 的許可。如需有關使用角色時所需許可的資訊，請參閱 使用 Amazon EC2 角色所需的許可。如需新增許可給使用者的詳細資訊，請參閱 管理 IAM 政策。

  如果您無法修改自己的許可，則必須聯絡可以操作 IAM 的管理員以更新您的許可。
- 如果使用 IAM CLI 或 API 建立角色，請確認下列事項：
  - 您已建立執行個體設定檔，並將角色新增至該執行個體設定檔。
  - 針對角色和執行個體設定檔使用相同的名稱。如果您以不同方式命名角色和執行個體設定檔，則在 Amazon EC2 主控台中將看不到正確的角色名稱。

  Amazon EC2 主控台中的 IAM 角色清單列出執行個體描述檔的名稱，而不是角色的名稱。您必須選擇包含所需角色的執行個體設定檔的名稱。如需執行個體設定檔的詳細資訊，請參閱 使用執行個體設定檔。

> **Note**
>
> 如果使用 IAM 主控台來建立角色，則無需使用執行個體描述檔。對於您在 IAM 主控台中建立的每個角色，將建立一個與角色同名的執行個體描述檔，該角色將自動新增到該執行個體描述檔中。執行個體描述檔只能包含一個 IAM 角色，並且無法增加該限制。

我的執行個體上的憑證針對錯誤的角色

在執行個體設定檔的角色最近可能已遭替換。若是如此，則您的應用程式將需要等待下一次自動排程的憑證輪換，您的角色憑證才可用。

若要強制變更，必須取消關聯執行個體描述檔，然後關聯執行個體描述檔，或者可以停止執行個體，然後重新啟動它。

當我嘗試呼叫 **AddRoleToInstanceProfile** 時，出現 **AccessDenied** 錯誤

如果您以 IAM 使用者身分提出請求，請確認您具有以下許可：

- `iam:AddRoleToInstanceProfile` 與資源符合執行個體設定檔 ARN (例如，`arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`)。

如需有關使用角色時所需許可的詳細資訊，請參閱 我該如何開始？。如需新增許可給使用者的詳細資訊，請參閱 管理 IAM 政策。

Amazon EC2：嘗試使用角色啟動執行個體時，出現 **AccessDenied** 錯誤。

請檢查以下內容：

- 啟動沒有執行個體設定檔的執行個體。這有助於確保問題僅限於 Amazon EC2 執行個體的 IAM 角色。
- 如果您以 IAM 使用者身分提出請求，請確認您具有以下許可：
  - `ec2:RunInstances` 使用萬用字元資源 (「*」)
  - `iam:PassRole` 與資源符合角色 ARN (例如，`arn:aws:iam::999999999999:role/ExampleRoleName`)
- 呼叫 IAM `GetInstanceProfile` 動作，以確保您使用的是有效的執行個體描述檔名稱或有效的執行個體描述檔 ARN。如需詳細資訊，請參閱在 Amazon EC2 執行個體中使用 IAM 角色。
- 呼叫 IAM `GetInstanceProfile` 動作，以確保執行個體描述檔具有角色。空白執行個體設定檔將失敗，並顯示 `AccessDenied` 錯誤。如需建立角色的詳細資訊，請參閱 IAM 角色建立。

如需有關使用角色時所需許可的詳細資訊，請參閱 我該如何開始？。如需新增許可給使用者的詳細資訊，請參閱 管理 IAM 政策。

我無法存取 EC2 執行個體上的暫時安全憑證

若要存取 EC2 執行個體上的暫時安全憑證，您必須先使用 IAM 主控台建立角色。然後啟動使用該角色的 EC2 執行個體，並檢查執行中的執行個體。如需詳細資訊，請參閱 使用 IAM 角色為在 Amazon EC2 執行個體上執行的應用程式授予許可 中的我該如何開始？

如果您仍然無法存取 EC2 執行個體上的暫時安全憑證，請檢查下列項目：

- 您是否可以存取 Instance Metadata Service (IMDS) 的另一部分？ 如果沒有，請檢查您沒有封鎖對 IMDS 請求存取的防火牆規則。

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/hostname; echo
```

- IMDS 的 `iam` 子目錄是否存在？ 如果不可以，請呼叫 EC2 `DescribeInstances` API 操作或使用 `aws ec2 describe-instances` CLI 命令，確認您的執行個體具有與其相關聯的 IAM 執行個體設定檔。

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- 檢查 IAM 子目錄中的 `info` 文件是否有錯誤。如果您有錯誤，請參閱 IAM 子目錄中的 `info` 文件的錯誤是什麼意思？ 以取得更多資訊。

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

IAM 子目錄中的 **`info`** 文件的錯誤是什麼意思？

此 **`iam/info`** 文件表示 **`"Code":"InstanceProfileNotFound"`**

您的 IAM 執行個體描述檔已被刪除，Amazon EC2 無法再為您的執行個體提供憑證。您必須將有效的執行個體描述檔連接至 Amazon EC2 執行個體。

如果存在具有該名稱的執行個體描述檔，則請檢查該執行個體描述檔是否已被刪除而建立了另一個同名的執行個體描述檔。

1. 呼叫 IAM `GetInstanceProfile` 操作以取得 `InstanceProfileId`。
2. 呼叫 Amazon EC2 `DescribeInstances` 操作以取得該執行個體的 `IamInstanceProfileId`。
3. 確認來自 IAM 操作的 `InstanceProfileId` 是否符合來自 Amazon EC2 操作的 `IamInstanceProfileId`。

如果 ID 不同，則連接到執行個體的執行個體設定檔不再有效。您必須將有效的執行個體設定檔連接至執行個體。

此 **`iam/info`** 文件表示成功，但指出 **`"Message":"Instance Profile does not contain a role..."`**

IAM `RemoveRoleFromInstanceProfile` 動作已從執行個體描述檔中移除該角色。您可以使用 IAM `AddRoleToInstanceProfile` 動作將角色連接到執行個體描述檔。您的應用程式需要等到下一次排程的重新整理才能存取該角色的憑證。

若要強制變更，必須取消關聯執行個體描述檔，然後關聯執行個體描述檔，或者可以停止執行個體，然後重新啟動它。

此 iam/security-credentials/[role-name] 文件表示 "Code":"AssumeRoleUnauthorizedAccess"

Amazon EC2 沒有可以擔任該角色的許可。擔任角色的許可由連接到角色的信任政策控制，如下面的範例所示。使用 IAM UpdateAssumeRolePolicy API 來更新信任政策。

JSON

```
{"Version":"2012-10-17",       "Statement": [{"Effect": "Allow","Principal":
 {"Service": ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

您的應用程式需要等到下一次自動排程的重新整理才能存取該角色的憑證。

若要強制變更，必須取消關聯執行個體描述檔，然後關聯執行個體描述檔，或者可以停止執行個體，然後重新啟動它。

對 IAM 和 Amazon S3 進行疑難排解

使用此處的資訊，協助您對使用 Amazon S3 和 IAM 時可能遇到的問題進行故障診斷和修正。

如何授與對 Amazon S3 儲存貯體的匿名存取權？

您使用在 principal 元素中指定萬用字元 (*) 的 Amazon S3 儲存貯體政策，這表示任何人都可以存取儲存貯體。透過匿名存取，任何人（包括沒有 的使用者 AWS 帳戶) 都可以存取儲存貯體。如需範例政策，請參閱 Amazon Simple Storage Service 使用者指南中的 Amazon S3 儲存貯體政策之範例案例。

我以 AWS 帳戶 根使用者身分登入。為何無法存取我帳戶下的 Amazon S3 儲存貯體？

在某些情況下，您可能擁有對 IAM 和 Amazon S3 具有完全存取權的 IAM 使用者。如果 IAM 使用者將儲存貯體政策指派給 Amazon S3 儲存貯體，並且未將 根使用者指定為主體，則該根使用者對該儲存貯體的存取將被拒絕。但作為根使用者，您仍然可以存取儲存貯體。若要這麼做，請修改儲存貯體政策來允許從 Amazon S3 主控台或 AWS CLI進行根使用者存取。請使用下列主體，將 *123456789012* 取代為 AWS 帳戶的 ID。

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

使用 IAM 對 SAML 聯合進行疑難排解

使用此處的資訊，來協助您針對在使用 SAML 2.0 和 AWS Identity and Access Management的聯合身分時所可能遇到的問題，進行故障診斷與排除。

主題

- 錯誤：您的請求包含無效的 SAML 回應。若要登出，請按一下這裡。
- 錯誤：在 AuthnResponse 中需要 RoleSessionName (服務：AWSSecurityTokenService；狀態代碼：400；錯誤代碼：InvalidIdentityToken)
- 錯誤：未獲授權，無法執行 sts:AssumeRoleWithSAML (服務：AWSSecurityTokenService；狀態代碼：403；錯誤代碼：AccessDenied)
- 錯誤：在 AuthnResponse 中的 RoleSessionName 必須符合 [a-zA-Z_0-9+=,.@-]{2,64} (服務：AWSSecurityTokenService；狀態代碼：400；錯誤代碼：InvalidIdentityToken)
- 錯誤：來源身分必須符合 【a-zA-Z_0-9+=，.@-】{2，256}，開頭不可為 "aws:" (服務：AWSSecurityTokenService；狀態碼：400；錯誤碼：InvalidIdentityToken)
- 錯誤：回應簽章無效 (服務：AWSSecurityTokenService；狀態代碼：400；錯誤代碼：InvalidIdentityToken)
- 錯誤：無效的私有金鑰。
- 錯誤：無法移除私有金鑰。
- 錯誤：無法移除私有金鑰，因為金鑰 ID 與私有金鑰不符。
- 錯誤：無法擔任角色：發行者不存在於指定的供應商 (服務：AWSOpenIdDiscoveryService；狀態代碼：400；錯誤代碼：AuthSamlInvalidSamlResponseException)
- 錯誤：無法剖析中繼資料。
- 錯誤：無法更新身分提供者。中繼資料或加密聲明未定義任何更新。
- 錯誤：無法將聲明加密模式設定為「必要」，因為未提供私有金鑰。
- 錯誤：無法在相同的請求中新增和移除私有金鑰。僅為兩個參數之一設定值。
- 錯誤：指定的供應商不存在。
- 錯誤：請求的 DurationSeconds 超過為此角色設定的 MaxSessionDuration。
- 錯誤：已達到私有金鑰限制 2。
- 錯誤：回應沒有包括必要的對象。

錯誤：您的請求包含無效的 SAML 回應。若要登出，請按一下這裡。

當身分提供者的 SAML 回應不包含 Name 設定為 `https://aws.amazon.com/SAML/Attributes/Role` 的屬性時，可能會發生此錯誤。該屬性必須包含一個或多個 `AttributeValue` 元素，每個元素都包含逗號分隔的字串對：

- 使用者可以映射到角色的 ARN
- SAML 供應商的 ARN

當身分提供者傳送的 SAML 屬性值包含前置或結尾空格，或 SAML 屬性值中包含其他無效字元時，也會發生此錯誤。如需有關 SAML 屬性預期值的詳細資訊，請參閱為身分驗證回應設定 SAML 聲明

如需詳細資訊，請參閱為身分驗證回應設定 SAML 聲明。若要在瀏覽器中查看 SAML 回應，請按照 在瀏覽器中檢視 SAML 回應 中列出的步驟操作。

錯誤：在 AuthnResponse 中需要 RoleSessionName (服務：AWSSecurityTokenService；狀態代碼：400；錯誤代碼：InvalidIdentityToken)

當身分提供者的 SAML 回應不包含 Name 設定為 `https://aws.amazon.com/SAML/Attributes/RoleSessionName` 的屬性時，可能會發生此錯誤。屬性值是使用者的識別符，通常是使用者 ID 或電子郵件地址。

如需更多詳細資訊，請參閱 為身分驗證回應設定 SAML 聲明。若要在瀏覽器中查看 SAML 回應，請按照 在瀏覽器中檢視 SAML 回應 中列出的步驟操作。

錯誤：未獲授權，無法執行 sts:AssumeRoleWithSAML (服務：AWSSecurityTokenService；狀態代碼：403；錯誤代碼：AccessDenied)

如果 SAML 回應中指定的 IAM 角色拼寫錯誤或不存在，則會發生此錯誤。請務必使用與您角色完全相同的名稱，因為角色名稱區分大小寫。修正 SAML 服務供應商組態中角色的名稱。

只有當您的角色信任政策包含 `sts:AssumeRoleWithSAML` 動作時，才允許您存取。如果您的 SAML 聲明設定為使用 `PrincipalTag` 屬性，則您的信任政策也必須包含 `sts:TagSession` 動作。如需有關工作階段標籤的詳細資訊，請參閱 在 中傳遞工作階段標籤 AWS STS。

如果您的角色信任政策中沒有 sts:SetSourceIdentity 許可，此錯誤可能發生。如果您的 SAML 聲明設定為使用 SourceIdentity 屬性，則您的信任政策也必須包含 sts:SetSourceIdentity 動作。如需來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。

如果聯合身分主體沒有擔任該角色的許可，也會發生此錯誤。該角色必須具有任政策，該政策將 IAM SAML 身分提供者的 ARN 指定為 Principal。該角色還包含控制哪些使用者可以擔任該角色的條件。確認您的使用者滿足條件的要求。

如果 SAML 回應不包括其中包含 NameID 的 Subject，也會發生此錯誤。

如需詳細資訊，請參閱啟用 SAML 2.0 聯合主體來存取 AWS 管理主控台及為身分驗證回應設定 SAML 聲明。若要在瀏覽器中查看 SAML 回應，請按照 在瀏覽器中檢視 SAML 回應 中列出的步驟操作。

錯誤：在 AuthnResponse 中的 RoleSessionName 必須符合 [a-zA-Z_0-9+=,.@-]{2,64} (服務：AWSSecurityTokenService；狀態代碼：400；錯誤代碼：InvalidIdentityToken)

如果 RoleSessionName 屬性值太長或包含無效的字元，則會發生此錯誤。最大有效長度為 64 個字元。

如需更多詳細資訊，請參閱 為身分驗證回應設定 SAML 聲明。若要在瀏覽器中查看 SAML 回應，請按照 在瀏覽器中檢視 SAML 回應 中列出的步驟操作。

錯誤：來源身分必須符合 【a-zA-Z_0-9+=，.@-】{2，256}，開頭不可為 **"aws:"** (服務：AWSSecurityTokenService；狀態碼：400；錯誤碼：InvalidIdentityToken)

如果 sourceIdentity 屬性值太長或包含無效的字元，則會發生此錯誤。有效長度上限為 256 個字元。如需來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。

如需建立 SAML 聲明的詳細資訊，請參閱 為身分驗證回應設定 SAML 聲明。若要在瀏覽器中查看 SAML 回應，請按照 在瀏覽器中檢視 SAML 回應 中列出的步驟操作。

錯誤：回應簽章無效 (服務：AWSSecurityTokenService；狀態代碼：400；錯誤代碼：InvalidIdentityToken)

當身分提供者的聯合中繼資料與 IAM 身分提供者的中繼資料不相符時，則會發生此錯誤。例如，身分服務供應商的中繼資料檔案可能已變更以更新過期的憑證。從身分服務供應商處下載更新的

SAML 中繼資料檔案。然後使用`aws iam update-saml-provider`跨平台 CLI 命令或 `Update-IAMSAMLProvider` PowerShell cmdlet，在您在 IAM 中定義的 AWS 身分提供者實體中更新它。

錯誤：無效的私有金鑰。

如果沒有正確格式化私有金鑰檔案，則可能會發生此錯誤。此錯誤可能會提供有關私有金鑰為何無效的其他詳細資訊：

- 金鑰已加密。
- 無法識別金鑰格式。私有金鑰檔案必須為 .pem 檔案。

當您在 在 IAM 中建立 SAML 身分提供者中時 AWS 管理主控台，您必須從身分提供者下載私有金鑰，以提供給 IAM 以啟用加密。私有金鑰必須是使用 AES-GCM 或 AES-CBC 加密演算法來解密 SAML 聲明的 .pem 檔案。

錯誤：無法移除私有金鑰。

當 SAML 加密設定為「必要」時，可能會發生此錯誤，您的請求會移除 IAM SAML 提供者的唯一私有解密金鑰。如需有關輪換私有金鑰的詳細資訊，請參閱 管理 SAML 加密金鑰。

錯誤：無法移除私有金鑰，因為金鑰 ID 與私有金鑰不符。

如果私有金鑰的 `keyId` 值不符合身分提供者私有金鑰檔案的金鑰 ID，則可能會發生此錯誤。

當您使用 update-saml-provider 或 UpdateSAMLProvider API 操作來移除 SAML 加密私有金鑰時，`RemovePrivateKey` 中的值必須是連接到身分提供者之私有金鑰的有效金鑰 ID。

錯誤：無法擔任角色：發行者不存在於指定的供應商 (服務：AWSOpenIdDiscoveryService；狀態代碼：400；錯誤代碼：AuthSamlInvalidSamlResponseException)

如果 SAML 回應中的發行者與在聯合中繼資料檔案內宣告的發行者不相符，便會發生此錯誤。當您在 IAM 中建立身分提供者 AWS 時，中繼資料檔案已上傳至 。

錯誤：無法剖析中繼資料。

如果沒有採用正確格式的中繼資料檔案，則可能會發生此錯誤。

當您在 中建立或管理 SAML 身分提供者時 AWS 管理主控台，您必須從身分提供者擷取 SAML 中繼資料文件。

此中繼資料檔案包括發行者名稱、過期資訊以及可用於驗證從 IdP 接收的 SAML 身分驗證回應 (聲明) 的金鑰。中繼資料檔案必須以 UTF-8 格式編碼，並且不含位元組順序記號 (BOM)。若要移除 BOM，您可以使用文字編輯工具，例如 Notepad++，將檔案編碼為 UTF-8。

包含在 SAML 中繼資料文件中的 X.509 憑證必須使用大小至少有 1024 位元的金鑰。此外，X.509 憑證也必須沒有任何重複的擴充。您可以使用擴充，但這些擴充只能在憑證中出現一次。如果 X.509 憑證不符合任一條件，IdP 建立會失敗，並傳回 "Unable to parse metadata" 錯誤。

如 SAML V2.0 Metadata Interoperability Profile Version 1.0 所定義，IAM 既不會評估 SAML 中繼資料文件中的 X.509 憑證是否過期，也不會對其採取動作。如果您擔心過期的 X.509 憑證，建議您監控憑證過期日期，並根據組織的治理和安全政策輪換憑證。

錯誤：無法更新身分提供者。中繼資料或加密聲明未定義任何更新。

如果使用 `update-saml-provider` CLI 或 `UpdateSAMLProvider` API 操作，但不在請求參數中提供更新值，則可能會發生此錯誤。如需有關更新 IAM SAML 提供者的詳細資訊，請參閱 在 IAM 中建立 SAML 身分提供者。

錯誤：無法將聲明加密模式設定為「必要」，因為未提供私有金鑰。

當您先前尚未上傳私有解密金鑰，且嘗試將 SAML 加密設定為「必要」，但未在請求中包含私有金鑰時，就會發生此錯誤。

使用 `create-saml-provider` CLI、`CreateSAMLProvider` API、`update-saml-provider` CLI 或 `UpdateSAMLProvider` API 操作來要求加密的 SAML 聲明時，務必確保已為您的 IAM SAML 提供者定義私有金鑰。

錯誤：無法在相同的請求中新增和移除私有金鑰。僅為兩個參數之一設定值。

如果新增和移除私有金鑰值都包含在相同請求中，則可能會發生此錯誤。

當您使用 update-saml-provider 或 UpdateSAMLProvider API 操作來輪換 SAML 加密私有金鑰檔案時，只能在請求中新增或移除私有金鑰。如果在移除私有金鑰時新增私有金鑰，操作會失敗。如需有關輪換私有金鑰的詳細資訊，請參閱 管理 SAML 加密金鑰。

錯誤：指定的供應商不存在。

如果 SAML 聲明中的提供者名稱與 IAM 中的提供者名稱不符，就可能發生此錯誤。如需檢視供應商名稱的詳細資訊，請參閱 在 IAM 中建立 SAML 身分提供者。

錯誤：請求的 DurationSeconds 超過為此角色設定的 MaxSessionDuration。

如果您從 AWS CLI 或 API 擔任角色，可能會發生此錯誤。

當您使用 assume-role-with-saml CLI 或 AssumeRoleWithSAML API 操作來擔任角色時，可以為 `DurationSeconds` 參數指定值。您可以指定從 900 秒 (15 分鐘) 到角色的最大工作階段持續時間設定的值。如果您指定高於此設定的值，操作會失敗。例如，如果您指定 12 小時的工作階段持續時間，但是您的管理員設定最大工作階段持續時間為 6 小時，則您的操作會失敗。若要了解如何檢視角色的最大值，請參閱 更新角色的最大工作階段持續時間。

錯誤：已達到私有金鑰限制 2。

如果嘗試將私有金鑰新增至身分提供者，可能會發生此錯誤。

可以為每個身分提供者儲存最多兩個私有金鑰。使用 update-saml-provider 或 UpdateSAMLProvider API 操作來新增第三個私有金鑰時，操作會失敗。

在新增新的私有金鑰之前，移除已過期的私有金鑰。如需有關輪換私有金鑰的詳細資訊，請參閱 管理 SAML 加密金鑰。

錯誤：回應沒有包括必要的對象。

如果對象 URL 與 SAML 組態中的身分提供者不相符，就可能會發生此錯誤。請確定您的身分提供者 (IdP) 依存方識別符完全符合 SAML 組態中提供的對象 URL (實體 ID)。

IAM 如何與其他 AWS 服務搭配使用

雖然 IAM 是您用來管理 IAM 資源的主要 AWS 服務，但所有其他 AWS 服務都與 IAM 搭配使用，以控制對您帳戶中資源的存取。

- CloudFormation

  CloudFormation 與 IAM 整合，可讓您在範本中 CloudFormation 定義和管理 IAM 資源。您可以使用 CloudFormation 為您佈建的其他 AWS 資源指定必要的 IAM 許可。 CloudFormation 也支援使用 IAM 角色來管理佈建和管理基礎設施 AWS 所需的登入資料，其偏離偵測功能可協助您維護 IAM 組態的完整性。

- AWS CloudShell

  當您存取 時 AWS CloudShell，您的身分驗證和授權會透過 IAM. AWS CloudShell runs 在指派給您的使用者或帳戶的 IAM 角色內容中處理。當您啟動時 AWS CloudShell，它會根據指派給您的 IAM 角色自動產生臨時安全登入資料。

- AWS SDKs

  AWS SDKs 透過處理身分驗證和授權程序、管理 AWS 登入資料，以及遵守 IAM 中定義的許可和政策來使用 IAM，以確保您的應用程式只能存取其獲授權使用的資源。SDK 提供取得和使用臨時安全憑證，以及驗證應用程式作業所需權限的機制。

如需使用 AWS IAM 的服務清單，以及服務支援的 IAM 功能，請參閱 AWS 使用 IAM 的 服務。

使用 建立 IAM 資源 CloudFormation

AWS Identity and Access Management 已與 整合 AWS CloudFormation，這項服務可協助您建立和設定 AWS 資源的模型，以減少建立和管理資源和基礎設施的時間。您可以建立範本來描述您想要的所有 AWS 資源 （例如存取金鑰、群組、群組政策、執行個體描述檔、受管政策、OIDC 供應商、內嵌政策、角色、角色政策、SAML 供應商、伺服器憑證、服務連結角色、使用者 （以及將使用者新增至群組）、使用者政策和虛擬 MFA 裝置），並為您 CloudFormation 佈建和設定這些資源。

使用 時 CloudFormation，您可以重複使用範本來一致且重複地設定 IAM 資源。描述您的資源一次，然後在多個 AWS 帳戶 和 區域中逐一佈建相同的資源。

IAM 和 CloudFormation 範本

若要佈建和設定 IAM 與相關服務的資源，您必須了解 CloudFormation 範本。範本是以 JSON 或 YAML 格式化的文本檔案。這些範本說明您要在 CloudFormation 堆疊中佈建的資源。如果您不熟悉 JSON 或 YAML，您可以使用 CloudFormation 設計工具來協助您開始使用 CloudFormation 範本。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》 中的什麼是 CloudFormation 設計器？。

IAM 支援在其中建立存取金鑰、群組、群組政策、執行個體描述檔、受管政策、OIDC 供應商、內嵌政策、角色、角色政策、SAML 供應商、伺服器憑證、服務連結角色、使用者 （以及將使用者新增至群組）、使用者政策和虛擬 MFA 裝置 CloudFormation。如需詳細資訊 (包括用於 IAM 資源的 JSON 和 YAML 範本範例)，請參閱《AWS CloudFormation 使用者指南》中的 AWS Identity and Access Management resource type reference。

您也可以建立範本來建立相關資源，例如角色和受管政策。

進一步了解 CloudFormation

若要進一步了解 CloudFormation，請參閱下列資源：

- AWS CloudFormation
- AWS CloudFormation 使用者指南
- CloudFormation API 參考
- AWS CloudFormation 命令列界面使用者指南

使用 AWS CloudShell 搭配 AWS Identity and Access Management

AWS CloudShell 是以瀏覽器為基礎的預先驗證 Shell，您可以直接從 啟動 AWS 管理主控台。您可以使用您偏好的 shell (Bash、PowerShell 或 Z shell) 對 AWS 服務 （包括 AWS Identity and Access Management) 執行 AWS CLI 命令。另外，您無需下載或安裝命令列工具即可執行此操作。

您可以從 啟動 AWS CloudShellAWS 管理主控台，而且您用來登入主控台的 AWS 登入資料會自動在新的 shell 工作階段中使用。此預先驗證 AWS CloudShell 使用者可讓您在使用 第 2 AWS CLI 版 （預先安裝在 Shell 的運算環境中） 與 IAM 等 AWS 服務互動時，略過設定登入資料。

取得 的 IAM 許可 AWS CloudShell

AWS Identity and Access Management管理員可以使用 提供的存取管理資源，將許可授予 IAM 使用者，讓他們可以存取 AWS CloudShell 和使用環境的功能。

管理員授予使用者存取權的最快速方法是透過 AWS 受管政策。AWS 受管政策是由 AWS建立並管理的獨立政策。下列適用於 CloudShell 的 AWS 受管政策可以連接到 IAM 身分：

- `AWSCloudShellFullAccess`：授予許可，以 AWS CloudShell 使用 並完整存取所有功能。

如果您想要限制 IAM 使用者可以執行的動作範圍 AWS CloudShell，您可以建立使用 `AWSCloudShellFullAccess`受管政策做為範本的自訂政策。如需限制 CloudShell 中使用者可用的動作的詳細資訊，請參閱AWS CloudShell 《 使用者指南》中的使用 IAM 政策管理 AWS CloudShell 存取和用量。

與 IAM 互動

AWS CloudShell 從 啟動後 AWS 管理主控台，您可以立即開始使用命令列界面與 IAM 互動。

> **Note**
>
> 在 AWS CLI 中使用 時 AWS CloudShell，您不需要下載或安裝任何其他資源。此外，因為您已經在 Shell 中驗證身分，因此無需設定憑證即可呼叫。

建立 IAM 群組，並使用 將 IAM 使用者新增至群組 AWS CloudShell

下列範例使用 CloudShell 建立 IAM 群組、將 IAM 使用者新增至群組，然後驗證命令是否成功。

1. 從 中 AWS 管理主控台，您可以選擇導覽列上可用的下列選項來啟動 CloudShell：
   - 選擇 CloudShell 圖示。
   - 開始在搜尋方塊中輸入「cloudshell」，然後選擇 CloudShell 選項。
2. 若要建立 IAM 群組，請在 CloudShell 命令列中輸入下列命令。在這個範例中，我們將群組命名為 east_coast：

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

   如果呼叫成功，命令列會顯示類似下列輸出的服務回應：

   ```
   {
      "Group": {
   ```

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

3. 若要將使用者新增至您建立的群組，請使用下列命令，指定群組名稱和使用者名稱。在此範例中，我們將群組命名為 east_coast，將使用者命名為 john：

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

4. 若要驗證使用者是否在群組中，請使用下列命令，指定群組名稱。在這個範例中，我們繼續使用群組 east_coast：

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

如果呼叫成功，命令列會顯示類似下列輸出的服務回應：

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

搭配 AWS SDK 使用此服務

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
AWS 適用於 Python v4 的 SDK	AWS 適用於 Python v4 的 SDK 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

如需此服務的特定範例，請參閱 使用 AWS SDKs IAM 程式碼範例。

可用性範例

找不到所需的內容嗎？ 請使用本頁面底部的提供意見回饋連結申請程式碼範例。

的參考資訊 AWS Identity and Access Management

使用本節中的這些主題，針對 IAM 和 AWS STS的各種面向來尋找詳細參考資料。

主題

- 使用 Amazon Resource Name (ARNs識別 AWS 資源
- IAM 識別碼
- IAM AWS STS 和配額
- 雙堆疊端點支援
- 介面 VPC 端點
- AWS 使用 IAM 的 服務
- AWS API 請求的簽章版本 4
- IAM JSON 政策參考

- 用於以程式設計方式進行存取的簡化 AWS 服務 資訊

使用 Amazon Resource Name (ARNs識別 AWS 資源

Amazon Resource Name (ARNs可唯一識別 AWS 資源。當您需要在所有 之間明確指定資源時 AWS，例如在 IAM 政策、Amazon Relational Database Service (Amazon RDS) 標籤和 API 呼叫中，我們需要 ARN。儘管 ARN 與任何識別資訊一樣，應該謹慎使用和共用，但不應將其視為秘密、敏感或機密資訊。

ARN 格式

以下是 ARN 的一般格式。特定格式視資源而定。若要使用 ARN，請以資源特定資訊取代*##*文字。請注意，有些資源的 ARN 會省略區域、帳戶 ID 或同時省略區域和帳戶 ID。

```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partition

資源所在的分割區。分割區是一組 AWS 區域。每個 AWS 帳戶的範圍都是一個分割區。

以下是支援的分割區：

- aws - AWS 區域
- aws-cn - 中國區域
- aws-us-gov - AWS GovCloud (US) 區域

service

識別 AWS 產品的服務命名空間。

region

區域代碼。例如，請為美國東部 (俄亥俄) 指定 us-east-2。如需區域代碼清單，請參閱 AWS 一般參考 中的區域端點。

account-id

擁有資源 AWS 的帳戶 ID，不含連字號。例如 123456789012。

resource-type

資源類型。例如，虛擬私有雲端 (VPC) 的 vpc。

resource-id

資源識別碼。這是資源名稱、資源 ID 或資源路徑。有些資源識別符包含父資源(sub-resource-type/parent-resource/sub-resource) 或限定詞，例如版本 (resource-type:resource-name:qualifier)。

範例

IAM 使用者

arn:aws:iam::*123456789012*:user/*john*

SNS 主題

arn:aws:sns:*us-east-1*:*123456789012*:*example-sns-topic-name*

VPC

arn:aws:ec2:*us-east-1*:*123456789012*:vpc/*vpc-0e9801d129EXAMPLE*

查詢資源的 ARN 格式

ARN 的確切格式取決於服務和資源類型。有些資源 ARN 可能包含路徑、變數或萬用字元。若要查詢特定 AWS 資源的 ARN 格式，請開啟服務授權參考、開啟服務的頁面，然後導覽至資源類型資料表。

ARN 中的路徑

資源 ARN 可以包含路徑。例如，在 Amazon S3 中，資源識別符是物件名稱，其中包含正斜線 (/) 來形成路徑。同樣地，IAM 使用者名稱和群組名稱可以包含路徑。IAM 路徑僅允許包含英數字元和下列字元：正斜線 (/)、加號 (+)、等號 (=)、逗號 (,)、句點 (.)、at 符號 (@)、底線 (_) 和連字號 (-)。

在路徑中使用萬用字元

路徑可以包含萬用字元，也就是星號 (*)。有些政策元素允許使用萬用字元，有些則不允許。萬用字元可以用於 Resource 或 NotResource 元素，但不能用於 Principal 或 NotPrincipal 元素。如需詳細資訊，請參閱IAM JSON 政策參考。

您可以指定 `role/*` 來表示帳戶 123456789012 中的所有角色，如下列範例所示：

```
arn:aws:iam::123456789012:role/*
```

也可以將資源名稱以萬用字元結尾。例如，可以指定 **`service-*`** 來表示所有以 `service` 開頭並以 `service-role1` 或 `service-test` 等不同字元結尾的角色：

```
arn:aws:iam::123456789012:role/service-*
```

下列範例展示了 Amazon S3 儲存貯體中物件的 ARN，其中資源名稱包含路徑。ARN `arn:aws:s3:::amzn-s3-demo-bucket/*` 適用於該儲存貯體中的所有物件，無論字首為何。ARN `arn:aws:s3:::amzn-s3-demo-bucket/Development/*` 則適用於在 **`/Development/`** 字首內建立的所有物件。

您也可以使用 ? 萬用字元來指定 ARN 中的一個字元。例如，您可以將下列 ARN 用於名為 amzn-s3-demo-bucket 的 S3 貯體中，所有以四個字元開頭並以 **`-test`** 結尾的資料夾。某些資料夾符合此要求，包括 `1234-test`、`2024-test` 或 `a100-test`。

```
arn:aws:s3:::amzn-s3-demo-bucket/????-test
```

您也可以在 ARN 的不同區段中使用萬用字元，以冒號「**`:`**」分隔。在下列範例中，使用了兩個萬用字元來比對帳戶 123456789012 所有區域中的所有 Amazon Q 應用程式及應用程式內的資源：

```
arn:aws:qbusiness:*:123456789012:*
```

同樣地，下列範例比對了帳戶 123456789012 所有區域中的所有 Amazon VPC：

```
arn:aws:ec2:*:123456789012:vpc/*
```

下列範例比對了帳戶 123456789012 所有區域中的所有 Amazon EBS 磁碟區：

```
arn:aws:ec2:*:123456789012:volume/*
```

ARN 內萬用字元用途限制

您無法使用 ARN 部分中指定資源類型的萬用字元。下列在資源類型內包含萬用字元的範例 ARN 無效：

```
arn:aws:lambda:us-east-2:123456789012:functi*:my-function <== not allowed
```

> **Note**
>
> 當您在身分型政策中指定不完整的 ARN（少於標準六個欄位）時，AWS 會自動將萬用字元 (*) 新增至所有缺少的欄位，以完成 ARN。例如，指定 `arn:aws:sqs` 等同於 `arn:aws:sqs:*:*:*`，這會授予所有區域和帳戶所有 Amazon SQS 資源的存取權。不過，傳遞至 AWS STS AssumeRole、AssumeRoleWithWebIdentity 和 AssumeRoleWithSAML 請求的工作階段政策不支援不完整ARNs。在工作階段政策中使用不完整的 ARN 將導致`MalformedPolicyDocumentException`錯誤。

您也無法在字首 ARN 中使用萬用字元，或在 ARN 的分區區段中使用萬用字元。

```
arn:aws:redshift:us-east-1:123456789012:? <== not allowed
```

IAM 識別碼

IAM 對使用者、IAM 群組、角色、政策和伺服器憑證使用一些不同的識別碼。本節介紹識別碼以及何時使用識別碼。

主題

- 易用名稱和路徑
- IAM ARN
- 唯一識別碼
- 了解唯一 ID 字首

- 獲取唯一識別碼

易用名稱和路徑

當您建立使用者、角色、使用者群組或政策，或上傳伺服器憑證時，請為其提供好記的名稱。例如 Bob、TestApp1、Developers、ManageCredentialsPermissions 或 ProdServerCert 等等。

如果您使用 IAM API 或 AWS Command Line Interface (AWS CLI) 來建立 IAM 資源，您可以新增選用路徑。您可以使用單一路徑，也可以巢狀處理多個路徑作為資料夾結構。例如，您可以使用巢狀路徑 `/division_abc/subdivision_xyz/product_1234/engineering/` 以符合您的公司的組織結構。然後，您可以建立一個政策，以允許該路徑中的所有使用者存取政策模擬器 API。若要查看此政策，請參閱 IAM：根據使用者路徑存取政策模擬器 API。如需如何指定易記名稱的資訊，請參閱使用者 API 文件。如需有關如何使用路徑的其他範例的詳細資訊，請參閱 IAM ARN。

當您使用 CloudFormation 建立資源時，您可以為使用者、IAM 群組和角色以及客戶受管政策指定路徑。

如果您的使用者和使用者群組位於相同路徑中，IAM 不會自動將使用者置於該使用者群組中。例如，您可以建立一個開發人員使用者群組並將其路徑指定為 `/division_abc/subdivision_xyz/product_1234/engineering/`。如果您建立一個名為 Bob 的使用者，並為他新增相同的路徑，則此操作不會自動將 Bob 置於 Developers 使用者群組。IAM 不會根據使用者或 IAM 群組的路徑在它們之前強制執行任何邊界。如果具有不同路徑的用戶被授予對這些資源的許可，則他們可以使用相同的資源。 AWS 帳戶中 IAM 資源的數量和大小有限。如需詳細資訊，請參閱IAM AWS STS 和配額。

IAM ARN

大多數的資源都有易用名稱 (例如，使用者名稱 Bob 或使用者群組名稱 Developers)。但是，許可政策語言要求您使用下列 Amazon Resource Name (ARN) 格式指定一或多個資源。

```
arn:partition:service:region:account:resource
```

其中：

- `partition` 辨識資源所在的分割區。對於標準 AWS 區域，分割區為 `aws`。如果您有其他分割區的資源，則該分割區為 `aws-partitionname`。例如，中國 (北京) 區域的資源分割區，即為 `aws-cn`。您無法在不同分割區的帳戶之間委派存取權。
- `service` 會識別 AWS 產品。IAM 資源一律使用 `iam`。
- `region` 會識別資源的區域。對 IAM 資源而言，此項目一律空白。

- `account` 指定沒有連字號的 AWS 帳戶 ID。
- `resource` 是按名稱辨識特定資源。

您可以使用下列語法指定 IAM 和 AWS STS ARNs。ARN 的區域部分是空白的，因為 IAM 資源是全域。

語法：

```
arn:aws:iam::account:root
arn:aws:iam::account:user/user-name-with-path
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:sts::account:self
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
arn:aws:iam::aws:contextProvider/context-provider-name
```

以下許多範例包括 ARN 的資源部分中的路徑。無法在 AWS 管理主控台中建立或操控路徑。若要使用路徑，您必須使用 AWS API、 AWS CLI或 Tools for Windows PowerShell 來使用 資源。

範例：

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/John
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
```

```
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/John
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Jane
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneMFA
arn:aws:iam::123456789012:u2f/user/John/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
arn:aws:iam::123456789012:oidc-provider/server.example.org
arn:aws:iam::aws:contextProvider/IdentityCenter
```

下列範例提供更多詳細資訊，協助您了解不同類型的 IAM 和資源的 ARN 格式 AWS STS 。

- 帳戶中的 IAM 使用者：

> **Note**
>
> 每個 IAM 使用者名稱都是唯一的。對於使用者，使用者名稱不區分大小寫 (例如在登入程序期間)，但在政策中或作為 ARN 的一部分使用時，該使用者名稱會區分大小寫。

```
arn:aws:iam::123456789012:user/John
```

- 另一個具有反映組織結構圖之路徑的使用者：

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
```

- IAM 使用者群組：

```
arn:aws:iam::123456789012:group/Developers
```

- 具有路徑的 IAM 使用者群組：

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- IAM 角色：

```
arn:aws:iam::123456789012:role/S3Access
```

- 服務連結角色：

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- 服務角色：

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- 受管政策：

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- 可以與 Amazon EC2 執行個體建立關聯的執行個體設定檔：

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- 在 AWS STS IAM 中識別為「Paulo」的聯合身分使用者主體：

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- 有人擔任「會計 - 角色」角色，角色工作階段名稱為 "Mary" 的活動工作階段：

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- 在 API 呼叫中使用呼叫者自己的工作階段做為資源時，例如在呼叫工作階段上操作的 AWS STS SetContext API：

```
arn:aws:sts::123456789012:self
```

- 指派給名為 Jorge 之使用者的多重要素驗證裝置：

```
arn:aws:iam::123456789012:mfa/Jorge
```

- 伺服器憑證：

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- 具有反映組織結構圖之路徑的伺服器憑證：

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- 身分提供者 (SAML 和 OIDC)：

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- OIDC 身分識別提供者，具有可反映 Amazon EKS OIDC 身分識別提供者 URL 的路徑：

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

- 與 AWS IAM Identity Center 信任的身分傳播搭配使用的內容提供者 ARN，從中產生信任的內容聲明：

```
arn:aws:iam::aws:contextProvider/IdentityCenter
```

另一個重要的 ARN 是根使用者 ARN。雖然這不是 IAM 資源，仍建議您熟悉此 ARN 的格式。它通常會用在資源型政策的 Principal 元素。

- AWS 帳戶 會顯示下列項目：

```
arn:aws:iam::123456789012:root
```

以下範例顯示您可以指派給 Richard 的政策，以允許他管理自己的存取金鑰。請注意，該資源是 IAM 使用者 Richard。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/
Richard"
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> 當您使用 ARN 識別 IAM 政策中的資源時，可以包含政策變數。政策變數可以包含執行時間資訊的預留位置 (例如使用者名稱)，作為 ARN 的一部分。如需詳細資訊，請參閱IAM 政策元素：變數與標籤

在 ARN 中使用萬用字元和路徑

您可以在 ARN 的##部分中使用萬用字元來指定多個使用者、IAM 群組或政策。例如，若要指定在 product_1234 上工作的所有使用者，則使用：

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

如果您有使用者的名稱是以字串 app_ 開頭，您可以使用以下 ARN 引用他們。

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

若要在 中指定所有使用者、IAM 群組或政策 AWS 帳戶，請分別在 ARN 的 user/、 group/或 policy/部分後面使用萬用字元。

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

如果您為使用者 arn:aws:iam::111122223333:user/* 指定以下 ARN，即符合以下兩個範例。

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

但是，如果您為使用者 arn:aws:iam::111122223333:user/division_abc* 指定以下 ARN，則符合第二個範例，但不符合第一個範例。

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Jane
```

不要在 ARN 的 user/、group/ 或 policy/ 部分使用萬用字元。例如，IAM 不允許下列項目：

```
arn:aws:iam::123456789012:u*
```

> **Note**
>
> 當您在身分型政策中指定不完整的 ARN （少於標準六個欄位） 時， AWS 會自動將萬用字元 (*) 新增至所有缺少的欄位，以完成 ARN。例如，指定 arn:aws:sqs 等同於 arn:aws:sqs:*:*:*，這會授予所有區域和帳戶所有 Amazon SQS 資源的存取權。不過，傳遞至 AWS STS AssumeRole、AssumeRoleWithWebIdentity 和 AssumeRoleWithSAML 請求的工作階段政策不支援不完整ARNs。在工作階段政策中使用不完整的 ARN 將導致MalformedPolicyDocumentException錯誤。

Example將路徑和 ARN 用於以專案為基礎之使用者群組的範例

無法在 AWS 管理主控台中建立或操控路徑。若要使用路徑，您必須使用 AWS API、 AWS CLI或 Tools for Windows PowerShell 來使用 資源。

在這個範例中，Marketing_Admin 使用者群組中的 Jules 在 /marketing/ 路徑中建立了專案類型使用者群組。Jules 會將來自公司不同部門的使用者指派至使用者群組。此範例說明使用者的路徑與使用者所在的使用者群組無關。

這個行銷群組有一個他們將推出的新產品，因此 Jules 在 /marketing/ path 中建立一個名為 Widget_Launch 的新的使用者群組。然後，Jules 將以下政策指派給使用者群組，讓該使用者群組存取屬於指定給此特定啟動之 example_bucket 的物件。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/
*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/
widget/*"}}
    }
  ]
}
```

然後，將此 Jules 將正在進行此次啟動的使用者指派給該使用者群組。這包括來自 /marketing/ 路徑的 Patricia 和 Eli。也包括來自 /sales/ 路徑的 Chris 和 Chloe，以及 /legal/ 路徑的 Aline 和 Jim。

唯一識別碼

當 IAM 建立使用者、使用者群組、角色、政策、執行個體設定檔或伺服器憑證時，它會指派給每個資源唯一的 ID。唯一 ID 類似於如下：

AIDAJQABLZS4A3QDU576Q

在大多數情況下，當您使用 IAM 資源時，使用的都是易記名稱和 ARN。如此即不需要知道特定資源的唯一 ID。但是，當使用易用名稱不可行時，唯一 ID 有時非常有用的。

其中一個範例會重複使用 中的易記名稱 AWS 帳戶。在您的帳戶、使用者、使用者群組、角色或政策的易用名稱必須是唯一的。例如，您可以建立名為 John 的 IAM 使用者。您的公司使用 Amazon S3，且有一個儲存貯體包含每位員工的資料夾。IAM 使用者 John 是 名為 User-S3-Access 的 IAM 使用者群組成員，其具有允許使用者僅存取儲存貯體中自己的資料夾的許可。如需如何建立身分型政策的範例，該政策允許 IAM 使用者採用使用者易記的名稱在 S3 中存取自己的儲存貯體物件，請參閱 Amazon S3：可讓 IAM 使用者以程式設計方式和在主控台存取其 S3 主目錄。

假設名為 John 的員工離開您的公司並刪除名為 John 的對應 IAM 使用者。但後來另一位名叫 John 的員工開始建立一個名為 John 的新 IAM 使用者。將名為 John 的 IAM 使用者新增至現有的 IAM 使用者群組 User-S3-Access。如果與使用者群組關聯的政策指定友好的 IAM 使用者名稱 John，則該政策會允許新的 John 存取前任 John 留下的資訊。

一般而言，我們建議您為政策中的資源指定 ARN，而非其唯一的 ID。但是，即使您建立一個重新使用之前刪除的易用名稱的新 IAM 使用者，每個 IAM 使用者都有一個唯一的 ID。在此範例中，舊 IAM 使用者 John 和新 IAM 使用者 John 具有不同的唯一 ID。您可以建立以資源為基礎的政策，依唯一 ID 授予存取權，而不僅僅依使用者名稱授予存取權。為免無意中將資訊存取權授予不該擁有此存取權的員工，這樣做可以降低這種情況發生的機率。

下列範例展示您如何在資源型政策的 Principal 元素中指定唯一的 ID。

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

下列範例展示您如何使用全域條件索引鍵 aws:userid 在政策的 Condition 元素中指定唯一的 ID。

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

使用者 ID 可以非常有用的另一個範例是，如果您維護自己的 IAM 使用者或角色資訊資料庫 (或其他存放區)。唯一 ID 可為您建立的每個 IAM 使用者或角色提供唯一識別碼。若您有重複使用名稱的 IAM 使用者或角色，就會出現這種情況，如上例所示。

了解唯一 ID 字首

IAM 使用以下字首，指示每個唯一的 ID 適用於哪種類型的資源。字首可能會因建立時間而有所不同。

字首	資源類型
ABIA	AWS STS 服務承載字符
ACCA	環境特定的憑證
AGPA	使用者群組
AIDA	IAM 使用者
AIPA	Amazon EC2 執行個體設定檔
AKIA	存取金鑰
ANPA	受管政策
ANVA	受管政策中的版本
APKA	公有金鑰
AROA	角色
ASCA	憑證
ASIA	暫時 (AWS STS) 存取金鑰 ID 使用此字首，但只有在與私密存取金鑰和工作階段權杖結合使用時才是唯一的。

獲取唯一識別碼

IAM 主控台中沒有 IAM 資源的唯一 ID。若要取得唯一 ID，您可以使用下列 AWS CLI 命令或 IAM API 呼叫。

AWS CLI:

- get-caller-identity
- get-group
- get-role
- get-user

- get-policy
- get-instance-profile
- get-server-certificate

IAM API：

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAM AWS STS 和配額

AWS Identity and Access Management (IAM) 和 AWS Security Token Service (STS) 具有限制物件大小的配額。這會影響您命名物件的方式、您可以建立的物件數，以及您在傳遞物件時可使用的字元數。

> **Note**
>
> 若要取得 IAM 用量和配額的帳戶層級資訊，請使用 GetAccountSummary API 操作或 get-account-summary AWS CLI 命令。

IAM 名稱需求

IAM 名稱具有下列需求與限制：

- 政策文件只能包含下列 Unicode 字元：水平定位字元 (U+0009)、換行字元 (U+000A)、歸位字元 (U+000D)，以及 U+0020 到 U+00FF 範圍內的字元。
- 使用者、群組、角色、政策、執行個體設定檔、伺服器憑證和路徑的名稱必須是英數字元，包括以下常見的字元：加號 (+)、等號 (=)、逗號 (,)、句號 (.)、at 符號 (@)、底線 (_) 和連字號 (-)。路徑名稱必須以正斜線 (/) 開頭和結尾。

- 使用者、群組、角色和執行個體設定檔的名稱在帳戶中必須是唯一的。名稱不分大小寫。例如：您無法同時建立名為 **ADMINS** 和 **admins** 的群組。
- 第三方用於擔任角色的外部 ID 值必須最少為 2 個字元，最多為 1,224 個字元。該值必須為英數字元，且不包含空格。也可以包含下列符號：加號 (+)、等號 (=)、逗號 (,)、句號 (.)、小老鼠 (@)、冒號 (:)、正斜線 (/) 和連字號 (-)。如需有關外部 ID 的詳細資訊，請參閱 存取第三方 AWS 帳戶 擁有的。
- 內嵌政策的政策名稱必須對其內嵌的使用者、群組或角色是唯一。此名稱可以包含任何基本拉丁文 (ASCII) 字元，但下列預留字元除外：反斜線 (\)、正斜線 (/)、星號 (*)、問號 (?) 和空格。這些字元是根據 RFC 3986 第 2.2 條予以保留。
- 使用者密碼 (登入設定檔) 可以包含任何基本拉丁文 (ASCII) 字元。
- AWS 帳戶 ID 別名在各 AWS 產品中必須是唯一的，而且必須依照 DNS 命名慣例為英數字元。別名必須為小寫，且不得以連字號開始或結束，不可包含兩個連續的連字號，也不能是 12 位數的號碼。

如需基本拉丁文 (ASCII) 字元的清單，請參閱國會圖書館基本拉丁文 (ASCII) 程式碼表。

IAM 物件配額

配額也稱為 中的限制 AWS，是 中資源、動作和項目的最大值 AWS 帳戶。使用 Service Quotas 來管理您的 IAM 配額。

如需 IAM 服務端點和 Service Quotas 的清單，請參閱 AWS 一般參考 中的 AWS Identity and Access Management 端點和配額。

請求提高配額

1. 按照《AWS 登入使用者指南》如何登入 AWS 主題中所述適合您的使用者類型之登入程序操作來登入 AWS 管理主控台。
2. 開啟 Service Quotas 主控台。
3. 在導覽窗格中，選擇 AWS services (AWS 服務)。
4. 在導覽列上，選擇 US East (N. Virginia) (美國東部 (維吉尼亞北部)) 區域。然後搜尋 **IAM**。
5. 選擇 AWS Identity and Access Management (IAM)、選擇配額，然後依照指示請求增加配額。

如需詳細資訊，請參閱《Service Quotas 使用者指南》中的請求增加配額。

如要查看如何使用 Service Quotas 主控台要求增加 IAM 配額的範例，請觀看下列影片。

使用 Service Quotas 主控台請求增加 IAM 配額。

您可以為可調整配額申請提升預設 IAM 配額。到達 maximum quota 的請求會自動核准，並在幾分鐘內完成。

下表列出可自動核准配額增加的資源。

資源	預設配額	最大配額
每個帳戶的客戶受管政策	1500	10000
每個帳戶的群組	300	500
每個帳戶的執行個體設定檔	1000	10000
每個角色的受管政策	20	25
每個使用者的受管政策	10	20
每個群組的受管政策	10	10
角色信任政策長度	2,048 個字元	8192 個字元
每個帳戶的角色	1000	10000
每個帳戶的伺服器憑證	20	20
每個帳戶的 OpenId 連線供應商	100	700

IAM Access Analyzer 配額

如需 IAM Access Analyzer 服務端點和 Service Quotas 的清單，請參閱 AWS 一般參考 中的 IAM Access Analyzer 端點和配額。

IAM Roles Anywhere 配額

如需 IAM Roles Anywhere 服務端點和 Service Quotas 的清單，請參閱 AWS 一般參考 中的 AWS Identity and Access Management Roles Anywhere 端點和配額。

STS 請求配額

AWS Security Token Service (AWS STS) 會強制執行下列請求配額。

對於使用AWS 登入資料提出 AWS STS 的請求，預設請求配額為每個區域每個帳戶每秒 600 個請求。下列 AWS STS 操作共用此配額：

- AssumeRole
- DecodeAuthorizationMessage
- GetAccessKeyInfo
- GetCallerIdentity
- GetFederationToken
- GetSessionToken

> **Note**
>
> AWS 服務主體對 AWS STS 的請求，例如用來擔任角色以與服務搭配使用 AWS 的請求，不會在您的帳戶中使用每秒 STS 請求配額。

例如，如果 會在相同區域中每秒 AWS 帳戶 發出 100 個 GetCallerIdentity 請求和每秒發出 100 個 AssumeRole 呼叫，則該帳戶會消耗該區域每秒 200 個可用的 600 個 STS 請求。

對於跨帳戶 AssumeRole 請求，只有發出 AssumeRole 請求的帳戶會影響 STS 配額。目標帳戶沒有消耗任何配額。

若要請求增加 STS 請求配額，請透過 AWS 支援來開立票證。

> **Note**
>
> 隨著 AWS STS 全球端點 (`https://sts.amazonaws.com`) 即將變更，對全球端點的請求不會與預設啟用區域中 AWS STS 的區域端點共用每秒請求數 (RPS) 配額。當全域 AWS STS 端點的請求來自單一區域時，它會計入全域端點的 RPS 配額。不過，如果請求來自多個區域，每個額外的區域都會獲得其獨立的 RPS 配額。如需 AWS STS 全域端點變更的詳細資訊，請參閱 AWS STS 全域端點變更。

IAM 和 STS 字元限制

以下是 IAM 和 AWS STS的字元數上限和大小上限：您無法請求提高下列限制。

說明	限制
AWS 帳戶 ID 的別名	3–63 個字元
對於內嵌政策	您可以新增任意數量的內嵌政策到 IAM 使用者、角色或群組。但是每個實體的總計彙總政策大小 (所有內嵌政策的大小總和) 不能超過下列限制： • 使用者政策大小不可超過 2,048 個字元。 • 角色政策大小不可超過 10,240 個字元。 • 群組政策大小不可超過 5,120 個字元。 **Note** 在根據這些限制計算政策的大小時，IAM 不會計算空格數。
對於客戶受管政策	• 每個受管政策的大小不可超過 6,144 個字元。 **Note** 在根據此限制計算政策的大小時，IAM 不會計算空格數。
Group name (群組名稱)	128 個字元
執行個體設定檔名稱	128 個字元
登入設定檔的密碼	1–128 個字元
路徑	512 個字元

說明	限制
政策名稱	128 個字元
角色名稱	64 個字元 **Important** 如果您想要在 中使用具有切換角色功能的角色 AWS 管理主控台，則合併後的 `Path`和 `RoleName`不得超過 64 個字元。
角色工作階段持續時間	12 小時 當您從 AWS CLI 或 API 擔任角色時，您可以使用 `duration-seconds` CLI 參數或 `DurationSeconds` API 參數來請求較長的角色工作階段。您可以指定值從 900 秒 (15 分鐘) 到角色的最長工作階段持續時間設定，範圍為 1–12 小時。如果您不為 `DurationSeconds` 參數指定一個值，則您的安全憑證有效期為 1 小時。在主控台中切換角色的 IAM 使用者會被授予最長工作階段持續時間或使用者工作階段中的剩餘時間，以較短者為準。工作階段持續時間設定上限不會限制 AWS 服務擔任的工作階段。若要了解如何檢視角色的最大值，請參閱 更新角色的最大工作階段持續時間。
角色工作階段名稱	64 個字元

說明	限制
角色工作階段政策	• 所傳遞的 JSON 政策文件和所有傳遞的受管政策 ARN 字元的大小總計不可超過 2,048 個字元。 • 當您建立工作階段時，最多可傳遞 10 個受管政策 ARN。 • 當您以程式設計方式為角色或聯合身分使用者主體建立暫時工作階段時，您只能傳遞一個 JSON AWS STS 政策文件。 • 此外，工作階段政策和工作階段標籤會新增至具有個別限制的工作階段字符大小。`SessionTokenUtilization` 回應元素會顯示字符使用的大小限制百分比。`PackedPolicySize` 回應元素已棄用。請改用 `SessionTokenUtilization`。 • 我們建議您使用 AWS CLI 或 AWS API 傳遞工作階段政策。 AWS 管理主控台 可能會透過新增工作階段資訊來增加工作階段字符大小。
角色工作階段標籤	• 工作階段標籤必須符合 128 個字元的標籤鍵限制，以及 256 個字元的標籤值限制。 • 您最多可以傳遞 50 個工作階段標籤。 • 工作階段政策和工作階段標籤會新增至具有個別限制的工作階段字符大小。您可以使用 AWS CLI 或 AWS API 傳遞工作階段標籤。`SessionTokenUtilization` 回應元素會顯示字符使用的大小限制百分比。`PackedPolicySize` 回應元素已棄用。請改用 `SessionTokenUtilization`。

說明	限制
SAML 驗證回應 base64 編碼	100,000 個字元 此字元限制適用於 `assume-role-with-saml` CLI 或 `AssumeRoleWithSAML` API 操作。
標籤鍵	128 個字元 此字元限制適用於 IAM 資源和工作階段標籤上的標籤。
標籤值	256 個字元 此字元限制適用於 IAM 資源和工作階段標籤上的標籤。 標籤值可為空，也就是說標籤值的長度為 0 個字元。
IAM 建立的唯一 ID	128 個字元。例如： • 以 AIDA 開頭的使用者 ID • 以 AGPA 開頭的群組 ID • 以 AROA 開頭的角色 ID • 以 ANPA 開頭的受管政策 ID • 以 ASCA 開頭的伺服器憑證 ID **Note** 這不是詳盡清單，也不保證特定類型的 ID 僅以指定的字母組合開頭。
使用者名稱	64 個字元

雙堆疊端點支援

雙堆疊端點可讓用戶端使用 IPv4 或 IPv6 地址與 AWS 服務通訊。IAM 和 都 AWS STS 提供雙堆疊端點支援。如需有關如何將 VPC 設定為雙堆疊模式的詳細資訊，請參閱 Amazon VPC User Guide 中的 IPv6 support for your VPC。

如果您使用包含 `aws:sourceIp`或 `aws:vpcSourceIp`條件金鑰的 IAM 政策，則需要更新這些政策以支援 IPv6 用戶端。如需有關為 VPC 進行 IPv6 定址的詳細資訊，請參閱 Amazon VPC User Guide 中的 IP addressing for your VPCs and subnets。

IAM 雙堆疊端點支援

IAM 提供同時支援 IPv4 和 IPv6 用戶端的雙堆疊公有端點。您也可以使用 從您的虛擬私有雲端 (VPC) 私下存取 IAM 雙堆疊公有端點 AWS PrivateLink。如需有關為 IAM 建立私有介面 VPC 端點的詳細資訊，請參閱建立 IAM 的 VPC 端點。

IAM 雙堆疊公有端點為 `https://iam.global.api.aws`。

與雙堆疊公有端點不同，位於 `https://iam.amazonaws.com` 的 IAM 公有端點僅支援 IPv4 用戶端。當透過 AWS PrivateLink從 VPC 進行私有存取時，該 IAM 公有端點可以同時支援 IPv4 和 IPv6 用戶端。

AWS STS 雙堆疊端點支援

AWS STS 提供支援 IPv4 和 IPv6 用戶端的雙堆疊區域端點。您也可以使用 從您的虛擬私有雲端 (VPC) 私下存取 AWS STS 雙堆疊區域端點 AWS PrivateLink。如需為 建立私有介面 VPC 端點的詳細資訊 AWS STS，請參閱 建立 的 VPC 端點 AWS STS。

下表顯示依分割區的 AWS STS 雙堆疊區域端點：

分割區	雙堆疊端點 URL
商業 AWS 區域	一般端點： `https://sts.` *`<region>`*`.api.aws` FIPS 端點： `https://sts-fips.` *`<region>`*`.api.aws`
AWS GovCloud (US) 區域	一般端點： `https://sts.` *`<region>`*`.api.aws`
中國區域	一般端點： `https://sts.` *`<region>`*`.api.amazonwebservices.com.cn`

> **Note**
>
> FIPS 端點僅適用於美國和加拿大區域。

限制

下列限制適用於 AWS STS 雙堆疊端點支援：

- 全域端點 不支援 IPv6 用戶端`https://sts.amazonaws.com`。您必須使用雙堆疊區域端點來支援 IPv6 用戶端。
- 不支援IPv6-only AWS STS VPC 端點。VPC 端點可以設定為 IPv4 或雙堆疊連線。

介面 VPC 端點

如果您使用 Amazon Virtual Private Cloud (Amazon VPC) 託管 AWS 資源，您可以在 VPC 與 AWS Identity and Access Management (IAM) 或 AWS Security Token Service () 之間建立私有連線AWS STS。您可以使用此連線來啟用 IAM 或 與 VPC 中的資源 AWS STS 通訊，而無需透過公有網際網路。

Amazon VPC 是一種 AWS 服務，可用來在您定義的虛擬網路中啟動 AWS 資源。您可利用 VPC 來控制您的網路設定，例如 IP 地址範圍、子網路、路由表和網路閘道。若要將 VPC 連線至 IAM 或 AWS STS，請為每個服務定義介面 VPC 端點。端點為 IAM 提供可靠、可擴展的連線，或 AWS STS 不需要網際網路閘道、網路位址轉譯 (NAT) 執行個體或 VPN 連線。如需詳細資訊，請參閱 Amazon VPC 使用者指南中的什麼是 Amazon VPC？。

介面 VPC 端點採用 AWS PrivateLink 一種 AWS 技術，可使用具有私有 IP 地址的彈性網路介面，在 AWS 服務之間進行私有通訊。如需詳細資訊，請參閱 AWS PrivateLink for AWS Services。

以下資訊適用於 Amazon VPC 的使用者。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 Amazon VPC 入門。

主題

- VPC 端點可用性
- 建立 IAM 的 VPC 端點
- 建立 的 VPC 端點 AWS STS
- 為 OIDC 探索建立 VPC AWS STS 端點

- AWS STS 使用 VPC 端點政策控制對 的存取
- 使用 VPC 條件金鑰來控制聯合存取

VPC 端點可用性

> **Important**
>
> 只能在 IAM 控制平面所在的區域中建立 IAM 的介面 VPC 端點。如果您的 VPC 位於與 IAM 控制平面區域不同的區域，您必須使用 AWS Transit Gateway 來允許從另一個區域存取 IAM 介面 VPC 端點。如需詳細資訊，請參閱建立 IAM 的 VPC 端點。

IAM 目前在下列區域支援 VPC 端點：

- 美國東部 (維吉尼亞北部)
- 中國 (北京)
- AWS GovCloud （美國西部）

AWS STS 目前在下列區域中支援 VPC 端點：

- 美國東部 (維吉尼亞北部)
- 美國東部 (俄亥俄)
- 美國西部 (加利佛尼亞北部)
- 美國西部 (奧勒岡)
- 非洲 (開普敦)
- 亞太地區 (香港)
- 亞太區域 (海德拉巴)
- 亞太地區 (雅加達)
- 亞太地區 (墨爾本)
- 亞太地區 (孟買)
- 亞太區域 (大阪)
- 亞太區域 (首爾)
- 亞太區域 (新加坡)

- 亞太地區 (雪梨)
- 亞太區域 (東京)
- 加拿大 (中部)
- 加拿大西部 (卡加利)
- 中國 (北京)
- 中國 (寧夏)
- 歐洲 (法蘭克福)
- 歐洲 (愛爾蘭)
- 歐洲 (倫敦)
- 歐洲 (米蘭)
- Europe (Paris)
- 歐洲 (西班牙)
- 歐洲 (斯德哥爾摩)
- 歐洲 (蘇黎世)
- 以色列 (特拉維夫)
- Middle East (Bahrain)
- 中東 (阿拉伯聯合大公國)
- 南美洲 (聖保羅)
- AWS GovCloud （美國東部）
- AWS GovCloud （美國西部）

建立 IAM 的 VPC 端點

若要開始使用 IAM 與 VPC，請建立 IAM 的介面 VPC 端點。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用介面 VPC 端點存取 AWS 服務。

只能在 IAM 控制平面所在的區域中建立 IAM 的介面 VPC 端點。在商業上 AWS 區域，IAM 控制平面位於美國東部（維吉尼亞北部）區域 (us-east-1)。IAM 的 AWS PrivateLink 介面 VPC 端點服務名稱為 `com.amazonaws.iam`。如需 AWS 區域 支援 IAM VPC 端點的清單，請參閱 VPC 端點可用性。

如果您的 VPC 位於與 IAM 控制平面區域不同的區域，則必須使用 AWS Transit Gateway 來允許從另一個區域存取 IAM 介面 VPC 端點。

使用 從不同區域中的 VPC 存取 IAM 介面 VPC 端點 AWS Transit Gateway

1. 建立傳輸閘道，或使用現有的傳輸閘道來互連虛擬私有雲端 (VPC)。每個區域均需要傳輸閘道。如需詳細資訊，請參閱《AWS Transit Gateway 指南》中的建立傳輸閘道。
2. 建立傳輸閘道 VPC 連接，將每個 VPC 連線到傳輸閘道。如需詳細資訊，請參閱《AWS Transit Gateway 指南》中的建立 VPC 的傳輸閘道連接。
3. 建立傳輸閘道 VPC 對等連接，以在對等 VPC 之間路由流量。如需詳細資訊，請參閱《AWS Transit Gateway 指南》中的建立對等連接。

> **Note**
>
> VPC 對等互連也可以在對等 VPC 之間路由流量，但如果存在大量 VPC 時，此方法無法很好地擴展。建議不要使用 VPC 對等互連，而是使用 AWS Transit Gateway 對等連接，它透過可擴展的中央樞紐來改善 VPC 和內部部署網路管理。如需 VPC 對等互連的詳細資訊，請參閱《Amazon VPC 對等互連指南》中的使用 VPC 對等互連。

建立 的 VPC 端點 AWS STS

若要開始 AWS STS 搭配 VPC 使用 ，請為 建立介面 VPC 端點 AWS STS。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用介面 VPC 端點存取 AWS 服務。

建立 VPC 端點之後，您必須使用相符的區域端點來傳送您的 AWS STS 請求。 AWS STS 建議使用 `setRegion`和 `setEndpoint`方法來呼叫區域端點。您可以單獨將 `setRegion` 方法用在手動啟用的區域，例如亞太區域 (香港)。在這種情況下，系統會將呼叫導向 STS 區域端點。如要了解如何手動啟用區域，請參閱 AWS 一般參考 中的管理 AWS 區域。如果您單獨將 `setRegion` 方法用在預設啟用的區域，系統會將呼叫導向 `https://sts.amazonaws.com` 的全域端點。

當您使用區域端點時， AWS 會使用公有端點或私有介面 VPC 端點 AWS STS 呼叫其他服務，以使用中者為準。例如，假設您已建立 的介面 VPC 端點， AWS STS 並已從位於 VPC 中的 AWS STS 資源請求臨時登入資料。在這種情況下，依預設這些憑證會開始流過界面 VPC 端點。如需使用 提出區域請求的詳細資訊 AWS STS，請參閱 在 AWS STS 中管理 AWS 區域。

為 OIDC 探索建立 VPC AWS STS 端點

AWS Security Token Service (AWS STS) OIDC 探索是一對公有端點，提供依賴各方驗證傳出聯合身分所發行 JSON Web 權杖 (JWTs) 所需的中繼資料和驗證金鑰。

當您啟用傳出聯合身分時， 會 AWS STS 發佈帳戶特定的發行者 URL。此 URL 託管兩個 OpenID Connect (OIDC) 探索端點。依賴方擷取這些端點，以取得驗證 JWTs GetWebIdentityToken 問題所需的中繼資料和公有金鑰：

- `{issuer_url}/.well-known/openid-configuration` — OIDC 探索文件
- `{issuer_url}/.well-known/jwks.json` — 包含字符簽章驗證金鑰的 JSON Web 金鑰集 (JWKS)

根據預設，從 VPC 到達這些端點需要網際網路閘道、NAT 裝置、VPN 連接或連線。若要改為在 AWS 網路上保留此流量，請為 STS OIDC 探索端點建立介面 VPC 端點。

> **Note**
>
> STS OIDC 探索端點僅提供上述兩個端點。它不提供任何 AWS STS API 操作。若要私下呼叫 `GetWebIdentityToken`、 `AssumeRole`或任何其他 AWS STS 操作，請使用 `com.amazonaws.region.sts`端點服務。請參閱 建立 的 VPC 端點 AWS STS。

選擇正確的端點服務

傳出聯合身分使用兩個不同的網路路徑。大多數同時具有最低和驗證權杖的工作負載都需要兩個端點服務。

任務	端點服務	身分驗證
減少 JWT (`sts:GetWebIdentityToken`)	`com.amazonaws.region.sts`	AWS SigV4
讀取 `/.well-known/openid-configuration`	`com.amazonaws.region.sts-oidc`	無（未驗證）
讀取 `/.well-known/jwks.json`	`com.amazonaws.region.sts-oidc`	無（未驗證）

如果驗證字符的唯一方是外部服務，則不需要 STS OIDC 探索端點 AWS。此服務會透過公有網際網路擷取發行者 URL，其流量不會進入您的 VPC。

考量事項

在為 STS OIDC 探索建立介面 VPC 端點之前，請考慮下列事項。

- 不支援 VPC 端點政策。這兩個探索 APIs都提供可公開取得的資料，而 STS OIDC 探索端點不會強制執行 VPC 端點政策。使用安全群組和子網路放置來控制哪些資源可以到達端點。
- 只會提供兩個路徑。端點`/.well-known/jwks.json`僅接受對 `/.well-known/openid-configuration`和 的 HTTPS `GET`請求。所有其他路徑和方法都會傳回錯誤。

為 STS OIDC 探索建立介面 VPC 端點

您可以使用 Amazon Virtual Private Cloud 主控台或 ，為 STS OIDC 探索端點建立 VPC 端點 AWS Command Line Interface。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用介面 VPC 端點存取 AWS 服務。

使用下列服務名稱為 STS OIDC 探索建立 VPC 端點：

- `com.amazonaws.`*`region`*`.sts-oidc`

如果您為端點啟用私有 DNS，您可以將請求傳送至 ，`sts-oidc.`*`region`*`.amazonaws.com`它們將解析為您 VPC 內端點的網路介面。

透過端點傳送請求

當 VPC 中的依賴方需要驗證 JWT 時，它會從發行者 URL 擷取 OIDC 探索文件和 JWKS。發行者主機名稱 （ 例如 *`uuid`*`.tokens.sts.global.api.aws`) 使用與端點 (`sts-oidc.`*`region`*`.amazonaws.com`) 不同的網域，因此預設不會解析為 VPC 端點。

若要透過端點路由發行者主機名稱的請求，請為您的發行者主機名稱建立 Amazon Route 53 私有託管區域，並新增指向 VPC 端點的別名記錄。建立託管區域之後，VPC 中的資源會將發行者主機名稱解析為端點的網路介面。然後，標準 OIDC 程式庫可以擷取沒有程式碼變更的探索文件。如需如何建立私有託管區域的詳細資訊，請參閱 Amazon Route 53 開發人員指南中的使用私有託管區域。

> **Note**
>
> 端點服務是區域性的，但單一端點可為分割區中的任何發行者 UUID 提供探索請求。在驗證者執行的區域中建立端點；每個發行者帳戶不需要一個端點。

AWS STS 使用 VPC 端點政策控制對 的存取

當您為 AWS Security Token Service (AWS STS) 建立介面 VPC 端點時，您可以連接端點政策。政策控制哪些主體可以使用端點，以及他們可以執行哪些 AWS STS 動作。如果您未連接政策，端點會使用預設政策，允許不受限制地存取所有委託人的所有 AWS STS 動作。

VPC 端點政策不會自行授予許可。它們可做為與其他政策一起運作的額外界限。端點政策和發起人的適用政策都必須允許請求成功。

如需 VPC 端點政策的詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用端點政策控制對 VPC 端點的存取。

主題

- 預設 VPC 端點政策
- AWS STS VPC 端點政策的重要考量事項
- 適用於 AWS STS VPC 端點政策的條件索引鍵
- 範例：允許組織的所有 AWS STS 動作
- 範例：允許特定帳戶的所有 AWS STS 動作
- 範例：限制為特定 AWS STS 動作
- 範例：拒絕非組織存取，同時允許聯合

預設 VPC 端點政策

如果您在建立端點時未連接自訂政策， 會 AWS 連接下列預設政策。此政策允許無限制存取端點。

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

若要限制對端點的存取，請連接自訂端點政策。

AWS STS VPC 端點政策的重要考量事項

AWS STS 處理來自兩種基本上不同類型來電者的請求。您的 VPC 端點政策必須考慮這兩種類型，以避免意外封鎖合法請求。

已驗證的 AWS 主體

使用 AWS Signature 第 4 版 (SigV4) 簽署請求的 IAM 使用者和 IAM 角色。這些呼叫者在請求內容中具有標準條件索引鍵`aws:PrincipalOrgID`，例如 `aws:PrincipalAccount`、 和 `aws:PrincipalArn` 。

聯合發起人

呼叫 `AssumeRoleWithSAML`或 的 SAML 2.0 和 OpenID Connect (OIDC) 主體`AssumeRoleWithWebIdentity`。這些發起人使用 SAML 聲明或 JSON Web 權杖 (JWTs) 進行身分驗證，而不是 SigV4 簽章。由於它們在請求時沒有 AWS 身分，因此請求內容不包含主體型條件索引鍵，例如 `aws:PrincipalOrgID`、 `aws:PrincipalAccount`和 `aws:PrincipalArn`。

> **Important**
>
> 如果您的 VPC 端點政策僅依賴 `aws:PrincipalOrgID`來允許存取，則會隱含拒絕聯合`AssumeRoleWithSAML`和`AssumeRoleWithWebIdentity`呼叫，因為非AWS 委託人沒有條件金鑰。

如何 AWS STS 評估聯合發起人的 VPC 端點政策

當聯合發起人`AssumeRoleWithWebIdentity`透過 VPC 端點叫用 `AssumeRoleWithSAML`或 時，適用下列情況：

- 發起人不是 AWS 委託人。請求內容中無法使用條件索引鍵`aws:PrincipalOrgID`，例如 `aws:PrincipalAccount`、 和 `aws:PrincipalArn` 。
- 聯合發起人在請求內容`aws:PrincipalIsAWSService`中沒有 條件索引鍵。
- 正在擔任的角色是 AWS 資源。`aws:ResourceAccount` 可使用 `aws:ResourceOrgID`和 等資源型條件索引鍵，並參考目標角色。
- 角色的信任政策仍然是聯合存取的主要授權閘道。VPC 端點政策提供額外的網路層級界限。

我們建議您在撰寫必須套用至聯合發起人的 VPC 端點政策陳述式aws:ResourceAccount時使用aws:ResourceOrgID或 ，因為這些發起人沒有可用的主體型條件金鑰。

適用於 AWS STS VPC 端點政策的條件索引鍵

下表顯示透過 AWS STS VPC 端點提出請求時，每個呼叫者類型的請求內容中常用的條件索引鍵及其可用性。除了此處列出的條件索引鍵之外，還有其他條件索引鍵可用。

依呼叫者類型的條件索引鍵可用性

條件鍵	已驗證的 AWS 主體	聯合發起人	說明
aws:PrincipalOrgID	是	否	呼叫主體的組織 ID
aws:PrincipalAccount	是	否	呼叫主體的帳戶 ID
aws:PrincipalArn	是	否	呼叫主體的 ARN
aws:PrincipalIsAWSService	是 （評估為 false)	否 （金鑰不存在）	發起人是否為 AWS 服務主體
aws:ResourceOrgID	是	是	擁有所請求資源之帳戶的組織 ID
aws:ResourceAccount	是	是	擁有所請求資源的帳戶 ID

> **Note**
>
> 對於已驗證的 AWS 主體 (IAM 使用者和角色) aws:PrincipalIsAWSService， 會出現在請求內容中，並評估為 false。對於聯合發起人，此金鑰完全不存在於請求內容中。檢查 的條

件"Bool": {"aws:PrincipalIsAWSService": "false"}不符合聯合發起人，因為金鑰不存在。

範例：允許組織的所有 AWS STS 動作

下列端點政策會將 AWS STS VPC 端點限制在您的組織，同時支援聯合存取。它允許組織中已驗證主體的所有 AWS STS 動作，並分別允許聯合發起人擔任組織中的角色。聯合發起人 (AssumeRoleWithSAML 和 AssumeRoleWithWebIdentity) 需要單獨的陳述式，因為他們在請求內容aws:PrincipalOrgID中沒有 。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrganizationPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
        {
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
```

```
        }
    ]
}
```

第一個陳述式使用 "Principal": {"AWS": "*"}搭配 aws:PrincipalOrgID來允許來自您組織的已驗證 AWS 主體。第二個陳述式使用 "Principal": "*"來比對聯合發起人，並使用 將目標角色限制為您的組織aws:ResourceOrgID。角色的信任政策仍然是可擔任哪些角色的聯合身分的主要控制項。

如需實作網路周邊控制的詳細資訊，請參閱 GitHub 網站上的在 上建置資料周邊 AWS和資料周邊政策範例。

範例：允許特定帳戶的所有 AWS STS 動作

下列端點政策允許特定帳戶中主體的所有 AWS STS 動作。當您的帳戶不屬於組織時，請使用此政策。在不同的陳述式中允許聯合發起人，因為他們在請求內容aws:PrincipalAccount中沒有 。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSpecificAccountPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
            }
        },
        {
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
```

```
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
            }
        }
    ]
}
```

第一個陳述式使用 "Principal": {"AWS": "*"}搭配 aws:PrincipalAccount來允許來自指定帳戶的已驗證 AWS 委託人。第二個陳述式使用 "Principal": "*"來比對聯合發起人，並使用 將目標角色限制為相同的帳戶aws:ResourceAccount。角色的信任政策仍然是可擔任哪些角色的聯合身分的主要控制項。

範例：限制為特定 AWS STS 動作

下列端點政策僅允許已AssumeRole驗證的委託人和AssumeRoleWithSAML聯合發起人，兩者都適用於組織中的角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAssumeRoleByOrgPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
```

```
        {
            "Sid": "AllowSAMLFederation",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "sts:AssumeRoleWithSAML",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

此政策AssumeRoleWithWebIdentity會透過此端點封鎖其他 AWS STS 動作GetCallerIdentity，例如 GetSessionToken、 和 。調整Action元素以符合您的需求。

範例：拒絕非組織存取，同時允許聯合

下列端點政策使用明確拒絕來封鎖組織外部已驗證的主體，同時保留聯合發起人的存取權。此方法從廣泛的允許開始，並新增目標拒絕陳述式。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAll",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
            "Resource": "*"
        },
        {
            "Sid": "DenyNonOrgPrincipals",
            "Effect": "Deny",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
```

```
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

拒絕陳述式使用 `"Principal": {"AWS": "*"}`，其範圍僅限於已驗證的 AWS 主體。聯合發起人 (SAML 和 OIDC) 不是 AWS 主體，且與此`Principal`元素不相符，因此拒絕不適用於它們。此方法可避免需要複雜`Null`或`Bool`條件來切出聯合發起人的例外狀況。

> **Note**
>
> 第一個陳述式允許所有主體的所有動作。第二個陳述式中的拒絕優先於組織外部已驗證的 AWS 主體。第一個陳述式允許聯合發起人，且不受拒絕影響，因為拒絕陳述式中的`Principal`元素與其不相符。

使用 VPC 條件金鑰來控制聯合存取

聯合身分使用者可以使用 `AssumeRoleWithWebIdentity` `AssumeRoleWithSAML`或透過 VPC 端點擔任角色。當它們這樣做時，您可以在角色的信任政策或資源控制政策 (RCPs) 中使用 VPC 特定的條件金鑰，以限制這些請求的來源。除了信任政策中的身分層級控制項之外，還提供網路層級界限。

主題

- VPC 條件金鑰如何用於聯合請求
- 範例：將 OIDC 聯合限制為特定 VPC
- 範例：將 SAML 聯合限制在特定 VPC 端點
- 範例：依 VPC 內的私有 IP 範圍限制 OIDC 聯合
- 範例：將 SAML 聯合限制為多個 VPCs
- 範例：結合 VPC 和提供者特定的條件
- 範例：使用 RCP 限制對特定 VPC 的聯合存取

VPC 條件金鑰如何用於聯合請求

請求內容中可用的條件索引鍵取決於聯合請求的網路路徑。

依網路路徑的 VPC 條件金鑰可用性

條件鍵	透過 VPC 端點	透過公有網際網路	說明
aws:SourceVpc	是	否	請求周遊的 VPC ID
aws:SourceVpcArn	是	否	請求周遊的 VPC ARN
aws:SourceVpce	是	否	請求周遊的 VPC 端點 ID
aws:VpcSourceIp	是	否	VPC 內呼叫者的私有 IP 地址
aws:SourceIp	否	是	發起人的公有 IP 地址

如果您的信任政策在 Allow陳述式aws:SourceVpce中使用 aws:SourceVpc、aws:SourceVpcArn或 ，且請求未透過 VPC 端點，則條件不相符並隱 AWS 含拒絕請求。這實際上需要聯合請求周遊 VPC 端點才能成功。

Important

對於透過 VPC 端點提出的請求， aws:SourceIp 不會填入。請改將 aws:VpcSourceIp用於以 IP 為基礎的限制。

我們建議在信任政策aws:SourceVpc中使用 aws:SourceVpcArn，而不是 。VPC IDs區域上是唯一的，但並非全域唯一，而 aws:SourceVpcArn包含區域和帳戶，可提供 VPC 的全域唯一識別。

Note

下列 OIDC 範例中的對象 (aud) 值使用 sts.amazonaws.com，這是 Amazon EKS 的預設值。對於其他 OIDC 供應商，請以為您的應用程式設定的對象值 （例如用戶端 ID 或應用程式 URI) 取代此值。

範例：將 OIDC 聯合限制為特定 VPC

下列信任政策允許 OIDC 提供者只在請求透過 VPC 端點來自特定 VPC 時擔任角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromVpc",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

AWS 隱含拒絕來自指定 VPC 外部的請求，因為aws:SourceVpcArn條件不相符。不需要明確拒絕陳述式。

範例：將 SAML 聯合限制在特定 VPC 端點

下列信任政策允許 SAML 提供者只在請求通過特定 VPC 端點時擔任角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromVpce",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
```

```
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpce": "vpce-0abcdef1234567890"
                }
            }
        }
    ]
}
```

當您在同一 VPC 中有多個 VPC 端點，並想要限制對特定端點的存取時，使用 aws:SourceVpce而非 aws:SourceVpc可提供更精細的控制。

範例：依 VPC 內的私有 IP 範圍限制 OIDC 聯合

下列信任政策會將聯合限制為來自 VPC 內特定子網路的請求。aws:VpcSourceIp 用於透過 VPC 端點提出請求的 IP 型限制。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromSubnet",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "IpAddress": {
                    "aws:VpcSourceIp": "10.0.1.0/24"
                }
            }
        }
    ]
}
```

> **Important**
>
> 對於透過 VPC 端點的請求，請勿aws:SourceIp搭配私有 IP 範圍使用 。aws:SourceIp 金鑰不適用於 VPC 端點請求。請改用 aws:VpcSourceIp。

範例：將 SAML 聯合限制為多個 VPCs

下列信任政策允許 SAML 提供者從多個 VPCs中的任何一個擔任角色。當您的組織在不同環境或區域中擁有多個 VPCs，而這些 VPC 需要聯合存取相同角色時，此功能非常有用。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromMultipleVpcs",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpcArn": [
                        "arn:aws:ec2:us-west-2:111122223333:vpc/vpc-111aaa11",
                        "arn:aws:ec2:us-east-1:111122223333:vpc/vpc-222bbb22"
                    ]
                }
            }
        }
    ]
}
```

範例：結合 VPC 和提供者特定的條件

您可以結合 VPC 條件金鑰與提供者特定的條件，例如對象 (aud) 和主體 (sub)，以進行精細存取控制。下列信任政策會將角色假設限制在特定聯合身分使用者，並要求請求來自特定 VPC。

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

```
    "Statement": [
        {
            "Sid": "AllowOIDCUserFromVpc",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "idp.example.com:sub": "user123",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

這結合了身分層級控制（僅限特定聯合身分使用者）和網路層級控制（僅限特定 VPC)，可提供深度防禦。

範例：使用 RCP 限制對特定 VPC 的聯合存取

下列資源控制政策 (RCP) 拒絕 AssumeRoleWithSAML 和 AssumeRoleWithWebIdentity請求，除非它們來自特定 VPC。與每個角色套用的信任政策不同，資源控制政策 (RCP) 適用於目標帳戶中的所有角色。這可提供集中式網路界限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyFederatedAccessOutsideVpc",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithWebIdentity",
                "sts:AssumeRoleWithSAML"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
```

```
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "Null": {
                    "aws:SourceVpcArn": "false"
                }
            }
        }
    ]
}
```

Null 條件可確保拒絕僅適用於aws:SourceVpcArn請求內容中存在 時 （即請求是透過 VPC 端點，但來自錯誤的 VPC)。如果沒有它， 也會拒絕透過公有網際網路 （其中 aws:SourceVpcArn 不存在 ） 提出的請求StringNotEquals。如果您移除Null檢查，拒絕也適用於未周遊任何 VPC 端點的請求，包括透過公有網際網路的請求。我們建議您在移除生產環境中的Null檢查之前，先徹底測試。

AWS 使用 IAM 的 服務

下列 AWS 服務會依字母順序分組，並包含其支援哪些 IAM 功能的相關資訊：

- 服務 – 您可以選擇服務的名稱，以檢視有關該服務的 IAM 授權和存取 AWS 的文件。
- 動作 – 您可以在政策中指定個別動作。如果服務不支援此功能，則會選取視覺化編輯器中的 All actions (所有動作)。在 JSON 政策文件中，您必須在 * 元素中使用 Action。如需每個服務的動作清單，請參閱 AWS 服務的動作、資源和條件金鑰。
- 資源層級許可 – 您可以使用 ARN 在政策中指定個別的資源。如果服務不支援此功能，則會選擇政策視覺化編輯器中的 所有資源。在 JSON 政策文件中，您必須在 * 元素中使用 Resource。有些動作，例如 List* 動作，不支援指定 ARN，因為它們是專為傳回多個資源而設計。如果服務因為某些資源但非其他而支援此功能，在表格中會以 Partial 表示它。如需詳細資訊，請參閱該服務的文件。
- 資源型政策 – 您可以將資源型政策連接到服務內的資源。資源型政策包括 Principal 元素以指定哪些 IAM 身分可以存取該資源。如需詳細資訊，請參閱以身分為基礎和以資源為基礎的政策。
- ABAC (依據標籤授權) – 若要依據標籤控制存取，應使用 aws:ResourceTag/*key-name*、aws:RequestTag/*key-name* 或 aws:TagKeys 條件鍵，在政策的條件元素中，提供標籤資訊。如果服務支援每個資源類型的全部三個條件鍵，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件鍵，則值為 Partial。

  如需有關根據屬性 (例如標籤) 定義許可的詳細資訊，請參閱 使用 ABAC 授權根據屬性定義許可。若要查看含有設定 ABAC 步驟的教學課程，請參閱使用屬性型存取控制 (ABAC)。

- 臨時登入資料 – 您可以使用使用 IAM Identity Center、帳戶存取管理員、在 主控台中切換角色，或您在 AWS CLI 或 AWS API AWS STS 中使用 產生的短期登入資料。僅在使用長期 IAM 使用者憑證時，您才可以使用 No 值來存取服務。這包括使用者名稱和密碼或您的使用者存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時安全憑證。
- 服務連結角色 – 服務連結角色是一種特殊類型的服務角色，提供服務代表您存取其他服務中資源的許可。選擇是或部分連結，參閱文件以取得支援這些角色的服務。此欄不會指出服務是否使用標準服務角色。如需詳細資訊，請參閱服務連結角色。
- 詳細資訊 – 如果服務不完全支援某功能，您可以檢閱項目的註腳以查看限制和相關資訊的連結。

可搭配 IAM 運作的服務

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
帳戶存取管理員	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	是
AWS 帳戶管理	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS 動作建議	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Activate Console	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon AI 操作	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Amplify 管理員	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Amplify	✓ 是	✓ 是	✗ 否	⚠ 部分	✓ 是	✗ 否
AWS Amplify UI Builder	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
適用於 Amazon MSK 叢集的 Apache Kafka API	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon API Gateway	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是
Amazon API Gateway Management	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon API Gateway Management V2	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS App Studio	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS App2Container	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS AppConfig	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS AppFabric	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon AppFlow	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon AppIntegrations	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Application Auto Scaling	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 應用程式探索兵工廠	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Application Discovery Service	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
AWS Transform MGN	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Application Recovery Controller - 區域轉移	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS 應用程式轉換服務	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS App Mesh	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS App Mesh 預覽版	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
AWS 應用程式執行器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon AppStream 2.0	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS AppSync	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon ARC 區域切換	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Artifact	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Athena	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Audit Manager	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Aurora DSQL	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Auto Scaling	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
AWS B2B 資料交換	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Backup	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
AWS Backup 閘道	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Backup 搜尋	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Backup 儲存體	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Batch	✓ 是	⚠ 部分	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Bedrock	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Bedrock Agentcore	✓ 是	✓ 是	✗ 否	⚠ 部分	✓ 是	✗ 否
AWS 帳單與成本管理	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 帳單與成本管理儀表板	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 帳單與成本管理 資料匯出	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 帳單與成本管理 定價計算器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS 帳單與成本管理 建議的動作	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Billing Conductor	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Braket	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Budget 服務	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Certificate Manager (ACM)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
聊天應用程式中的 Amazon Q Developer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Chime	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Clean Rooms	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Clean Rooms ML	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Client VPN	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
AWS Cloud9	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
AWS 雲端 控制 API	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon 雲端目錄	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
CloudFormation	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudFront	✓ 是	✓ 是	✗ 否	⚠ 部分	✓ 是	✓ 是
Amazon CloudFront KeyValueStore	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS CloudHSM	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS Cloud Map	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudSearch	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS CloudShell	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS CloudTrail	✓ 是	✓ 是	⚠ 部分 (資訊)	✓ 是	✓ 是	✓ 是
AWS CloudTrail 資料	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudWatch	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	⚠ 部分 (資訊)
Amazon CloudWatch Application Insights	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon CloudWatch Application Signals	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudWatch Internet Monitor	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudWatch Logs	✓ 是	✓ 是	✓ 是	⚠ 部分	✓ 是	✓ 是
Amazon CloudWatch 網絡合成監視器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudWatch Observability Access Manager	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudWatch 可觀測性管理服務	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon CloudWatch RUM	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon CloudWatch Synthetics	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS CodeArtifact	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
AWS CodeBuild	✓ 是	✓ 是	✓是 (資訊)	⚠ 部分 (資訊)	✓ 是	✗ 否
Amazon CodeCatalyst	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS CodeCommit	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS CodeConnections	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS CodeDeploy	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS CodeDeploy 安全主機命令服務	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon CodeGuru Profiler	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon CodeGuru Reviewer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon CodeGuru Security	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS CodePipeline	✓ 是	⚠ 部分	✗ 否	✓ 是	✓ 是	✗ 否
AWS CodeStar	✓ 是	⚠ 部分	✗ 否	✓ 是	✓ 是	✗ 否
AWS CodeStar 連線	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS CodeStar 通知	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon CodeWhisperer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Cognito	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Cognito Sync	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Cognito 使用者集區	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Comprehend	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Comprehend Medical	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Compute Optimizer	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
AWS Config	✓ 是	⚠ 部分 (資訊)	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Connect 客戶	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Connect 客戶案例	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Connect 客戶設定檔	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Connect Customer Outbound Campaigns	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Connect Customer Voice ID	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Connect 運作狀態	✓ 是	✓ 是	✗ 否	⚠ 部分	✓ 是	✗ 否
AWS Console Mobile Application	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Consolidated Billing (合併帳單)	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Control Catalog	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Control Tower	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Cost and Usage Report	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS Cost Explorer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 成本最佳化中心	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 客戶驗證服務	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Database Migration Service	✓ 是	✓ 是	✗ 無 (資訊)	✓ 是	✓ 是	✓ 是
Database Query Metadata Service	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Data Exchange	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Data Lifecycle Manager	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Data Pipeline	✓ 是	✓ 是	✗ 否	⚠ 部分	✓ 是	✗ 否
AWS DataSync	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon DataZone	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 截止日期雲端	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS DeepRacer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Detective	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Device Farm	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon DevOps Guru	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
AWS 診斷工具	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Direct Connect	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Directory Service	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS 目錄服務資料	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon DocumentDB Elastic Clusters	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon DynamoDB Accelerator (DAX)	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
Amazon DynamoDB	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
Amazon Elastic Compute Cloud (Amazon EC2)	✓ 是	⚠ 部分	✗ 否	✓ 是	✓ 是	⚠ 部分 (資訊)
Amazon EC2 Auto Scaling	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
EC2 Image Builder	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon EC2 Instance Connect	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Elastic Beanstalk	✓ 是	⚠ 部分	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Elastic Block Store (Amazon EBS)	✓ 是	⚠ 部分	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Elastic Container Registry (Amazon ECR)	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
Amazon Elastic Container Registry 公有 (Amazon ECR 公有)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Elastic Container Service (Amazon ECS)	✓ 是	⚠ 部分 (資訊)	✗ 否	✓ 是	✓ 是	✓ 是
AWS 彈性災難復原	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Elastic File System (Amazon EFS)	✓ 是	✓ 是	✓ 是	⚠ 部分	✓ 是	✓ 是
Amazon Elastic Kubernetes Service (Amazon EKS)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Elastic Kubernetes Service (Amazon EKS) 授權	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Elastic Load Balancing	✓ 是	⚠ 部分	✗ 否	⚠ 部分	✓ 是	✓ 是
Amazon Elastic VMware Service	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon ElastiCache	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Elemental Appliances and Software Activation Service	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Elemental 設備與軟體	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Elemental MediaConnect	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
AWS Elemental MediaConvert	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Elemental MediaLive	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Elemental MediaPackage	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	⚠ 部分 (資訊)
AWS Elemental MediaPackage V2	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Elemental MediaPackage VOD	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	⚠ 部分 (資訊)
AWS Elemental MediaTailor	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS 元素支援案例	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 元素支援內容	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon EMR	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon EMR on EKS	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon EMR Serverless	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS End User Messaging SMS 和 Voice V2	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 最終使用者傳訊社交	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Entity Resolution	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
Amazon EventBridge	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
Amazon EventBridge Pipes	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon EventBridge 排程器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon EventBridge Schemas	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
AWS 故障注入服務	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon FinSpace	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon FinSpace API	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Firewall Manager	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	⚠ 部分
Fleet Hub for AWS IoT Device Management	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Forecast	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Fraud Detector	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
FreeRTOS	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 免費方案	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon FSx	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon GameLift 伺服器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon GameLift Streams	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Global Accelerator	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Glue	✓ 是	✓ 是	⚠ 部分	⚠ 部分	✓ 是	✗ 否
AWS Glue DataBrew	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Ground Station	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Ground Truth 標記	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon GuardDuty	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Health APIs和通知	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS HealthImaging	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS HealthLake	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS HealthOmics	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS IAM Identity Center	✓ 是	✓ 是	✗ 否	⚠ 部分	✓ 是	✓ 是
IAM Identity Center 目錄	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
IAM Identity Center 身分存放區	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
IAM Identity Center OIDC 服務	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Identity and Access Management (IAM)	✓ 是	✓ 是	⚠ 部分 (資訊)	⚠ 部分 (資訊)	⚠ 部分 (資訊)	✗ 否
AWS Identity and Access Management 和 Access Analyzer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	⚠ 部分
AWS Identity and Access Management Roles Anywhere	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 身分存放區驗證	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 身分同步	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Import/Export	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Inspector	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Inspector Classic	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
Amazon InspectorScan	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Interactive Video Service	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Interactive Video Service Chat	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Invoicing	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS IoT Analytics	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS IoT	✓ 是	✓ 是	⚠ 部分 (資訊)	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS IoT Core Device Advisor	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS IoT Device Tester	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS IoT Events	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS IoT FleetWise	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS IoT Greengrass	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS IoT Greengrass V2	✓ 是	✓ 是	✗ 否	⚠ 部分	✓ 是	✗ 否
AWS IoT Jobs DataPlane	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS IoT 受管整合服務	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS IoT SiteWise	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS IoT TwinMaker	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS IoT Wireless	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Kendra	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Kendra Intelligent Ranking	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Key Management Service (AWS KMS)	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
Amazon Keyspaces (適用於 Apache Cassandra)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Managed Service for Apache Flink	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Managed Service for Apache Flink V2	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Data Firehose	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Kinesis Data Streams	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
Amazon Kinesis Video Streams	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Lake Formation	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
AWS Lambda	✓ 是	✓ 是	✓ 是	⚠ 部分 (資訊)	✓ 是	⚠ 部分 (資訊)
AWS Launch Wizard	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Lex	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Lex V2	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是
AWS License Manager	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS License Manager Linux 訂閱管理員	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS License Manager 使用者訂閱	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
Amazon Lightsail	✓ 是	⚠ 部分 (資訊)	✗ 否	⚠ 部分 (資訊)	✓ 是	✓ 是
Amazon Location Service	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Location Service Maps	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Location Service Places	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Location Service Routes	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Lookout for Equipment	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Lookout for Metrics	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Machine Learning	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Macie	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Mainframe Modernization	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Mainframe Modernization 應用程式測試	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Managed Blockchain	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Managed Blockchain Query	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Managed Grafana	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Managed Service for Prometheus	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Managed Streaming for Apache Kafka (MSK)	✓ 是	✓ 是	⚠ 部分 (資訊)	✓ 是	✓ 是	✓ 是
Amazon Managed Streaming for Kafka Connect	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Managed Workflows for Apache Airflow	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Marketplace	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS Marketplace 目錄	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Marketplace Commerce Analytics	✓ 是	✗ 否	✗ 否	✗ 否	✗ 否	✗ 否
AWS Marketplace 服務部署	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Marketplace 探索	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace Entitlement Service	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace 映像建置服務	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace 管理入口網站	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace Metering Service	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace 私有市集	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace 採購系統整合	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace 報告	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace 賣方報告	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Marketplace 廠商洞見	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon MediaImport	✓ 是	✗ 否	✗ 否	✗ 否	✗ 否	✗ 否
Amazon MemoryDB	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Message Delivery Service	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Message Gateway Service	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Microservice Extractor for .NET	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Migration Acceleration 計劃點數	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Migration Hub	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
AWS Migration Hub 協調器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Migration Hub Refactor Spaces	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
AWS Migration Hub Strategy Recommendations	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
Amazon Monitron	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon MQ	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Neptune	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
多方核准	✓ 是	✓ 是	✗ 否	⚠ 部分 (資訊)	✓ 是	✗ 否
Amazon Neptune Analytics	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Network Firewall	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
網路流量監控	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Network Manager	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是 (資訊)
AWS Network Manager 聊天	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Nimble Studio	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon One Enterprise	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon OpenSearch	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon OpenSearch Ingestion	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon OpenSearch Serverless	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon OpenSearch Service	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
Oracle Database@AWS	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS OpsWorks	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS OpsWorks 組態管理	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Organizations	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是
AWS Outposts	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 平行運算服務	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Partner Central 帳戶管理	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 合作夥伴中央銷售	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Payment Cryptography	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 付款	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Performance Insights (績效詳情)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Personalize	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Pinpoint	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Pinpoint 電子郵件服務	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Pinpoint 簡訊和語音服務	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Polly	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS 價格表	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 私有 5G	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 私有 CA Connector for Active Directory	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 私有 CA Connector for SCEP	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 私有憑證授權單位 (AWS 私有 CA)	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
AWS PrivateLink	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Proton	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 採購訂單主控台	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Q Business	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Q Business Q 應用程式	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Q Developer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Q in Connect	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Quantum Ledger Database (Amazon QLDB)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Quick	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon RDS Data API	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon RDS IAM 身分驗證	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS 資源回收筒	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Redshift	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Redshift 資料 API	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Redshift Serverless	✓ 是	✓ 是	✓ 是 (資訊)	✓ 是	✓ 是	✗ 否
Amazon Rekognition	✓ 是	✓ 是	⚠ 部分 (資訊)	✓ 是	✓ 是	✗ 否
Amazon Relational Database Service (Amazon RDS) (資訊)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS re:Post 私有	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Resilience Hub	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Resource Access Manager (AWS RAM)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 資源總管	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Resource Groups	✓ 是	✓ 是	✗ 否	✓ 是	⚠ 部分 (資訊)	✓ 是

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS Resource Groups Tagging API	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon RHEL 知識庫入口網站	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS RoboMaker	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Route 53	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Route 53 網域	✓ 是	✗ 否	✗ 否	✗ 否	✗ 否	✗ 否
Amazon Route 53 Profiles	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Route 53 Recovery Cluster	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Route 53 Recovery Control Config	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Route 53 Recovery Readiness	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Route 53 Resolver	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS RTB Fabric	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon S3 Express	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
Amazon Glacier	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是	⚠ 部分
Amazon S3 Tables	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是	✗ 否
Amazon S3 Vectors	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon SageMaker AI	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	⚠ 部分 (資訊)

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon SageMaker AI 資料科學助理	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon SageMaker AI 地理空間功能	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon SageMaker AI 與 MLflow	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS Savings Plans	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Secrets Manager	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
AWS Security Hub CSPM	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 安全事件回應	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Security Lake	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
AWS Security Token Service (AWS STS)	✓ 是	⚠ 部分 (資訊)	✗ 否	✓ 是	⚠ 部分 (資訊)	✗ 否
AWS Serverless Application Repository	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是	✗ 否
AWS Service Catalog	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 管理 AWS 主控台使用者體驗功能的服務	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Service Quotas	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Shield	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Shield 網絡安全分析機	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Signer	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS 登入	✓ 是	✗ 否	✓ 是	✗ 否	✓ 是	✗ 否
Amazon SimpleDB	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Simple Email Service - 郵件管理器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon Simple Email Service (Amazon SES) v2	✓ 是	⚠ 部分 (資訊)	✓ 是	✓ 是	⚠ 部分 (資訊)	✓ 是
Amazon Simple Notification Service (Amazon SNS)	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✗ 否
Amazon Simple Queue Service (Amazon SQS)	✓ 是	✓ 是	✓ 是	✓ 部分	✓ 是	✗ 否
Amazon Simple Storage Service (Amazon S3)	✓ 是	✓ 是	✓ 是	⚠ 部分 (資訊)	✓ 是	⚠ 部分 (資訊)
Amazon Simple Storage Service (Amazon S3) 物件 Lambda	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
上的 Amazon Simple Storage Service (Amazon S3) AWS Outposts	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是
Amazon Simple Workflow Service (Amazon SWF)	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS SimSpace Weaver	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Site-to-Site VPN	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✓ 是
AWS Snowball Edge	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Snowball Edge Edge	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
AWS Snowball Edge 裝置管理	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS SQL Workbench	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Step Functions	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Storage Gateway	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Supply Chain	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 支援	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✓ 是
AWS Support App in Slack	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 支援 主控台	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 支援 計劃	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 永續性	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Systems Manager	✓ 是	✓ 是	⚠ 部分	✓ 是	✓ 是	✓ 是
AWS Systems Manager 適用於 SAP 的	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Systems Manager GUI Connect	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Systems Manager Incident Manager	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是	✓ 是
AWS Systems Manager Incident Manager 聯絡案例	✓ 是	✓ 是	✓ 是	✗ 否	✓ 是	✗ 否
AWS Systems Manager 快速設定	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
標籤編輯器	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 稅務設定	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS 電信網路建置器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Textract	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Timestream	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon Timestream Influxdb	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Tiros API （適用於 Reachability Analyzer)	✓ 是	✗ 否	✗ 否	✗ 否	✗ 否	✗ 否
Amazon Transcribe	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Transfer Family	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Transform (變形)	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Translate	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Trusted Advisor	⚠ 部分 (資訊)	✓ 是	✗ 否	✗ 否	⚠ 部分	✓ 是
AWS 使用者通知	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS 使用者通知聯絡人	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS 使用者訂閱	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
AWS Verified Access	✓ 是	✗ 否	✗ 否	✗ 否	✓ 是	✗ 否
Amazon Verified Permissions	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否

服務	動作	資源層級許可	資源型政策	ABAC	暫時性憑證	服務連結角色
Amazon Virtual Private Cloud (Amazon VPC)	✓ 是	⚠ 部分 (資訊)	⚠ 部分 (資訊)	✓ 是	✓ 是	⚠ 部分 (資訊)
Amazon VPC Lattice	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon VPC Lattice Services	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
AWS WAF	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS WAF Classic	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS WAF Regional	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
AWS Well-Architected Tool	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS Wickr	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon WorkMail	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon WorkMail Message Flow	✓ 是	✓ 是	✗ 否	✗ 否	✓ 是	✗ 否
Amazon WorkSpaces	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS WorkSpaces 受管執行個體	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
Amazon WorkSpaces 安全瀏覽器	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✓ 是
Amazon WorkSpaces 精簡型客戶端	✓ 是	✓ 是	✗ 否	✓ 是	✓ 是	✗ 否
AWS X-Ray	✓ 是	⚠ 部分 (資訊)	✗ 否	⚠ 部分 (資訊)	✓ 是	✗ 否

其他資訊

AWS CloudTrail

CloudTrail 支援 CloudTrail Lake 事件資料存放區、儀表板和通道上的資源型政策，用於與 外部的事件來源整合 AWS。

Amazon CloudWatch

無法使用 建立 CloudWatch 服務連結角色 AWS 管理主控台，並僅支援警示動作功能。

AWS CodeBuild

CodeBuild 支援使用 AWS RAM的跨帳戶資源共用。

CodeBuild 針對以專案為基礎的動作支援 ABAC。

AWS Config

AWS Config 支援多帳戶多區域資料彙總和 AWS Config 規則的資源層級許可。如需支援的資源清單，請參閱 AWS Config API 指南的多帳戶多區域資料彙總與 AWS Config 規則章節。

AWS Database Migration Service

您可以建立和修改連接到您建立的 AWS KMS 加密金鑰的政策，以加密遷移到支援目標端點的資料。支援的目標端點包含 Amazon Redshift 和 Amazon S3。如需詳細資訊，請參閱AWS Database Migration Service 《 使用者指南》中的建立和使用 AWS KMS 金鑰來加密 Amazon Redshift 目標資料和建立 AWS KMS 金鑰來加密 Amazon S3 目標物件。

Amazon Elastic Compute Cloud

Amazon EC2 服務連結角色只能用於下列功能：Spot 執行個體請求、Spot 機群請求、Amazon EC2 Fleet 以及快速啟動 Windows 執行個體。

Amazon Elastic Container Service

僅部分 Amazon ECS 動作支援資源層級的許可。

AWS Elemental MediaPackage

MediaPackage 支援服務連結角色，可將客戶存取日誌發佈到 CloudWatch，但不支援其他 API 動作。

AWS Identity and Access Management

僅支援一種稱為角色信任政策，且已連接到 IAM 角色的以資源為基礎的政策。如需詳細資訊，請參閱向使用者授予切換角色的許可。

IAM 支援大多數 IAM 資源的以標籤為基礎的存取控制。如需詳細資訊，請參閱AWS Identity and Access Management 資源的標籤。

只有 IAM 的部分 API 動作可使用暫時憑證來呼叫。如需詳細資訊，請參閱比較 API 選項。

AWS IoT

連線至 的裝置 AWS IoT 會使用 X.509 憑證或使用 Amazon Cognito 身分進行身分驗證。您可以將 AWS IoT 政策連接至 X.509 憑證或 Amazon Cognito Identity，以控制裝置獲授權執行的操作。如需詳細資訊，請參閱AWS IoT 開發人員指南中的 AWS IoT的安全與身分。

AWS Lambda

針對函數、事件來源映射和程式碼簽署組態，Lambda 支援屬性型存取控制 (ABAC)，但不支援層。如需詳細資訊，請參閱 Using attribute-based access control in Lambda。

Lambda 沒有服務連結角色，但 Lambda@Edge 則有。如需詳細資訊，請參閱《Amazon CloudFront 開發人員指南》中的 Lambda@Edge 的服務連結角色。

Amazon Lightsail

Lightsail 部分支援資源層級許可和 ABAC。如需詳細資訊，請參閱適用於 Amazon Lightsail 的動作、資源和條件鍵。

Amazon Managed Streaming for Apache Kafka (MSK)

可以將叢集政策連接至已設定用於多 VPC 連線的 Amazon MSK 叢集。

多方核准

對於多方核准身分來源與核准團隊，可以使用所有三個 ABAC 條件索引鍵：`aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 和 `aws:TagKeys`。

對於工作階段，只能使用 `aws:ResourceTag/key-name` 條件索引鍵，因為：

- 工作階段並非可直接標記的資源。
- 工作階段會從與其相關聯的核准團隊繼承標籤。

AWS Network Manager

AWS Cloud WAN 也支援服務連結角色。如需詳細資訊，請參閱《Amazon VPC AWS AWS Cloud WAN 指南》中的 Cloud WAN 服務連結角色。

Amazon Redshift Serverless

Redshift Serverless 支援資源型政策來共用快照。

Amazon Relational Database Service

Amazon Aurora 為全受管關聯式資料庫引擎，可與 MySQL 和 PostgreSQL 相容。透過 Amazon RDS 設定新的資料庫伺服器時，您可以選擇 Aurora MySQL 或 Aurora PostgreSQL 做為資料庫引擎選項。如需詳細資訊，請參閱《Amazon Aurora 使用者指南》中的 適用於 Amazon Aurora 的 Identity and Access Management。

Amazon Rekognition

僅在複製 Amazon Rekognition 自訂標籤模型時才支援以資源為基礎的政策。

AWS Resource Groups

使用者可以使用允許資源群組操作的政策來擔任角色。

Amazon SageMaker AI

服務連結角色目前可用於 SageMaker AI Studio 和 SageMaker AI 訓練職位。

AWS Security Token Service

AWS STS 沒有「資源」，但允許以類似的方式限制使用者存取。如需詳細資訊，請參閱拒絕依名稱存取暫時安全憑證。

只有部分 API 操作 AWS STS 支援使用臨時登入資料呼叫 。如需詳細資訊，請參閱比較 API 選項。

Amazon Simple Email Service

您可以在參考與傳送電子郵件相關動作的政策陳述式中只使用資源層級的許可，例如 `ses:SendEmail` 或 `ses:SendRawEmail`。對於參考任何其他動作的政策陳述式，資源元素只能包含 *。

只有 Amazon SES API 支援暫時安全憑證。Amazon SES SMTP 界面不支援自臨時安全憑證所衍生的 SMTP 憑證。

Amazon Simple Storage Service

針對目錄儲存貯體、物件資源、Amazon S3 Storage Lens、Storage Lens 群組和存取點，Amazon S3 支援以標籤為基礎的授權。

Amazon S3 支援 Amazon S3 Storage Lens 的服務連結角色。

AWS Trusted Advisor

對 的 API 存取 Trusted Advisor 是透過 支援 API，並由 IAM 支援 政策控制。

Amazon Virtual Private Cloud

Amazon VPC 支援將單一資源政策連接至 VPC 端點，以限制可透過該端點存取的項目。如需有關使用資源型政策控制從特定 Amazon VPC 端點存取資源的詳細資訊，請參閱AWS PrivateLink 指南中的使用端點政策控制對服務的存取。

AWS X-Ray

X-Ray 未針對所有動作支援資源層級的許可。

X-Ray 支援以標籤為基礎的群組和抽樣規則的存取控制。

AWS API 請求的簽章版本 4

> **Important**
>
> 如果您使用 AWS SDK （ 請參閱範例程式碼和程式庫) 或 AWS Command Line Interface (AWS CLI) 工具來傳送 API 請求 AWS，您可以略過簽章程序，因為 SDK 和 CLI 用戶端會使用您提供的存取金鑰來驗證您的請求。除非有充分的理由不這樣做，否則建議您始終使用 SDK 或 CLI。
> 在支援多個簽章版本的區域中，手動簽署要求意味著您必須指定要使用的簽章版本。當您向多區域存取點提供請求時，SDK 和 CLI 會自動切換為使用第 4A 版簽署程序，而不需其他組態。

您在請求中傳送的身分驗證資訊必須包含簽章。 AWS Signature 第 4 版 (SigV4) 是將身分驗證資訊新增至 AWS API 請求的 AWS 簽署通訊協定。

您不會使用私密存取金鑰來簽署請求。相反，您使用 SigV4 簽署程序。簽署請求涉及：

1. 根據請求詳細資訊來建立正式請求。

2. 使用您的 AWS 登入資料計算簽章。
3. 將此簽章作為授權標頭新增至請求。

AWS 然後， 會複寫此程序，並驗證簽章、授予或拒絕相應存取。

對稱 SigV4 要求您在特定日期，在特定 AWS 區域中衍生範圍限定為單一 AWS 服務的金鑰。這使得每個區域的金鑰和計算簽章不同，意味著您必須知道簽章的目的地區域。

非對稱簽章程序第 4 版 (SigV4a) 是支援使用新演算法進行簽署的延伸版，並產生可在多個 AWS 區域中驗證的個別簽章。使用 SigV4a，您可以簽署多個區域的請求，並在區域之間進行無縫路由和容錯移轉。當您使用 AWS SDK 或 AWS CLI 叫用需要多區域簽署的功能時，簽章類型會自動變更為使用 SigV4a。如需詳細資訊，請參閱如何運作 AWS SigV4a。

How AWS SigV4 運作方式

下列步驟說明使用 SigV4 來計算簽章的一般程序：

1. 要簽署的字串取決於請求類型。例如，當您使用 HTTP 授權標頭或查詢參數進行驗證時，請使用請求元素組合來建立要簽署的字串。對於 HTTP POST 請求，請求中的 POST 政策是您簽署的字串。
2. 簽署金鑰是一系列計算，每個步驟的結果會饋送至下一個步驟。最後的步驟是簽署金鑰。
3. 當 AWS 服務收到已驗證的請求時，它會使用請求中包含的身分驗證資訊重新建立簽章。如果簽章相符，則服務會處理請求。否則，它會拒絕請求。

如需詳細資訊，請參閱AWS API 請求簽章的元素。

如何運作 AWS SigV4a

SigV4a 使用基於公有/私有金鑰密碼的非對稱簽章。SigV4a 會經歷與 SigV4 類似的範圍憑證衍生程序，但 SigV4a 使用相同的金鑰來簽署所有請求，而不需要根據日期、服務和區域衍生不同的簽署金鑰。橢圓曲線數位簽章演算法 (ECDSA) 金鑰對可以從您現有的 AWS 私密存取金鑰衍生。

系統使用非對稱密碼編譯來驗證多區域簽章，因此 AWS 僅需要存放您的公有金鑰。公有金鑰不是秘密，無法用於簽署請求。多區域 API 請求需要非對稱簽章，例如使用 Amazon S3 多區域存取點。

下列步驟說明使用 SigV4a 運算簽章的一般程序：

1. 要簽署的字串取決於請求類型。例如，當您使用 HTTP 授權標頭或查詢參數進行驗證時，請使用請求元素組合來建立要簽署的字串。對於 HTTP POST 請求，請求中的 POST 政策是您簽署的字串。

2. 簽署金鑰是透過一系列計算衍生自 AWS 私密存取金鑰，每個步驟的結果會饋送至下一個步驟。最後一個步驟會產生金鑰對。
3. 當 AWS 服務收到使用 SigV4a 簽署的請求時， 只會使用金鑰對的公有半部來 AWS 驗證簽章。如果簽章有效，則會驗證請求，並且服務會處理請求。具有無效簽章的請求會遭到拒絕。

如需有關多區域 API 請求的 SigV4a 詳細資訊，請參閱 GitHub 上的 sigv4a-signing-examples 專案。

簽署請求的時機

當您編寫傳送 API 請求的自訂程式碼時 AWS，必須包含簽署請求的程式碼。您可能會因為以下原因編寫自訂程式碼：

- 您使用的程式設計語言沒有 AWS 軟體開發套件。
- 您需要完全控制請求的傳送方式 AWS。

當 API 請求使用 AWS SigV4 來驗證存取時 AWS CLI ， AWS SDKs會使用您提供的存取金鑰來驗證您的請求。如需使用 AWS SDKs和 驗證 的詳細資訊 AWS CLI，請參閱 其他資源。

為什麼要簽署請求

簽署程序有助於以下列方式保護請求的安全：

- 驗證請求者的身分

  經過驗證的請求需要您使用存取金鑰 (存取金鑰 ID、私密存取金鑰) 建立的簽章。如果您使用暫時安全憑證，則簽章計算還需要安全字符。如需詳細資訊，請參閱 AWS 安全憑證程式設計存取權。
- 保護傳輸中的資料

  為了防止傳送中的請求遭到竄改，有些請求元素可用來計算請求的雜湊 (摘要)，而產生的雜湊值會包含在請求中。當 AWS 服務 收到請求時，它會使用相同的資訊來計算雜湊，並將其與請求中的雜湊值相符。如果值不相符， 會 AWS 拒絕請求。
- 抵禦潛在重播攻擊的侵害

  在大多數情況下，請求必須在請求中時間戳記的五分鐘 AWS 內到達。否則， 會 AWS 拒絕請求。

AWS SigV4 可以在 HTTP 授權標頭中表示，或在 URL 中表示為查詢字串。如需詳細資訊，請參閱身分驗證方法。

其他資源

- 如需不同服務的 SigV4 簽署程序的詳細資訊，請參閱 請求簽章範例。
- 若要設定 CLI AWS 的程式設計存取憑證，請參閱《 AWS 命令列界面使用者指南》中的身分驗證和存取憑證。
- 此 AWS SDKs包含 GitHub 上用於簽署 AWS API 請求的原始碼。如需程式碼範例，請參閱 AWS 範例儲存庫中的範例專案。
  - 適用於 .NET 的 AWS SDK – AWSSigV4Signer.cs
  - 適用於 C++ 的 AWS SDK – AWSAuthV4Signer.cpp
  - 適用於 Go 的 AWS SDK – sigv4.go
  - 適用於 Java 的 AWS SDK – BaseAws4Signer.java
  - 適用於 JavaScript 的 AWS SDK – signature-v4
  - 適用於 PHP 的 AWS SDK – SignatureV4.php
  - 適用於 Python (Boto) 的 AWS SDK – https：//signers.py
  - 適用於 Ruby 的 AWS SDK – signer.rb
- 如果您在簽署請求時遇到錯誤，請參閱 針對 AWS API 請求的 Signature 第 4 版簽署進行故障診斷。

AWS API 請求簽章的元素

> **Important**
>
> 除非您使用 AWS SDKs或 CLI，否則您必須撰寫程式碼來計算在請求中提供身分驗證資訊的簽章。Signature 第 4 版中的 AWS 簽章計算可能是一項複雜的任務，我們建議您盡可能使用 AWS SDKs或 CLI。

凡是使用 Signature Version 4 簽署的每個 HTTP/HTTPS 請求都必須包含下列元素。

元素

- 端點規格
- Action
- 動作參數
- 日期

- 身分驗證資訊

端點規格

指定您要向其傳送請求的端點的 DNS 名稱。此名稱通常包含服務代碼和區域。例如，us-east-1 區域中 Amazon DynamoDB 的端點為 dynamodb.us-east-1.amazonaws.com。

若為 HTTP/1.1 請求，必須包含 Host 標頭。若為 HTTP/2 請求，則可包含 :authority 標頭或 Host 標頭。為符合 HTTP/2 規格，應僅使用 :authority 標頭。並非所有服務都支援 HTTP/2 請求。

如需每個服務支援的端點，請參閱 AWS 一般參考 中的服務端點和配額。

Action

指定服務的 API 動作。例如，DynamoDB CreateTable 動作或 Amazon EC2 DescribeInstances 動作。

如需每個服務支援的動作，請參閱服務授權參考。

動作參數

指定請求中指定的動作參數。每個 AWS API 動作都有一組必要和選用的參數。API 版本通常是必要參數。

如需 API 動作支援的參數，請參閱服務的《API 參考》。

日期

指定請求的日期和時間。請求中附上日期和時間有助於防止第三方攔截您的請求再於稍後重新提交。您在憑證範圍中指定的日期必須與請求日期相符。

此時間戳記必須為 UTC 時間，而且必須使用下列 ISO 8601 格式：YYYYMMDDTHHMMSSZ。例如 20220830T123600Z。不包含時間戳記的毫秒數。

您可以使用 date 標頭或 x-amz-date 標頭，或包含 x-amz-date 作為查詢參數。如果找不到 x-amz-date 標頭，那麼我們會尋找 date 標頭。

身分驗證資訊

您傳送的每個請求必須包含以下資訊。 AWS 會使用此資訊來確保請求的有效性和真實性。

- 演算法 – 簽署程序中所使用的演算法。
  - SigV4 – 使用 AWS4-HMAC-SHA256 來指定採用 HMAC-SHA256 雜湊演算法的第 4 版簽署程序。
  - SigV4a – 使用 AWS4-ECDSA-P256-SHA256 指定 ECDSA-P256-SHA-256 雜湊演算法。
- 憑證 – 透過串連存取金鑰 ID 和憑證範圍元件所形成的字串。
  - SigV4 – 憑證範圍包括存取金鑰 ID、YYYYMMDD 格式的日期、區域代碼、服務代碼和 aws4_request 終止字串，以斜線 (/) 分隔。區域代碼、服務代碼和終止字串必須使用小寫字元。

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
    ```

  - SigV4a – 憑證範圍包括 YYYYMMDD 格式的日期、服務名稱和 aws4_request 終止字串，以斜線 (/) 分隔。請注意，憑證範圍不包括區域，因為區域涵蓋在單獨的標頭 X-Amz-Region-Set 中。

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/service/aws4_request
    ```

- 已簽署的標頭 – 要包含在簽章中的 HTTP 標頭，以分號 (;) 分隔。例如 host;x-amz-date。

  對於 SigV4a，您必須包含區域集標頭，指定請求將在其中生效的區域集。標頭 X-Amz-Region-Set 將指定為逗號分隔值的清單。下列範例展示了區域標頭，其允許在 us-east-1 和 us-west-1 區域中發出請求。

  ```
  X-Amz-Region-Set=us-east-1,us-west-1
  ```

  您可以在區域中使用萬用字元 (*) 來指定多個區域。在下列範例中，標頭允許在 us-west-1 和 us-west-2 中發出請求。

  ```
  X-Amz-Region-Set=us-west-*
  ```

- 簽章 – 表示已計算出之簽章的十六進位編碼字串。您必須使用 Algorithm 參數所指定的演算法計算簽章。

如需詳細資訊，請參閱身分驗證方法

身分驗證方法

> **Important**
>
> 除非您使用 AWS SDKs或 CLI，否則您必須撰寫程式碼來計算在請求中提供身分驗證資訊的簽章。Signature 第 4 版中的 AWS 簽章計算可能是一項複雜的任務，我們建議您盡可能使用 AWS SDKs或 CLI。

您可以使用下列其中一種方法來表示身分驗證資訊。

HTTP 授權標頭

HTTP `Authorization` 標頭是驗證請求的最常見方法。所有 REST API 操作 (使用 `POST` 請求的以瀏覽器為基礎的上傳除外) 需要此標頭。

下列範例展示了 SigV4 和 SigV4a 的 `Authorization` 標頭值。為了便於閱讀，向此範例新增了分行符號。在程式碼中，標頭必須是一個連續字串。演算法與憑證之間沒有逗號，但其他元素必須以逗號分隔。

Example SigV4

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Example SigV4a

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/s3/aws4_request,
SignedHeaders=host;range;x-amz-date;x-amz-region-set,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

下表描述上述範例中授權標頭值的各種元件：

元件	說明
Authorization	用於計算簽章的演算法。

元件	說明
	• SigV4 – 使用 AWS4-HMAC-SHA256 。此字串識別 AWS SigV4 (AWS4) 和HMAC-SHA256 演算法。 • SigV4a – 使用 AWS4-ECDSA-P256-SHA256 。此字串識別 AWS SigV4 (AWS4) 和ECDSA-P256-SHA-256 演算法。

Credential	存取金鑰 ID 和範圍資訊。
	• SigV4 – 包含用於計算簽章的日期、區域和服務。此字串的格式如下：
	`<your-access-key-id>/<date>/<aws-region>/<aws-service>/aws4_request`
	• SigV4a – 包含用於計算簽章的日期和服務。此字串的格式如下：
	`<your-access-key-id>/<date>/<aws-service>/aws4_request`
	使用 YYYYMMDD 格式指定 <date> 值。將請求傳送至 Amazon S3 時 <aws-service> 值為 S3。

SignedHeaders	用於計算簽章的請求標頭清單 (以分號分隔)。此清單僅包含標頭名稱，且標頭名稱必須為小寫。例如：`host;range;x-amz-date` 對於 SigV4a，您必須包含區域集標頭，指定請求將在其中生效的區域集。標頭 X-Amz-Region-Set 將指定為逗號分隔值的清單。

元件	說明
簽章	256 位元簽章以 64 個小寫十六進位字元表示。例如：`fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024` 請注意，簽章計算因您選擇傳輸承載的選項而異。

查詢字串參數

您可以使用查詢字串以完全在 URL 中表達請求。在此情況下，您可以使用查詢參數來提供請求資訊，包括驗證資訊。由於請求簽章是 URL 的一部分，因此此類型 URL 通常稱為預先簽章的 URL。您可以使用預先簽章的 URL 在 HTML 中嵌入可點選連結，有效期最長可達七天。如需詳細資訊，請參閱《Amazon S3 API 參考》中的驗證請求：使用查詢參數AWS（簽章第 4 版）。

下列範例展示了 SigV4 和 SigV4a 預先簽章的 URL。為了便於閱讀，向此範例新增了分行符號：

Example SigV4

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/<region>/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

Example SigV4a

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256 &
X-Amz-Credential=<your-access-key-id>/20240721/s3/aws4_request &
X-amz-Region-Set=<regionset> &
X-Amz-Date=20240721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host;x-amz-region-set &
X-Amz-Signature=<signature-value>
```

> **Note**
>
> URL 中的 X-Amz-Credential 值僅為了便於閱讀而顯示 "/" 字元。實際上，應將其編碼為 %2F。例如：
> &X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request

下表描述 URL 中提供驗證資訊的查詢參數。

查詢字串參數名稱	說明
X-Amz-Algorithm	AWS 簽章的版本，以及您用來計算簽章的演算法。 • SigV4 – 使用 AWS4-HMAC-SHA256 。此字串識別 AWS SigV4 (AWS4) 和HMAC-SHA256 演算法。 • SigV4a – 使用 AWS4-ECDSA-P256-SHA256 。此字串識別 AWS SigV4 (AWS4) 和ECDSA-P256-SHA-256 演算法。
X-Amz-Credential	除了存取金鑰 ID 外，此參數還提供了簽章有效的範圍。此值必須與您在簽章計算中使用的範圍相符，這將在下一節中討論。 • SigV4 – 此參數值的一般格式如下：<your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request 例如：AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request • SigV4a – 此參數值的一般格式如下：<your-access-key-id>/<date>/<AWS-service>/aws4_request

查詢字串參數名稱	說明
	例如：AKIAIOSFODNN7EXAMPLE/20130721/s3/aws4_request SigV4a 的區域在區域集標頭 X-Amz-Region-Set 中定義。 如需 AWS 區域字串的清單，請參閱《AWS 一般參考》中的區域端點。
X-Amz-Region-Set	請求將在其中生效的區域集。標頭 x-amz-region-set 將指定為逗號分隔值的清單。
X-Amz-Date	日期和時間格式必須遵循 ISO 8601 標準，且必須使用 yyyyMMddTHHmmssZ 格式進行格式化。例如，如果日期和時間是 "08/01/2016 15:32:41.982-700"，則必須先將其轉換為 UTC (國際標準時間)，然後以 "20160801T223241Z" 形式提交。
X-Amz-Expires	提供所產生預先簽章的 URL 有效的時段 (以秒為單位)。例如，86400 (24 小時)。此值為整數。您可以設定的最小值為 1，最大值為 604,800 (七天)。預先簽章的 URL 的有效期最長為七天，因為您在簽章計算中使用的簽署金鑰有效期最長為七天。

查詢字串參數名稱	說明
X-Amz-SignedHeaders	列出您用來計算簽章的標頭。簽章計算中需要下列標頭： • HTTP 主機標頭。 • 您計劃新增至請求的任何 x-amz-* 標頭。 • 對於 SigV4a，需要 X-Amz-Region-Set 來指定可在其中發出請求的區域。 為了提高安全性，您應簽署計劃包含在請求中的所有請求標頭。
X-Amz-Signature	提供簽章以驗證您的請求。此簽章必須與服務計算的簽章相符；否則，服務會拒絕請求。例如 `733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7` 簽章計算將在下一節中描述。
X-Amz-Security-Token	選用的憑證參數 (如果使用來自 STS 服務的憑證)。

建立已簽署的 AWS API 請求

> **Important**
>
> 如果您使用 AWS SDK （請參閱範例程式碼和程式庫) 或 AWS Command Line Interface (AWS CLI) 工具來傳送 API 請求 AWS，您可以略過本節，因為 SDK 和 CLI 用戶端會使用您提供的存取金鑰來驗證您的請求。除非有充分的理由不這樣做，否則建議您始終使用 SDK 或 CLI。在支援多個簽章版本的區域中，手動簽署要求意味著您必須指定要使用的簽章版本。當您向多區域存取點提供請求時，SDK 和 CLI 會自動切換為使用第 4A 版簽署程序，而不需其他組態。

您可以使用 AWS SigV4 簽署通訊協定來建立 AWS API 請求的已簽署請求。

1. 根據請求詳細資訊來建立正式請求。
2. 使用您的 AWS 登入資料計算簽章。
3. 將此簽章作為授權標頭新增至請求。

AWS 然後， 會複寫此程序，並驗證簽章、授予或拒絕相應存取。

若要了解如何使用 AWS SigV4 簽署 API 請求，請參閱 請求簽章範例。

下表描述了在建立已簽署請求的過程中所使用的函數。您需要為這些函數實作程式碼。如需詳細資訊，請參閱 AWS SDK 中的程式碼範例。

函式	說明
`Lowercase()`	將字串轉換成小寫。
`Hex()`	小寫基數 16 編碼。
`SHA256Hash()`	安全雜湊演算法 (SHA) 加密雜湊函數。
`HMAC-SHA256()`	使用 SHA256 演算法搭配提供的簽署金鑰來計算 HMAC。這是使用 SigV4 簽署時的最終簽章。
`ECDSA-Sign`	橢圓曲線數位簽章演算法 (ECDSA) 簽章，使用以公有-私有金鑰密碼編譯為基礎的非對稱簽章進行計算。
`KDF(K, Label, Context, L)`	使用 PRF 函數 HMAC-SHA256 的計數器模式下的 NIST SP800-108 KDF，如 NIST SP 800-108r1 中所定義。
`Oct2Int(byte[ ])`	八位元轉整數函數，如 ANSI X9.62 中所述。
`Trim()`	移除任何前置和結尾空格。
`UriEncode()`	URI 編碼每個位元組。UriEncode() 必須強制執行下列規則： • URI 對除未預留字元之外的每個位元組進行編碼：'A'-'Z', 'a'-'z', '0'-'9', '-', '.', '_', and '~'。

函式	說明
	• 空格字元是預留字元，必須編碼為 "%20" (而不是 "+")。 • 每個 URI 編碼的位元組都是由 '%' 和位元組的兩位數十六進位值組成。 • 十六進位值中的字母必須為大寫，例如 "%1A"。 • 對所有位置的正斜線字元 '/' 進行編碼 (物件索引鍵名稱除外)。例如，如果物件索引鍵名稱為 `photos/Jan/sample.jpg` ，則不會對索引鍵名稱中的正斜線進行編碼。 **Important** 由於實作的差異和基礎 RFCs 中的相關模棱兩可性，開發平台提供的標準 UriEncode 函數可能無法運作。我們建議您編寫自己的自訂 UriEncode 函數，以確保您的編碼有效。 若要查看 Java 中的 UriEncode 函數範例，請參閱 GitHub 網站上的 Java 公用程式。

Note

簽署請求時，您可以使用 AWS SigV4 或 AWS SigV4a。兩者之間的主要差異取決於簽章的計算方式。對於 SigV4a，區域集包含在要簽署的字串中，但不是憑證衍生步驟的一部分。

具有臨時安全憑證的簽署請求

您可以使用由 AWS Security Token Service () 提供的臨時安全登入資料，而不是使用長期登入資料來簽署請求AWS STS。

當您使用臨時安全憑證時，必須將 X-Amz-Security-Token 新增至授權標頭或將它包含在查詢字串中，以保留工作階段權杖。某些服務會要求您將 X-Amz-Security-Token 新增至正式請求。其他服務只需要您在計算簽章之後於結尾新增 X-Amz-Security-Token。檢查每個 的文件 AWS 服務 ，了解特定需求。

簽署步驟摘要

建立正式請求

將您的請求內容 (主機、動作、標頭等) 編排為標準正式格式。正式請求就是其中一個用於建立登入字串的輸入。如需有關建立正式請求的詳細資訊，請參閱 AWS API 請求簽章的元素。

建立正式請求的雜湊

使用與您用來建立承載雜湊相同的演算法，對正式請求進行雜湊。正式請求的雜湊是小寫十六進位字元字串。

建立要簽署的字串

使用正式請求和額外資訊，例如演算法、請求日期、憑證範圍及正式請求的雜湊，建立登入字串。

衍生簽署金鑰

使用私密存取金鑰衍生用於簽署請求的金鑰。

計算簽章

使用衍生的簽署金鑰作為雜湊金鑰，對要簽署的字串執行金鑰式雜湊操作。

將簽章新增至請求

將計算簽章新增至 HTTP 標頭，或新增至請求的查詢字串。

建立正式請求

若要建立正式請求，請串連以下字串 (以換行符號字元分隔)。這有助於確保您計算的簽章可以符合 AWS 計算的簽章。

```
<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
```

```
<HashedPayload>
```

- *HTTPMethod* – HTTP 方法，例如 GET、PUT、HEAD 和 DELETE。
- *CanonicalUri*– 絕對路徑元件 URI 的 URI 編碼版本，以 / 開頭，後接網域名稱，一直到字串結尾，或者如果具有查詢字串參數，則一直到問號字元 (?)。如果絕對路徑空白，請使用斜線字元 (/)。下列範例中的 URI /amzn-s3-demo-bucket/myphoto.jpg 是絕對路徑，您不會在絕對路徑中編碼 / ：

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket/myphoto.jpg
  ```

- *CanonicalQueryString* – URI 編碼的查詢字串參數。您可以單獨對每個名稱和值進行 URI 編碼。您還必須依索引鍵名稱的字母順序來排序正式查詢字串中的參數。編碼後進行排序。下列 URI 範例中的查詢字串為：

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket?prefix=somePrefix&marker=someMarker&max-keys=2
  ```

  正式查詢字串如下所示 (為了便於閱讀，向此範例新增了分行符號)：

  ```
  UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
  UriEncode("max-keys")+"="+UriEncode("20") + "&" +
  UriEncode("prefix")+"="+UriEncode("somePrefix")
  ```

  請求以子資源為目標時，對應的查詢參數值將是空字串 ("")。例如，下列 URI 會識別 amzn-s3-demo-bucket 儲存貯體上的 ACL 子資源：

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket?acl
  ```

  在這種情況下，CanonicalQueryString 為：

  ```
  UriEncode("acl") + "=" + ""
  ```

  如果 URI 不包含 ?，則請求中沒有查詢字串，並且您將正式查詢字串設定為空字串 ("")。您仍然需要包含換行符號字元 ("\n")。

- *CanonicalHeaders* – 請求標頭清單及其值。個別標頭名稱和值對以換行符號字元 ("\n") 分隔。以下是 CanonicalHeader 的範例：

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

CanonicalHeaders 清單必須包含以下內容：

- HTTP host 標頭。
- 如果 Content-Type 標頭存在於請求中，您必須將其新增至 *CanonicalHeaders* 清單。
- 還必須新增您計劃包含在請求中的任何 x-amz-* 標頭。例如，如果您使用的是暫時安全憑證，需要將 x-amz-security-token 包含在您的請求中。您必須將此標頭新增在 *CanonicalHeader* 的清單中。
- 對於 SigV4a，您必須包含區域集標頭，指定請求將在其中生效的區域集。標頭 X-Amz-Region-Set 將指定為逗號分隔值的清單。下列範例展示了區域標頭，其允許在 us-east-1 和 us-west-1 區域中發出請求。

  ```
  X-Amz-Region-Set=us-east-1,us-west-1
  ```

  您可以在區域中使用萬用字元 (*) 來指定多個區域。在下列範例中，標頭允許在 us-west-1 和 us-west-2 中發出請求。

  ```
  X-Amz-Region-Set=us-west-*
  ```

> **Note**
>
> Amazon S3 AWS 請求需要 x-amz-content-sha256 標頭。其提供請求承載的雜湊。如果沒有承載，您必須提供空字串的雜湊。

每個標頭名稱必須：

- 使用小寫字元。
- 依字母順序顯示。
- 後接冒號 (:)。

對於值，您必須：

- 修剪任何前置或結尾空格。
- 將連續空格轉換為單一空格。

- 使用逗號分隔多值標頭的值。
- 簽章中必須包含主機標頭 (HTTP/1.1) 或 :authority 標頭 (HTTP/2)，以及簽章中的任何 x-amz-* 標頭。可以選擇性地在簽章中包含其他標準標頭，例如 content-type。

本範例中使用的 Lowercase() 和 Trim() 函數在上一節中進行了描述。

以下是 CanonicalHeaders 字串範例。標頭名稱為小寫且已排序。

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> 對於計算授權簽章而言，僅需要主機和任何 x-amz-* 標頭；但是，為了防止資料竄改，應考慮在簽章計算中包含其他標頭。
> 請勿包含跨複雜系統傳輸期間經常變更的逐個躍點標頭。這包括所有易變的傳輸標頭，這些標頭可能會由代理、負載平衡器和分散式系統中的節點變更，包括 connection、x-amzn-trace-id、user-agent、keep-alive、transfer-encoding、TE、trailer、upgrade、proxy-authorization 和 proxy-authenticate。

- *SignedHeaders* – 依字母順序排序、以分號分隔的小寫請求標頭名稱清單。清單中的請求標頭與您在 CanonicalHeaders 字串中包含的標頭相同。對於上一個範例，*SignedHeaders* 值將如下所示：

```
host;x-amz-content-sha256;x-amz-date
```

- *HashedPayload* – 使用 HTTP 請求內文中的承載作為雜湊函數的輸入所建立的字串。此字串使用小寫十六進位字元。

```
Hex(SHA256Hash(<payload>))
```

如果請求中沒有承載，您可以計算空字串的雜湊，例如當您使用 GET 請求擷取物件時，承載中沒有任何內容。

```
Hex(SHA256Hash(""))
```

> **Note**
>
> 對於 Amazon S3，請在建構正式請求時包含文字字串 UNSIGNED-PAYLOAD，並在傳送請求時設定與 x-amz-content-sha256 標頭值相同的值。
> Hex(SHA256Hash("UNSIGNED-PAYLOAD"))

建立正式請求的雜湊

使用與您用來建立承載雜湊相同的演算法，建立正式請求的雜湊 (摘要)。正式請求的雜湊是小寫十六進位字元字串。

建立要簽署的字串

若要建立要簽署的字串，請串連以下字串 (以換行符號字元分隔)。請勿以換行符號字元結束此字串。

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *Algorithm* - 用來建立正式請求雜湊的演算法。
  - SigV4 – 使用 AWS4-HMAC-SHA256 來指定 HMAC-SHA256 雜湊演算法。
  - SigV4a – 使用 AWS4-ECDSA-P256-SHA256 指定 ECDSA-P256-SHA-256 雜湊演算法。
- *RequestDateTime* - 憑證範圍中使用的日期和時間。此值是 ISO 8601 格式的目前 UTC 時間 (例如，20130524T000000Z)。
- *CredentialScope* – 憑證範圍會將產生的簽章限制到指定的區域和服務。
  - SigV4 – 憑證包括存取金鑰 ID、YYYYMMDD 格式的日期、區域代碼、服務代碼和 aws4_request 終止字串，以斜線 (/) 分隔。區域代碼、服務代碼和終止字串必須使用小寫字元。此字串採用下列格式：YYYYMMDD/region/service/aws4_request。
  - SigV4a – 憑證包括 YYYYMMDD 格式的日期、服務名稱和 aws4_request 終止字串，以斜線 (/) 分隔。請注意，憑證範圍不包括區域，因為區域涵蓋在單獨的標頭 X-Amz-Region-Set 中。此字串採用下列格式：YYYYMMDD/service/aws4_request。
- *HashedCanonicalRequest* – 在上一步驟中計算的正式請求的雜湊。

以下是登入字串範例。

```
"<Algorithm>" + "\n" +
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(<Algorithm>(<CanonicalRequest>))
```

衍生簽署金鑰

若要衍生簽署金鑰，請選擇下列其中一個程序來計算 SigV4 或 SigV4a 的簽署金鑰。

衍生 SigV4 的簽署金鑰

若要衍生 SigV4 的簽署金鑰，請在請求日期、區域和服務上執行連續的金鑰雜湊操作 (HMAC)，並將 AWS 私密存取金鑰作為初始雜湊操作的金鑰。

對於每個步驟，使用所需金鑰和資料呼叫雜湊函數。對雜湊函數的每次呼叫結果都會變成雜湊函數的下一次呼叫輸入。

下列範例顯示如何衍生此程序下一節中使用的 SigningKey，顯示您的輸入串連和雜湊的順序。HMAC-SHA256 是用於雜湊資料的雜湊函數，如下所示。

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

需要輸入

- Key – 包含私密存取金鑰的字串。
- Date – 包含憑證範圍中所使用日期的字串，格式為 YYYYMMDD。
- Region – 包含區域代碼的字串 (例如 us-east-1)。

  如需區域字串清單，請參閱《AWS 一般參考》中的區域端點。
- Service – 包含服務代碼的字串 (例如 ec2)。
- 在之前步驟中建立的登入字串。

衍生 SigV4 的簽署金鑰

1. 串連 "AWS4" 和私密存取金鑰。呼叫雜湊函數，將串連的字串作為索引鍵，日期字串做為資料。

```
DateKey = hash("AWS4" + Key, Date)
```

2. 呼叫雜湊函數，將上一次呼叫的結果作為索引鍵，區域字串作為資料。

```
DateRegionKey = hash(kDate, Region)
```

3. 呼叫雜湊函數，將上一次呼叫的結果作為索引鍵，服務字串作為資料。

   服務代碼由服務定義。可以使用 AWS Pricing CLI 中的 get-products 來傳回服務的服務代碼。

```
DateRegionServiceKey = hash(kRegion, Service)
```

4. 呼叫雜湊函數，將上一次呼叫的結果作為索引鍵，"aws4_request" 作為資料。

```
SigningKey = hash(kService, "aws4_request")
```

衍生 SigV4a 的簽署金鑰

若要建立 SigV4a 的簽署金鑰，請使用下列程序從私密存取金鑰衍生金鑰對。如需此衍生的實作範例，請參閱AWS 用戶端身分驗證的 C99 程式庫實作

```
n = [NIST P-256 elliptic curve group order]
G = [NIST P-256 elliptic curve base point]
label = "AWS4-ECDSA-P256-SHA256"

akid = [AWS access key ID as a UTF8 string]
sk = [AWS secret access Key as a UTF8 Base64 string]

input_key = "AWS4A" || sk
count = 1
while (counter != 255) {
  context = akid || counter // note: counter is one byte
  key = KDF(input_key, label, context, 256)
  c = Oct2Int(key)
  if (c > n - 2) {
    counter++
  } else {
    k = c + 1   // private key
    Q = k * G   // public key
  }
}
```

```
if (c < 255) {
  return [k, Q]
} else {
  return FAILURE
}
```

計算簽章

在您衍生簽署金鑰之後，請計算要新增至請求的簽章。此程序會根據您使用的簽章版本而有所不同。

計算 SigV4 的簽章

1. 呼叫雜湊函數，將上一次呼叫的結果作為索引鍵，待簽署字串作為資料。使用衍生的金鑰做為此操作的雜湊金鑰。結果是作為二進位值的簽章。

   ```
   signature = hash(SigningKey, string-to-sign)
   ```

2. 將簽章的表示形式從二進位轉換為十六進位，使用小寫字元。

計算 SigV4a 的簽章

1. 使用數位簽署演算法 (ECDSA P-256)，簽署上一步驟中建立的待簽署字串。用於此簽章的金鑰是衍生自上述私密存取金鑰的私有非對稱金鑰。

   ```
   signature = base16(ECDSA-Sign(k, string-to-sign))
   ```

2. 將簽章的表示形式從二進位轉換為十六進位，使用小寫字元。

將簽章新增至請求

將計算簽章新增至您的請求。

Example範例：授權標頭

SigV4

下列範例顯示使用 AWS SigV4 之`DescribeInstances`動作的 `Authorization`標頭。為了便於閱讀，此範例使用換行符號進行格式化。在代碼中，這必須是一個連續字串。演算法和 `Credential` 之間沒有逗號。但是，必須使用逗號分隔其他元素。

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

SigV4a

下列範例顯示使用 AWS SigV4a `CreateBucket`之動作的授權標頭。為了便於閱讀，此範例使用換行符號進行格式化。在代碼中，這必須是一個連續字串。演算法和憑證之間沒有逗號。但是，必須使用逗號分隔其他元素。

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request,
SignedHeaders=host;x-amz-date;x-amz-region-set,
Signature=calculated-signature
```

Example範例：查詢字串中包含身分驗證參數的請求

SigV4

下列範例顯示使用 AWS SigV4 `DescribeInstances`之動作的查詢，其中包含身分驗證資訊。為了便於閱讀，此範例使用換行符號進行格式化，而不是 URL 編碼。在程式碼中，查詢字串必須是 URL 編碼的連續字串。

```
https://ec2.amazonaws.com/?
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

SigV4a

下列範例展示了針對 `CreateBucket` 動作使用 AWS SigV4a 的查詢，其中包含身分驗證資訊。為了便於閱讀，此範例使用換行符號進行格式化，而不是 URL 編碼。在程式碼中，查詢字串必須是 URL 編碼的連續字串。

```
https://ec2.amazonaws.com/?
```

```
Action=CreateBucket&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request&
X-Amz-Region-Set=us-west-1&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date;x-amz-region-set&
X-Amz-Signature=calculated-signature
```

請求簽章範例

下列 AWS 簽署請求範例示範如何使用 SigV4 簽署在沒有 AWS SDK 或 AWS 命令列工具的情況下傳送的請求。

使用 HTTP POST 的基於瀏覽器的 Amazon S3 上傳

驗證請求：基於瀏覽器的上傳描述簽章和相關資訊，Amazon S3 在收到請求時會使用其來計算簽章。

範例：使用 HTTP POST 的瀏覽器型上傳 （使用 AWS Signature 第 4 版） 提供更多資訊，其中包含範例 POST 政策和可用來上傳檔案的表單。範例政策和虛擬憑證會顯示工作流程以及產生的簽章和政策雜湊。

VPC Lattice 驗證的請求

第 4 版簽署程序 (SigV4) 驗證的請求範例提供了 Python 和 Java 範例，顯示如何在使用和不使用自訂攔截器的情況下執行請求簽署。

搭配使用第 4 版簽署程序與 Amazon Translate

Metaverse 中的即時轉譯示範如何建置應用程式，以產生近乎即時的翻譯解決方案。此speech-to-speech翻譯程式解決方案在事件串流編碼中使用 AWS SigV4 來產生即時轉錄。

搭配使用第 4 版簽署程序與 Neptune

範例：搭配使用 Python 與第 4 版簽署程序來連接到 Neptune 說明如何使用 Python 向 Neptune 發出已簽署的請求。此範例包括使用存取金鑰或暫時憑證的變化情況。

簽署對 Amazon Glacier 的 HTTP 請求

串流 API 的簽章計算範例將逐步說明為上傳封存 (POST 封存) 建立簽章的詳細資訊，這是 Amazon Glacier 中的兩個串流 API 之一。

向 Amazon SWF 發出 HTTP 請求

向 Amazon SWF 發出 HTTP 請求顯示向 Amazon SWF 發出 JSON 請求的標頭內容。

Amazon OpenSearch Service 中的串流 API 的簽章計算

使用適用於 PHP 的 AWS SDK 第 3 版簽署 Amazon OpenSearch Service 搜尋請求包括如何將已簽署 HTTP 請求傳送至 Amazon OpenSearch Service 的範例。

AWS 範例儲存庫中的範例專案

下列範例專案示範如何簽署請求，以對 Python、Node.js、Java、C#、Go 和 Rust 等常用語言 AWS 的服務提出 Rest API 請求。

4a 版簽署程序專案

sigv4-signing-examples 專案提供如何使用 SigV4 簽署請求的範例，以 AWS 服務 使用 Python、Node.js、Java、C#、Go 和 Rust 等常用語言向 發出 Rest API 請求。

sigv4a-signing-examples 專案提供簽署多區域 API 請求的範例，例如 Amazon S3 中的多區域存取點。

發佈至 AWS IoT Core

AWS IoT Core 使用 HTTPs 通訊協定發佈至 的 Python 程式碼提供如何使用 AWS IoT Core HTTPS 通訊協定和 AWS SigV4 身分驗證將訊息發佈至 的指引。兩種參考實作分別採用 Python 和 NodeJs 語言。

AWS IoT Core 使用 HTTPs 通訊協定發佈至 的 .Net Framework 應用程式提供如何使用 HTTPS 通訊協定和 AWS SigV4 身分驗證將訊息發佈至 AWS IoT Core 的指引。此專案還包括 .NET 核心同等實作。

針對 AWS API 請求的 Signature 第 4 版簽署進行故障診斷

> **Important**
>
> 除非您使用 AWS SDKs或 CLI，否則您必須撰寫程式碼來計算在請求中提供身分驗證資訊的簽章。SigV4 簽章計算可能是一項複雜的任務，我們建議您盡可能使用 AWS SDKs或 CLI。

當您開發建立已簽署請求的程式碼時，您可能會`SignatureDoesNotMatch`從中收到 HTTP 403 AWS 服務。這些錯誤表示 HTTP 請求中的簽章值與 AWS 服務 計算的簽章 AWS 不相符。當許可不允許呼叫者發出請求時，將傳回 HTTP 401 `Unauthorized` 錯誤。

如有以下情形，API 請求可能傳回錯誤：

- API 請求未簽署，且 API 請求使用 IAM 身分驗證。
- 用於簽署請求的 IAM 憑證不正確或沒有調用 API 的許可。
- 已簽署 API 要求的簽章與 AWS 服務計算的簽章不符。
- API 請求標頭不正確。

> **Note**
>
> 在探索其他錯誤解決方案之前，將您的簽署通訊協定從 AWS Signature 第 2 版 (SigV2) 更新為 AWS Signature 第 4 版 (SigV4)。服務 (例如 Amazon S3) 和區域不再支援 SigV2 簽署。

可能原因

- 憑證錯誤
- 正式請求與簽署字串錯誤
- 憑證範圍錯誤
- 金鑰簽署錯誤

憑證錯誤

確保使用 SigV4 簽署 API 請求。如果 API 請求未簽署，則您有可能收到錯誤：`Missing Authentication Token`。新增缺少的簽章並重新傳送請求。

驗證存取金鑰和私密金鑰的身分驗證憑證是正確的。如果存取金鑰錯誤，則您有可能收到錯誤：`Unauthorized`。確保用於簽署請求的實體有權發出請求。如需詳細資訊，請參閱對拒絕存取錯誤訊息進行疑難排解。

正式請求與簽署字串錯誤

如果您錯誤計算 建立正式請求的雜湊 或 建立要簽署的字串 中正式請求，則該服務執行的簽章驗證步驟便會失敗，並顯示錯誤訊息：

```
The request signature we calculated does not match the signature you provided
```

當 AWS 服務收到已簽署的請求時，它會重新計算簽章。如果值之間有差異，則簽名不匹配。使用錯誤訊息中的值，比對正式請求和字串以及您的已簽署請求。若有任何差異，請修改簽署程序。

> **Note**
>
> 您也可以驗證您未透過可修改標頭或請求的代理傳送請求。

Example正式請求範例

```
GET                                                    -------- HTTP method
/                                                      -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
                                                       -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                          -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com    -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                           -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501       -------- Hash
 of the payload
```

要驗證私密金鑰是否與存取金鑰 ID 相符，可以使用已知有效的實作來測試它們。例如，使用 AWS SDK 或 AWS CLI 向 提出請求 AWS。

API 請求標頭

當授權標頭為空、憑證金鑰或簽章遺失或不正確、標頭不是以演算法名稱開頭，或金鑰值對不包含等號時，您會收到下列其中一個錯誤：

- 授權標頭不能為空白。
- 授權標頭需要 'Credential' 參數。
- 授權標頭需要 'Signature' 參數。
- 簽章在授權標頭中包含無效的 key=value 對 (缺少等號)。

確保您在 計算簽章 中新增的 SigV4 授權標頭包含正確的憑證金鑰，且還包含使用 HTTP Date 或 x-amz-date 標頭的請求日期。

如果您收到 IncompleteSignatureException 錯誤且簽章的建構正確，您可以透過計算用戶端請求中授權標頭的 SHA-256 雜湊和 B64 編碼 AWS 服務 ，來驗證授權標頭未在傳輸到 時遭到修改。

1. 取得您在請求中傳送的授權標頭。授權標頭看起來類似下列範例：

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=example-generated-signature
```

2. 計算授權標頭的 SHA-256 雜湊。

```
hashSHA256(rawAuthorizationHeader) = hashedAuthorizationHeader
```

3. 將雜湊授權標頭編碼為 Base64 格式。

```
base64(hashedAuthorizationHeader) = encodedHashedAuthorizationHeader
```

4. 將剛計算的雜湊和編碼字串與錯誤訊息中收到的字串進行比較。錯誤訊息應類似於如下範例：

```
com.amazon.coral.service#IncompleteSignatureException:
The signature contains an in-valid key=value pair (missing equal-sign)
in Authorization header (hashed with SHA-256 and encoded with Base64):
'9c574f83b4b950926da4a99c2b43418b3db8d97d571b5e18dd0e4f3c3ed1ed2c'.
```

- 如果兩個雜湊不同，授權標頭的某些部分會在傳輸中變更。此變更可能是由於您的網路或用戶端處理常式連接了已簽章的標頭，或以某種方式變更授權標頭。
- 如果兩個雜湊相符，您在請求中傳送的授權標頭會符合 AWS 收到的內容。檢閱收到的錯誤訊息，以判斷問題是否是因為憑證或簽章不正確所致。此頁面的其他區段會涵蓋這些錯誤。

憑證範圍錯誤

您在 建立要簽署的字串 中建立的憑證範圍會將簽章限制在特定日期、區域和服務。此字串的格式如下所示：

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> 如果您使用的是 SigV4a，區域不包含在憑證範圍之中。

日期

如果憑證範圍未指定與 x-amz-date 標頭相同的日期，則簽名驗證步驟會失敗，並顯示下列錯誤訊息：

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

如果請求指定未來的時間，則簽章驗證步驟會失敗，並顯示下列錯誤訊息：

```
Signature not yet current: date is still later than date
```

如果請求已過期，簽章驗證步驟會失敗，並顯示下列錯誤訊息：

```
Signature expired: date is now earlier than date
```

區域

如果憑證範圍未指定與請求相同的區域，則簽名驗證步驟會失敗，並顯示下列錯誤訊息：

```
Credential should be scoped to a valid Region, not region-code
```

服務

如果憑證範圍未指定與 host 標頭相同的服務，則簽名驗證步驟會失敗，並顯示下列錯誤訊息：

```
Credential should be scoped to correct service: 'service'
```

終止字串

如果憑證範圍沒有以 aws4_request 結尾，則簽名驗證步驟會失敗，並顯示下列錯誤訊息：

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

金鑰簽署錯誤

簽署金鑰衍生不正確或不當使用密碼編譯而造成的錯誤，較難進行故障診斷。確認正式字串和登入字串正確無誤之後，您也可以檢查下列其中一個問題：

- 私密存取金鑰與您指定的存取金鑰 ID 不符。
- 您的金鑰衍生程式碼發生問題。

要驗證私密金鑰是否與存取金鑰 ID 相符，可以使用已知有效的實作來測試它們。例如，使用 AWS SDK 或 AWS CLI 向 提出請求 AWS。如需取得範例，請參閱「請求簽章範例」。

IAM JSON 政策參考

本節介紹 IAM 中 JSON 政策的元素、變數和評估邏輯的詳細語法、描述和範例。如需一般詳細資訊，請參閱JSON 政策概觀。

本參考包括下列章節。

- IAM JSON 政策元素參考 — 進一步了解有關在建立政策時可以使用的元素。查看其他政策範例和了解條件，支援的資料類型以及它們在各種服務中的使用方式。
- 政策評估邏輯 — 本節說明 AWS 請求的身分驗證方式，以及 AWS 如何使用政策來判斷對 資源的存取。
- IAM JSON 政策語言的文法 — 本節介紹用來在 IAM 中建立政策的語言的正式語法。
- AWS 任務函數的 受管政策 — 本節列出了直接對應到 IT 產業中常見任務功能的所有 AWS 受管政策。使用這些政策來授予執行特定任務角色中的某人所預期的任務所需的許可權。這些政策將許多服務的許可合整合到單一政策中。
- AWS 全域條件內容索引鍵 — 本節包含可用於限制 IAM 政策中許可的所有 AWS 全域條件金鑰清單。
- IAM 和 AWS STS 條件內容索引鍵 — 本節包含所有 IAM 和 AWS STS 條件索引鍵的清單，可用於限制 IAM 政策中的許可。
- AWS 服務的動作、資源和條件金鑰 — 本節提供您可以在 IAM 政策中用作許可的所有 AWS API 操作的清單。它還包括可用來進一步強化請求的服務專用的條件索引鍵。

IAM JSON 政策元素參考

JSON 政策文件由元素組成。這些元素按照在政策中使用的大致順序列出。元素的順序不會有影響，例如，`Resource` 元素可排在 `Action` 元素之前。您不需要指定政策中的任何 `Condition` 元素。如需進一步了解 JSON 政策文件的一般結構和目的，請參閱 JSON 政策概觀。

部分 JSON 政策元素會相互排斥。這表示您不能建立同時使用兩個元素的政策。例如，您不能在同一政策陳述式中同時使用 `Action` 與 `NotAction`。相互排斥的其他組合包括 `Principal/NotPrincipal` 以及 `Resource/NotResource`。

政策中所包含的具體內容因服務而異，取決於服務所提供的動作、內含的資源類型等。當您針對特定服務編寫政策時，參考該服務的範例將會有所幫助。若需要支援 IAM 之所有服務的清單，以及探討這些服務的 IAM 和政策的文件連結，請參閱 AWS 使用 IAM 的 服務。

當您建立或編輯 JSON 政策時，IAM 可以執行政策驗證以協助您建立有效的政策。IAM 會識別 JSON 語法錯誤，而 IAM Access Analyzer 會提供額外的政策檢查及建議，協助您進一步改良政策。若要進一步了解政策驗證的資訊，請參閱 IAM 政策驗證。若要進一步了解 IAM Access Analyzer 政策檢查和可動作的建議，請參閱 IAM Access Analyzer 政策驗證。

主題

- IAM JSON 政策元素：Version
- IAM JSON 政策元素：Id
- IAM JSON 政策元素：Statement
- IAM JSON 政策元素：Sid
- IAM JSON 政策元素：Effect
- AWS JSON 政策元素： Principal
- AWS JSON 政策元素： NotPrincipal
- IAM JSON 政策元素：Action
- IAM JSON 政策元素：NotAction
- IAM JSON 政策元素：Resource
- IAM JSON 政策元素：NotResource
- IAM JSON 政策元素：Condition
- IAM 政策元素：變數與標籤
- IAM JSON 政策元素：支援的資料類型

IAM JSON 政策元素：Version

> **消除歧義注意事項**
>
> 此 Version JSON 政策元素與「政策版本」不同。Version 政策元素是在政策內使用，並定義政策語言的版本。另一方面，政策版本會在您在 IAM 中變更客戶受管政策時建立。變更的政策不會覆寫現有的政策。IAM 反而會建立新版本的受管政策。如果您在搜尋有關受管政策可用的多版本支援的詳細資訊，請參閱the section called “版本控制 IAM 政策”。

Version 政策元素指定用於處理政策的語言語法規則。若要使用所有可用的政策功能，請在所有政策的 Statement 元素之外包含下列 Version 元素。

JSON

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

IAM 支援下列 Version 元素值：

- 2012-10-17. 這是政策語言的目前版本，您應該一律包含 Version 元素，並將其設定為 2012-10-17。否則，您無法使用此版本所引進的功能 (例如政策變數)。
- 2008-10-17. 這是舊版本的政策語言。您可以在較舊的現有政策上看到此版本。對於任何新的政策，或您更新任何現有政策時，請不要使用此版本。較新的功能 (例如政策變數) 將無法使用您的政策。例如，${aws:username} 這類變數無法辨識為變數，而是視為政策中的文字字串。

IAM JSON 政策元素：Id

Id 元素指定政策的選用識別符。該 ID 在不同的服務中的使用各有不同。以資源為基礎的政策中允許使用 ID，但是以身分為基礎的政策不允許使用 ID。

對於可讓您設定 ID 元素的服務，我們建議您對該值使用 UUID (GUID)，或者將 UUID 做為 ID 的一部分合併以確保唯一性。

JSON

```
{
  "Version":"2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 某些 AWS 服務 （例如 Amazon SQS 或 Amazon SNS) 可能需要此元素，並對其具有唯一性要求。如需有關編寫政策的服務特定資訊，請參閱您正在使用的服務文件。

IAM JSON 政策元素：Statement

`Statement` 元素是政策的主要元素。此元素為必要。`Statement` 元素可包含單一陳述式，或是個別陳述式的陣列。每個個別的陳述式區塊都必須用大括號 { } 括起。針對多個陳述式，陣列必須用方括號 [ ] 括起。

```
"Statement": [{...},{...},{...}]
```

以下範例顯示政策，其中包含在單一 `Statement` 元素中的一系列三個陳述式 (政策可讓您存取 Amazon S3 主控台中您自己的「主資料夾」)。政策包含 `aws:username` 變數，這會在政策評估期間以請求的使用者名稱取代。如需詳細資訊，請參閱簡介。

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

IAM JSON 政策元素：Sid

您可以提供 Sid (陳述式 ID) 作為政策陳述式的可選識別符。您可以將 Sid 值指派給陳述式陣列中的每個陳述式。您可以使用 Sid 值做為政策陳述式的描述。在允許您指定 ID 元素的服務 (例如 SQS 和 SNS) 中，Sid 值只是政策文件 ID 的子 ID。在 IAM 中，Sid 值在 JSON 政策中必須是唯一的。

JSON

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

```
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

Sid 元素支援 ASCII 大寫字母 (A-Z)、小寫字母 (A-Z) 和數字 (0-9)。

IAM 未在 IAM API 中公開 Sid。您無法根據此 ID 擷取特定陳述式。

> **Note**
>
> 某些 AWS 服務 （例如 Amazon SQS 或 Amazon SNS) 可能需要此元素，並對其具有唯一性要求。如需有關編寫政策的服務特定資訊，請參閱您正在使用的服務文件。

IAM JSON 政策元素：Effect

Effect 元素是必要的，且指定陳述式會導致允許或明確拒絕。Effect 的有效值為 Allow 和 Deny。Effect 值會區分大小寫。

```
"Effect":"Allow"
```

在預設情況下，拒絕存取資源。若要允許存取資源，您必須將 Effect 元素設定為 Allow。若要覆寫允許 (例如，以強制方式覆寫允許)，您必須將 Effect 元素設定為 Deny。如需詳細資訊，請參閱政策評估邏輯。

AWS JSON 政策元素： Principal

使用資源型 JSON 政策中的 Principal 元素來指定允許或拒絕存取資源的主體。

您可以在資源型政策中使用 Principal 元素。許多服務支援資源型政策，包括 IAM。IAM 資源型政策類型是角色信任政策。在 IAM 角色中，使用角色信任政策中的 Principal 元素來指定誰可擔任該角色。對於跨帳戶存取，您必須指定信任帳戶的 12 位數識別碼。若要了解在您信任區域 (受信任組織或帳戶) 外帳戶中的主體是否具有擔任您角色的許可，請參閱什麼是 IAM Access Analyzer？。

> **Note**
>
> 在您建立角色後，您可以變更帳戶為「*」，以允許每個人擔任該角色。若執行此操作，我們強烈建議您限制誰可以透過其他方式存取角色，例如 Condition 元素會限制只能存取特定 IP 地址。切勿讓任何人都能存取您的角色！

支援資源型政策的其他資源範例包括 Amazon S3 儲存貯體或 AWS KMS key。

您不能使用身分型政策中的 Principal 元素。身分型政策是指您連接到 IAM 身分 (使用者、群組或角色) 的許可政策。在這些情況中，主體由政策連接的身分隱含識別。

主題

- 如何指定主體
- AWS 帳戶 主體
- IAM 角色主體
- 角色工作階段主體
- OIDC 聯合身分主體
- SAML 聯合身分主體
- IAM 使用者主體
- IAM Identity Center 主體
- AWS STS 聯合身分使用者主體
- AWS 服務主體
- AWS 選擇加入區域中的服務主體
- 所有主體
- 其他資訊

如何指定主體

您可以在資源型政策的 Principal 元素中指定主體，或在支援主體的條件金鑰中指定。

您可以在政策中指定以下任何主體：

- AWS 帳戶 和根使用者
- IAM 角色

- 角色工作階段
- IAM 使用者
- 聯合身分使用者主體
- AWS 服務
- 所有主體

您無法將使用者群組識別為政策 (例如資源型政策) 中的主體，因為群組與許可 (而非驗證) 相關，並且主體是經過驗證的 IAM 實體。

您可以使用陣列在以下幾節中為每個主體類型指定一個以上的主體類型。陣列可以使用一個或多個值。當您在元素中指定多個主體時，便會授予每個主體許可。這是邏輯 OR，而不是邏輯 AND，因為您一次只會驗證為一位主體。如果包含多個值，請使用方括號 ([和])，並以逗號分隔陣列的每個項目。下列範例政策會定義 123456789012 帳戶或 555555555555 帳戶的許可。

```
"Principal" : {
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> 您不能使用萬用字元來符合部分主體名稱或 ARN。

AWS 帳戶 主體

您可以在資源型政策的 `Principal`元素或支援主體的條件索引鍵中指定 AWS 帳戶 識別符。這會委派帳戶的授權。當您允許存取其他帳戶時，該帳戶中的管理員必須接著授予該帳戶中身分 (IAM 使用者或角色) 的存取權。當您指定 時 AWS 帳戶，您可以使用帳戶 ARN (arn：aws：iam：：*account-ID*：root)，或由帳戶 ID 後面加上 `"AWS":`字首的縮短表單。

例如，指定帳戶 ID `123456789012`，您就可以使用以下其中一種方法來指定 `Principal` 元素中的帳戶：

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

帳戶 ARN 和簡化帳戶 ID 的行為方式相同。兩者都會委派帳戶的許可。使用 Principal 元素中的帳戶 ARN 不會將許可限制為帳戶的根使用者。

> **Note**
>
> 當您儲存包含簡化帳戶 ID 的資源型政策時，服務可能會將其轉換為主體 ARN。如此並不會變更政策的功能。

有些 AWS 服務支援指定帳戶委託人的其他選項。例如，Amazon S3 可讓您使用以下格式指定正式使用者 ID：

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

您也可以使用 陣列 AWS 帳戶，將多個 （或正式使用者 ID) 指定為委託人。例如，您可以使用全部三種方法在儲存貯體政策中指定主體。

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
}
```

IAM 角色主體

您可以在資源型政策的 Principal 元素中指定 IAM 角色主體 ARN，或在支援主體的條件金鑰中指定。IAM 角色是身分。在 IAM 中，身分是您可以指派許可的資源。角色信任另一個已驗證的身分來擔任該角色。這包括 AWS 中的主體或來自外部身分提供者 (IdP) 的使用者。當主體或身分擔任角色時，他們會收到具有擔任角色許可的暫時安全憑證。當他們使用這些工作階段登入資料在 中執行操作時 AWS，就會成為角色工作階段主體。

當您在資源型政策中指定角色主體時，主體的有效許可會受限於限制角色許可的任何政策類型。其中包括工作階段政策和許可界限。如需有關如何評估角色工作階段的有效許可的詳細資訊，請參閱 政策評估邏輯。

若要在 Principal 元素中指定角色 ARN，請使用以下格式：

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> 如果角色信任政策中 Principal 元素包含指向特定 IAM 角色的 ARN，則該 ARN 會在儲存政策時轉換為角色的唯一主體 ID。如果有人希望藉由刪除並重新建立角色來提升特權，這麼做可有助於減輕此類風險。您通常不會在主控台中看到此 ID，因為在顯示信任政策時，IAM 會反向轉換回角色 ARN。
> 不過，如果您刪除角色，則關係會中斷。即使重新建立角色，您的政策都不再適用，因為新角色有不符合信任政策中所儲存的主體 ID 的新主體 ID。發生這種情況時，委託人 ID 會出現在以資源為基礎的政策中，因為 AWS 無法再將其對應回有效的 ARN。
> 結果是，如果您刪除並重新建立信任政策Principal元素中參考的角色，則必須編輯政策中的角色，以將主體 ID 取代為正確的 ARN。當您儲存政策時，ARN 再次轉換為角色的新主體 ID。如需詳細資訊，請參閱了解 AWS處理政策中已刪除的 IAM 角色。

或者，您可以指定角色主體作為資源型政策中的主體，或使用 aws:PrincipalArn 條件金鑰建立廣泛許可政策。當您使用此索引鍵時，角色工作階段主體會根據擔任的角色 ARN 授予許可，而不是根據所產生之工作階段的 ARN 授予許可。由於 AWS 不會將條件金鑰 ARNs 轉換為 IDs，因此如果您刪除角色，然後使用相同名稱建立新角色，則授予角色 ARN 的許可會保留。在資源型政策的 Principal 元素中，使用 aws:PrincipalArn 條件金鑰及萬用字元 (*) 授予的許可，不受身分型政策類型 (例如許可界限或工作階段政策) 的限制，除非身分型政策包含明確拒絕。

角色工作階段主體

您可以在資源型政策的 Principal 元素中指定角色工作階段，或在支援主體的條件金鑰中指定。當主體或身分擔任角色時，他們會收到具有擔任的角色許可的暫時安全憑證。當他們使用這些工作階段登入資料在 中執行操作時 AWS，就會成為角色工作階段主體。

您用於角色工作階段主體的格式取決於用來擔任角色 AWS STS 的操作。

> **Important**
>
> AWS 建議您盡可能在政策中使用 IAM 角色主體，而不是角色工作階段主體。使用Condition陳述式和條件索引鍵，在必要時進一步縮小存取範圍。

若要在 Principal元素中指定角色工作階段主體 ARN，請使用下列格式：

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

此外，系統管理員可以設計程序來控制如何發出角色工作階段。例如，他們可以為使用者提供一鍵式解決方案，以建立可預測的工作階段名稱。如果您的系統管理員執行這項操作，則可以在政策或條件金鑰中使用角色工作階段主體。否則，您可以在 aws:PrincipalArn 條件金鑰中將角色 ARN 指定為主體。將角色指定為主體的方式可以變更產生工作階段的有效許可。如需詳細資訊，請參閱IAM 角色主體。

OIDC 聯合身分主體

OIDC 聯合委託人是使用 JSON Web 字符 (JWT) 從 OIDC 相容 IDP 呼叫 AWS STS AssumeRoleWithWebIdentity API 時所使用的委託人，也稱為 OpenID 提供者 (OP)，以請求臨時 AWS 憑證。OIDC 聯合委託人可以代表您 AWS 帳戶中的 OIDC IDP，或 4 個內建身分提供者：Login with Amazon、Facebook、 Google和 Amazon Cognito。

從其 OIDC IDP 發出 JWT 的使用者、工作負載或系統可以使用 AssumeRoleWithWebIdentity JWT 呼叫 ，為設定為信任發出 JWT 的 OIDC IDP 的 IAM 角色請求臨時 AWS 安全登入資料。只要符合 AWS STS列出的需求，JWT 可以是 ID 權杖、存取權杖或透過任何其他方法提供的 JWT 權杖。如需詳細資訊，請參閱常見案例和透過 OIDC 提供者請求憑證。

在您的角色信任政策中使用此委託人類型，以允許或拒絕AssumeRoleWIthWebIdentity使用 中存在的 OIDC IDP AWS 帳戶或四個內建 IDPs許可。若要在角色信任政策的 Principal 元素中指定 OIDC 聯合身分主體 ARN，請針對內建的 OIDC IDP，使用下列四種格式之一：

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

使用新增至帳戶的 OIDC 提供者 (例如 GitHub) 時，需要在角色的信任政策中指定提供者的 ARN。此組態可讓您撰寫 IAM 政策，專門控制透過自訂身分提供者進行驗證之使用者的存取權。

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/full-OIDC-identity-provider-URL" }
```

例如，如果 GitHub 是受信任的 Web 身分提供者，則角色信任政策的 Principal 元素中的 OIDC 角色工作階段 ARN 會使用下列格式：

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/tokens.actions.githubusercontent.com" }
```

如需詳細資訊，請參閱 Configuring OpenID Connect in Amazon Web Services。

角色信任政策以外的政策類型不支援 OIDC 聯合身分主體。

SAML 聯合身分主體

SAML 聯合委託人是呼叫 AWS STS AssumeRoleWithSAML API 以使用 SAML 聲明請求臨時 AWS 登入資料的委託人。您可以使用 SAML 身分提供者 (IdP) 登入，然後使用此操作擔任 IAM 角色。與 類似AssumeRoleWithWebIdentity， AssumeRoleWithSAML 不需要 AWS 登入資料即可進行身分驗證。反之，使用者會先驗證其 SAML 身分提供者，然後使用其 SAML 聲明進行 AssumeRoleWithSAML API 呼叫，或重新導向至 AWS 登入/SAML 頁面以登入 AWS 管理主控台。如需哪些主體可以使用此操作擔任角色的詳細資訊，請參閱 比較 AWS STS 登入資料。

在角色信任政策中使用此主體類型，以根據受信任的 SAML 身分提供者允許或拒絕許可。若要在角色信任政策的 Principal 元素中指定 SAML 身分角色工作階段 ARN，請使用下列格式：

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

IAM 使用者主體

您可以在資源型政策的 Principal 元素中指定 IAM 使用者，或在支援主體的條件金鑰中指定。

> **Note**
>
> 在 Principal 元素中，Amazon Resource Name (ARN) 的使用者名稱部分區分大小寫。

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

當您在 Principal 元素中指定使用者時，您無法使用萬用字元 (*) 來表示「所有使用者」。主體必須一律指定特定的使用者。

> **Important**
>
> 如果角色信任政策中您的 Principal 元素包含指向特定 IAM 使用者的 ARN，則該 ARN 會在儲存政策時將 ARN 轉換為使用者的唯一主體 ID。如果有人希望藉由刪除並重新建立使用者來提升特權，這麼做可有助於減輕此類風險。您通常不會在主控台中看到此 ID，因為在顯示信任政策時還會反向轉換回使用者的 ARN。
> 不過，如果您刪除使用者，則關係會中斷。即使重新建立使用者，政策都不再適用。這是因為新的使用者有新的主體 ID，其不符合儲存在信任政策中的 ID。發生這種情況時，委託人 ID 會出現在以資源為基礎的政策中，因為 AWS 無法再將其對應回有效的 ARN。
> 結果是，如果您刪除並重新建立了信任政策的 Principal 元素所引用的使用者，您必須編輯角色來以正確 ARN 替換不正確的主體 ID。當您儲存政策時，IAM 再次將 ARN 轉換為使用者的新主體 ID。

IAM Identity Center 主體

在 IAM Identity Center 中，資源型政策中的主體必須定義為 AWS 帳戶 主體。若要指定存取權，請參照條件區塊中許可集合的角色 ARN。如需詳細資訊，請參閱《IAM Identity Center 使用者指南》中的資源政策、Amazon EKS 和 AWS KMS中的參考許可集合。

AWS STS 聯合身分使用者主體

您可以在資源型政策的 Principal 元素中指定聯合身分使用者工作階段，或在支援主體的條件金鑰中指定。

> **Important**
>
> AWS 建議您限制使用 AWS STS 聯合身分使用者工作階段。改為使用 IAM 角色。

聯合身分使用者主體是透過使用長期 AWS STS IAM 憑證呼叫 GetFederationToken的操作建立。聯合身分使用者許可是呼叫 GetFederationToken 的主體與作為參數傳遞至 GetFederationToken API 的工作階段政策之間的交集。

在 中 AWS，IAM 使用者或 AWS 帳戶根使用者 可以使用長期存取金鑰進行身分驗證。如需哪些主體可以使用此操作聯合身分的詳細資訊，請參閱 比較 AWS STS 登入資料。

- IAM 聯合身分使用者 – IAM 使用者使用 GetFederationToken 操作進行聯合，該操作會產生此 IAM 使用者的聯合身分使用者工作階段。
- 聯合身分根使用者 – 根使用者使用 GetFederationToken 操作進行聯合，該操作會產生此根使用者的聯合身分使用者工作階段。

當 IAM 使用者或根使用者請求 AWS STS 使用此操作的臨時登入資料時，他們會開始臨時聯合身分使用者工作階段。此工作階段的 ARN 是以聯合的原始身分為基礎。

若要在 Principal 元素中指定聯合身分使用者工作階段 ARN，請使用以下格式：

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

AWS 服務主體

您可以在資源型政策的 Principal元素或支援主體的條件金鑰中指定 AWS 服務。服務主體是服務的識別碼。

服務可擔任的 IAM 角色 AWS 稱為服務角色。服務角色必須包含信任政策。信任政策是連接到角色的資源型政策，這些政策定義可擔任該角色的主體。有些服務角色具有已預先定義的信任政策。不過，在某些情況下，您必須在信任政策中指定服務主體。IAM 政策中的服務主體不能為 "Service": "*"。

> **Important**
>
> 服務主體的識別碼包含服務名稱，而且通常採用以下格式：
> *service-name*.amazonaws.com

服務主體是由服務定義。您可以藉由開啟 AWS 使用 IAM 的 服務，檢查服務連結角色資料欄中是否為是，並開啟是連結，以檢視該服務的服務連結角色文件，從而尋找某些服務的服務主體。尋找該服務的 Service-Linked Role Permissions (服務連結角色許可) 區段，以檢視服務主體。

以下範例顯示一個可以連接到服務角色的政策。政策啟用兩種服務，分別為 Amazon ECS 和 Elastic Load Balancing，來擔任角色。然後服務可以執行指派給角色的許可政策所授予的任何任務 (不顯示)。若要指定多個服務主體，不用指定兩個 `Service` 元素；您可以只使用一個該元素。實際上，您可以將一組多個服務主體作為單一 `Service` 元素的值。

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

AWS 選擇加入區域中的服務主體

您可以在數個 AWS 區域中啟動資源，以及您必須選擇加入的部分區域。如需您必須選擇加入的區域完整清單，請參閱 AWS 一般參考指南中的管理 AWS 區域。

當選擇加入區域中 AWS 的服務在相同區域內提出請求時，服務主體名稱格式會識別為其服務主體名稱的非區域化版本：

service-name.amazonaws.com

當選擇加入區域中 AWS 的服務向另一個區域提出跨區域請求時，服務主體名稱格式會識別為其服務主體名稱的區域化版本：

service-name.*{region}*.amazonaws.com

例如，您有一個位於區域 `ap-southeast-1` 中的 Amazon SNS 主題，而 Amazon S3 儲存貯體位於選擇加入區域 `ap-east-1`。您想要設定 S3 儲存貯體通知，將訊息發佈到 SNS 主題。若要允許 S3 服務將訊息發佈至 SNS 主題，您必須透過主題的資源型存取政策授予 S3 服務主體`sns:Publish`許可。

如果您指定 S3 服務主體的非區域化版本 `s3.amazonaws.com`，則在存取政策主題中，從儲存貯體到主題的 `sns:Publish` 要求將會失敗。下列範例會在 SNS 主題存取政策的 `Principal` 政策元素中指定非區域化 S3 服務主體。

```
"Principal": { "Service": "s3.amazonaws.com" }
```

由於儲存貯體位於選擇加入的區域，而且要求是在同一個區域之外發出，因此 S3 服務主體會顯示為區域化服務主體名稱 `s3.ap-east-1.amazonaws.com`。當選擇加入區域中的服務向另一個區域發出請

求時，您必須使用區域化 AWS 服務主體名稱。指定區域化服務主體名稱後，如果儲存貯體向位於另一個區域的 SNS 主題發出 sns:Publish 要求，則該要求將會成功。下列範例會在 SNS 主題存取政策的 Principal 政策元素中指定區域化 S3 服務主體。

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

只有在您指定區域化服務主體名稱時，從選擇加入區域至另一個區域的跨區域要求的資源政策或服務主體型允許清單才會成功。

> **Note**
>
> 對於 IAM 角色信任政策，建議使用非區域化服務主體名稱。IAM 資源是全球性的，因此可以在任何區域中使用相同的角色。

所有主體

可使用萬用字元 (*) 在資源型政策的 Principal 元素中或支援主體的條件金鑰中指定所有主體。資源型政策 授予許可和條件金鑰用於限制政策陳述式的條件。

> **Important**
>
> 強烈建議您不要在資源型政策的 Principal 中搭配使用萬用字元 (*) 與 Allow 效果，除非您打算授予公開或匿名存取。否則，請指定預定的主體、服務或 Principal 元素中的 AWS 帳戶，然後進一步限制 Condition 元素中的存取。IAM 角色信任政策尤其如此，因為它們允許其他主體成為您帳戶中的主體。

對於資源型政策，搭配使用萬用字元 (*) 與 Allow 效果可授權存取所有使用者，包括匿名使用者 (公開存取)。對於您帳戶中的 IAM 使用者和角色主體，不需要其他許可。對於其他帳戶中的主體，他們也必須在其帳戶中具有身分型許可，以允許他們存取您的資源。這稱為跨帳戶存取。

對於匿名使用者，下列元素相同：

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

您不能使用萬用字元來比對部分主體名稱或 ARN。

以下範例顯示可用於明確拒絕所有主體的資源型政策 (而不是 AWS JSON 政策元素： NotPrincipal)，但在 Condition 元素中指定的主體除外。此政策應新增至 Amazon S3 儲存貯體。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

其他資訊

如需詳細資訊，請參閱下列內容：

- Amazon Simple Storage Service 使用者指南中的儲存貯體政策範例
- Amazon Simple Notification Service 開發人員指南中的 Amazon SNS 政策範例
- Amazon Simple Queue Service 開發人員指南中的 Amazon SQS 政策範例
- AWS Key Management Service 開發人員指南中的索引鍵政策
- AWS 一般參考 中的帳戶識別符
- OIDC 聯合身分

AWS JSON 政策元素： NotPrincipal

NotPrincipal 元素使用 "Effect":"Deny" 來拒絕存取所有主體，但 NotPrincipal 元素中指定的主體除外。委託人可以是 IAM 使用者、 AWS STS 聯合身分使用者委託人、IAM 角色、擔任的角色工作階段 AWS 帳戶、 AWS 服務或其他委託人類型。如需有關主體的資訊，請參閱 AWS JSON 政策元素： Principal。

NotPrincipal 必須與 "Effect":"Deny" 搭配使用。不支援將其與 "Effect":"Allow" 搭配使用。

> **Important**
>
> 不建議將 NotPrincipal 用於新的資源型政策，作為安全和授權策略的一部分。使用 NotPrincipal 時，會難以疑難排解多個政策類型的影響。我們建議改為將 aws:PrincipalArn 內容索引鍵與 ARN 條件運算子搭配使用。

重點

- 某些 AWS 服務 (包括 VPC 端點) 的資源型政策支援 NotPrincipal 元素。資源型政策是您直接內嵌在資源中的政策。您無法在以 IAM 身分為基礎的政策或 IAM 角色信任政策中使用 NotPrincipal 元素。
- 對於已附加許可界限政策的 IAM 使用者或角色，請勿使用所含 NotPrincipal 政策元素具有 Deny 效果的資源型政策陳述式。無論在 NotPrincipal 元素中指定何值，具有 Deny 效果的 NotPrincipal 元素將始終拒絕任何已附加許可界限政策的 IAM 主體。這會造成部分本可存取資源的 IAM 使用者或角色失去存取權。我們建議您變更資源型政策陳述式，將條件運算子 ArnNotEquals 與 aws:PrincipalArn 內容索引鍵搭配使用來限制存取，而不是使用 NotPrincipal 元素。如需有關許可界限的資訊，請參閱 IAM 實體的許可界限。
- 使用 NotPrincipal 時，還必須指定未遭拒之主體的帳戶 ARN。否則，政策可能會拒絕存取包含主體的整個帳戶。根據您包含在政策中的服務， AWS 可能先驗證帳戶，再來才是使用者。如果正在評估擔任角色的使用者 （使用角色的使用者）， AWS 則 可能會先驗證帳戶，然後驗證角色，然後驗證擔任角色使用者。擔任角色的使用者由該使用者擔任角色時指定的角色工作階段名稱來進行識別。因此，我們強烈建議您明確包含使用者帳戶的 ARN，或同時包含角色 ARN 和包含該角色的帳戶的 ARN。
- 服務控制政策 (SCP) 和資源控制政策 (RCP) 不支援 NotPrincipal 元素。

`NotPrincipal` 元素的替代方案

在 中管理存取控制時 AWS，在某些情況下，您可能需要明確拒絕所有委託人存取資源，但您指定的一或多個委託人除外。 AWS 建議使用具有全域條件內容索引鍵的拒絕陳述式，以便更精確地控制和更輕鬆地進行故障診斷。下列範例展示了使用條件運算子 (例如 `StringNotEquals` 或 `ArnNotEquals`) 來拒絕存取所有主體 (Condition 元素中指定的主體除外) 的替代方法。

使用 IAM 角色的案例範例

您可以將資源型政策與拒絕陳述式搭配使用，來防止所有 IAM 角色 (Condition 元素中指定的角色除外) 存取或操作資源。此方法遵循 AWS 安全原則，明確拒絕一律優先於任何允許陳述式，並有助於維護整個 AWS 基礎設施的最低權限原則。

建議使用帶有全域條件內容索引鍵和條件運算子 (例如 ArnNotEquals) 的拒絕陳述式，而不是使用 `NotPrincipal`，來明確允許 IAM 角色存取資源。下列範例使用了 aws:PrincipalArn 來明確允許角色 `read-only-role` 存取 `Bucket_Account_Audit` 資料夾中的 Amazon S3 儲存貯體。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyCrossAuditAccess",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::Bucket_Account_Audit",
        "arn:aws:s3:::Bucket_Account_Audit/*"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:role/read-only-role"
        }
      }
    }
  ]
}
```

使用服務主體的案例範例

您可以使用拒絕陳述式來防止所有服務主體 (Condition 元素中指定的主體除外) 存取或操作資源。當您需要實作精細存取控制或在 AWS 環境中的不同服務和應用程式之間建立安全邊界時，此方法特別有用。

建議使用帶有全域條件內容索引鍵和條件運算子 (例如 StringNotEquals) 的拒絕陳述式，而不是使用 NotPrincipal，來明確允許服務主體存取資源。下列範例使用了 aws:PrincipalServiceName 來明確允許 AWS CodeBuild 服務主體存取 BUCKETNAME 資料夾中的 Amazon S3 儲存貯體。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyNotCodeBuildAccess",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME",
        "arn:aws:s3:::BUCKETNAME/*"
      ],
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:PrincipalServiceName": "codebuild.amazonaws.com"
        }
      }
    }
  ]
}
```

IAM JSON 政策元素：Action

Action 元素描述了將允許或拒絕的特定動作。陳述式必須包含 Action 或 NotAction 元素。每個 AWS 服務都有自己的一組動作，描述您可以使用該服務執行的任務。例如，Amazon S3 的動作清單可以在 Amazon Simple Storage Service 使用者指南中的在政策中指定許可中找到，Amazon EC2 的動

作清單可以在 Amazon EC2 API 參考中找到，而 的動作清單 AWS Identity and Access Management 可以在 IAM API 參考中找到。若要尋找其他服務的動作清單，請參閱該服務的 API 參考文件。

AWS 也提供 JSON 格式的服務參考資訊，以簡化政策管理工作流程的自動化。透過 服務參考資訊，您可以從 AWS 服務 機器可讀取的檔案跨 存取可用的動作、資源和條件索引鍵。如需詳細資訊，請參閱 Service Authorization Reference 中的 Simplified AWS 服務 information for programmatic access。

您可以使用服務命名空間做為動作字首 (iam、ec2、sqs、sns、s3 等) 來指定值，其後跟隨要允許或拒絕的動作名稱。該名稱必須符合所支援的服務的動作。字首和動作名稱不區分大小寫。例如，iam:ListAccessKeys 與 IAM:listaccesskeys 相同。以下範例顯示不同服務的 Action 元素。

Amazon SQS 動作

```
"Action": "sqs:SendMessage"
```

Amazon EC2 動作

```
"Action": "ec2:StartInstances"
```

IAM 動作

```
"Action": "iam:ChangePassword"
```

Amazon S3 動作

```
"Action": "s3:GetObject"
```

您可以為 Action 元素指定多個值。

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

您可以使用多字元比對萬用字元 (*) 和單一字元比對萬用字元 (?)，讓 存取特定 AWS 產品提供的所有動作。例如，以下 Action 元素適用於所有 S3 動作。

```
"Action": "s3:*"
```

您也可以將萬用字元 (* 或 ?) 用作動作名稱的一部分。例如，以下 Action 元素適用於包含字串 AccessKey 的所有 IAM 動作，包括 CreateAccessKey、DeleteAccessKey、ListAccessKeys 和 UpdateAccessKey。

```
"Action": "iam:*AccessKey*"
```

有些服務可讓您限制可用的動作。例如，Amazon SQS 可讓您僅提供所有可能的 Amazon SQS 動作的子集。在這種情況下，* 萬用字元不允許完全控制佇列；它只允許您已共用的部分動作。如需詳細資訊，請參閱《Amazon Simple Queue Service 開發人員指南》中的了解許可。

IAM JSON 政策元素：NotAction

NotAction 是進階政策元素，明確符合「除了」指定的動作清單外的所有內容。使用 NotAction 可以透過僅列出一些不相符的動作來產生較短的政策，而不是包括相符的長動作清單。在 NotAction 中指定的動作不受政策陳述式中的 Allow 或 Deny 效果的影響。反之，這表示如果使用 Allow 效果，則允許未列出的所有適用動作或服務。此外，如果您使用 Deny 效果，則會拒絕此類未列出的動作或服務。將 NotAction 與 Resource 元素一起使用時，可以為政策提供範圍。這是 AWS 決定哪些動作或服務適用的方式。如需詳細資訊，請參閱下列範例政策。

帶允許的 NotAction

您可以使用 陳述式中的 NotAction元素"Effect": "Allow"來提供 AWS 服務中所有動作的存取權，但 中指定的動作除外NotAction。您可以將其與 Resource 元素一起使用，以便為政策提供範圍，將允許的動作限制為可以對指定資源執行的動作。

以下範例可讓使用者存取除了刪除儲存貯體之外可在任何 S3 資源上執行的所有 Amazon S3 動作。此政策也不允許在其他服務中執行動作，因為其他服務動作不適用於 S3 資源。

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

有時，您可能想要允許存取大量動作。透過使用 NotAction 元素，您可以有效地反轉陳述式，從而產生更短的動作清單。例如，因為 AWS 有這麼多的服務，您可能想要建立一個政策，允許使用者執行除存取 IAM 動作以外的一切。

下列範例允許使用者存取每個 AWS 服務中的每個動作，IAM 除外。

```
"Effect": "Allow",
"NotAction": "iam:*",
```

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

在相同陳述式中或在政策中的不同陳述式小心使用 NotAction 元素和 "Effect": "Allow"。NotAction 符合未明確列出或適用於指定資源的所有服務和動作，並可能導致授予使用者比您預期更多的許可。

帶拒絕的 NotAction

您可以在陳述式中使用 NotAction 元素 "Effect": "Deny" 拒絕存取除 NotAction 元素中指定的動作之外的所有列出的資源。此組合不允許列出的項目，而是明確拒絕未列出的動作。您仍必須允許您要允許的動作。

如果使用者未使用 MFA 登入，則以下條件範例拒絕存取非 IAM 動作。如果使用者使用 MFA 登入，則 "Condition" 測試失敗，最終的 "Deny" 陳述式沒有影響。但請注意，這不會授予使用者存取任何動作的許可；它只會明確拒絕除 IAM 動作以外的所有其他動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

如需拒絕特定區域以外動作存取 (特定服務除外) 的範例政策，請參閱 AWS： AWS 根據請求的區域拒絕對 的存取。

IAM JSON 政策元素：Resource

IAM 政策陳述式中的 Resource 元素可指定套用該陳述式的一個或多個物件。陳述式必須包含 Resource 或 NotResource 元素。

您可以使用 Amazon Resource Name (ARN) 來指定資源。ARN 的格式取決於您參考的 AWS 服務 和特定資源。雖然 ARN 格式不同，但務必使用 ARN 來識別資源。如需有關 ARN 格式的詳細資訊，請參閱 IAM ARN。如需有關如何指定資源的詳細資訊，請參閱您要撰寫陳述式的服務文件。

> **Note**
>
> 有些 AWS 服務 不允許您指定個別資源的動作。在這些情況下，您在 Action 或 NotAction 元素中列出的任何動作都會套用至該服務中的所有資源。在這種情況下，您可以在 Resource 元素中使用萬用字元 (*)。

以下範例參考特定的 Amazon SQS 佇列。

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

以下範例參考 AWS 帳戶中名為 Bob 的 IAM 使用者。

> **Note**
>
> 在 Resource 元素中，IAM 使用者名稱區分大小寫。

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

在資源 ARN 中使用萬用字元

可以在 ARN 的個別區段內 (各部分以冒號分隔) 使用萬用字元 (* 和 ?)。

- 字元的任意組合 (*)
- 任何單一字元 (?)

您可以在每個區段中使用多個 * 或 ? 字元。如果萬用字元 * 是資源 ARN 區段的最後一個字元，則其可以展開，以符合冒號界限以外的內容。我們建議您在由冒號分隔的 ARN 區段內使用萬用字元 (* 和 ?)。

> **Note**
>
> 您無法在識別 AWS 產品的服務區段中使用萬用字元。如需 ARN 客群的詳細資訊，請參閱 使用 Amazon Resource Name (ARNs識別 AWS 資源。

以下範例參考路徑為 /accounting 的所有 IAM 使用者。

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

以下範例參考特定 Amazon S3 儲存貯體中的所有項目。

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
```

星號 (*) 字元可以展開以取代區段內的所有項目，包括如正斜線 (/) 等字元，這些字元可能會顯示為指定服務命名空間內的分隔符號。例如，假設下列 Amazon S3 ARN，因為相同的萬用字元擴充邏輯適用於所有服務。

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*/test/*"
```

ARN 中的萬用字元適用於儲存貯體中的所有以下物件，而不僅僅是列出的第一個物件。

```
amzn-s3-demo-bucket/1/test/object.jpg
amzn-s3-demo-bucket/1/2/test/object.jpg
amzn-s3-demo-bucket/1/2/test/3/object.jpg
amzn-s3-demo-bucket/1/2/3/test/4/object.jpg
amzn-s3-demo-bucket/1///test///object.jpg
amzn-s3-demo-bucket/1/test/.jpg
amzn-s3-demo-bucket//test/object.jpg
amzn-s3-demo-bucket/1/test/
```

考慮上一個清單中的最後兩個物件。Amazon S3 物件名稱可以傳統分隔符號正斜線 (/) 字元開頭或結尾。雖然 / 作為分隔符，但在資源 ARN 中使用此字元時卻沒有特定的意義。它被視為與任何其他有效字元相同。ARN 不符合下列物件：

```
amzn-s3-demo-bucket/1-test/object.jpg
amzn-s3-demo-bucket/test/object.jpg
amzn-s3-demo-bucket/1/2/test.jpg
```

指定多個資源

您可以使用 ARNs 陣列，在 `Resource` 元素中指定多個資源。以下範例參考兩個 DynamoDB 資料表。

```
"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
```

```
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

在資源 ARN 中使用政策變數

在 Resource 元素中，您可以在 ARN 中用於識別特定資源 (即 ARN 的尾隨部分) 的部分中使用 JSON 政策變數。例如，您可以使用索引鍵 {aws:username} 做為資源 ARN 的一部分，以表示目前使用者的名稱應做為資源名稱的一部分包含在內。以下範例顯示如何在 {aws:username} 元素中使用 Resource 索引鍵。此政策允許存取與目前使用者名稱符合的 Amazon DynamoDB 資料表。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "dynamodb:*",
        "Resource": "arn:aws:dynamodb:us-east-2:111122223333:table/
${aws:username}"
    }
}
```

如需有關 JSON 政策變數的詳細資訊，請參閱 IAM 政策元素：變數與標籤。

IAM JSON 政策元素：NotResource

NotResource 是進階政策元素，會明確比對除指定資源以外的每項資源。使用 NotResource 可以透過僅列出一些不相符的資源來產生較短的政策，而不是包括相符的長資源清單。這對套用在單一 AWS 服務中的政策特別有用。

例如，假設您有一個名為 HRPayroll 的群組。不應允許 HRPayroll 成員存取除 HRBucket 儲存貯體中的 Payroll 資料夾以外的任何 Amazon S3 資源。以下政策明確拒絕存取除列出的資源以外的所有 Amazon S3 資源。不過，請注意，此政策不授予使用者存取任何資源的許可。

JSON

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

```
  "Statement": {
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
    ]
  }
}
```

一般而言，要明確拒絕對資源的存取，您可以編寫使用 "Effect":"Deny" 的政策，其中包含個別列出每個資料夾的 Resource 元素。但是，在這種情況下，每次向 HRBucket 新增資料夾或向 Amazon S3 新增不應存取的資源時，都必須將其名稱新增至 Resource 元素中的清單。如果您使用 NotResource 元素，則會自動拒絕使用者存取新資料夾，除非您將資料夾名稱新增到 NotResource 元素。

使用 NotResource 時，請記住，此元素中指定的資源是「唯一」不受限的資源。因此，這會限制可能套用至動作的所有資源。在上述範例中，政策只影響 Amazon S3 動作，因此只會影響 Amazon S3 資源。如果 Action 元素也包含 Amazon EC2 動作，則該政策將拒絕存取未在 NotResource 元素中指定的任何 EC2 資源。若要了解服務中的哪些動作允許指定資源的 ARN，請參閱 AWS 服務的動作、資源和條件金鑰。

NotResource 與其他元素

"Effect": "Allow"、"Action": "*" 和 "NotResource": "arn:aws:s3:::HRBucket" 元素絕對不能一起使用。此陳述式非常危險，因為它允許 S3 HRBucket 儲存貯體以外所有資源 AWS 上的 中的所有動作。 S3 它甚至允許使用者自行新增允許他們存取 HRBucket 的政策。請勿執行此作業。

在相同陳述式中或在政策中的不同陳述式小心使用 NotResource 元素和 "Effect": "Allow"。NotResource 允許未明確列出的所有服務和資源，並可能導致授予使用者比您預期的更多許可。在相同陳述式中使用 NotResource 元素和 "Effect": "Deny" 會拒絕未明確列出的服務和資源。

IAM JSON 政策元素：Condition

Condition 元素 (或 Condition 區塊) 可讓您於政策生效時指定條件。Condition 元素是選用的。在 Condition 元素中，您所建置的表達式使用條件運算子 (等於、小於等等) 來比對政策中的內容索引鍵和值，以及請求內容中的索引鍵和值。若要進一步了解請求內容，請參閱 請求的元件。

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

您在政策條件中指定的內容索引鍵可以是全域條件內容索引鍵或服務特定的內容索引鍵。全域條件內容索引鍵包含 aws: 字首。服務特定的內容索引鍵包含服務的字首。例如，Amazon EC2 可讓您使用 ec2:InstanceType 內容索引鍵撰寫條件，此索引鍵對於該服務是唯一的。若要檢視包含 iam: 字首的服務特定的 IAM 內容索引鍵，請參閱 IAM 和 AWS STS 條件內容索引鍵。

內容索引鍵名稱不區分大小寫。例如，包括 aws:SourceIP 內容索引鍵等同於對 AWS:SourceIp 的測試。內容索引鍵值是否區分大小寫取決於您使用的條件運算子。例如，下列條件包含 StringEquals 運算子，以確保僅 john 所做的請求會相符。使用者指定的 John 會受到存取遭拒。

```
"Condition" : { "StringEquals" : { "aws:username" : "john" }}
```

以下條件使用 StringEqualsIgnoreCase 運算子，以符合使用者指定的 john 或 John。

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "john" }}
```

部分內容索引鍵支援索引鍵值對，可允許您指定部分索引鍵名稱。範例包括aws:RequestTag/*tag-key*內容索引鍵、 AWS KMS kms:EncryptionContext:*encryption_context_key*和多個 服務支援ResourceTag/*tag-key*的內容索引鍵。

- 如果您對 Amazon EC2 之類的服務使用 ResourceTag/*tag-key* 內容索引鍵，則必須為 tag-key 指定索引鍵名稱。
- 金鑰名稱不區分大小寫。這表示如果您在政策的條件元素中指定 "aws:ResourceTag/TagKey1": "Value1"，則該條件會符合名為 TagKey1 或 tagkey1 的資源標籤鍵 (但不會同時符合兩者)。
- AWS 支援這些屬性的 服務可能允許您建立多個金鑰名稱，這些名稱僅因案例而異。例如，可能使用 ec2=test1 和 EC2=test2 標記 Amazon EC2 執行個體。當您使用條件 (例如，"aws:ResourceTag/EC2": "test1") 以允許對該資源的存取，則金鑰名稱同時符合兩個標籤，但只有一個值符合。這會導致意外的條件失敗。

> **Important**
>
> 根據最佳實務，確保帳戶成員在命名鍵值對屬性時遵守一致的命名慣例。範例包括標籤或 AWS KMS 加密內容。您可以使用aws:TagKeys內容金鑰進行標記，或使用 kms:EncryptionContextKeys進行 AWS KMS 加密內容。

- 如需所有條件運算子的清單及其運作方式的描述，請參閱條件運算子。
- 除非另有說明，否則，所有內容索引鍵都可以具有多個值。如需有關如何處理具有多個值的內容索引鍵的描述，請參閱 用於多值內容索引鍵的集運算子
- 如需有關所有全域可用內容索引鍵的清單，請參閱 AWS 全域條件內容索引鍵。
- 如需每個服務定義的條件內容金鑰，請參閱 AWS 服務的動作、資源和條件金鑰。

請求內容

當委託人向 提出請求時 AWS， 會將請求資訊 AWS 收集到請求內容中。請求內容包含有關主體、資源、動作和其他環境屬性的資訊。政策評估會將政策中的屬性與請求中傳送的屬性比對，以評估和授權可在 AWS中執行的動作。

您可以使用 JSON 政策的 Condition 元素來針對請求內容測試特定內容索引鍵。例如，您可以建立使用 aws:CurrentTime 內容索引鍵的政策，來僅允許使用者在特定日期範圍內執行動作。

下列範例展示了當 Martha Rivera 傳送請求以停用其 MFA 裝置時的請求內容表示。

```
Principal: AROA123456789EXAMPLE
Action: iam:DeactivateMFADevice
Resource: arn:aws:iam::user/martha
Context:
  - aws:UserId=AROA123456789EXAMPLE:martha
  - aws:PrincipalAccount=1123456789012
  - aws:PrincipalOrgId=o-example
  - aws:PrincipalARN=arn:aws:iam::1123456789012:assumed-role/TestAR
  - aws:MultiFactorAuthPresent=true
  - aws:MultiFactorAuthAge=2800
  - aws:CurrentTime=...
  - aws:EpochTime=...
  - aws:SourceIp=...
```

該請求內容與一項政策相符，該政策允許使用者移除自己的多重要素驗證 (MFA) 裝置，但前提是他們必須在過去一小時 (3,600 秒) 內使用 MFA 登入過。

JSON

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

```
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

在此範例中，政策與請求內容相符：動作相同，資源與 "*" 萬用字元相符，且 `aws:MultiFactorAuthAge` 的值為 2800，小於 3600，因此政策允許此授權請求。

AWS 會評估政策中的每個內容索引鍵，並傳回 true 或 false 的值。請求中不存在的內容索引鍵視為不相符。

請求內容可傳回下列值：

- True (真) – 若申請者在過去一個小時或一個小時內使用 MFA 登入，條件便會傳回 true。
- False (偽) – 若申請者在過去超過一個小時中使用 MFA 登入，條件便會傳回 false。
  - 不存在 – 如果請求者在 AWS CLI 或 AWS API 中使用其 IAM 使用者存取金鑰提出請求，則金鑰不存在。在此情況下，由於索引鍵不存在，因此不會相符。

> **Note**
>
> 在某些情況下，當條件索引鍵值不存在時，條件仍然可以傳回 true。例如，如果新增 `ForAllValues` 限定詞，則當請求中不存在內容索引鍵時，請求會傳回 true。若要防止遺失內容索引鍵或具有空值的內容索引鍵評估為 true，您可以在政策中包含值為 `false` 的 Null 條件運算子，以檢查內容索引鍵是否存在且其值不為 Null。

條件區塊

以下範例顯示 `Condition` 元素的基本格式：

```
"Condition": {"StringLike": {"s3:prefix": ["jane/*"]}}
```

來自請求的值由內容索引鍵表示，在本例中為 s3:prefix。內容金鑰值會與您指定為常值的值進行比較，例如 jane/*。要進行的比較類型由 條件運算子 (在這裡為StringLike) 指定。您可以使用典型的布林比較 (例如，等於、大於和小於) 來建立比較字串、日期、數字等條件。當您使用字串運算子或 ARN 運算子時，您也可以在內容索引鍵值中使用政策變數。下列範例包含 aws:username 變數。

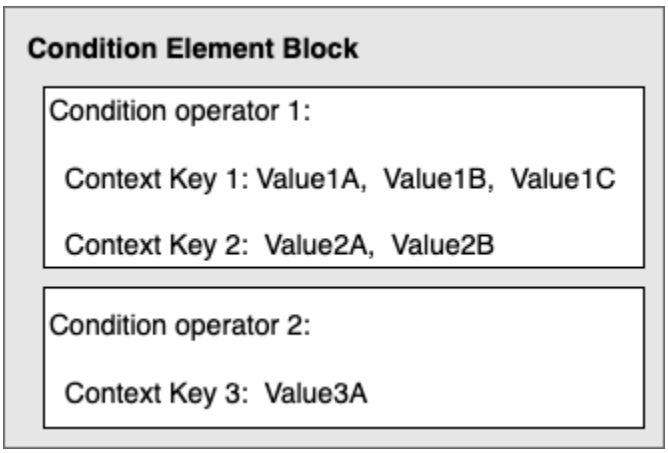
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

在某些情況下，內容索引鍵可以包含多個值。例如，對 Amazon DynamoDB 的請求可能會要求從資料表中傳回或更新多個屬性。用於存取 DynamoDB 資料表的政策可以包括 dynamodb:Attributes 內容索引鍵，其包含請求中列出的所有屬性。您可以使用 Condition 元素中的設定運算子，根據政策中允許的屬性清單測試請求中的多個屬性。如需詳細資訊，請參閱用於多值內容索引鍵的集運算子。

在請求期間評估政策時， 會將金鑰 AWS 取代為請求中的對應值。（在此範例中， AWS 將使用請求的日期和時間。) 負責評估條件，以傳回 true 或 false，然後將該因素考慮為整個政策是否允許或拒絕請求。

條件中的多個值

Condition 元素可以包含多個條件運算子，而且每個條件運算子都可以包含多個內容索引鍵值對。下圖說明了這一點。



如需詳細資訊，請參閱用於多值內容索引鍵的集運算子。

IAM JSON 政策元素：條件運算子

在 Condition 元素中使用條件運算子來將政策中的條件鍵和值與請求內容中的值進行比對。如需有關 Condition 元素的詳細資訊，請參閱 IAM JSON 政策元素：Condition。

您可以在政策中使用的條件運算子取決於您選擇的條件索引鍵。您可以選擇全域條件索引鍵或服務限定條件索引鍵。若要了解針對全域條件索引鍵您可以使用的條件運算子，請參閱 AWS 全域條件內容索

引鍵。若要了解您可以用於服務特定條件金鑰的條件運算子，請參閱 AWS 服務的動作、資源和條件金鑰，然後選擇您要檢視的服務。

> **Important**
>
> 如果您在政策條件中指定的索引鍵不存在於請求內容中，則值不相符且條件為 false。如果政策條件要求索引鍵為不相符，例如 `StringNotLike` 或 `ArnNotLike`，並且正確的索引鍵不存在，則條件為 true。除了 ...IfExist 和 Null 檢查以外，此邏輯適用於所有條件運算子。這些運算子測試索引鍵是否存在於請求內容中。

條件運算子可分成以下幾種類別：

- ：字串
- 數值
- 日期和時間
- ：布林值
- ：二進位
- IP 地址
- Amazon Resource Name (ARN) (僅可用於部分服務。)
- ...IfExists (檢查鍵值是否做為另一檢查的一部分存在)
- Null 檢查 (檢查鍵值是否做為獨立檢查存在)

字串條件運算子

運用字串條件運算子，您可以建構以索引鍵與字串值的對比為基礎來限制存取的 `Condition` 元素。

- 政策變數 – 支援
- 萬用字元 – 支援

條件運算子	描述
`StringEquals`	完全相符，區分大小寫
`StringNotEquals`	否定相符

條件運算子	描述
StringEqualsIgnoreCase	完全相符，不區分大小寫
StringNotEqualsIgnoreCase	否定相符，不區分大小寫
StringLike	大小寫相符。值可以在字串的任何位置包含多字元比對萬用字元 (*) 和單一字元比對萬用字元 (?)。您必須指定萬用字元才能達到部分字串相符。 **Note** 如果索引鍵包含多個值，StringLike 可使用集合運算子限定 ForAllValues:StringLike 和 ForAnyValue:StringLike。如需詳細資訊，請參閱用於多值內容索引鍵的集運算子。
StringNotLike	否定大小寫相符。值可以在字串的任何位置包含多字元比對萬用字元 (*) 或單一字元比對萬用字元 (?)。

Example字串條件運算子

例如，下列陳述式包含的 Condition 元素使用 aws:PrincipalTag 索引鍵，以指定提出請求的主體必須以 iamuser-admin 任務類別標記。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalTag/job-category": "iamuser-admin"
```

```
            }
        }
    }
}
```

如果您在政策條件中指定的索引鍵不存在於請求內容中，則值不相符。在此範例中，如果主體使用的 IAM 使用者已連接標籤，則 aws:PrincipalTag/job-category 索引鍵存在於請求內容中。如果主體使用的 IAM 角色有連接的標籤或工作階段標籤，則也包含此索引鍵。如果不具有標籤的使用者嘗試檢視或編輯存取金鑰，此條件會傳回 false，且此陳述式會隱含拒絕請求。

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	`aws:PrincipalTag/job-category: - iamuser-admin`	匹配
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	`aws:PrincipalTag/job-category: - dev-ops`	無相符項目
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	請求內容中沒有 aws:PrincipalTag/job-category 。	無相符項目

Example將政策變數與字串條件運算子搭配使用

下列範例使用 StringLike 條件運算子執行與政策變數的字串配對來建立政策，該政策允許 IAM 使用者使用 Amazon S3 主控台管理其 Amazon S3 儲存貯體中的「主目錄」。該政策允許對 S3 儲存貯體執行指定操作，前提是 s3:prefix 與任一指定模式相符。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "",
            "home/",
            "home/${aws:username}/"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

下表顯示 如何根據請求內容中的aws:username值，評估不同使用者的 AWS 此政策。

政策條件	請求內容	結果
`"StringLike": { "s3:prefix": [ "home/", "home/${aws:username}/" ] }`	`aws:username: - martha_rivera`	`"StringLike": { "s3:prefix": [ "home/", "home/martha_rivera /" ] }`
`"StringLike": { "s3:prefix": [ "home/", "home/${aws:username}/" ] }`	`aws:username: - nikki_wolf`	`"StringLike": { "s3:prefix": [ "home/", "home/nikki_wolf /" ] }`
`"StringLike": { "s3:prefix": [ "home/", "home/${aws:username}/" ] }`	請求內容中沒有 `aws:username` 。	無相符項目

如需關於如何使用 Condition 元素根據 OIDC 聯合身分的應用程式 ID 和使用者 ID 來限制存取資源的政策範例，請參閱 Amazon S3：可讓 Amazon Cognito 使用者存取其儲存貯體中的物件。

多值字串條件運算子

如果請求中的索引鍵包含多個值，則可以使用集運算子 ForAllValues 和 ForAnyValue 來限定字串運算子。如需有關多個內容索引鍵或值之評估邏輯的詳細資訊，請參閱用於多值內容索引鍵的集運算子。

條件運算子	說明
ForAllValues:StringEquals ForAllValues:StringEqualsIgnoreCase	請求中條件索引鍵的所有值都必須與政策中的至少一個值相符。
ForAnyValue:StringEquals ForAnyValue:StringEqualsIgnoreCase	請求中至少有一個條件索引鍵值必須與政策中的其中一個值相符。
ForAllValues:StringNotEquals ForAllValues:StringNotEqualsIgnoreCase	否定相符。 請求中內容索引鍵的所有值都不與政策中的任何內容索引鍵值相符。
ForAnyValue:StringNotEquals ForAnyValue:StringNotEqualsIgnoreCase	否定相符。 請求中至少有一個內容索引鍵值不得與政策中內容索引鍵中的任何值相符。
ForAllValues:StringLike	請求中條件索引鍵的所有值都必須與政策中的至少一個值相符。
ForAnyValue:StringLike	請求中至少有一個條件索引鍵值必須與政策中的其中一個值相符。
ForAllValues:StringNotLike	否定相符。 請求中內容索引鍵的所有值都不與政策中的任何內容索引鍵值相符。
ForAnyValue:StringNotLike	否定相符。 請求中至少有一個內容索引鍵值不得與政策中內容索引鍵中的任何值相符。

Example將 ForAnyValue 與字串條件運算子搭配使用

此範例會示範如何建立身分型政策，它允許使用 Amazon EC2 `CreateTags` 動作以將標籤連接至執行個體。使用 `StringEqualsIgnoreCase` 時，僅當標籤包含 `environment` 鍵且值為 `preprod` 或 `storage` 時，您才能連接標籤。將 `IgnoreCase` 附加至運算子時，可以允許任何現有的標籤值大寫 (例如 `preprod`、`Preprod` 和 `PreProd`) 都解析為 true。

將 `ForAnyValue` 修飾詞與 aws:TagKeys 條件索引鍵搭配使用時，請求中至少有一個標籤索引鍵值必須與值 `environment` 相符。`ForAnyValue` 比較會區分大小寫，這可以阻止使用者使用錯誤大小寫的標籤索引鍵，例如使用 `Environment` 而不是 `environment`。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:*:*:instance/*",
    "Condition": {
      "StringEqualsIgnoreCase": {
        "aws:RequestTag/environment": [
          "preprod",
          "storage"
        ]
      },
      "ForAnyValue:StringEquals": {
        "aws:TagKeys": "environment"
      }
    }
  }
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"StringEqualsIgnor eCase": {`	`aws:TagKeys:` `  - environment`	匹配

政策條件	請求內容	結果
"aws:RequestTag/environment": ["preprod", "storage"] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }	aws:RequestTag/ environment: - preprod	
"StringEqualsIgnoreCase": { "aws:RequestTag/environment": ["preprod", "storage"] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }	aws:TagKeys: - environment - costcenter aws:RequestTag/environment: - PreProd	匹配
"StringEqualsIgnoreCase": { "aws:RequestTag/environment": ["preprod", "storage"] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }	aws:TagKeys: - Environment aws:RequestTag/ Environment: - preprod	無相符項目

政策條件	請求內容	結果
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - costcenter aws:RequestTag/environment: - preprod`	無相符項目
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	請求內容中沒有 aws:TagKeys 。 `aws:RequestTag/environment: - storage`	無相符項目

政策條件	請求內容	結果
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment` 請求內容中沒有 aws:RequestTag/environment 。	無相符項目
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	請求內容中沒有 aws:TagKeys 。 請求內容中沒有 aws:RequestTag/environment 。	無相符項目

萬用字元比對

字串條件運算子會執行不強制執行預先定義格式的無模式比對。ARN 和 Date 條件運算子是字串運算子的子集，它們會對條件索引鍵值強制執行一種結構。

建議使用與索引鍵要比較的值對應的條件運算子。例如，將索引鍵與字串值進行比較時，應該使用 the section called “字串條件運算子”。同樣地，將索引鍵與 ARN 值進行比較時，應該使用 the section called “Amazon Resource Name (ARN) 條件運算子”。

Example

此範例說明如何圍繞組織中的資源建立邊界。此政策中的條件將拒絕存取 Amazon S3 動作，除非正在存取的資源位於 AWS Organizations中的一組特定組織單位中。 AWS Organizations 路徑是組織實體結構的文字表示。

條件要求 aws:ResourceOrgPaths 包含任何列出的組織單位路徑。由於 aws:ResourceOrgPaths 是多值條件，因此政策會使用 ForAllValues:StringNotLike 運算子來比較 aws:ResourceOrgPaths 的值與政策中的組織單位清單。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyS3AccessOutsideMyBoundary",
      "Effect": "Deny",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:ResourceOrgPaths": [
            "o-acorg/r-acroot/ou-acroot-mediaou/",
            "o-acorg/r-acroot/ou-acroot-sportsou/*"
          ]
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"ForAllValues:StringNotLike": {`	`aws:ResourceOrgPaths:`	匹配

政策條件	請求內容	結果
"aws:ResourceOrgPaths": ["o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*"] }	- o-acorg/r-acroot/ou-acroot-sportsou/costcenter/	
"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": ["o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*"] }	aws:ResourceOrgPaths: - o-acorg/r-acroot/ou-acroot-mediaou/costcenter/	無相符項目
"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": ["o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*"] }	請求中沒有 aws:ResourceOrgPaths: 。	無相符項目

數位條件運算子

利用數位條件運算子，您可以建構以索引鍵與整數或小數值的對比來限制存取的 Condition 元素。

- 政策變數 – 不支援
- 萬用字元 – 不支援

條件運算子	描述
NumericEquals	相符
NumericNotEquals	否定相符
NumericLessThan	「小於」相符
NumericLessThanEqu als	「小於或等於」相符
NumericGreaterThan	「大於」相符
NumericGreaterThan Equals	「大於或等於」相符

例如，下列聲明包含一個 Condition 元素，該元素使用 NumericLessThanEquals 條件運算子與 s3:max-keys 金鑰來指定申請者一次「最多」可在 amzn-s3-demo-bucket 內列出 10 個物件。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
}
```

如果您在政策條件中指定的索引鍵不存在於請求內容中，則值不相符。在此範例中，當您執行 s3:max-keys 操作時，ListBucket 索引鍵永遠存在於請求中。如果此政策允許所有 Amazon S3 操作，則只有包含 max-keys 內容索引鍵且值小於或等於 10 的操作，才允許執行。

日期條件運算子

運用日期條件運算子，您可以建構以索引鍵與日期/時間值的對比為基礎來限制存取的 Condition 元素。您同時使用這些條件運算子與 aws:CurrentTime 索引鍵或 aws:EpochTime 索引鍵。您必須指定日期/時間值，且其中一個 W3C 實作要採用 ISO 8601 日期格式或 epoch (UNIX) 時間格式。

- 政策變數 – 不支援
- 萬用字元 – 不支援

條件運算子	描述
DateEquals	符合特定日期
DateNotEquals	否定相符
DateLessThan	在特定日期與時間前相符
DateLessThanEquals	在特定日期與時間時或之前相符
DateGreaterThan	在特定日期與時間後相符
DateGreaterThanEqu als	在特定日期與時間時或之後相符

例如，下列陳述式包含的 Condition 元素將 DateGreaterThan 條件運算子與 aws:TokenIssueTime 索引鍵搭配使用。此條件指定用來提出請求的臨時安全憑證已於 2020 年發行。此政策可以每天以程式設計方式更新，以確保帳戶成員使用全新的憑證。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "DateGreaterThan": {
```

```
                "aws:TokenIssueTime": "2020-01-01T00:00:01Z"
            }
        }
    }
}
```

如果您在政策條件中指定的索引鍵不存在於請求內容中，則值不相符。只有在主體使用臨時憑證發出請求時，aws:TokenIssueTime 索引鍵才存在於請求內容中。金鑰不存在於使用存取金鑰提出的 AWS CLI API AWS 或 AWS SDK 請求中。在此範例中，如果 IAM 使用者嘗試檢視或編輯存取金鑰，則會拒絕請求。

布林值條件運算子

運用布林值條件，您可以建構 Condition 元素，這些元素會根據將索引鍵與 true 或 false 進行比較的結果來限制存取權。

如果索引鍵包含多個值，布林運算子可以使用集運算子 ForAllValues 和 ForAnyValue 進行限定。如需有關多個內容索引鍵或值之評估邏輯的詳細資訊，請參閱用於多值內容索引鍵的集運算子。

- 政策變數 – 支援
- 萬用字元 – 不支援

條件運算子	描述
Bool	布林值相符
ForAllValues:Bool	與 Array of Bool 資料類型搭配使用。內容索引鍵值中的所有布林值必須與政策中的布林值相符。 若要防止 ForAllValues 運算子將遺失內容索引鍵或具有空值的內容索引鍵評估為「已允許」，您可以在政策中包含 Null 條件運算子。
ForAnyValue:Bool	與 Array of Bool 資料類型搭配使用。內容索引鍵值中至少有一個布林值必須與政策中的布林值相符。

Example布林值條件運算子

下列身分型政策透過將 Bool 條件運算子和 aws:SecureTransport 索引鍵搭配使用，拒絕當請求不在 SSL 上時，將物件和物件標籤複寫至目的地儲存貯體及其內容。

> **Important**
>
> 此政策不允許任何動作。將此政策與允許特定動作的其他政策結合使用。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"Bool": {`	`aws:SecureTransport:` `  - false`	匹配

政策條件	請求內容	結果
"aws:SecureTranspo rt": "false" }		

"Bool": {
"aws:SecureTranspo
rt": "false"
}
	aws:SecureTransport:
	- true
		無相符項目

"Bool": {
"aws:SecureTranspo
rt": "false"
}
	請求內容中沒有 aws:Secur
	eTransport 。
		無相符項目

二進位條件運算子

運用 BinaryEquals 條件運算子，您可以建構用於測試二進位格式索引鍵值的 Condition 元素。它會比較指定金鑰位元組的值與政策中 base-64 編碼表示的二進位值。如果您在政策條件中指定的索引鍵不存在於請求內容中，則值不相符。

- 政策變數 – 不支援
- 萬用字元 – 不支援

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

政策條件	請求內容	結果
"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA=="	key: - QmluYXJ5VmFsdWVJbk Jhc2U2NA==	匹配

政策條件	請求內容	結果
} 		

"BinaryEquals": {
  "key" : "QmluYXJ5
VmFsdWVJbkJhc2U2NA=="
}
	key:
	  - ASIAIOSFODNN7EXAMP
	LE
		無相符項目

"BinaryEquals": {
  "key" : "QmluYXJ5
VmFsdWVJbkJhc2U2NA=="
}
	請求內容中沒有 key。	無相符項目

IP 地址條件運算子

運用 IP 地址條件運算子，您可以建構 Condition 元素，它們會以索引鍵與 IPv4 或 IPv6 地址或 IP 地址範圍的對比來限制存取。您可以搭配 aws:SourceIp 索引鍵來使用這些運算子。該值必須採用標準的 CIDR 格式 (例如 203.0.113.0/24 或 2001:DB8:1234:5678::/64)。如果您指定的 IP 地址沒有關聯的路由字首，IAM 將使用 /32 做為預設字首值。

有些 AWS 服務支援 IPv6，使用 ：： 代表 0 的範圍。若要了解某項服務是否支援 IPv6，請參閱該服務的文件。

- 政策變數 – 不支援
- 萬用字元 – 不支援

條件運算子	描述
IpAddress	指定的 IP 地址或範圍
NotIpAddress	除指定 IP 地址或範圍外的所有 IP 地址

Example IP 位址條件運算子

下列陳述式將 IpAddress 條件運算子與 aws:SourceIp 索引鍵搭配使用，來指定請求必須來自 IP 範圍 203.0.113.0 至 203.0.113.255。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "IpAddress": {
                "aws:SourceIp": "203.0.113.0/24"
            }
        }
    }
}
```

aws:SourceIp 條件索引鍵解析為發出請求的 IP 地址。如果請求源自 Amazon EC2 執行個體，則 aws:SourceIp 計算為執行個體的公有 IP 地址。

如果您在政策條件中指定的索引鍵不存在於請求內容中，則值不相符。aws:SourceIp 索引鍵永遠存在於請求內容中，但請求者使用 VPC 端點提出請求時為例外。在此情況下，此條件會傳回 false，且此陳述式會隱含拒絕請求。

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"IpAddress": { "aws:SourceIp": "203.0.113.0/24" }`	`aws:SourceIp: - 203.0.113.1`	匹配
`"IpAddress": {`	`aws:SourceIp: - 198.51.100.1`	無相符項目

政策條件	請求內容	結果
"aws:SourceIp": "203.0.113.0/24" }		

以下範例顯示如何混合使用 IPv4 與 IPv6 地址，以涵蓋您組織的所有有效 IP 地址。建議您使用您的 IPv6 地址範圍以及已擁有的 IPv4 範圍來更新組織的政策，以確保政策在您過渡到 IPv6 時繼續有效。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
        ]
      }
    }
  }
}
```

如果以使用者身分直接呼叫測試的 API，aws:SourceIp 條件索引鍵僅在 JSON 政策中有效。如果改為使用服務代表您呼叫目標服務，則目標服務看到的是進行呼叫的服務的 IP 地址而不是來源使用者的 IP 地址。例如，如果使用 AWS CloudFormation 呼叫 Amazon EC2 來建構執行個體，則會發生這種情況。目前，無法透過進行呼叫的服務將來源 IP 地址傳遞給目標服務以在 JSON 政策中進行評估。對於這些服務 API 呼叫類型，請勿使用 aws:SourceIp 條件索引鍵。

Amazon Resource Name (ARN) 條件運算子

運用 Amazon Resource Name (ARN) 條件運算子，您可以建構以索引鍵與 ARN 的對比為基礎來限制存取的 Condition 元素。ARN 被視為一個字串。

- 政策變數 – 支援
- 萬用字元 – 支援

條件運算子	描述
ArnEquals ，ArnLike	ARN 大小寫相符。ARN 的六個由冒號分隔開的部分都要單獨檢查，每一個部分都可包括一個多字元比對萬用字元 (*) 或一個單字元比對萬用字元 (?)。ArnEquals 和 ArnLike 條件運算子的行為完全相同。
ArnNotEquals ，ArnNotLike	ARN 否定相符。ArnNotEquals 和 ArnNotLike 條件運算子的行為完全相同。

Example ARN 條件運算子

下列以資源為基礎的政策範例顯示連接至 Amazon SQS 佇列的政策 (您想要將 SNS 訊息傳送到此佇列)。它授予 Amazon SNS 將訊息傳送到所選佇列的許可，但僅在該服務代表特定 Amazon SNS 主題傳送這些訊息時才授予此許可。您在 Resource 欄位中指定佇列，Amazon SNS 主題則作為 SourceArn 索引鍵的值。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "sns.amazonaws.com"
        },
        "Action": "SQS:SendMessage",
        "Resource": "arn:aws:sqs:us-east-1:123456789012:QUEUE-ID",
        "Condition": {
            "ArnEquals": {
                "aws:SourceArn": "arn:aws:sns:us-east-1:123456789012:TOPIC-ID"
            }
        }
    }
}
```

僅當資源觸發服務代表資源擁有者呼叫另一項服務時，請求內容中才存在 aws:SourceArn 索引鍵。如果 IAM 使用者嘗試直接執行此操作，此條件會傳回 false，且此陳述式會隱含拒絕請求。

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`aws:SourceArn: - arn:aws:sns:us-west-2:123456789012:TOPIC-ID`	匹配
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`aws:SourceArn: - arn:aws:sns:us-west-2:777788889999:TOPIC-ID`	無相符項目
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	請求內容中沒有 aws:SourceArn 。	無相符項目

多值 ARN 條件運算子

如果請求中的索引鍵包含多個值，則可以使用集運算子 ForAllValues 和 ForAnyValue 來限定 ARN 運算子。如需有關多個內容索引鍵或值之評估邏輯的詳細資訊，請參閱用於多值內容索引鍵的集運算子。

條件運算子	說明
ForAllValues:ArnEquals	請求內容中的所有 ARN 都必須與政策中的至少一個 ARN 模式相符。

條件運算子	說明
ForAllValues:ArnLike	
ForAnyValue:ArnEquals ForAnyValue:ArnLike	請求內容中至少有一個 ARN 必須與政策中的其中一個 ARN 模式相符。
ForAllValues:ArnNotEquals	否定相符。
ForAllValues:ArnNotLike	請求內容中的所有 ARN 都不與政策中的任何字串 ARN 模式相符。
ForAnyValue:ArnNotEquals	否定相符。
ForAnyValue:ArnNotLike	請求內容中必須至少有一個 ARN 不得與政策中的任何 ARN 模式相符。

Example將 ForAllValues 與 ARN 條件運算子搭配使用

下列範例使用 ForAllValues:ArnLike 來建立或更新 Amazon CloudWatch Logs 日誌的邏輯交付來源。條件區塊包含條件索引鍵 logs:LogGeneratingResourceArns，用於篩選請求中傳遞的日誌產生資源 ARN。使用此條件運算子，請求中的所有 ARN 都必須與政策中的至少一個 ARN 相符。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "logs:PutDeliverySource",
            "Resource": "arn:aws:logs:us-east-1:123456789012:delivery-source:*",
            "Condition": {
                "ForAllValues:ArnLike": {
                    "logs:LogGeneratingResourceArns": [
                        "arn:aws:cloudfront::123456789012:distribution/*",
                        "arn:aws:cloudfront::123456789012:distribution/support*"
                    ]
                }
```

```
            }
        }
    ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter`	匹配
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter - arn:aws::cloudfront:123456789012:distribution/support2025`	匹配
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter`	匹配

政策條件	請求內容	結果
"arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*"] }	- arn:aws::cloudfront:123456789012:distribution/admin	

"ForAllValues:ArnLike": {
"logs:LogGeneratingResourceArns": [
"arn:aws::cloudfront:123456789012:distribution/*",
"arn:aws::cloudfront:123456789012:distribution/support*"
]
}
	logs:LogGeneratingResourceArns:
	- arn:aws::cloudfront:777788889999:distribution/costcenter
		無相符項目

"ForAllValues:ArnLike": {
"logs:LogGeneratingResourceArns": [
"arn:aws::cloudfront:123456789012:distribution/*",
"arn:aws::cloudfront:123456789012:distribution/support*"
]
}
	請求內容中沒有 logs:LogGeneratingResourceArns 。	匹配

如果請求中沒有內容索引鍵，或內容索引鍵值解析為 Null 資料集 (例如空白字串)，則 ForAllValues 限定詞會傳回 true。若要防止遺失內容索引鍵或具有空值的內容索引鍵評估為 true，您可以在政策中包含值為 false 的 Null 條件運算子，以檢查內容索引鍵是否存在且其值不為 Null。

...IfExists 條件運算子

除 IfExists 條件外，您可在任何條件運算子名稱的尾端加入 Null，例如 StringLikeIfExists。執行此操作，並表明如果請求的內容中存在條件索引鍵，則依照政策所述來處理索引鍵。如果該索引鍵不存在，則評估條件元素為 true。」 陳述式中其他條件因素仍然可以導致不相符，但使用 ...IfExists 檢查時並非使用遺失索引鍵。如果使用 "Effect": "Deny" 元素搭配否定條件運算子，例如 StringNotEqualsIfExists，即使條件索引鍵不存在，仍然會拒絕請求。

使用 **IfExists** 的範例

許多條件索引鍵描述有關特定類型的資源的資訊，僅當存取該類型的資源時才存在。這些條件索引鍵在其他類型的資源上不存在。當政策陳述式僅適用於一種類型的資源時，這不會導致問題。但是，有時單一陳述式可以適用於多種類型的資源，例如當政策陳述式從多個服務引用操作時，或是當服務中的給定操作存取同一服務中的多種不同資源類型時。在這種情況下，在政策陳述式中包含僅適用於一種資源的條件索引鍵可能會導致政策陳述式中的 Condition 元素失敗，從而使陳述式的 "Effect" 不適用。

例如，請設想以下政策範例：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
      "m3.*"
    ]}}
  }
}
```

上述政策的意圖是讓使用者可以啟動類型為 t1、t2 或 m3 的任何執行個體。但是，啟動執行個體要求存取除了執行個體本身之外的許多資源；例如映射、金鑰對、安全群組等。會針對啟動執行個體所需的每個資源來評估整個陳述式。這些其他資源沒有 ec2:InstanceType 條件金鑰，因此 StringLike 檢查會失敗，並且不會向使用者授予啟動「任何」執行個體類型的能力。

若要解決此問題，請改用 StringLikeIfExists 條件運算子。如此一來，僅有在條件索引鍵存在時才會進行測試。您會讀到以下政策：「如果正在檢查的資源有 "ec2:InstanceType" 條件索引鍵，那麼只會在鍵值以 t1.、t2. 或 m3. 開頭時才允許該動作。如果正在檢查的資源沒有該條件索引鍵，則無需擔心它。』 與 StringLikeIfExists 條件運算子搭配使用時，條件索引鍵值中的星號 (*) 會被解譯為萬用字元，以實現部分字串相符。DescribeActions 陳述式包含在主控台檢視執行個體所需的動作。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RunInstance",
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
        "StringLikeIfExists": {
          "ec2:InstanceType": [
            "t1.*",
            "t2.*",
            "m3.*"
          ]
        }
      }
    },
    {
      "Sid": "DescribeActions",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeImages",
        "ec2:DescribeInstances",
        "ec2:DescribeVpcs",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	`ec2:InstanceType: - t1.micro`	匹配
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	`ec2:InstanceType: - m2.micro`	無相符項目
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	請求內容中沒有 `ec2:InstanceType` 。	匹配

用於檢查條件索引鍵是否存在的條件運算子

使用 Null 條件運算子檢查授權時是否沒有條件索引鍵。在政策陳述式中使用 true (索引鍵不存在 - 為 null) 或 false (索引鍵存在且值不為 null)。

您不可使用政策變數與 Null 條件運算子。

例如，您可以使用此條件運算子來確定使用者使用的是臨時憑證還是自己的憑證來發出請求。如果使用者使用的是臨時憑證，則索引鍵 aws:TokenIssueTime 存在並具有一個值。下列範例展示了一個條件，該條件規定使用者必須使用臨時憑證 (必須存在索引鍵) 才能使用 Amazon EC2 API。

JSON

```
{
  "Version":"2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"false"}}
  }
}
```

具有多個內容索引鍵或值的條件

您可以使用政策的 Condition 元素來測試請求中單一內容索引鍵的多個內容索引鍵或多個值。當您以程式設計方式或透過 提出請求時 AWS， AWS 管理主控台您的請求會包含有關委託人、操作、標籤等資訊。您可以使用內容索引鍵來測試請求中相符的內容索引鍵的值，以及政策條件中指定的內容索引鍵。若要了解請求中包含的資訊和資料，請參閱 請求內容。

主題

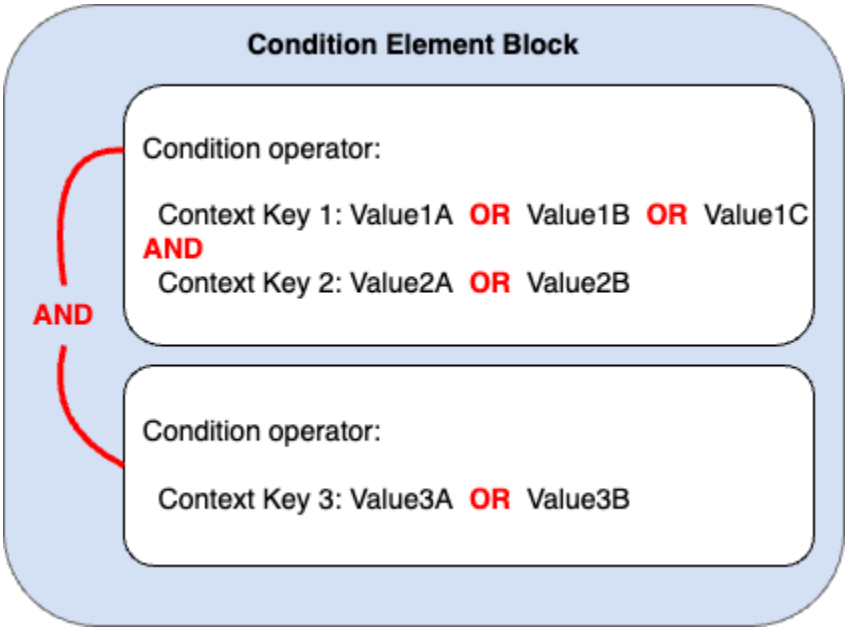
- 多個內容索引鍵或值的評估邏輯
- 否定相符條件運算子的評估邏輯

多個內容索引鍵或值的評估邏輯

Condition 元素可以包含多個條件運算子，而且每個條件運算子都可以包含多個內容索引鍵值對。除非另有指定，否則大多數內容索引鍵都支援使用多個值。

- 如果您的政策陳述式有多個條件運算子，則使用邏輯 AND 評估條件運算子。
- 如果您的政策陳述式已將多個內容索引鍵連接至單一條件運算子，則使用邏輯 AND 評估內容索引鍵。
- 如果單一條件運算子包含一個內容索引鍵的多個值，則使用邏輯 OR 評估這些值。
- 如果單一否定相符條件運算子包含一個內容索引鍵的多個值，則使用邏輯 NOR 評估這些值。

條件元素區塊中的所有內容索引鍵都必須解析為 true 才能調用所需的 `Allow` 或 `Deny` 效果。下圖說明具有多個條件運算子和內容索引鍵值對的條件的計算邏輯。



例如，下列 S3 儲存貯體政策說明上圖在政策中的表示方式。條件區塊包含條件運算子 `StringEquals` 和 `ArnLike`，以及內容索引鍵 `aws:PrincipalTag` 和 `aws:PrincipalArn`。若要調用所需的 `Allow` 或 `Deny` 效果，條件區塊中的所有內容索引鍵都必須解析為 true。發出請求的用戶必須同時具有部門和角色這兩個主體標籤索引鍵，其中包含政策中指定的其中一個標籤索引鍵值。此外，發出請求之使用者的主體 ARN 必須符合政策中指定的其中一個 `aws:PrincipalArn` 值才會評估為 true。

JSON

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

```
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary`	相符

政策條件	請求內容	結果

"audience",
"security"
]
},
"ArnLike": {
"aws:PrincipalArn":
[
"arn:aws:
iam::222222222222:
user/Ana",
"arn:aws:
iam::222222222222:
user/Mary"
]
}		

"StringEquals": {
"aws:PrincipalTag/
department": [
"finance",
"hr",
"legal"
],
"aws:PrincipalTag/
role": [
"audit",
"security"
]
},
"ArnLike": {
"aws:PrincipalArn":
[
"arn:aws:
iam::222222222222:
user/Ana",
"arn:aws:
iam::222222222222:
user/Mary"
]
}
	aws:PrincipalTag/d
	epartment: hr
	aws:PrincipalTag/role:
	audit
	aws:PrincipalArn:
	arn:aws:iam::22222
	2222222:user/ *Nikki*
		沒有相符項目

政策條件	請求內容	結果

"StringEquals": {
  "aws:PrincipalTag/department": [
  "finance",
  "hr",
  "legal"
  ],
  "aws:PrincipalTag/role": [
  "audit",
  "security"
  ]
},
"ArnLike": {
  "aws:PrincipalArn":
 [
  "arn:aws:iam::222222222222:user/Ana",
  "arn:aws:iam::222222222222:user/Mary"
  ]
}
	aws:PrincipalTag/department: hr
	aws:PrincipalTag/role: *payroll*
	aws:PrincipalArn:
	  arn:aws:iam::222222222222:user/Mary
		沒有相符項目

政策條件	請求內容	結果
"StringEquals": { "aws:PrincipalTag/department": ["finance", "hr", "legal"], "aws:PrincipalTag/role": ["audit", "security"] }, "ArnLike": { "aws:PrincipalArn": ["arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary"] }	請求內容中沒有 aws:PrincipalTag/role 。 aws:PrincipalTag/department: hr aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary	沒有相符項目

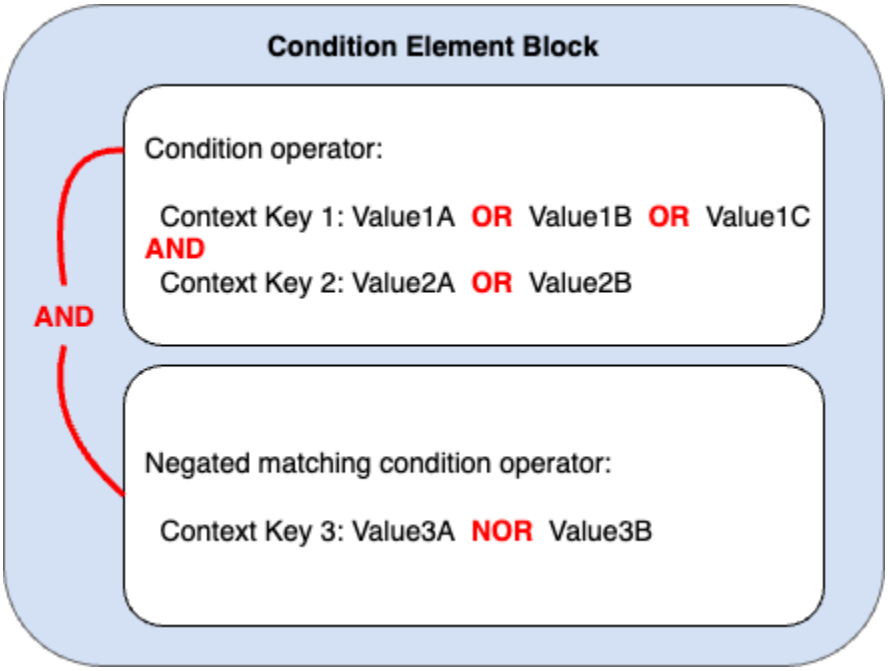
政策條件	請求內容	結果

<pre>"StringEquals": {
  "aws:PrincipalTag/
department": [
    "finance",
    "hr",
    "legal"
  ],
  "aws:PrincipalTag/
role": [
    "audit",
    "security"
  ]
},
"ArnLike": {
  "aws:PrincipalArn":
 [
      "arn:aws:
iam::222222222222:
user/Ana",
      "arn:aws:
iam::222222222222:
user/Mary"
  ]
}</pre>
	請求內容中沒有 aws:PrincipalTag 。
	<pre>aws:PrincipalArn:
	  arn:aws:iam::22222
	2222222:user/Mary</pre>
		沒有相符項目

否定相符條件運算子的評估邏輯

部分條件運算子 (例如 StringNotEquals 或 ArnNotLike) 使用否定相符將政策中的內容索引鍵值對與請求中的內容金鑰值對進行比較。使用否定相符條件運算子為政策中的單一內容索引鍵指定多個值時，有效許可的工作方式類似於邏輯 NOR。在否定相符中，僅當所有值都評估為 false 時，邏輯 NOR 或 NOT OR 才傳回 true。

下圖說明具有多個條件運算子和內容索引鍵值對的條件的計算邏輯。此圖包含內容索引鍵 3 的否定相符條件運算子。



例如，下列 S3 儲存貯體政策說明上圖在政策中的表示方式。條件區塊包含條件運算子 `StringEquals` 和 `ArnNotLike`，以及內容索引鍵 `aws:PrincipalTag` 和 `aws:PrincipalArn`。若要調用所需的 `Allow` 或 `Deny` 效果，條件區塊中的所有內容索引鍵都必須解析為 true。發出請求的用戶必須同時具有部門和角色這兩個主體標籤索引鍵，其中包含政策中指定的其中一個標籤索引鍵值。由於 `ArnNotLike` 條件運算子使用否定相符，因此發出請求的使用者的主體 ARN 必須符合要評估為 true 的政策中指定的任何 `aws:PrincipalArn` 值。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
```

```
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana",`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	相符

政策條件	請求內容	結果
"arn:aws:iam::222222222222:user/Mary"] }		

"StringEquals": {
"aws:PrincipalTag/department": [
"finance",
"hr",
"legal"
],
"aws:PrincipalTag/role": [
"audit",
"security"
]
},
"ArnNotLike": {
"aws:PrincipalArn": [
"arn:aws:iam::222222222222:user/Ana",
"arn:aws:iam::222222222222:user/Mary"
]
}
	aws:PrincipalTag/department: hr
	aws:PrincipalTag/role: audit
	aws:PrincipalArn: arn:aws:iam::222222222222:user/ *Mary*
		沒有相符項目

政策條件	請求內容	結果

"StringEquals": {
  "aws:PrincipalTag/department": [
  "finance",
  "hr",
  "legal"
  ],
  "aws:PrincipalTag/role": [
  "audit",
  "security"
  ]
},
"ArnNotLike": {
  "aws:PrincipalArn": [
  "arn:aws:iam::222222222222:user/Ana",
  "arn:aws:iam::222222222222:user/Mary"
  ]
}
	aws:PrincipalTag/department: hr
	aws:PrincipalTag/role: *payroll*
	aws:PrincipalArn:
	  arn:aws:iam::22222222222222:user/Nikki
		沒有相符項目

政策條件	請求內容	結果

`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`>`
	
	請求內容中沒有 `aws:PrincipalTag/role` 。
	
	`aws:PrincipalTag/department: hr aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`
		沒有相符項目

政策條件	請求內容	結果

<pre>"StringEquals": {
  "aws:PrincipalTag/
department": [
    "finance",
    "hr",
    "legal"
  ],
  "aws:PrincipalTag/
role": [
    "audit",
    "security"
  ]
},
"ArnNotLike": {
  "aws:PrincipalArn":
 [
      "arn:aws:
iam::222222222222:
user/Ana",
      "arn:aws:
iam::222222222222:
user/Mary"
  ]
}</pre>
	請求內容中沒有 aws:PrincipalTag 。
	<pre>aws:PrincipalArn:
	  arn:aws:iam::22222
	2222222:user/Nikki</pre>
		沒有相符項目

單值與多值的內容索引鍵

單值與多值內容索引鍵之間的差異在於請求內容中值的數目，而非政策條件中值的數目。

- 單值條件內容索引鍵在請求內容中最多有一個值。例如，當您在 中標記資源時 AWS，每個資源標籤都會儲存為索引鍵/值對。由於資源標籤金鑰只能有單一標籤值，因此 the section called “ResourceTag” 是單值內容索引鍵。請勿使用具有單值內容索引鍵的條件集運算子。
- 多值條件內容索引鍵在請求內容中可以擁有多個值。例如，當您在 中標記資源時 AWS，您可以在單一請求中包含多個標籤鍵/值對。因此，the section called “TagKeys” 是多值內容索引鍵。多值內容索引鍵需要條件集運算子。

例如，請求可以來自最多一個 VPC 端點，因此 the section called “SourceVpce” 是單值內容索引鍵。由於服務可以有多個屬於該服務的服務主體名稱，因此 the section called “PrincipalServiceNamesList” 是多值內容索引鍵。

> **Important**
>
> 單值與多值內容索引鍵之間的差異取決於請求內容中值的數目，而非政策條件中值的數目。

如需示範具有多個內容索引鍵和值之條件的政策範例，請參閱 條件政策範例。

重點

- 單值和多值分類包含在 AWS 全域條件內容索引鍵 主題中作為值類型的每個條件內容索引鍵的描述中。
- Service Authorization Reference 中的多值內容索引鍵使用 `ArrayOf` 字首，後接條件運算子類別類型 (例如 `ArrayOfString` 或 `ArrayOfARN`)，表示請求可能包含條件內容索引鍵的多個值。
- 可以使用任何可用的單值內容索引鍵作為政策變數，但不能使用多值內容金鑰作為政策變數。如需有關政策變數的詳細資訊，請參閱 IAM 政策元素：變數與標籤。
- 使用包含金鑰/值對的內容索引鍵時，請務必注意，即使可以有多個標籤鍵值，每個 `tag-key` 只能有一個值。
  - the section called “PrincipalTag”、the section called “RequestTag” 和 the section called “ResourceTag” 是單值內容索引鍵。
  - the section called “TagKeys” 定義了請求中允許的標籤索引鍵，但不包含標籤索引鍵值。由於您可以在請求中包含多個標籤索引鍵值對，因此 `aws:TagKeys` 是多值內容索引鍵。
- 多值內容索引鍵需要條件集運算子。請勿將條件集運算子 `ForAllValues` 或 `ForAnyValue` 與單值內容索引鍵搭配使用。使用條件集運算子搭配單值內容索引鍵可能會導致過度寬鬆的政策。

用於多值內容索引鍵的集運算子

若要將您的條件內容索引鍵與具有多值的請求內容索引鍵進行比較，您必須使用 `ForAllValues` 或 `ForAnyValue` 集運算子。這些集運算子用於比較兩組值，例如請求中的標籤集和政策條件中的標籤集。

`ForAllValues` 和 `ForAnyValue` 限定詞將集操作功能新增到了條件運算子，允許您針對政策條件中的多內容索引鍵值來測試具有多值的請求內容索引鍵。此外，如果您在政策中使用萬用字元或變數包含

多值字串內容索引鍵，您也必須使用 StringLike 條件運算子。多個條件索引鍵值必須像陣列那樣用方括號括住，例如 "Key2":["Value2A", "Value2B"]。

ForAllValues

ForAllValues 限定詞會測試請求內容中每個成員的值是否與限定詞後面的條件運算子相符。如果請求中每個內容索引鍵值都與政策中的內容索引鍵值相符，則條件會傳回 true。如果請求中沒有內容索引鍵，也會傳回 true。

> **Important**
>
> 如果將 ForAllValues 與 Allow 效果搭配使用，請務必小心，因為如果請求內容中意外出現遺失內容索引鍵，則可能導致政策過度寬鬆。您應該始終在政策中包含具有 false 值的 Null 條件運算子，以檢查內容索引鍵是否存在且其值不為 Null。如需範例，請參閱 根據標籤索引鍵控制存取權限。

ForAllValues 集運算子範例

在下列範例中，ForAllValues 會與 aws:TagKeys 搭配使用，以允許使用者刪除指派給 EC2 執行個體的特定標籤。此政策僅允許使用者刪除 environment 和 cost-center 標籤。您可以將其單獨刪除，也可以同時刪除。請求中的標籤索引鍵必須與政策中指定的索引鍵完全相符。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                },
                "Null": {
                    "aws:TagKeys": "false"
```

```
                }
            }
        }
    ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - environment`	相符
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - cost-center`	相符
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ]`	`aws:TagKeys: - environment - cost-center`	相符

政策條件	請求內容	結果
}, "Null": { "aws:TagKeys": "false" }		
"ForAllValues:StringEquals": { "aws:TagKeys": ["environment", "cost-center"] }, "Null": { "aws:TagKeys": "false" }	aws:TagKeys: - environment - dept	沒有相符項目
"ForAllValues:StringEquals": { "aws:TagKeys": ["environment", "cost-center"] }, "Null": { "aws:TagKeys": "false" }	請求內容中沒有 aws:TagKeys 。	沒有相符項目

請注意，在最後一個範例中，結果為「無相符項目」，因為 Null 條件檢查會在遺失內容索引鍵時防止相符。這是避免政策過度寬鬆的最佳實務。

ForAnyValue

ForAnyValue 限定詞會測試這組請求內容索引鍵值是否至少有一個成員與政策條件中這組內容索引鍵值的至少一個成員相符。如果請求中任一內容索引鍵值與政策中的任一內容索引鍵值相符，則條件會傳回 true。如果沒有相符的內容索引鍵，或者如果索引鍵不存在，則條件會傳回 false。

> **Important**
>
> 將 ForAnyValue 與 Deny 效果搭配使用時，如果請求中不存在內容索引鍵，政策會評估為無相符項目。為實現一致的行為，請在政策中新增明確的 Null 條件檢查，以驗證內容索引鍵是否存在。如需詳細資訊，請參閱用於檢查條件索引鍵是否存在的條件運算子。

ForAnyValue 集運算子範例

在下列範例中，ForAnyValue 會與 aws:TagKeys 搭配使用，以允許使用者刪除指派給 EC2 執行個體的特定標籤。如果請求中指定的標籤索引鍵包含 environment 或 cost-center，則此政策允許使用者刪除執行個體的標籤。請求可以包含政策中指定標籤索引鍵之外的其他標籤索引鍵，但必須包含至少一個指定索引鍵以符合條件。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                }
            }
        }
    ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment`	相符
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - cost-center`	相符
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - cost-center`	相符
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - dept`	相符
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center"`	`aws:TagKeys: - dept`	沒有相符項目

政策條件	請求內容	結果
] }		

"ForAnyValue:StringEquals": {
"aws:TagKeys": [
"environment",
"cost-center"
]
}
	請求內容中沒有 aws:TagKeys 。	沒有相符項目

條件政策範例

在 IAM 政策中，您可以為單值和多值內容索引鍵指定多個值，以便與請求內容進行比較。下列一組政策範例示範具有多個內容索引鍵和值的政策條件。

> **Note**
>
> 如果您想提交要包含在本參考指南中的政策，請使用此頁面底部的 Feedback (意見回饋) 按鈕。如需以身分為基礎的 IAM 政策範例，請參閱 以身分為基礎的 IAM 政策範例。

條件政策範例：單值內容索引鍵

- 具有單一值內容索引鍵的多個條件區塊
- 具有多個單一值內容索引鍵和值的一個條件區塊

條件政策範例：多值內容索引鍵

- 使用條件集運算子 ForAllValues 拒絕政策
- 使用條件集運算子 ForAnyValue 拒絕政策

多值內容索引鍵範例

下列一組政策範例示範如何使用多值內容索引鍵建立政策條件。

範例：使用條件集運算子 ForAllValues 的拒絕政策

下列範例說明如何使用身分型政策，在請求中包含特定標籤索引鍵字首時，拒絕使用 IAM 標記動作。aws:TagKeys 的值包含用於部分字串比對的萬用字元 (*)。此政策包含具有內容索引鍵 aws:TagKeys 的 ForAllValues 集運算子，因為請求內容索引鍵可以包含多值。為了使內容索引鍵 aws:TagKeys 符合，請求內容中的每個值都必須與政策中的至少一個值相符。

如果請求中沒有內容索引鍵，ForAllValues 集運算子也會傳回 true。

您可以透過在政策中包含一個值為 false 的 Null 條件運算子，來檢查請求中的內容索引鍵是否存在且其值不為 Null，從而防止遺失內容索引鍵或具有空值的內容索引鍵評估為 true。如需詳細資訊，請參閱用於檢查條件索引鍵是否存在的條件運算子。

> **Important**
>
> 此政策不允許任何動作。將此政策與允許特定動作的其他政策結合使用。

Example拒絕多值內容索引鍵的單一政策條件值

在下列範例中，該政策會拒絕請求，其中請求中的 aws:TagKeys 值不包含字首 key1。請求內容可以包含多個值，但由於使用了 ForAllValues 條件集運算子，請求內容中的所有標籤索引鍵值都必須以字首 key1 開頭。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
```

```
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": "key1*"
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。對於拒絕陳述式，「相符」表示「遭拒」，「無相符項目」表示「未遭拒」，且可能由其他陳述式允許。

政策條件	請求內容	結果
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:legal`	沒有相符項目 可能由其他陳述式允許。
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:hr - key1:personnel`	沒有相符項目 可能由其他陳述式允許。
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key2:audit`	相符
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	請求內容中沒有 aws:TagKeys 。	相符

Example拒絕多值內容索引鍵的多個政策條件值

在下列範例中，該政策會拒絕請求，其中請求中的 aws:TagKeys 值不包含字首 key1 或 key2。請求內容可以包含多個值，但由於使用了 ForAllValues 條件集運算子，請求內容中的所有標籤索引鍵值都必須以字首 key1 或 key2 開頭。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*"
          ]
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。對於拒絕陳述式，「相符」表示「遭拒」，「無相符項目」表示「未遭拒」，且可能由其他陳述式允許。

政策條件	請求內容	結果
`"ForAllValues:StringNotLike": {`	`aws:TagKeys:`	沒有相符項目

政策條件	請求內容	結果

<pre>"aws:TagKeys": [
"key1*",
"key2*"
]
}</pre>
	<pre>- key1:legal</pre>	可能由其他陳述式允許。

<pre>"ForAllValues:StringNotLike": {
"aws:TagKeys": [
"key1*",
"key2*"
]
}</pre>
	<pre>aws:TagKeys:
	- key1:hr
	- key1:personnel</pre>
		沒有相符項目
		
		可能由其他陳述式允許。

<pre>"ForAllValues:StringNotLike": {
"aws:TagKeys": [
"key1*",
"key2*"
]
}</pre>
	<pre>aws:TagKeys:
	- key1:hr
	- key2:audit</pre>
		沒有相符項目
		
		可能由其他陳述式允許。

<pre>"ForAllValues:StringNotLike": {
"aws:TagKeys": [
"key1*",
"key2*"
]
}</pre>
	<pre>aws:TagKeys:
	- *key3*:legal</pre>
		相符

<pre>"ForAllValues:StringNotLike": {
"aws:TagKeys": [
"key1*",
"key2*"
]
}</pre>
	請求內容中沒有 aws:TagKeys 。	相符

範例：使用條件集運算子 ForAnyValue 拒絕政策

如果使用政策中指定的其中一個標籤索引鍵 (environment 或 webserver) 標記任何快照，下列以身分為基礎的政策範例將拒絕建立 EC2 執行個體磁碟區的快照。此政策包含具有內容索引鍵 aws:TagKeys 的 ForAnyValue 集運算子，因為請求內容索引鍵可以包含多值。如果您的標記請求包含政策中指定的任何一個標籤索引鍵值，aws:TagKeys 內容索引鍵將傳回 true 以調用拒絕政策效果。

> **Important**
>
> 此政策不允許任何動作。將此政策與允許特定動作的其他政策結合使用。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "ec2:CreateSnapshot",
        "ec2:CreateSnapshots"
      ],
      "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": "webserver"
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。對於拒絕陳述式，「相符」表示「遭拒」，「無相符項目」表示「未遭拒」，且可能由其他陳述式允許。

政策條件	請求內容	結果
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - webserver`	相符
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - environment - webserver - test`	相符
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - environment - test`	沒有相符項目 可能由其他陳述式允許。
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	請求內容中沒有 `aws:TagKeys` 。	沒有相符項目 可能由其他陳述式允許。

單值內容索引鍵政策範例

下列一組政策範例示範如何使用單值內容索引鍵建立政策條件。

範例：具有單值內容索引鍵的多個條件區塊

當條件區塊具有多個條件時，每個條件都有單一內容索引鍵，所有內容索引鍵都必須解析為 true 才能調用所需的 `Allow` 或 `Deny` 效果。當您使用否定相符條件運算子時，會反轉條件值的評估邏輯。

下列範例可讓使用者建立 EC2 磁碟區，並在磁碟區建立期間將標籤套用至磁碟區。請求內容必須包含內容索引鍵 `aws:RequestTag/project` 的值，以及內容索引鍵 `aws:ResourceTag/environment` 的值可以是生產以外的任何值。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

請求內容必須包含專案標籤值，且無法為生產資源建立以調用 Allow 效果。下列 EC2 磁碟區已成功建立，因為專案名稱為 Feature3，資源標籤為 QA。

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
```

```
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

範例：具有多個單值內容索引鍵和值的一個條件區塊

當條件區塊包含多個內容索引鍵，且每個內容索引鍵都具有多個值時，每個內容索引鍵都必須解析為 true，以便至少一個金鑰值能夠調用所需的 `Allow` 或 `Deny` 效果。當您使用否定相符條件運算子時，會反轉內容索引鍵值的評估邏輯。

下列範例可讓使用者在 Amazon Elastic Container Service 叢集上啟動和執行任務。

- 對於 `aws:RequestTag/environment` 內容索引鍵 AND，請求內容必須包含 `production` OR `prod-backup`。
- `ecs:cluster` 內容索引鍵確保任務在 `default1` OR `default2` ARN ECS 叢集上執行。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
```

```
        }
      }
    }
  ]
}
```

下表顯示 如何根據請求中的條件索引鍵值來 AWS 評估此政策。

政策條件	請求內容	結果
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: environment:production ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default1`	匹配
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1",`	`aws:RequestTag: environment:prod-backup ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default2`	匹配

政策條件	請求內容	結果
"arn:aws:ecs:us-east-1:111122223333:cluster/default2"] }		

"StringEquals": {
"aws:RequestTag/environment": [
"production",
"prod-backup"
]
},
"ArnEquals": {
"ecs:cluster": [
"arn:aws:ecs:us-east-1:111122223333:cluster/default1",
"arn:aws:ecs:us-east-1:111122223333:cluster/default2"
]
}
	aws:RequestTag:
	webserver:production
	ecs:cluster:
	arn:aws:ecs:us-east-1:111122223333:cluster/default2
		無相符項目

政策條件	請求內容	結果
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	請求內容中沒有 `aws:RequestTag` 。 `ecs:cluster arn:aws:ecs:us-east-1:111122223333:cluster/default2`	無相符項目

IAM 政策元素：變數與標籤

當您在撰寫政策時不知道資源或條件索引鍵的確切值時，請使用 AWS Identity and Access Management (IAM) 政策變數做為預留位置。

> **Note**
>
> 如果 AWS 無法解析變數，這可能會導致整個陳述式無效。例如，如果您使用 `aws:TokenIssueTime` 變數，此變數只會在請求者使用臨時憑證驗證時解析值 (IAM 角色)。為避免變數造成陳述式無效，請使用 ... IfExists 條件運算子。

主題

- 簡介
- 在政策中使用變數
- 做為政策變數的標籤
- 在此您可以使用政策變數

- 沒有值的政策變數
- 您可以使用適用於政策變數的請求資訊
- 指定預設值
- 如需詳細資訊

簡介

在 IAM 政策中，許多動作允許您為要控制存取的特定資源提供名稱。例如，下列政策可讓使用者為 `marketing` 專案列出、讀取和寫入 S3 儲存貯體 `amzn-s3-demo-bucket` 中的物件。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/marketing/*"]
    }
  ]
}
```

在某些情況下，您在編寫政策時可能不知道資源的確切名稱。您可能想要一般化政策，以使其適用於許多使用者，而不必為每個使用者製作政策的唯一副本。建議您建立適用於該群組中任何使用者的單一群組政策，而不是為每個使用者建立單獨政策。

在政策中使用變數

您可以使用在政策中設定預留位置的政策變數來定義政策內的動態值。

變數使用 **$** 字首後接一對大括號 (**{ }**) 進行標記，其中包含請求中的值的變數名稱。

評估政策時，政策變數將取代為來自請求中傳遞的條件式內容金鑰的值。變數可用於以身分為基礎的政策、資源政策、服務控制政策、工作階段政策以及 VPC 端點政策。用作許可界限的以身分為基礎的政策也支援政策變數。

全域條件內容索引鍵可以用作跨 AWS 服務請求中的變數。與 AWS 資源互動時，服務特定條件索引鍵也可以用作變數，但只有在針對支援其資源提出請求時才可用。如需每個 AWS 服務和資源可用的內容金鑰清單，請參閱服務授權參考。在某些情況下，您無法使用值填入全域條件內容索引鍵。若要進一步了解每個索引鍵，請參閱 AWS 全域條件內容索引鍵。

> **Important**
>
> - 索引鍵名稱不區分大小寫。例如，`aws:CurrentTime` 等同於 `AWS:currenttime`。
> - 您可以使用任何單一值條件鍵作為變數。您無法使用多重值條件鍵做為變數。

下面範例顯示了 IAM 角色或使用者的政策，它用政策變數替換特定資源名稱。您可以利用 `aws:PrincipalTag` 條件索引鍵來重複使用此政策。此政策被評估時，僅在儲存貯體名稱以 `team` 主體標籤中的團隊名稱作為結尾時，`${aws:PrincipalTag/team}` 才會允許該動作。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/team}/*"]
```

```
    }
  ]
}
```

該變數使用 $ 字首標記，後跟一對大括號 ({ })。在 ${ } 字元內，您可以包含要在政策中想要使用的請求中的值的名稱。您可以使用的值將在本頁稍後討論。

如需有關此全域條件索引鍵的詳細資訊，請參閱全域條件金鑰清單中的 aws:PrincipalTag/tag-key。

> **Note**
>
> 若要使用政策變數，必須在陳述式中包含 Version 元素，並且必須將版本設定為支援政策變數的版本。變數已導入版本 2012-10-17。舊版的政策語言不支援政策變數。如果不包含 Version 元素，並將其設定為適當的版本，則 ${aws:username} 等變數將被視為政策中的常值字串。
>
> Version 政策元素與政策版本不同。Version 政策元素是在政策內使用，並定義政策語言的版本。另一方面，政策版本會在您在 IAM 中變更客戶受管政策時建立。變更的政策不會覆寫現有的政策。IAM 反而會建立新版本的受管政策。若要進一步了解 Version 政策元素，請參閱 the section called “Version”。若要進一步了解政策版本，請參閱 the section called “版本控制 IAM 政策”。

允許主體從 S3 儲存貯體的 /David 路徑取得物件之政策如下所示：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/David/*"
      ]
    }
```

```
  ]
}
```

如果此政策附加至使用者 David，該使用者會從其自己的 S3 儲存貯體取得物件，但您必須為包含使用者名稱的每個使用者建立獨立政策。您接著會將每個政策連接到個別使用者。

透過使用政策變數，您可以建立可重複使用的政策。如果 aws:PrincipalTag 的標籤鍵值與請求中傳遞的標籤鍵 owner 值相符，下列政策將允許使用者從 Amazon S3 儲存貯體取得物件。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/owner": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

像這樣使用政策變數來替代使用者時，您不必為每個單獨使用者設定獨立的政策。在下列範例中，政策會附加至由產品經理使用臨時安全憑證擔任的 IAM 角色。使用者提出新增 Amazon S3 物件的請求時，IAM 會將目前請求中的 dept 標籤值替換為 ${aws:PrincipalTag} 變數，並評估該政策。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOnlyDeptS3Prefix",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/dept}/*"
            ]
        }
    ]
}
```

做為政策變數的標籤

在某些 AWS 服務中，您可以將自己的自訂屬性連接到這些服務建立的資源。例如，您可以將標籤套用至 Amazon S3 儲存貯體或 IAM 使用者。這些標籤均為鍵值組。您定義標籤鍵名稱以及與該鍵名稱關聯的值。例如，您可以建立一個具有 **department** 索引鍵和 **Human Resources** 值的標籤。如需有關標記 IAM 實體的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。有關標記其他 AWS 服務建立之資源的資訊，請參閱該服務的文件。如需有關使用標籤編輯器的詳細資訊，請參閱 AWS 管理主控台 使用者指南中的使用標籤編輯器。

您可以標記 IAM 資源以簡化探索、整理和追蹤您的 IAM 資源。您也可以標記 IAM 身分來控制存取資源或標記本身。若要進一步了解有關使用標籤以控制存取的詳細資訊，請參閱使用標籤控制對 IAM 使用者和角色的存取。

在此您可以使用政策變數

您可以在 Resource 元素中使用政策變數，也可以在 Condition 元素中使用字串比較。

資源元素

您可以在 Resource 元素中使用政策變數，但只能在 ARN 的資源部分中使用政策變數。ARN 的這個部分會出現在第 5 個冒號 (:) 之後。您無法使用變數來取代第 5 個冒號之前的 ARN 部分，例如服務或帳戶。如需有關 ARN 格式的詳細資訊，請參閱 IAM ARN。

若要以標籤值取代 ARN 的一部分，請以 ${ } 包住字首與索引鍵名稱。例如，以下資源元素僅指其名稱與請求之使用者的部門標籤值相同的儲存貯體。

```
"Resource": ["arn:aws::s3:::amzn-s3-demo-bucket/
${aws:PrincipalTag/department}"]
```

許多 AWS 資源使用包含使用者建立名稱的 ARNs。下列 IAM 政策可確保只有具有相符 access-project、access-application 和 access-environment 標籤值的特定使用者才能修改其資源。此外，使用 * 萬用字元比對時，它們能允許自訂資源名稱尾碼。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
      ]
    }
  ]
}
```

條件元素

在涉及字串運算子或 ARN 運算子的任何條件下，您都可以將政策變數用於 Condition 值。字串運算子包括 StringEquals、StringLike 和 StringNotLike。ARN 運算子包括 ArnEquals 和 ArnLike。您無法使用政策變數搭配其他運算子，例如 Numeric、Date、Boolean、Binary、IP Address 或 Null 運算子。如需有關條件運算子的詳細資訊，請參閱 IAM JSON 政策元素：條件運算子。

在 Condition 元素表達式中參照標籤時，請使用相關的字首和索引鍵名稱做為條件金鑰。然後，使用您想在條件值中測試的值。

例如，下列政策範例允許完整存取使用者，但僅限在標籤 costCenter 連接至使用者時。此標籤也必須有一個 12345 或 67890 的值。如果此標籤沒有值，或有任何其他值，請求將會失敗。

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

沒有值的政策變數

當政策變數參考的條件內容索引鍵沒有值或不存在於請求授權內容中時，該值實際上是空值。沒有相等或類似的值。在以下情況下，授權內容中可能不存在條件內容索引鍵：

- 您正在對不支援該條件索引鍵的資源請求中使用服務特定條件內容索引鍵。
- IAM 主體、工作階段、資源或請求的標籤不存在。
- 在 AWS 全域條件內容索引鍵 中列出了每個全域條件內容金鑰的其他情況。

當您在 IAM 政策的條件元素中使用沒有值的變數時 (IAM JSON 政策元素：條件運算子，例如 StringEquals 或 StringLike 不匹配)，政策陳述式不會生效。

反向條件運算子 (例如 StringNotEquals 或 StringNotLike) 確實與空值匹配，因為其正在測試的條件索引鍵的值不等於或不類似於有效空值。

在下列範例中，aws:principaltag/Team 必須等於 s3:ExistingObjectTag/Team 才能允許存取。未設定 aws:principaltag/Team 時，系統會明確拒絕存取。如果在授權內容中沒有值的變數用作政策的 Resource 或 NotResource 元素的一部分，則包含沒有值之政策變數的資源將不會與任何資源相匹配。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
      }
    }
  ]
}
```

您可以使用適用於政策變數的請求資訊

您可以使用 JSON 政策的 Condition 元素，來比較請求內容中的金鑰和您在政策中指定的鍵值。當您使用政策變數時， 會從請求內容索引鍵 AWS 取代 值，以取代政策中的 變數。

主體索引鍵值

aws:username、aws:userid 及 aws:PrincipalType 的值取決於啟動請求之主體的類型。例如，請求可能是使用 IAM 使用者、IAM 角色或 AWS 帳戶根使用者的憑證提出。下列清單顯示不同類型之主體的鍵值。

- AWS 帳戶根使用者
  - aws:username：(不存在)
  - aws:userid： AWS 帳戶 ID
  - aws:PrincipalType: Account
- IAM 使用者
  - aws:username：*IAM-user-name*
  - aws:userid：唯一 ID
  - aws:PrincipalType: User
- AWS STS 聯合身分使用者主體

  - aws:username：(不存在)
  - aws:userid：*account*:*caller-specified-name*
  - aws:PrincipalType: FederatedUser
- OIDC 和 SAML 聯合身分主體

> **Note**
>
> 有關使用 OIDC 聯合時可用的政策金鑰的詳細資訊，請參閱 OIDC 聯合身分。

  - aws:username：(不存在)
  - aws:userid：(不存在)
  - aws:PrincipalType: AssumedRole
- 擔任的角色
  - aws:username：(不存在)
  - aws:userid：*role-id*:*caller-specified-role-name*
  - aws:PrincipalType: Assumed role
- 指派給 Amazon EC2 執行個體的角色
  - aws:username：(不存在)
  - aws:userid：*role-id*:*ec2-instance-id*
  - aws:PrincipalType: Assumed role
- 匿名呼叫者 (僅限 Amazon SQS、Amazon SNS 和 Amazon S3)
  - aws:username：(不存在)
  - aws:userid：(不存在)
  - aws:PrincipalType: Anonymous

對於此、清單中的項目，請注意下列事項：

- 「不存在」表示該值不在目前請求資訊中，並且任何符合它的嘗試都會失敗並導致陳述式無效。
- *role-id* 是在建立時指派給每個角色的唯一識別符。您可以使用 AWS CLI 命令顯示角色 ID： `aws iam get-role --role-name` *rolename*
- *caller-specified-name* 和 *caller-specified-role-name* 是呼叫程序 (例如，應用程式或服務) 在呼叫取得暫時憑證時所傳遞的名稱。

- *ec2-instance-id* 是在啟動時指派給執行個體的值，並顯示在 Amazon EC2 主控台的 Instances (執行個體) 頁面上。您也可以執行 AWS CLI 命令來顯示執行個體 ID： `aws ec2 describe-instances`

聯合身分主體在請求中可用的資訊

聯合身分主體是使用 IAM 以外的系統進行身分驗證的使用者。例如，公司可能有內部呼叫 的應用程式以供內部使用 AWS。向使用該應用程式的每個公司使用者提供 IAM 身分可能是不切實際的。相反地，公司可能會使用具有單一 IAM 身分的代理 (中間層) 應用程式，或者公司可能使用 SAML 身分提供者 (IdP)。代理應用程式或 SAML IdP 使用公司網路對個別使用者進行身分驗證。然後，代理應用程式可以使用其 IAM 身分來取得個別使用者的臨時安全憑證。SAML IdP 可以有效地交換 AWS 臨時安全登入資料的身分資訊。然後，臨時登入資料可用於存取 AWS 資源。

同樣，您可以為應用程式需要存取 AWS 資源的行動裝置建立應用程式。在這種情況下，您可以使用 OIDC 聯合，其中應用程式使用 Login with Amazon、Amazon Cognito、Facebook 或 Google 等知名身分提供者對使用者進行身分驗證。然後，應用程式可以使用來自這些提供者的使用者身分驗證資訊來取得用來存取 AWS 資源的臨時安全憑證。

使用 OIDC 聯合的建議方法是利用 Amazon Cognito 和 AWS 行動 SDKs。如需詳細資訊，請參閱下列內容：

- Amazon Cognito 使用者指南
- 暫時性憑證的常見案例

特殊字元

有幾個特殊的預先定義的政策變數具有固定值，使您能夠表示具有特殊含意的字元。如果這些特殊字元是字串的一部分，那麼您正在嘗試比對，並且從字面上插入它們會被轉譯。例如，在字串中插入 * 星號將被轉譯為萬用字元，比對任何字元，而不是做為文字 *。在這些情況下，您可以使用以下預定義的政策變數：

- ${*} - 在需要 * 星號字元的地方使用。
- ${?} - 在需要 ? 問號字元的地方使用。
- ${$} - 在需要 $ 貨幣符號字元的地方使用。

這些預先定義的政策變數可以在可以使用一般政策變數的任何字串中使用。

指定預設值

若要將預設值新增到一個變數，請以單引號 (' ') 括住預設值，並使用逗號和空格 (,) 分隔變數文字和預設值。

例如，如果主體標記為 team=yellow，他們可以存取名為 amzn-s3-demo-bucket-yellow 的 ExampleCorp's Amazon S3 儲存貯體。具有此資源的政策可讓團隊成員存取其團隊儲存貯體，但不能存取其他團隊的儲存貯體。對於沒有團隊標籤的使用者，它會將 company-wide 的預設值設定為儲存貯體名稱。這些使用者只能存取 amzn-s3-demo-bucket-company-wide 儲存貯體，其中他們可以檢視廣泛的資訊，例如加入團隊的指示。

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

如需詳細資訊

如需有關政策的詳細資訊，請參閱以下項目：

- 中的政策和許可 AWS Identity and Access Management
- 以身分為基礎的 IAM 政策範例
- IAM JSON 政策元素參考
- 政策評估邏輯
- OIDC 聯合身分

IAM JSON 政策元素：支援的資料類型

本部分列出在 JSON 政策中指定值時所支援的資料類型。政策語言不支援每個政策元素的所有類型；有關每個元素的相關資訊，請參閱上一個部分。

- Strings
- 編號 (Ints 和 Floats)
- Boolean
- Null
- 清單
- 地圖
- 結構 (只是巢狀地圖)

下表將每種資料類型對應至序列化。請注意，所有政策必須採用 UTF-8。如需 JSON 資料類型的相關資訊，請參閱 RFC 4627。

Type	JSON
字串	字串
整數	Number
Float	Number
Boolean	true false
Null	null
日期	字串遵守 ISO 8601 的 W3C 描述檔
IpAddress	字串遵守 RFC 4632
列出	Array
物件	物件

政策評估邏輯

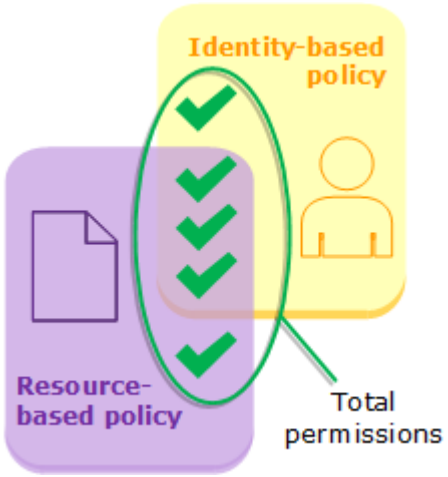
當委託人嘗試使用 AWS 管理主控台、 AWS API 或 時 AWS CLI，該委託人會傳送請求至 AWS。當 AWS 服務收到請求時， AWS 會完成數個步驟，以決定是否允許或拒絕請求。

1. 身分驗證 – 如有必要， AWS 首先驗證提出請求的委託人。有少數幾個服務並不需要這個步驟，例如 Amazon S3，它允許匿名使用者的一些請求。
2. 處理請求內容 – AWS 處理請求中收集的資訊，以判斷哪些政策適用於請求。
3. AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求 – AWS 評估所有政策類型和政策的順序會影響它們的評估方式。 AWS 然後， 會根據請求內容處理政策，以判斷是否允許或拒絕請求。

搭配以資源為基礎的政策來評估以身分為基礎的政策

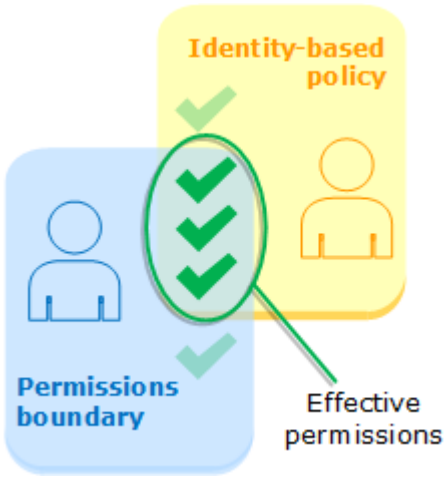
以身分為基礎的政策和以資源為基礎的政策，可為其連接到其中的身分或資源授予許可。當 IAM 實體 （使用者或角色） 請求存取相同帳戶中的資源時， 會 AWS 評估身分型和資源型政策授予的所有

許可。最終產生的許可是這兩種許可的聯合。如果身分型政策、資源型政策或兩者都允許動作，則 會 AWS 允許該動作。在這些政策的明確拒絕會覆寫允許。



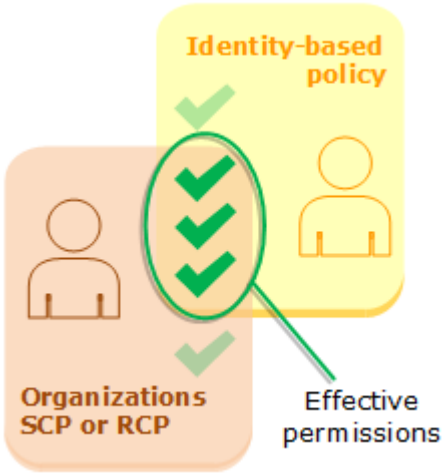
搭配許可界限來評估以身分為基礎的政策

當 AWS 評估使用者的身分型政策和許可界限時，產生的許可是兩個類別的交集。這表示，當您新增許可界限到已有現有以身分為基礎之政策的使用者時，您可能會減少使用者可執行的動作。或者，當您移除使用者的許可界限時，您可能會增加他們可以執行的動作。在這些政策的明確拒絕會覆寫允許。若要檢視有關如何搭配許可界限來評估其他政策類型的詳細資訊，請參閱評估含界限的有效許可。



使用 AWS Organizations SCPs或 RCPs 評估身分型政策

當使用者屬於組織成員帳戶，並存取未設定資源型政策的資源時，產生的許可是使用者政策、服務控制政策 (SCP) 和資源控制政策 (RCP) 的交集。這表示動作必須得到所有三種政策類型的允許。身分型政策、SCP 或 RCP 中的明確拒絕會覆寫允許。



您可以了解您的帳戶是否為 AWS Organizations中組織的成員。組織成員可能會受到 SCP 或 RCP 的影響。若要使用 AWS CLI 命令或 AWS API 操作檢視此資料，您必須擁有 AWS Organizations 實體 `organizations:DescribeOrganization`動作的許可。您必須擁有其他許可，才能在 AWS Organizations 主控台中執行 操作。若要了解 SCP 或 RCP 是否拒絕存取特定請求，或變更您的有效許可，請聯絡您的 AWS Organizations 管理員。

處理請求內容

當 AWS 評估並授權請求時，它會將請求資訊組合成請求內容。請求內容包含任何可用於政策評估的資訊。

- 主體 – 傳送請求的使用者、角色或聯合身分使用者主體。有關主體的資訊，包括與該主體相關聯的政策。
- 動作 – 主體想要執行的一個或多個動作。
- 資源 – 執行動作或操作的一或多個 AWS 資源物件。
- 資源資料 – 與所請求資源相關的資料。這可以包括諸如 DynamoDB 資料表名稱或 Amazon EC2 執行個體上之標籤的資訊。
- 環境資料 – IP 地址、使用者代理程式、啟用 SSL 的狀態或一天中時間的相關資訊。

此資訊會與適用的政策進行比較，以決定是允許還是拒絕請求。您可以使用主體、動作、資源和條件 (PARC) 模型來組織此屬性資訊，以進一步了解如何評估 AWS 政策。

了解 PARC 模型

PARC 模型根據政策語言中的四個 JSON 元素來表示請求內容：

- Principal – 發出請求的實體。主體代表可以進行身分驗證並授權在 AWS 帳戶中執行動作的人類使用者或程式設計式工作負載。
- Action – 正在執行的操作。動作通常會映射至 API 動作。
- Resource – 正在對其執行動作的 AWS 資源。
- Condition – 請求得到允許所必須滿足的其他限制。

下列範例說明 PARC 模型如何表示請求內容：

```
Principal: AIDA123456789EXAMPLE
Action: s3:CreateBucket
Resource: arn:aws:s3:::amzn-s3-demo-bucket1
Context:
- aws:UserId=AIDA123456789EXAMPLE:BobsSession
- aws:PrincipalAccount=123456789012
- aws:PrincipalOrgId=o-example
- aws:PrincipalARN=arn:aws:iam::AIDA123456789EXAMPLE:role/HR
- aws:MultiFactorAuthPresent=true
- aws:CurrentTime=...
- aws:EpochTime=...
- aws:SourceIp=...
- aws:PrincipalTag/dept=123
- aws:PrincipalTag/project=blue
- aws:RequestTag/dept=123
```

請求內容的重要性

理解請求內容及其與政策評估的互動方式，對於下列項目至關重要：

- 對存取問題進行疑難排解
- 設計有效且安全的政策
- 了解政策授予的許可的完整範圍
- 在不同案例中預測政策評估的結果

透過使用 PARC 模型視覺化請求內容，您可以更輕鬆地了解 如何 AWS 更有效地制定授權決策和設計政策。

AWS 如何使用請求內容

評估政策時， 會將請求內容中的資訊與所有適用政策中指定的資訊 AWS 進行比較。這包括身分型政策、資源型政策、IAM 許可界限、Organizations SCP、Organizations RCP 和工作階段政策。

對於每個政策類型， AWS 會使用請求內容來檢查：

- 政策是否根據主體套用至請求。
- 請求的動作是否允許在指定的資源上執行。
- 請求內容是否符合政策中指定的任何條件。

如何 AWS 評估政策取決於套用至請求內容的政策類型。這些政策類型可在單一 AWS 帳戶內使用。如需有關這些政策類型的詳細資訊，請參閱 中的政策和許可 AWS Identity and Access Management。若要了解如何 AWS 評估跨帳戶存取的政策，請參閱 跨帳戶政策評估邏輯。

- AWS Organizations 資源控制政策 RCPs) – AWS Organizations RCPs 指定組織或組織單位 (OU) 中帳戶內資源的可用許可上限。RCPs適用於成員帳戶中的資源，並影響委託人的有效許可，包括 AWS 帳戶根使用者，無論委託人是否屬於您的組織。RCP 不適用於組織管理帳戶中的資源，以及由服務連結角色進行的呼叫。如果 RCP 存在，則只有在 RCP 允許該動作時，身分型和資源型政策向成員帳戶中的資源授予許可才有效。
- AWS Organizations 服務控制政策 SCPs) – AWS Organizations SCPs 指定組織或組織單位 (OU) 中帳戶內主體的可用許可上限。SCPs適用於成員帳戶中的主體，包括每個主體 AWS 帳戶根使用者。如果 SCP 存在，則只有在 SCP 允許該動作時，身分型和資源型政策授予給成員帳戶中主體的許可才有效。唯一的例外是組織管理帳戶中的主體和服務連結角色。
- 資源型政策：資源型政策會為政策中指定的主體授予許可。這些許可會定義主體可以對政策連接於其中的資源做哪些動作。
- 許可界限 – 許可界限是一種功能，負責設定身分型政策可向 IAM 實體 (使用者或角色) 授予的許可上限。當您為實體設定許可界限時，實體只能執行由身分型政策和其許可界限同時允許的動作。在某些情況下，許可界限中的隱含拒絕會限制資源型政策所授予的許可。如需詳細資訊，請參閱AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求。
- 以身分為基礎的政策 – 以身分為基礎的政策會連接到 IAM 身分 (使用者、使用者群組或角色)，並且授予許可給 IAM 實體 (使用者與角色)。如果只有身分型政策適用於請求，則 AWS 會檢查所有這些政策中至少有一個 `Allow`。
- 工作階段政策 – 工作階段政策是在以程式設計方式為角色或聯合身分使用者工作階段建立臨時工作階段時作為參數傳遞的政策。若要以程式設計方式建立角色工作階段，請使用其中一種 `AssumeRole*` API 操作。當您這麼做且傳遞工作階段政策時，所產生工作階段的許可會是 IAM 實

體的身分類型政策和工作階段政策的交集。若要建立聯合身分使用者工作階段，您要使用 IAM 使用者存取金鑰，以程式設計的方式來呼叫 GetFederationToken API 操作。如需詳細資訊，請參閱工作階段政策。

請記住，所有這類政策中的明確拒絕都會覆寫該允許。

> **Note**
>
> AWS Organizations 宣告政策可讓您集中宣告和強制執行整個組織中指定 AWS 服務 所需的組態。由於宣告式政策直接在服務層級套用，因此不會直接影響政策評估請求，也不會包含在請求內容中。如需詳細資訊，請參閱 AWS Organizations 《 使用者指南》中的宣告政策。

使用 PARC 模型的政策評估範例

為了說明請求內容如何與政策評估互動，我們來看一個政策範例：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:CreateBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/dept": "123"
        }
      }
    }
  ]
}
```

在此範例中，該政策僅在請求內容包含的 aws:PrincipalTag/dept 值為 "123"，且資源符合 amzn-s3-demo-bucket1 儲存貯體名稱時，才會允許 CreateBucket 動作。下表說明 AWS 如何使用請求內容來評估此政策並作出授權決策。

政策	請求內容	評估結果

`"StringEquals": { "aws:PrincipalTag/dept": "123" }`	`Principal: AIDA12345 6789EXAMPLE`
	`Action: s3:Create Bucket`
	`Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1`
	`Context: - aws:PrincipalTag/ dept=123`
		相符

`"StringEquals": { "aws:PrincipalTag/dept": "123" }`	`Principal: AIDA12345 6789EXAMPLE`
	`Action: s3: DeleteBuc ket`
	`Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1`
	`Context: - aws:PrincipalTag/ dept=123`
		沒有相符項目

`"StringEquals": { "aws:PrincipalTag/dept": "123" }`	`Principal: AIDA12345 6789EXAMPLE`
	`Action: s3:Create Bucket`
	`Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1`
	`Context: - aws:PrincipalTag/ dept= 321`
		沒有相符項目

`"StringEquals": { "aws:PrincipalTag/dept": "123" }`	`Principal: AIDA12345 6789EXAMPLE` `Action: s3:Create Bucket`	沒有相符項目

政策	請求內容	評估結果

	`Resource: arn:aws:s3:::amzn-s3-demo-bucket1`
	`Context:`
	
	請求中沒有 `aws:PrincipalTag` 。	

AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求

AWS 強制執行程式碼會決定是否 AWS 應該允許或拒絕傳送至 的請求。 會 AWS 評估適用於請求內容的所有政策。以下是 AWS 政策評估邏輯的摘要。

- 根據預設，所有的要求會一律拒絕 (AWS 帳戶根使用者提出的要求例外，該使用者具有完整存取權)。
- 允許以下評估邏輯後，請求必須得到某個政策或一組政策的明確允許。
- 明確拒絕會覆寫明確允許。

政策評估可能會因請求是在單一帳戶內還是跨帳戶而有所不同。如需有關如何為單一帳戶內的 IAM 角色或使用者作出政策評估決策的詳細資訊，請參閱單一帳戶中請求的政策評估。如需有關如何針對跨帳戶請求作出政策評估決策的詳細資訊，請參閱跨帳戶政策評估邏輯。

- 拒絕評估 – 根據預設，所有的請求一律拒絕。這稱為隱含拒絕。 AWS 強制執行程式碼會評估帳戶內套用至請求的所有政策。這些包括 AWS Organizations SCPs和 RCPs、資源型政策、身分型政策、IAM 許可界限和工作階段政策。在所有這些政策中，強制執行程式碼會尋找套用到請求的 `Deny` 陳述式。此稱為明確拒絕。如果強制執行程式碼找到一個適用的明確拒絕，則強制執行程式碼會傳回拒絕這一最後決定。如果沒有明確拒絕，該強制執行程式碼評估會持續執行。
- AWS Organizations RCPs – 強制執行程式碼會評估套用至請求 AWS Organizations 的資源控制政策 (RCPs)。RCP 適用於 RCP 所連接的帳戶資源。如果強制執行程式碼在 RCP 中找不到任何適用的 `Allow` 陳述式，則強制執行程式碼會傳回拒絕這一最終決定。請注意，啟用 RCP 時，會自動建立名為 AWS 的 `RCPFullAWSAccess` 受管政策並連接到組織中的每個實體，包括根實體、每個 OU 以及 AWS 帳戶 。`RCPFullAWSAccess` 無法分離，因此始終會有 `Allow` 陳述式。如果 RCP 不存在，或 RCP 允許請求的動作，強制執行程式碼評估就會繼續執行。

- AWS Organizations SCPs – 強制執行程式碼會評估套用至請求 AWS Organizations 的服務控制政策 (SCPs)。SCP 會套用至連接 SCP 的帳戶之主體。如果強制執行程式碼在 SCP 中找不到任何適用的 `Allow` 陳述式，則強制執行程式碼會傳回拒絕這一最終決定。如果 SCP 不存在，或是 SCP 允許請求動作，強制執行程式碼評估就會繼續執行。
- 資源型政策 – 在同一帳戶內，資源型政策會根據存取資源的主體類型以及資源型政策中允許的主體，以不同的方式影響政策評估。根據主體類型，資源型政策中的 `Allow` 可能會導致 `Allow` 的最終決定，即使身分型識別政策、許可界限或工作階段政策中有隱含拒絕。

  針對大部分資源，您只需要在身分型政策或資源型政策中對主體提供明確 `Allow`，即可授予存取權。IAM 角色信任政策和 KMS 金鑰政策是此邏輯的例外，因為這些政策必須明確允許存取主體。對於 IAM 和 AWS KMS 以外的服務，資源型政策可能也需要在同一帳戶內具有明確的 `Allow` 陳述式才能授予存取權。如需詳細資訊，請參閱適用於您所使用之特定服務的文件。

  對於單一帳戶政策評估請求，如果指定的主體是 IAM 使用者、IAM 角色或工作階段主體，資源型政策邏輯會與其他政策類型有所不同。工作階段主體包括 IAM 角色工作階段或 AWS STS 聯合身分使用者主體。如果資源型政策直接將許可授予 IAM 使用者或提出要求的工作階段主體，則身分識別型政策、許可界限或工作階段政策中的隱含拒絕不會影響最終決定。

  - IAM 角色 – 將許可授予 IAM 角色 ARN 的資源型政策受限於許可界限或工作階段政策中隱含拒絕的限制。您可以在主體元素或 `aws:PrincipalArn` 條件索引鍵中指定角色 ARN。在這兩種情況下，發出請求的主體是 IAM 角色工作階段。

    許可界限和工作階段政策不會限制在主體元素中使用 `aws:PrincipalArn` 條件索引鍵與萬用字元 (*) 所授予的許可，除非身分型政策包含明確拒絕。如需詳細資訊，請參閱IAM 角色主體。

    角色 ARN 範例

    ```
    arn:aws:iam::111122223333:role/examplerole
    ```

  - IAM 角色工作階段 – 在同一帳戶內，將許可授予 IAM 角色工作階段 ARN 的資源型政策直接授予許可給擔任的角色工作階段。直接授予工作階段的許可不會受到身分識別型政策、許可界限或工作階段政策中隱含拒絕的限制。當您擔任角色並提出要求時，提出要求的主體是 IAM 角色工作階段 ARN，而不是角色本身的 ARN。如需詳細資訊，請參閱角色工作階段主體。

    角色工作階段 ARN 範例

    ```
    arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
    ```

  - IAM 使用者 – 在同一帳戶內，將許可授予 IAM 使用者 ARN (非聯合身分使用者工作階段) 的資源型政策不受身分識別型政策或許可界限中隱含拒絕的限制。

IAM 使用者 ARN 範例

```
arn:aws:iam::111122223333:user/exampleuser
```

- AWS STS 聯合身分使用者主體 – 聯合身分使用者工作階段是透過呼叫 建立的工作階段GetFederationToken。當聯合身分使用者提出要求時，提出要求的主體是聯合身分使用者 ARN，而不是聯合身分 IAM 使用者的 ARN。在相同的帳戶中，授予許可給聯合身分使用者 ARN 的資源型政策會直接將許可授予工作階段。直接授予工作階段的許可不會受到身分識別型政策、許可界限或工作階段政策中隱含拒絕的限制。

  但是，如果資源型政策向聯合身分 IAM 使用者 ARN 授予許可，則聯合身分使用者在工作階段期間提出的要求會受到許可界限或工作階段政策中隱含拒絕的限制。

  聯合身分使用者工作階段 ARN 範例

  ```
  arn:aws:sts::111122223333:federated-user/exampleuser
  ```

- 身分型政策：強制執行程式碼會檢查主體的身分型政策。若為 IAM 使用者，這些政策會包含使用者政策，以及使用者所屬群組的政策。如果沒有任何身分型政策或其中的任何陳述式允許請求的動作，則表示請求已遭隱含拒絕，且強制執行程式碼會傳回拒絕這一最終決定。如果任何適用的身分型政策中有任一陳述式允許請求的動作，則程式碼評估會繼續執行。
- IAM 許可界限：強制執行程式碼會檢查主體所使用的 IAM 實體是否有許可界限。如果用來設定許可界限的政策不允許該請求動作，則表示請求已遭隱含拒絕。強制執行程式碼傳回拒絕的最後決定。如果許可界限不存在，或是許可界限允許請求的動作，程式碼評估就會繼續執行。
- 工作階段政策：強制執行程式碼會檢查主體是否為工作階段主體。工作階段主體包括 IAM 角色工作階段或 AWS STS 聯合身分使用者工作階段。如果主體不是工作階段主體，則強制執行程式碼會傳回允許的最終決定。

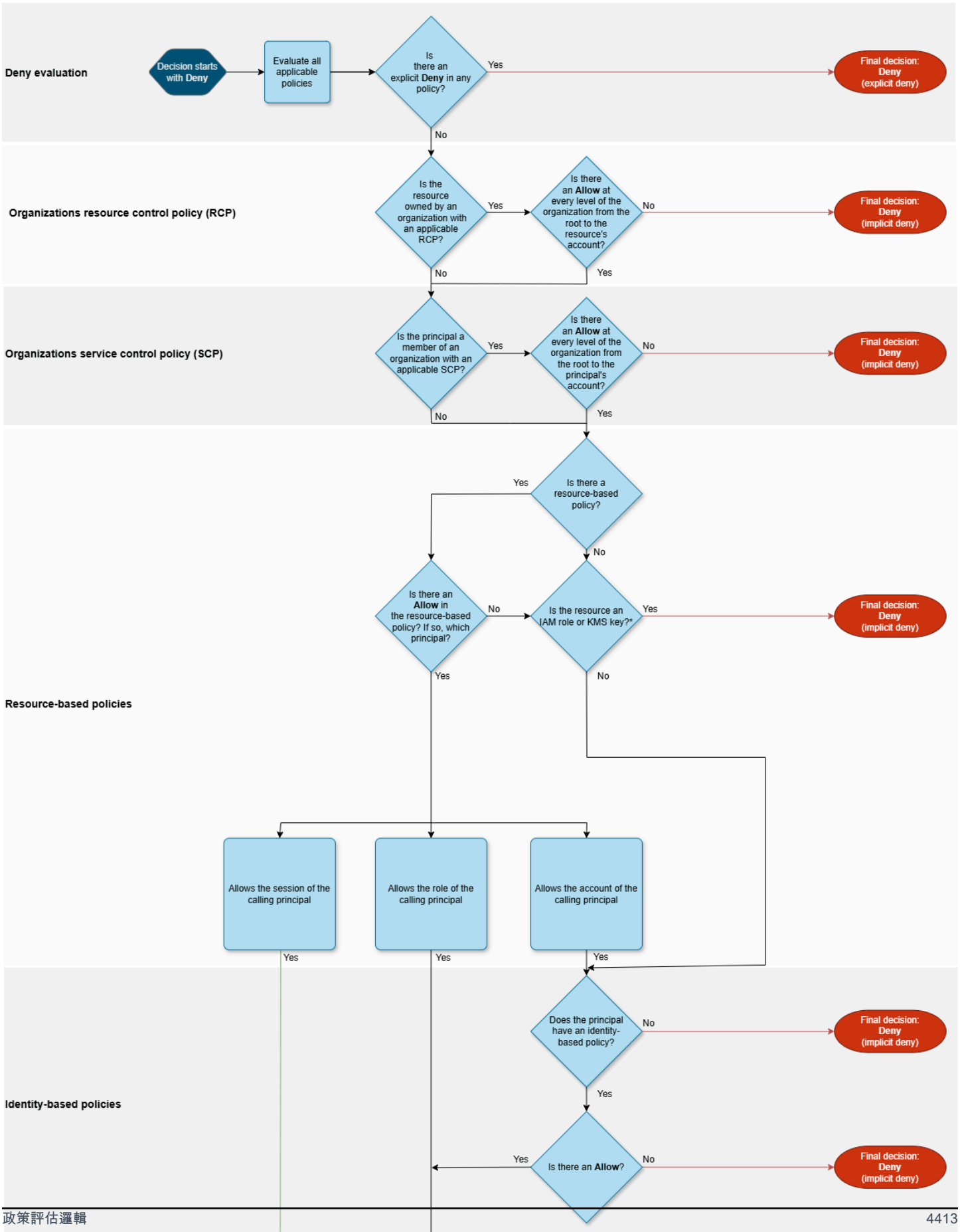
  對於工作階段主體，強制執行程式碼會檢查工作階段政策是否已在請求中傳遞。您可以在使用 AWS CLI 或 AWS API 取得角色或 AWS STS 聯合身分使用者主體的臨時登入資料時，傳遞工作階段政策。如果未傳遞工作階段政策，則會建立預設工作階段政策，且強制執行程式碼會傳回允許這一最終決定。

  - 如果工作階段政策存在，且不允許該要求動作，則表示要求已遭隱含拒絕。強制執行程式碼傳回拒絕的最後決定。
  - 強制執行程式碼會檢查主體是否為角色工作階段。如果主體是角色工作階段，則請求將為已允許。否則，會隱含拒絕請求，強制執行程式碼會傳回拒絕這一最終決定。
  - 如果有工作階段政策，且政策允許該要求動作，則強制執行程式碼會傳回允許的最終決定。

單一帳戶中請求的政策評估

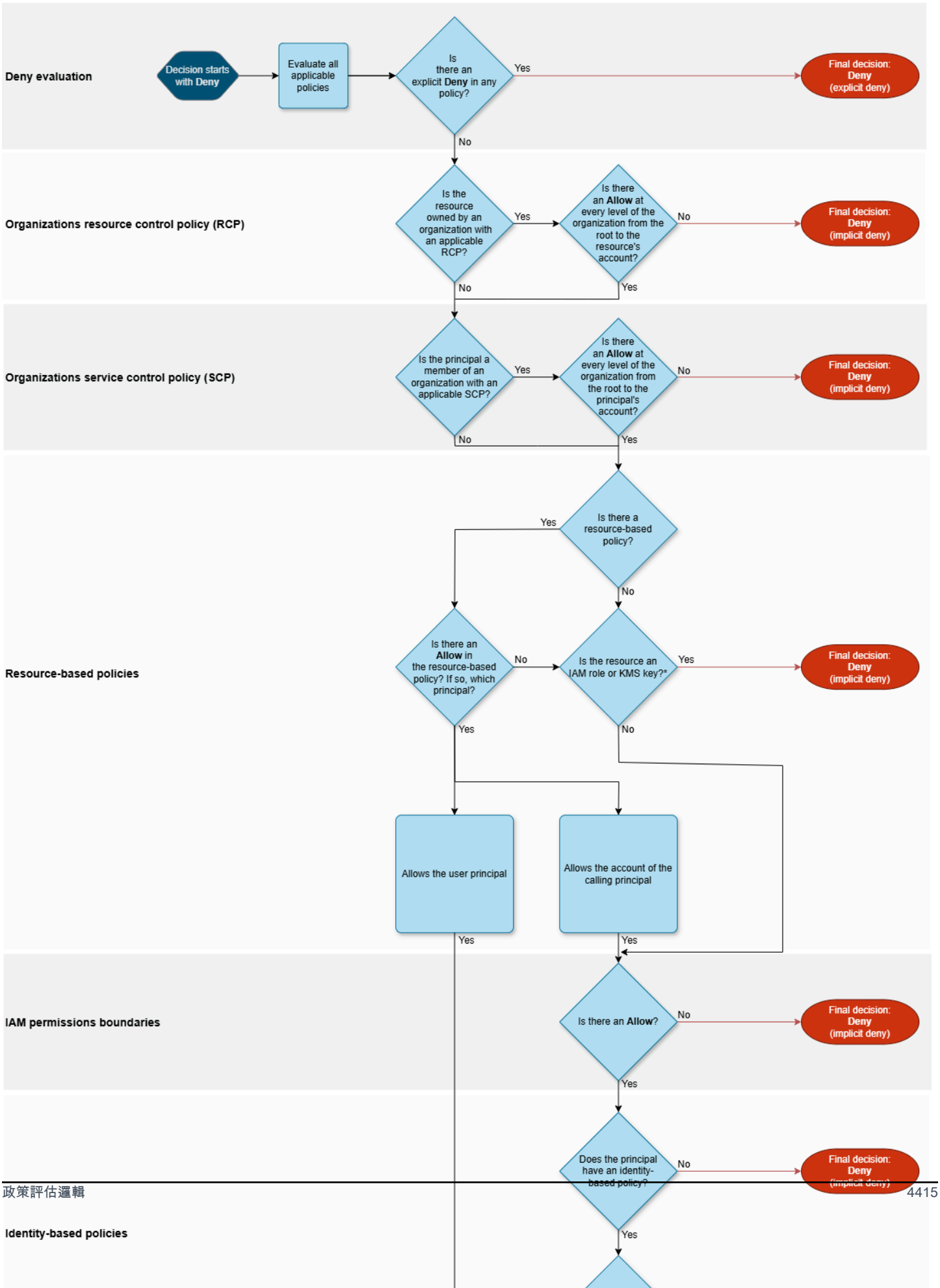
IAM 角色的政策評估

下列流程圖詳細說明如何針對單一帳戶內的 IAM 角色作出政策評估決策。


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Is there a resource-based policy?
Yes
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the session of the calling principal
Allows the role of the calling principal
Allows the account of the calling principal
Yes
Yes
Yes
Identity-based policies
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes
Yes
Is there an Allow?
No
Final decision: Deny (implicit deny)
IAM permissions boundaries
Does the principal have a permissions
Yes
Is there an Allow?
No
Final decision: Deny

IAM 使用者的政策評估

下列流程圖詳細說明如何針對單一帳戶內的 IAM 使用者作出政策評估決策。


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Is there a resource-based policy?
Yes
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the user principal
Allows the account of the calling principal
Yes
Yes
IAM permissions boundaries
Is there an Allow?
No
Final decision: Deny (implicit deny)
Yes
Identity-based policies
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes

以身分為基礎和以資源為基礎的政策的範例

最常見的政策類型是以身分為基礎的政策和以資源為基礎的政策。請求存取資源時， 會 AWS 評估相同帳戶中至少一個允許之政策授予的所有許可。所有政策中的明確拒絕都會覆寫該允許。

> **Important**
>
> 如果在相同帳戶中，身分型政策和資源型政策中的任意一項政策允許請求而另一項不允許，則請求仍將被允許。

假設 Carlos 有使用者名稱 `carlossalazar`，他嘗試儲存檔案到 `amzn-s3-demo-bucket-carlossalazar-logs` Amazon S3 儲存貯體。

另外，也假設以下政策已連接到 `carlossalazar` IAM 使用者。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        },
```

```
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

此政策中的 AllowS3ListRead 陳述式允許 Carlos 檢視帳戶中的所有儲存貯體的清單。AllowS3Self 陳述式允許 Carlos 完整存取名稱和他的使用者名稱完全相同的儲存貯體。DenyS3Logs 陳述式拒絕 Carlos 存取名稱中包含 log 的任何 S3 儲存貯體。

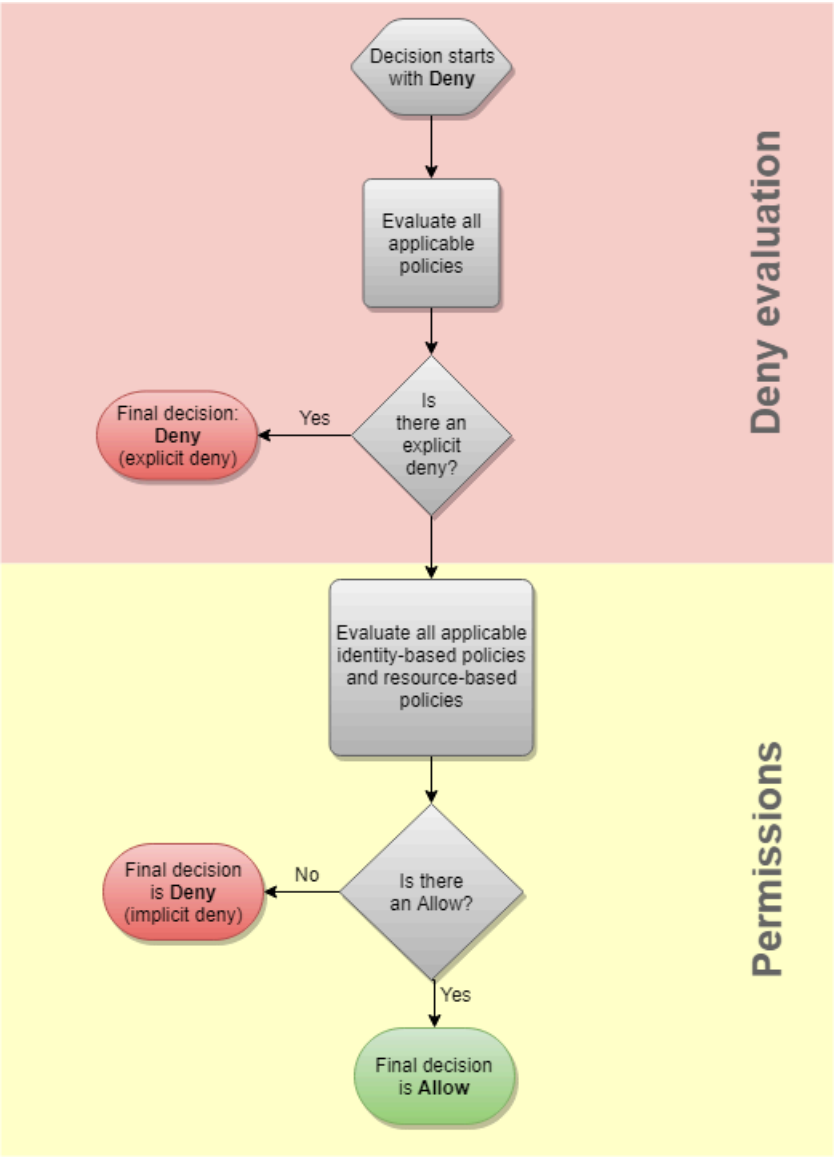
此外，以下以資源為基礎的政策 (稱為儲存貯體政策) 已連接到 amzn-s3-demo-bucket-carlossalazar 儲存貯體。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        }
    ]
}
```

此政策指定只有 carlossalazar 使用者可以存取 amzn-s3-demo-bucket-carlossalazar 儲存貯體。

當 Carlos 提出將檔案儲存至儲存amzn-s3-demo-bucket-carlossalazar-logs貯體的請求時，AWS 會決定哪些政策適用於請求。在這種情況下，只有以身分為基礎的政策和以資源為基礎的政策適用。這兩者都是許可政策。由於未套用許可界限，評估邏輯會降低到以下邏輯。



AWS 會先檢查套用到請求內容的Deny陳述式。它找到一個，因為以身分為基礎的政策明確拒絕 Carlos 存取任何用於記錄的 S3 儲存貯體。Carlos 存取遭拒。

假設他接著發現自己的錯誤，並嘗試將檔案儲存至儲存amzn-s3-demo-bucket-carlossalazar貯體。 會 AWS 檢查陳述式，但找不到Deny陳述式。接著檢查許可政策。身分類型政策和資源類型政策都允許請求。因此， AWS 允許請求。如果其中一個元素明確拒絕陳述式、請求會被拒絕。如果其中一種政策類型允許請求，但另一種則不允許，則仍會允許請求。

跨帳戶政策評估邏輯

您可以允許一個帳戶中的主體存取第二個帳戶中的資源。這稱為跨帳戶存取。當您允許跨帳戶存取時，主體所在的帳戶稱為受信任帳戶。資源所在的帳戶是信任帳戶。

若要允許跨帳戶存取，請將以資源為基礎的政策連接至您要共用的資源。您也必須將身分型政策連接至在請求中扮演主體的身分。信任帳戶中的資源型政策，必須指定具有資源存取權的受信任帳戶主體。您可以指定整個帳戶或其 IAM 使用者、 AWS STS 聯合身分使用者主體、IAM 角色或擔任角色的工作階段。您也可以將 AWS 服務指定為委託人。如需詳細資訊，請參閱如何指定主體。

主體的以身分為基礎的政策，必須允許要求存取信任服務中的資源。可以指定資源的 ARN 來執行此操作。

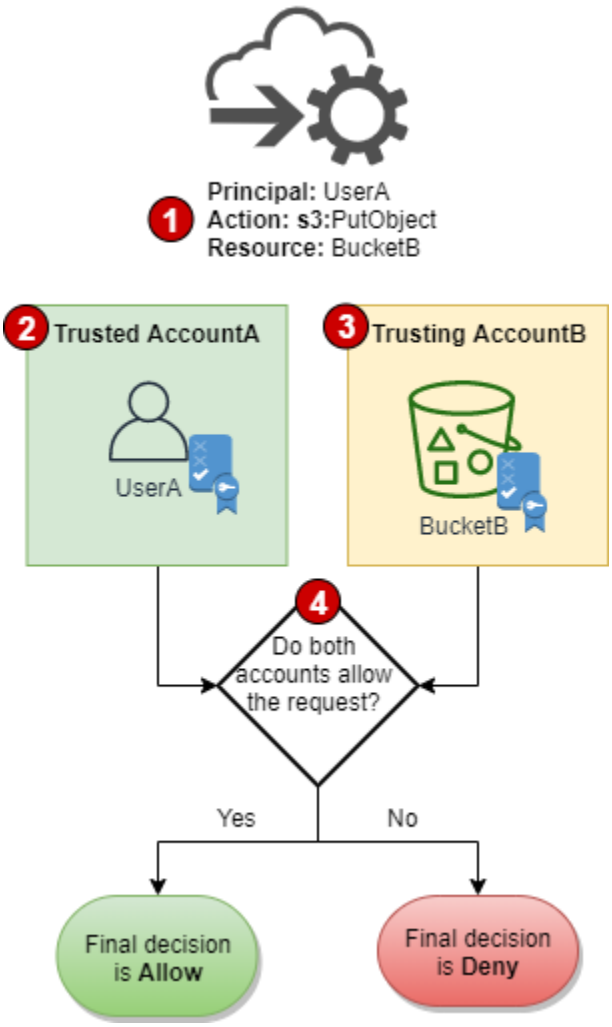
在 IAM 中，您可以將以資源為基礎的政策連接至 IAM 角色，以允許其他帳戶中的主體擔任該角色。角色的以資源為基礎的政策稱為角色信任政策。擔任該角色之後，允許的主體可以使用產生的臨時憑證，以存取您帳戶中的多個資源。此存取是在角色的以身分為基礎的許可政策中定義。若要了解使用角色允許跨帳戶存取，與使用其他以資源為基礎的政策允許跨帳戶存取有何不同，請參閱 IAM 中的跨帳戶資源存取。

> **Important**
>
> 其他服務會影響政策評估邏輯。例如， AWS Organizations 支援可套用至一或多個帳戶中主體和資源的服務控制政策和資源控制政策。 AWS Resource Access Manager 支援政策片段，以控制允許主體在與其共用的資源上執行的動作。 https://docs.aws.amazon.com/ram/latest/userguide/permissions.html

決定是否允許跨帳戶請求

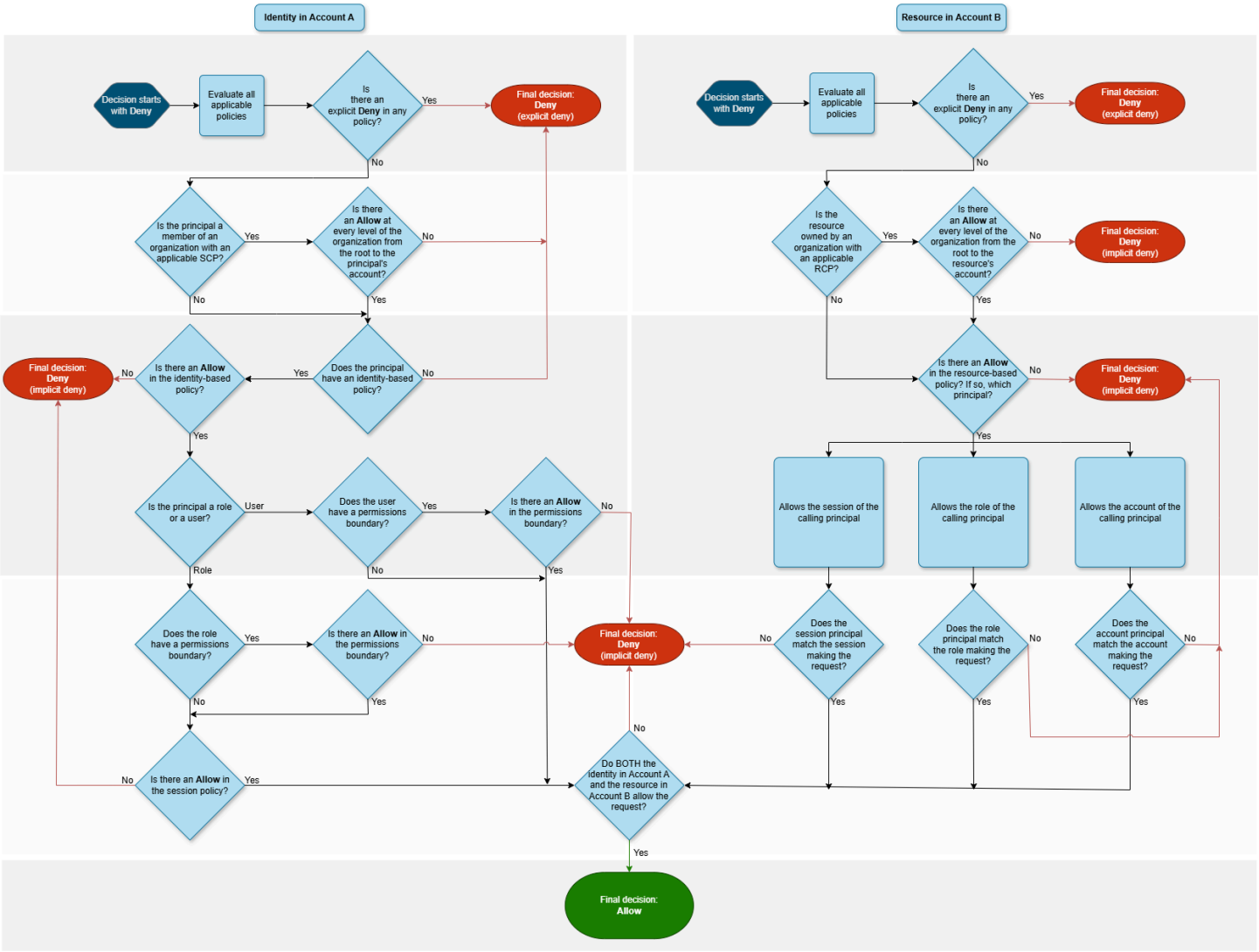
對於跨帳戶請求，受信任 `AccountA` 中的請求者必須具有以身分為基礎的政策。該政策必須允許他們對信任 `AccountB` 中的資源提出請求。此外，`AccountB` 中以資源為基礎的政策，必須允許 `AccountA` 中的請求者存取資源。

當您提出跨帳戶請求時， 會 AWS 執行兩項評估。 會 AWS 評估信任帳戶和信任帳戶中的請求。如需有關如何在單一帳戶內評估請求的詳細資訊，請參閱 AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求。只有當兩項評估都傳回 `Allow` 決定時，才允許此請求。



1. 當一個帳戶中的主體請求存取另一個帳戶中的資源時，就稱為跨帳戶請求。
2. 提出請求的主體存在於受信任帳戶中 (AccountA)。 AWS 評估此帳戶時會檢查以身分為基礎的政策，以及可限制以身分為基礎的政策的任何政策。如需詳細資訊，請參閱搭配許可界限來評估以身分為基礎的政策。
3. 所請求的資源存在於信任帳戶中 (AccountB)。 AWS 評估此帳戶時會檢查連接至所請求資源的以資源為基礎的政策，以及可限制以資源為基礎的政策的任何政策。如需詳細資訊，請參閱搭配以資源為基礎的政策來評估以身分為基礎的政策。
4. AWS 只有在兩個帳戶政策評估都允許請求時， 才會允許請求。

下列流程圖更詳細地說明如何針對跨帳戶請求作出政策評估決策。同樣地，只有在兩個帳戶政策評估都 AWS 允許請求時， 才會允許請求。



跨帳戶政策評估範例

下列範例示範一個帳戶中的角色由第二個帳戶中資源型政策授予許可的案例。

假設 Carlos 是開發人員，而且他在帳戶 111111111111 中是名為 Demo 的 IAM 角色。他想將檔案儲存至帳戶 222222222222 中的 `amzn-s3-demo-bucket-production-logs` Amazon S3 儲存貯體。

另外，也假設以下政策已連接到 Demo IAM 角色。

JSON

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

```
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

此政策中的 AllowS3ListRead 陳述式允許 Carlos 檢視 Amazon S3 中所有儲存貯體的清單。AllowS3ProductionObjectActions 陳述式允許 Carlos 完整存取 amzn-s3-demo-bucket-production 儲存貯體中的物件。

此外，以下以資源為基礎的政策 (稱為儲存貯體政策) 已連接至帳戶 222222222222 中的 amzn-s3-demo-bucket-production 儲存貯體。

JSON

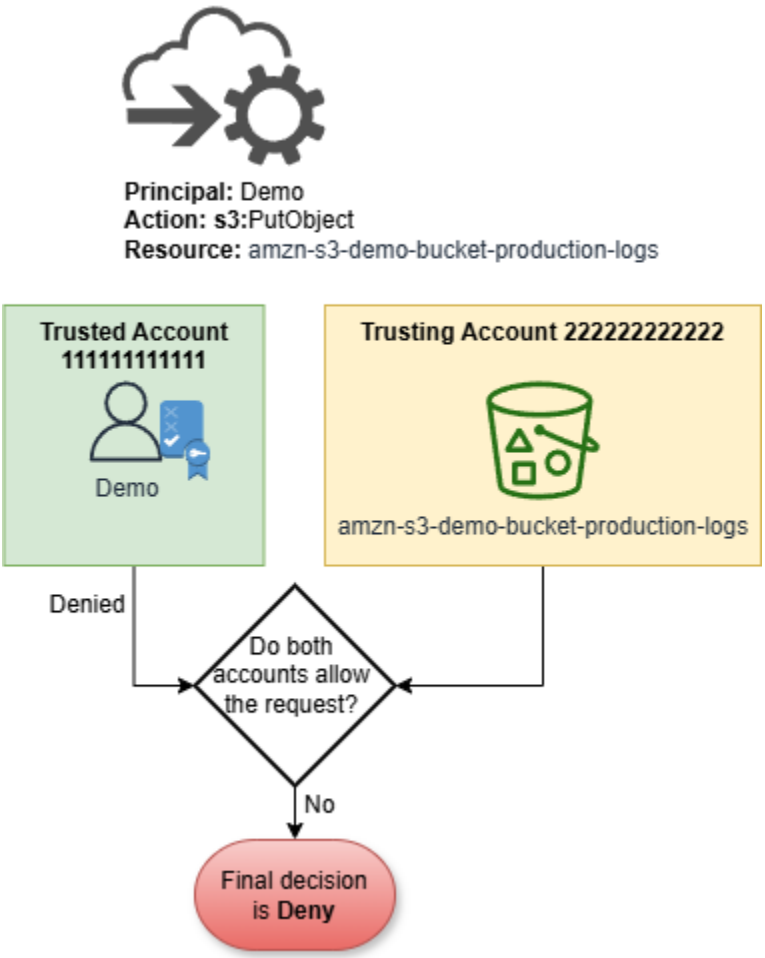
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
```

```
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:role/Demo" },
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        }
    ]
}
```

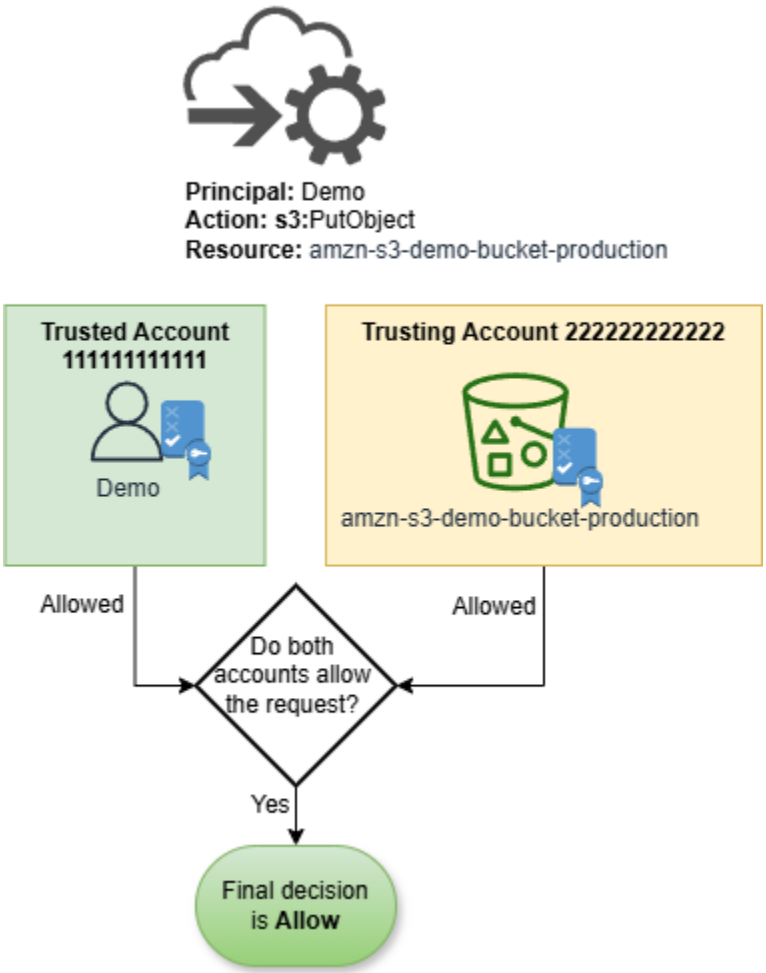
此政策允許 Demo 角色存取 amzn-s3-demo-bucket-production 儲存貯體中的物件。角色可建立與編輯儲存貯體中的物件，但不能刪除。角色無法管理儲存貯體本身。

當 Carlos 提出將檔案儲存至儲存amzn-s3-demo-bucket-production-logs貯體的請求時， AWS 會決定哪些政策適用於請求。在此情況下，連接至 Demo 角色的身分型政策是帳戶 111111111111 中唯一套用的政策。在帳戶 222222222222 中，沒有任何以資源為基礎的政策連接至 amzn-s3-demo-bucket-production-logs 儲存貯體。當 AWS 評估帳戶 時111111111111，會傳回 的決策Deny。這是因為以身分為基礎的政策中的 DenyS3Logs 陳述式，明確拒絕存取任何日誌儲存貯體。如需有關如何在單一帳戶內評估請求的詳細資訊，請參閱 AWS 強制執行程式碼邏輯如何評估允許或拒絕存取的請求。

因為其中一個帳戶內明確拒絕請求，所以最終決定是拒絕請求。



假設 Carlos 接著發現他的錯誤，並嘗試將檔案儲存到 `Production`儲存貯體。 AWS 首先 會檢查帳戶111111111111以判斷是否允許請求。只有身分型政策適用，且允許 request. AWS then 檢查帳戶222222222222。只有連接至 `Production` 儲存貯體的以資源為基礎的政策才會套用，並允許請求。因為這兩個帳戶都允許請求，所以最終決定是允許請求。



明確和隱含拒絕之間的差異

如果適用的請求包含 Deny 陳述式，請求會導致明確拒絕。如果套用到請求的政策包含 Allow 陳述式和 Deny 陳述式，則 Deny 陳述式勝過 Allow 陳述式。請求明確遭拒。

如果沒有適用的 Deny 陳述式，也沒有適用的 Allow 陳述式，則會發生暗中拒絕。由於預設拒絕存取 IAM 主體，他們必須明確獲允許執行動作。否則，會暗中拒絕他們存取。

當您設計授權策略時，您必須建立具 Allow 陳述式的政策，讓您的主體成功發出請求。不過，您可以選擇明確和暗中拒絕的任意組合。

例如，您可以建立下列政策，其中包含允許的動作、隱含拒絕的動作以及明確拒絕的動作。此 AllowGetList 陳述式允許對以字首 Get 和 List 開頭的 IAM 動作的唯讀存取。IAM 中的所有其他動作，例如 iam:CreatePolicy，則被隱含拒絕。此 DenyReports 陳述式明確拒絕存取 IAM 報告，方法是拒絕存取包含 Report 尾碼的動作，例如 iam:GetOrganizationsAccessReport。如果有人向此主體新增其他政策，以授予他們對 IAM 報

告的存取權，例如 `iam:GenerateCredentialReport`，則由於存在此明確拒絕，報告相關的要求仍然被拒絕。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenyReports",
            "Effect": "Deny",
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

IAM JSON 政策語言的文法

此頁面介紹用來在 IAM 中建立 JSON 政策語言的正式文法。我們展示此文法，以便您可以了解如何建構和驗證政策。

如需政策的範例，請參閱下列主題：

- 中的政策和許可 AWS Identity and Access Management
- 以身分為基礎的 IAM 政策範例
- 使用 Amazon EC2 主控台的範例政策，以及 Amazon Amazon EC2 使用者指南中的使用 AWS CLI、Amazon EC2 CLI 或 AWS SDK 的範例政策。 Amazon EC2
- Amazon Simple Storage Service 使用者指南中的儲存貯體政策範例和使用者政策範例。

如需其他 AWS 服務中使用的政策範例，請前往這些服務的文件。

主題

- 政策語言和 JSON
- 此文法中使用的慣例
- 文法
- 政策文法備註

政策語言和 JSON

以 JSON 表達的政策。當您建立或編輯 JSON 政策時，IAM 可以執行政策驗證以協助您建立有效的政策。IAM 會識別 JSON 語法錯誤，而 IAM Access Analyzer 會提供額外的政策檢查及建議，協助您進一步改良政策。若要進一步了解政策驗證的資訊，請參閱 IAM 政策驗證。若要進一步了解 IAM Access Analyzer 政策檢查和可動作的建議，請參閱 IAM Access Analyzer 政策驗證。

在本文件中，我們不提供哪些資料構成有效 JSON 的完整說明。不過，以下是一些基本 JSON 規則：

- 個別實體間允許空格。
- 值以雙引號括住。雙引號對於數字和布林值為選用。
- 許多元素 (例如，`action_string_list` 和 `resource_string_list`) 可將 JSON 陣列當作值。陣列可以使用一個或多個值。如果包含多個值，陣列會在方括號 ([和]) 中並以逗號分隔，如以下範例所示：

  ```
  "Action" : ["ec2:Describe*","ec2:List*"]
  ```
- 基本 JSON 資料類型 (布林值、數字及字串) 是在 RFC 7159 中定義。

此文法中使用的慣例

此文法中使用以下慣例：

- 下列字元是 JSON 權杖，並且「包含」在政策中：

  ```
  { } [ ] " , :
  ```
- 下列字元在文法中為特殊字元，並且「不」包含在政策中：

  ```
  = < > ( ) |
  ```
- 如果某個元素允許多個值，它會使用重複值、逗號分隔符號和省略符號 (`...`) 來表示。範例：

```
[<action_string>, <action_string>, ...]

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
```

如果允許多個值，只包含一個值也屬有效。對於僅有一個值，必須省略尾端逗號。如果元素使用陣列 (以 [和] 標註)，但只包含一個值，則括號為選用。範例：

```
"Action": [<action_string>]

"Action": <action_string>
```

- 某個元素後的問號 (?) 表示該元素為選用。範例：

```
<version_block?>
```

不過，有關選用元素的詳細資訊，請務必參照文法清單後的注意事項。

- 元素之間的垂直線 (|) 表示替代方案。在文法中，括號定義替代方案的範圍。範例：

```
("Principal" | "NotPrincipal")
```

- 必須為文字字串的元素會加上雙引號 (")。範例：

```
<version_block> = "Version" : ("2008-10-17" | "2012-10-17" )
```

如需其他注意事項，請參閱文法清單後的政策文法備註。

文法

以下清單說明政策語言文法。如需清單中所使用的慣例，請參閱上一節。如需其他資訊，請參閱後續的注意事項。

> **Note**
>
> 此文法描述以版本 `2008-10-17` 和 `2012-10-17` 標記的政策。`Version` 政策元素與政策版本不同。`Version` 政策元素是在政策內使用，並定義政策語言的版本。另一方面，政策版本會在您在 IAM 中變更客戶受管政策時建立。變更的政策不會覆寫現有的政策。IAM 反而會建立新版本的受管政策。若要進一步了解 `Version` 政策元素，請參閱 IAM JSON 政策元素：Version。若要進一步了解政策版本，請參閱 the section called “版本控制 IAM 政策”。

```
policy  = {
```

```
     <version_block?>,
     <id_block?>,
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17"        | "2012-10-17"       )

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
    <condition_block?>
}

<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | <action_string> | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
```

```
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

政策文法備註

- 單一政策可包含一系列的陳述式。
- 政策大小上限為 2048 個字元和 10,240 個字元，取決於政策所連接的實體。如需詳細資訊，請參閱 IAM AWS STS 和配額。政策大小的計算不包含空格字元。
- 個別元素不得包含相同索引鍵的多個執行個體。例如，您不能在同一個陳述式中包含兩次 `Effect` 區塊。
- 區塊可以任何順序顯示。例如，政策中 `version_block` 可以在 `id_block` 之後。同樣地，`effect_block`、`principal_block`、`action_block` 可以在陳述式內以任何順序顯示。
- `id_block` 在以資源為基礎的政策中為選用。身分類型政策「不」得包含它。
- 以資源為基礎的政策和 IAM 角色的信任政策需要 `principal_block` 元素 (例如，在 Amazon S3 儲存貯體政策中)。身分類型政策「不」得包含它。
- Amazon S3 儲存貯體政策中的 `principal_map` 元素可以包含 `CanonicalUser` ID。大多數以資源為基礎的政策不支援此映射。若要進一步了解有關在儲存貯體政策中使用正式使用者 ID，請參閱 Amazon Simple Storage Service 使用者指南中的在政策中指定主體。
- 每個字串值 (`policy_id_string`、`sid_string`、`principal_id_string`、`action_string`、`resource_stri` 和字串版本的 `condition_value`) 可以有自己的最低和最大長度限制、特定允許的值，或是必要的內部格式。

有關字串值的備註

本節提供有關在政策的不同元素中使用字串值的其他資訊。

`action_string`

包含服務命名空間、冒號和動作的名稱。動作名稱可以包含萬用字元。範例：

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
  "ec2:StopInstances"
]
```

```
"Action":"cloudformation:*"

"Action":"*"

"Action":[
  "s3:Get*",
  "s3:List*"
]
```

policy_id_string

提供方式以包含有關政策的資訊。有些服務 (例如 Amazon SQS 和 Amazon SNS) 在預留方式中使用 Id 元素。除非按個別服務進行限制，否則 policy_id_string 可以包含空格。有些服務需要此值在 AWS 帳戶中為唯一。

> **Note**
>
> 以資源為基礎的政策中允許 id_block，但是以身分為基礎的政策不允許。

長度並沒有限制，雖然此字串對政策的整體長度有影響，但是有限的。

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

sid_string

提供方式以包含有關個別陳述式的資訊。對於 IAM 政策，Sid 值只允許使用基本英數字元 (A-Z、a-z、0-9)。支援資源政策的其他 AWS 服務可能對 Sid 值有其他要求。例如，某些服務要求此值在 中是唯一的 AWS 帳戶，而某些服務允許額外的字元，例如Sid值中的空格。

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

principal_id_string

提供使用 、IAM 使用者 AWS 帳戶、IAM 角色、聯合身分使用者或擔任角色使用者 的 Amazon Resource Name (ARN) 來指定委託人的方法。對於 AWS 帳戶，您也可以使用簡短格

式，AWS:*accountnumber*而不是完整的 ARN。如需包含 AWS 服務、擔任角色等等所有選項，請參閱如何指定主體。

請注意，您只能使用 * 來指定「每個人/匿名」。您不能使用它來指定 ARN 的部分名稱。

resource_string

在大部分情況下，包含 Amazon Resource Name (ARN)。您可以在 ARN 的資源部分中使用萬用字元 (* 和 ？)。如需有關在 ARN 中使用萬用字元的詳細資訊，請參閱在路徑中使用萬用字元。

> **Note**
>
> 當您在身分型政策中指定不完整的 ARN （少於標準六個欄位） 時， AWS 會自動將萬用字元 (*) 新增至所有缺少的欄位，以完成 ARN。例如，指定 `arn:aws:sqs` 等同於 `arn:aws:sqs:*:*:*`，這會授予所有區域和帳戶所有 Amazon SQS 資源的存取權。不過，傳遞至 AWS STS AssumeRole、AssumeRoleWithWebIdentity 和 AssumeRoleWithSAML 請求的工作階段政策不支援不完整ARNs。在工作階段政策中使用不完整的 ARN 將導致`MalformedPolicyDocumentException`錯誤。

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::amzn-s3-demo-bucket/*"
```

condition_type_string

識別要測試條件的類型，例如 `StringEquals`、`StringLike`、`NumericLessThan`、`DateGreaterThanEquals`、`Bool`、`BinaryE` 等。如需資料類型的完整清單，請參閱 IAM JSON 政策元素：條件運算子。

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}
```

```
"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

condition_key_string

識別要測試其值以判斷是否符合條件的條件金鑰。 AWS 定義一組可用於所有 AWS 服務的條件金鑰，包括 aws:PrincipalType、 aws:SecureTransport和 aws:userid。

如需 AWS 條件索引鍵的清單，請參閱 AWS 全域條件內容索引鍵。如需服務特定的條件金鑰，請參閱該服務的文件，例如下列：

- Amazon Simple Storage Service 使用者指南中的在政策中指定條件
- 《Amazon EC2 使用者指南》中的 Amazon EC2 的 IAM 政策。

```
"Condition":{
  "Bool": {
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

condition_value_string

識別決定是否符合條件的 condition_key_string 的值。如需條件類型有效值的完整清單，請參閱 IAM JSON 政策元素：條件運算子。

```
"Condition":{
```

```
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
   "ID",
   "PostDateTime"
         ]
  }
}
```

AWS 任務函數的 受管政策

我們建議使用授予最低權限的政策，或僅授予執行任務所需的許可。授予最低權限的最安全方式是撰寫僅具有團隊所需許可的自訂政策。您必須建立程序，以允許您的團隊在必要時要求更多許可。建立 IAM 客戶受管政策需要時間和專業知識，而受管政策可為您的團隊提供其所需的許可。

若要開始將許可新增至 IAM 身分（使用者、使用者群組和角色），您可以使用 AWS 受管政策。 AWS 受管政策涵蓋常見的使用案例，並可在 中使用 AWS 帳戶。 AWS 受管政策不會授予最低權限許可。若授予主體的許可超出執行任務所需的許可，您必須考量其相關的安全性風險。

您可以將 AWS 受管政策，包括任務函數，連接到任何 IAM 身分。若要切換到最低權限許可，您可以執行 AWS Identity and Access Management 和 Access Analyzer 以使用 AWS 受管政策監控主體。了解他們正在使用哪些許可後，您可以撰寫自訂政策或產生僅具有團隊所需許可的政策。這較不安全，但在您了解團隊使用的方式時提供更多彈性 AWS。

AWS 工作職能的 受管政策旨在與 IT 產業中的常見工作職能密切一致。您可以使用這些政策來授予執行特定任務角色中的某人所預期的任務所需的許可權。這些政策將許多服務的許可整合到單一政策，以便更輕鬆地使用分散在許多政策中的許可。

使用角色來結合服務

有些政策使用 IAM 服務角色來協助您利用其他 AWS 服務中找到的功能。這些政策會授予 的存取權`iam:passrole`，這可讓具有 政策的使用者將角色傳遞給 AWS 服務。此角色會將 IAM 許可委派給 AWS 服務，以代表您執行動作。

您必須根據您的需求建立角色。例如，網路管理員政策可讓具有政策的使用者將名為「flow-logs-vpc」的角色傳遞到 Amazon CloudWatch 服務。CloudWatch 使用該角色來記錄和擷取 IP 流量和使用者所建立的 VPC。

為了遵守安全性最佳實務，工作職能的政策包含篩選條件，其會限制可傳遞的有效角色的名稱。這有助於避免授予不必要的許可。如果您的使用者確實選用的服務角色，則您必須建立遵照政策中指定的命名

慣例的角色。然後，將許可授予角色。完成後，使用者就可以設定服務以使用角色，授予它角色提供的許可。

在以下區段，每個政策的名稱是連到 AWS 管理主控台中政策詳細資訊頁面的連結。您可以在那裡查看政策文件並檢閱其授予的許可。

管理員任務角色

AWS 受管政策名稱： AdministratorAccess

使用案例：此使用者具有完整的存取權，可委派許可給 AWS中的每個服務和資源。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策會針對帳戶中的所有 AWS 服務和所有資源授予所有動作。如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 AdministratorAccess。

> **Note**
>
> 您必須先啟用 IAM 使用者和角色存取， IAM 使用者或角色才能使用此政策中的許可存取 AWS 帳單與成本管理 主控台。若要執行此操作，請遵循將存取權授予帳單主控台中的指示，將存取權委派給帳單主控台。

帳單任務角色

AWS 受管政策名稱：帳單

使用案例：此使用者需要檢視帳單資訊、設定付款和授權付款。使用者可以監控整個 AWS 服務累積的成本。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予完整許可，以管理帳單、成本、付款方法、預算和報告。如需其他成本管理政策範例，請參閱《AWS 帳單與成本管理 使用者指南》中的 AWS Billing 政策範例 如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 Billing。

> **Note**
>
> 您必須先啟用 IAM 使用者和角色存取， IAM 使用者或角色才能使用此政策中的許可存取 AWS 帳單與成本管理 主控台。若要執行此操作，請遵循將存取權授予帳單主控台中的指示，將存取權委派給帳單主控台。

資料庫管理員任務角色

AWS 受管政策名稱：DatabaseAdministrator

使用案例：此使用者在 AWS 雲端中設定、設定和維護資料庫。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予許可，以建立、設定和維護資料庫。它包括對 AWS 資料庫服務的存取，例如 Amazon DynamoDB、Amazon Relational Database Service (RDS) 和 Amazon Redshift。查看政策以了解此政策支援之資料庫服務的完整清單。如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 DatabaseAdministrator。

此任務函數政策支援將角色傳遞至 AWS 服務的能力。政策只針對下表中所命名的那些角色允許 `iam:PassRole` 動作。如需詳細資訊，請參閱本主題稍後的 建立角色和連接政策 (主控台)。

使用案例	角色名稱 (* 是萬用字元)	要選擇的服務角色類型	選取此 AWS 受管政策
允許使用者監控 RDS 資料庫	rds-monitoring-role	增強監控的 Amazon RDS 角色	AmazonRDS EnhancedMonitoring Role
允許 AWS Lambda 監控您的資料庫並存取外部資料庫	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess
允許 Lambda 使用 DynamoDB 將檔案上傳	lambda_exec_role	AWS Lambda	如 AWS 大數據部落格中所定義建立新的受管政策

使用案例	角色名稱 (* 是萬用字元)	要選擇的服務角色類型	選取此 AWS 受管政策
至 Amazon S3 和 Amazon Redshift 叢集			
允許 Lambda 函數作為您 DynamoDB 表格的觸發條件	lambda-dynamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole
允許 Lambda 函數存取 VPC 中的 Amazon RDS	lambda-vpc-execution-role	如 AWS Lambda 開發人員指南 中所定義使用信任政策建立角色	AWSLambda VPCAccessExecutionRole
允許 AWS Data Pipeline 存取您的 AWS 資源	DataPipelineDefaultRole	如 AWS Data Pipeline 開發人員指南 中所定義使用信任政策建立角色	AWS Data Pipeline 文件會列出此使用案例所需的許可。請參閱 的 IAM 角色 AWS Data Pipeline
允許在 Amazon EC2 執行個體上執行的應用程式存取您的 AWS 資源	DataPipelineDefaultResourceRole	如 AWS Data Pipeline 開發人員指南 中所定義使用信任政策建立角色	AmazonEC2RoleforDataPipelineRole

資料科學家任務角色

AWS 受管政策名稱：DataScientist

使用案例：此使用者執行 Hadoop 工作和查詢。使用者也會存取和分析資料分析和商業智慧的資訊。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予許可，以在 Amazon EMR 叢集上建立、管理和執行查詢，並使用 Amazon QuickSight 等工具執行資料分析。此政策包括存取其他資料科學家服務，例如 Amazon EC2 AWS

Data Pipeline、Amazon Kinesis、Amazon Machine Learning 和 SageMaker AI。查看政策以了解此政策支援之資料科學家服務的完整清單。如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 DataScientist。

此任務函數政策支援將角色傳遞至 AWS 服務的能力。一個陳述式允許傳遞任何角色到 SageMaker AI。另外一個陳述式只針對下表中所命名的那些角色允許 `iam:PassRole` 動作。如需詳細資訊，請參閱本主題稍後的 建立角色和連接政策 (主控台)。

使用案例	角色名稱 (* 是萬用字元)	要選擇的服務角色類型	AWS 要選取的 受管政策
允許 Amazon EC2 執行個體存取適合叢集的服務和資源	EMR-EC2_DefaultRole	Amazon EMR for EC2	AmazonElasticMapReduceforEC2Role
允許 Amazon EMR 存取適用於叢集的 Amazon EC2 服務和資源	EMR_DefaultRole	Amazon EMR	AmazonEMRServicePolicy_v2
允許 Kinesis Managed Service for Apache Flink 存取串流資料來源	kinesis-*	如 AWS 大數據部落格中所定義使用信任政策建立角色。	請參閱 AWS 大數據部落格，其根據您的使用案例概述四個可能的選項
允許 AWS Data Pipeline 存取您的 AWS 資源	DataPipelineDefaultRole	如 AWS Data Pipeline 開發人員指南 中所定義使用信任政策建立角色	AWS Data Pipeline 文件會列出此使用案例所需的許可。請參閱 的 IAM 角色 AWS Data Pipeline
允許在 Amazon EC2 執行個體上執行的應用程式存取您的 AWS 資源	DataPipelineDefaultResourceRole	如 AWS Data Pipeline 開發人員指南 中所定義使用信任政策建立角色	AmazonEC2RoleforDataPipelineRole

開發人員進階使用者任務角色

AWS 受管政策名稱：PowerUserAccess

使用案例：此使用者執行應用程式開發任務，並可以建立和設定支援 AWS 感知應用程式開發的資源和服務。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策的第一個陳述式使用 `NotAction`元素來允許所有 AWS 服務和所有資源的所有動作 AWS Identity and Access Management，但 AWS Organizations和 除外 AWS 帳戶管理。第二個陳述式則會授予可建立服務連結角色的 IAM 許可。如果某些服務必須存取其他服務中的資源 (例如 Amazon S3 儲存貯體)，則這是必要的。

它還授予許可 AWS Organizations ，以檢視有關使用者組織的資訊，包括管理帳戶電子郵件和組織限制。雖然此政策會限制 IAM AWS Organizations，但如果啟用 IAM Identity Center，則會允許使用者執行所有 IAM Identity Center 動作。它還授予帳戶管理許可，以檢視為帳戶啟用或停用哪些 AWS 區域。

網路管理員任務角色

AWS 受管政策名稱：NetworkAdministrator

使用案例：此使用者的任務是設定和維護 AWS 網路資源。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予在 Auto Scaling、Amazon EC2 AWS Direct Connect、Route 53、Amazon CloudFront、Elastic Load Balancing AWS Elastic Beanstalk、Amazon SNS、CloudWatch、CloudWatch Logs、Amazon S3、IAM 和 Amazon Virtual Private Cloud 中建立和維護網路資源的許可。如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 NetworkAdministrator。

此任務函數需要將角色傳遞至 AWS 服務的能力。政策只針對下表中所命名的那些角色授予 `iam:GetRole` 和 `iam:PassRole`：如需詳細資訊，請參閱本主題後面部分的 建立角色和連接政策 (主控台)。

使用案例	角色名稱 (* 是萬用字元)	要選擇的服務角色類型	AWS 要選取的 受管政策
可讓 Amazon VPC 代表使用者在 CloudWatch Logs 中建	flow-logs-*	如 Amazon VPC 使用者指南 中所定義	此使用案例沒有現有的 AWS 受管政

使用案例	角色名稱 (* 是萬用字元)	要選擇的服務角色類型	AWS 要選取的 受管政策
立和管理日誌，以監控進出 VPC 的 IP 流量		使用信任政策建立角色	策，但文件會列出所需的許可。請參閱 Amazon VPC 使用者指南。

唯讀存取

AWS 受管政策名稱：ReadOnlyAccess

使用案例:此使用者需要對 AWS 帳戶中的每個資源進行唯讀存取。

> **Important**
>
> 此使用者也將有權讀取 Amazon S3 儲存貯體和 Amazon DynamoDB 資料表等儲存服務中的資料。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策說明：此政策會授予列出、取得、描述及檢視資源及其屬性的許可。不包括像建立或刪除變動職能。此政策確實包含安全相關 AWS 服務的唯讀存取權，例如 AWS Identity and Access Management 和 AWS 帳單與成本管理。

查看政策以了解此政策支援之服務和動作的完整清單。如需有關受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 ReadOnlyAccess。如果您需要類似政策，但此政策不會授予存取權來讀取儲存服務中的資料，請參閱僅限檢視使用者任務角色。

MCP 服務動作完整存取

AWS 受管政策名稱：AWSMcpServiceActionsFullAccess

使用案例：此使用者需要使用 AWS MCP 伺服器存取 AWS 服務。此政策不會將 MCP 服務所採取動作的存取權授予其他服務 AWS 。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予呼叫任何 AWS MCP 服務動作的許可。當您不需要指定每個 AWS MCP 服務的許可時，可以使用 。它不會將 MCP 服務所採取動作的許可授予其他服務 AWS ，除了 MCP 服務動作之外，還必須一律單獨授予這些許可。如需 受管政策的詳細資訊，請參閱《 AWS 受管政策參考指南》中的 AWSMcpServiceActionsFullAccess。

安全稽核人員任務角色

AWS 受管政策名稱：SecurityAudit

使用案例：此使用者監控帳戶是否符合安全需求。這個使用者可以存取日誌和事件，以調查潛在安全漏洞或潛在惡意活動。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予許可，以檢視許多 AWS 服務的組態資料和檢閱其日誌。如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 SecurityAudit。

支援使用者任務角色

AWS 受管政策名稱：AWSSupportAccess

使用案例：此使用者聯絡 AWS Support、建立支援案例，以及檢視現有案例的狀態。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予建立和更新 支援 案例的許可。如需有關受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 AWSSupportAccess。

系統管理員任務角色

AWS 受管政策名稱：SystemAdministrator

使用案例：此使用者為開發操作設定和維護資源。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策授予許可，以跨各種 AWS 服務建立和維護資源，包括 Amazon CloudWatch AWS CloudTrail、 AWS CodeCommit AWS CodeDeploy、 AWS Config、 AWS Directory Service、Amazon EC2 AWS Identity and Access Management、 AWS Key Management Service、AWS Lambda、Amazon RDS、Route 53、Amazon S3、Amazon SES、Amazon SQS AWS Trusted Advisor和 Amazon VPC。如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 SystemAdministrator。

此任務函數需要將角色傳遞至 AWS 服務的能力。政策只針對下表中所命名的那些角色授予 `iam:GetRole` 和 `iam:PassRole`：如需詳細資訊，請參閱本主題後面部分的 建立角色和連接政策 (主控台)。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

使用案例	角色名稱 (* 是萬用字元)	要選擇的服務角色類型	AWS 要選取的 受管政策
允許在 Amazon ECS 叢集的 EC2 執行個體中執行的應用程式存取 Amazon ECS	ecr-sysadmin-*	EC2 容器服務的 Amazon EC2 角色	AmazonEC2ContainerServiceforEC2Role
允許使用者監控資料庫	rds-monitoring-role	增強監控的 Amazon RDS 角色	AmazonRDSEnhancedMonitoringRole
允許在 EC2 執行個體中執行的應用程式存取 AWS 資源。	ec2-sysadmin-*	Amazon EC2	授予 S3 儲存貯體存取權的角色的範例政策，如《Amazon EC2 使用者指南》https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html中所示；可視需要自訂

使用案例	角色名稱 (* 是萬用字元)	要選擇的服務角色類型	AWS 要選取的 受管政策
允許 Lambda 讀取 DynamoDB Streams 和寫入 CloudWatch Logs	lambda-sysadmin-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole

僅限檢視使用者任務角色

AWS 受管政策名稱：ViewOnlyAccess

使用案例：此使用者可以檢視 帳戶中跨 服務 AWS 的資源和基本中繼資料清單。使用者無法讀取超出配額的資源內容或中繼資料，以及列出資源的資訊。

政策更新： AWS 維護和更新此政策。如需此政策的變更記錄，請在 IAM 主控台中檢視政策，然後選擇 Policy versions (政策版本) 索引標籤。如需有關任務角色政策更新的詳細資訊，請參閱 工作職能的 AWS 受管政策更新。

政策描述：此政策會授予 `List*`、`View*`、、 `Describe*` `Get*`和 `Lookup*`存取 AWS 服務的資源。若要查看此政策針對每個服務包含哪些動作，請參閱 ViewOnlyAccess。如需受管政策的詳細資訊，請參閱 AWS Managed Policy Reference Guide 中的 ViewOnlyAccess。

工作職能的 AWS 受管政策更新

這些政策都由 維護， AWS 並保持在最新狀態，以在 AWS 服務新增時包含對新服務和新功能的支援。這些政策無法由客戶修改。您可以複製政策，然後修改副本，但該副本不會自動更新，因為 AWS 引入新的服務和 API 操作。

對於任務角色政策，您可以在 IAM 主控台中檢視版本歷程記錄，以及每次更新的時間和日期。若要執行此操作，請使用此頁面上的連結來檢視政策詳細資訊。然後選擇 Policy versions (政策版本) 索引標籤以檢視版本。此頁面會顯示政策的最近 25 個版本。若要檢視政策的所有版本，請呼叫 get-policy-version AWS CLI 命令或 GetPolicyVersion API 操作。

> **Note**
>
> 您最多可以有五個版本的客戶受管政策，但 會 AWS 保留 AWS 受管政策的完整版本歷史記錄。

建立角色和連接政策 (主控台)

一些先前列出的政策授予使用角色設定 AWS 服務的能力，這些角色可讓這些服務代表您執行操作。工作職能政策會指定確切的角色名稱，您必須使用或至少包含字首，其指定可用名稱的第一個部分。若要建立這些角色之一，請執行下列程序中的步驟。

為 AWS 服務 (IAM 主控台) 建立角色

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇角色，然後選擇建立角色。
3. 對於 Trusted entity type (信任的實體類型)，請選擇 AWS 服務。
4. 針對服務或使用案例，選擇服務，然後選擇使用案例。服務會定義使用案例，以包含服務所需的信任政策。
5. 選擇下一步。
6. 對於許可政策，選項取決於您選取的使用案例：
   - 如果服務定義了角色的許可，則您無法選取許可政策。
   - 從一組有限的許可政策中選取。
   - 從所有許可政策中選取。
   - 選取無許可政策，在建立角色之後建立政策，然後將政策連接到角色。
7. (選用) 設定許可界限。這是進階功能，可用於服務角色，而不是服務連結的角色。
   a. 開啟設定許可界限區段，然後選擇使用許可界限來控制角色許可上限。

      IAM 包含您帳戶中 AWS 受管和客戶受管政策的清單。
   b. 選取用於許可界限的政策。
8. 選擇下一步。
9. 對於角色名稱，選項取決於服務：
   - 如果服務定義了角色名稱，則無法編輯角色名稱。
   - 如果服務定義了角色名稱的字首，則可以輸入選用字尾。
   - 如果服務未定義角色名稱，則可以為角色命名。

> **Important**
>
> 當您命名角色時，請注意下列事項：
>
> - 角色名稱在您的 中必須是唯一的 AWS 帳戶，而且無法依大小寫設為唯一。
>
> 例如，不要同時建立名為 **`PRODROLE`** 和 **`prodrole`** 的角色。當角色名稱用於政策或 ARN 的一部分時，角色名稱會區分大小寫，但是當角色名稱在主控台中顯示給客戶時，例如在登入過程中，角色名稱不會區分大小寫。
> - 因為其他實體可能會參考角色，所以在建立角色之後，就無法編輯其名稱。

10. (選用) 在說明中，輸入角色的說明。
11. (選用) 若要編輯使用案例和角色許可，請在步驟 1：選取受信任的實體或者步驟 2：新增許可區段中選擇編輯。
12. (選用) 若要協助識別、組織或搜尋角色，請將標籤新增為索引鍵值對。如需在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的AWS Identity and Access Management 資源的標籤。
13. 檢閱角色，然後選擇 Create role (建立角色)。

範例 1：設定使用者做為資料庫管理員 (主控台)

此範例顯示設定 IAM 使用者 Alice 作為資料庫管理員所需的步驟。您使用該區段中表格第一列的資訊，並允許使用者啟用 Amazon RDS 監控。您將 DatabaseAdministrator 政策連接到 Alice 的 IAM 使用者，讓他們可以管理 Amazon 資料庫服務。該政策也允許 Alice 將稱為 `rds-monitoring-role` 的角色傳遞到 Amazon RDS 服務，讓該服務代表他們來監控 Amazon RDS 資料庫。

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇政策，在搜尋方塊中輸入 **`database`**，然後按下 Enter 鍵。
3. 選取 DatabaseAdministrator 政策的選項按鈕，選擇動作，然後選擇連接。
4. 在實體清單中，選取 Alice，然後選擇連接政策。Alice 現在可管理 AWS 資料庫。不過，為允許 Alice 來監控這些資料庫，您必須設定服務角色。
5. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
6. 選擇 AWS Service (服務) 角色類型，然後選擇 Amazon RDS。
7. 選擇 Amazon RDS Role for Enhanced Monitoring (增強監控的 Amazon RDS 角色) 使用案例。
8. Amazon RDS 定義角色的許可。選擇 Next: Review (下一步：檢閱) 以繼續進行。

9. 角色名稱必須是 Alice 現在所有的 DatabaseAdministrator 政策所指定的其中一個。其中一個是 **rds-monitoring-role**。為 Role name (角色名稱) 輸入該名稱。
10. (選用) 在 Role description (角色說明) 中，輸入新角色的說明。
11. 檢閱詳細資訊之後，選擇 Create role (建立角色)。
12. Alice 現在可以在 Amazon RDS 主控台的 Monitoring (監控) 區段中啟用 RDS Enhanced Monitoring (RDS 增強型監控)。例如，他們在建立資料庫執行個體、建立讀取複本或修改資料庫執行個體時，他們可能執行此操作。在將 Enable Enhanced Monitoring (啟用增強型監控) 設定為 Yes (是) 時，他們必須在 Monitoring Role (監控角色) 方塊中輸入他們建立的角色名稱 (rds-monitoring-role)。

範例 2：設定使用者做為網路管理員 (主控台)

此範例顯示設定 IAM 使用者 Jorge 作為網路管理員所需的步驟。它使用該區段中表格的資訊，允許 Jorge 監控進出 VPC 的 IP 流量。它也允許 Jorge 在 CloudWatch Logs 中擷取日誌中的資訊。您可將 NetworkAdministrator 政策連接到 Jorge 的 IAM 使用者，讓他們可以設定 AWS 網路資源。該政策也會在您建立流程日誌時，讓 Jorge 將名稱開頭為 `flow-logs*` 的角色傳遞到 Amazon EC2。在這個案例中，不像範例 1，沒有預先定義的服務角色類型，所以您必須分別地執行幾個步驟。

1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇政策，然後在搜尋方塊中輸入 **network**，再按下 Enter。
3. 選取 NetworkAdministrator 政策旁的選項按鈕，選擇動作，然後選擇連接。
4. 在使用者清單中，選取 Jorge 旁的核取方塊，然後選擇連接政策。Jorge 現在可以管理 AWS 網路資源。不過，為啟用監控 VPC 中的 IP 流量，您必須設定服務角色。
5. 由於您需要建立的服務角色，沒有預先定義的受管政策，您必須先建立它。在導覽窗格中，選擇 Policies (政策)，然後選擇 Create policy (建立政策)。
6. 在政策編輯器區段中，選擇 JSON 選項，並從下列 JSON 政策文件複製文字。將此文字貼上至 JSON 文字方框中。

   JSON

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

```
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

7. 解決政策驗證期間產生的任何安全性警告、錯誤或一般性警告，然後選擇 Next (下一步)。

> **Note**
>
> 您可以隨時切換視覺化與 JSON 編輯器選項。不過，如果您進行變更或在視覺化編輯器中選擇下一步，IAM 就可能會調整您的政策結構，以便針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱政策結構調整。

8. 在檢視與建立頁面上，針對政策名稱輸入 **`vpc-flow-logs-policy-for-service-role`**。檢視此政策中定義的許可以查看政策授與的許可，然後選擇建立政策來儲存您的工作。

   新的政策會出現在受管政策清單中，並且已準備好連接。

9. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
10. 選擇 AWS Service (服務) 角色類型，然後選擇 Amazon EC2。
11. 選擇 Amazon EC2 使用案例。
12. 在 Attach permissions policies (連接許可政策) 頁面上，選擇您稍早建立的政策 vpc-flow-logs-policy-for-service-role，然後選擇 Next: Review (下一步：檢閱)。
13. 角色名稱必須經過 Jorge 現在所有的 NetworkAdministrator 政策許可。任何以 `flow-logs-` 開頭的名稱均允許。在此範例中，為 Role name (角色名稱) 輸入 **`flow-logs-for-jorge`**。
14. (選用) 在 Role description (角色說明) 中，輸入新角色的說明。
15. 檢閱詳細資訊之後，選擇 Create role (建立角色)。
16. 現在您可以設定此案例所需的信任政策。在角色頁面上，選擇 flow-logs-for-jorge 角色 (名稱，而非核取方塊)。針對您的新角色，在詳細資訊頁面上，選擇 Trust relationships (信任關係) 標籤，然後選擇 Edit trust relationship (編輯信任關係)。
17. 將「Service (服務)」行讀取為如下所示，取代 `ec2.amazonaws.com` 的項目：

```
        "Service": "vpc-flow-logs.amazonaws.com"
```

18. Jorge 現在可在 Amazon EC2 主控台中為 VPC 或子網路建立流程日誌。當您建立流程日誌時，指定 flow-logs-for-jorge 角色。該角色具有許可，以建立日誌和寫入資料。

AWS 全域條件內容索引鍵

當委託人向 提出請求時 AWS， 會將請求資訊 AWS 收集到請求內容中。您可以使用 JSON 政策的 `Condition` 元素，來比較請求內容中的鍵和您在政策中指定的鍵值。請求資訊由不同的來源提供，包括提出請求的主體、提出請求的資源，以及請求本身的中繼資料。

全域條件索引鍵可用於所有 AWS 服務。雖然這些條件索引鍵可用於所有政策，但不是每個請求內容都提供該索引鍵。例如，只有在 AWS 服務主體直接呼叫您的資源時，`aws:SourceAccount` 條件索引鍵才可用。若要進一步了解全域金鑰包含在請求內容中的情況，請參閱每個金鑰的可用性資訊。

有些個別服務會建立其自己的條件索引鍵，可在其他服務的請求內容中使用。跨服務條件索引鍵是一種全域條件索引鍵類型，其中包含符合服務名稱的字首，例如 `ec2:` 或 `lambda:`，但可在其他服務中使用。

服務特定條件金鑰已定義用於個別 AWS 服務。例如，Amazon S3 可讓您使用 `s3:VersionId` 條件索引鍵撰寫政策，以限制對特定 Amazon S3 物件版本的存取。此條件索引鍵對服務是唯一的，這表示它僅適用於對 Amazon S3 服務的請求。如需服務特定的條件金鑰，請參閱 AWS 服務的動作、資源和條件金鑰，然後選擇您要檢視其金鑰的服務。

> **Note**
>
> 如果您使用只能在一些情況下可用的條件鍵，則可以使用條件運算子的 IfExists 版本。如果請求內容中缺少條件鍵，則政策可能會讓評估失敗。例如，搭配使用以下條件區塊與 `...IfExists` 運算子，以在請求來自特定 IP 範圍或特定 VPC 時進行比對。若其中一個鍵，或兩者都不包含在請求內容中，條件仍會傳回 `true`。只有在請求內容中包含指定鍵時，才會檢查值。如需有關在運算子不存在金鑰時如何評估政策的詳細資訊，請參閱條件運算子。
>
> ```
> "Condition": {
>     "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
>     "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
> }
> ```

> **Important**
>
> 若要將您的條件與具有多個鍵值的請求內容進行比較，您必須使用 ForAllValues 或 ForAnyValue 設定運算子。只能將集合運算子與多值條件鍵搭配使用。請勿將集合運算子與單一值條件鍵搭配使用。如需詳細資訊，請參閱用於多值內容索引鍵的集運算子。

主體的屬性	角色工作階段的屬性	網路的屬性	資源的屬性	請求的屬性
aws:PrincipalArn	aws:AssumedRoot	aws:SourceIp	aws:ResourceAccount	aws:CalledVia
aws:PrincipalAccount	aws:FederatedProvider	aws:SourceVpc	aws:ResourceOrgID	aws:CalledViaFirst
aws:PrincipalOrgPaths	aws:TokenIssueTime	aws:SourceVpcArn	aws:ResourceOrgPaths	aws:CalledViaLast
aws:PrincipalOrgID	aws:SignInSessionArn	aws:SourceVpce	aws:ResourceTag/ *tag-key*	aws:CalledViaAWSMCP
aws:PrincipalTag/ *tag-key*	aws:MultiFactorAuthAge	aws:VpceAccount		aws:ViaAWSService
aws:PrincipalIsAWSService	aws:MultiFactorAuthPresent	aws:VpceOrgID		aws:ViaAWSMCPService
aws:PrincipalServiceName	aws:ChatbotSourceArn	aws:VpceOrgPaths		aws:CurrentTime
aws:PrincipalServi		aws:VpcSourceIp		aws:EpochTime
				aws:referer
				aws:RequestedRegion

主體的屬性	角色工作階段的屬性	網路的屬性	資源的屬性	請求的屬性
ceNamesList aws:PrincipalType aws:userid aws:username	aws:Ec2InstanceSourceVpc aws:Ec2InstanceSourcePrivateIPv4 aws:SourceIdentity aws:ViaCustomerDomain ec2:RoleDelivery ec2:SourceInstanceArn glue:RoleAssumedBy glue:CredentialIssuingService codebuild:BuildArn codebuild:ProjectArn			aws:RequestTag/ *tag-key* aws:TagKeys aws:SecureTransport aws:SourceAccount aws:SourceArn aws:SourceOrgID aws:SourceOrgPaths aws:UserAgent aws:IsMcpServiceAction

主體的屬性	角色工作階段的屬性	網路的屬性	資源的屬性	請求的屬性
	lambda:SourceFunctionArn ssm:SourceInstanceArn identitystore:UserId			

敏感條件索引鍵

下列條件索引鍵視為敏感資訊。在這些條件索引鍵中使用萬用字元沒有任何有效的使用案例，即使具有萬用字元的索引鍵值子字串也一樣。這是因為萬用字元可能會將條件索引鍵與任何值相符，這可能會構成安全風險。

- aws:PrincipalAccount
- aws:PrincipalOrgID
- aws:ResourceAccount
- aws:ResourceOrgID
- aws:SourceAccount
- aws:SourceOrgID
- aws:SourceVpc
- aws:SourceVpce
- aws:VpceAccount
- aws:VpceOrgID

主體的屬性

使用下列條件索引鍵，將提出請求的主體詳細資訊與您在政策中指定的主體屬性進行比較。如需可以提出請求的主體清單，請參閱 如何指定主體。

aws:PrincipalArn

使用此鍵來將提出請求主體的 Amazon Resource Name (ARN) 與您在政策中所指定的 ARN 進行比較。針對 IAM 角色，請求內容會傳回角色的 ARN，而非取得角色使用者的 ARN。

- 可用性 – 此鍵會包含在所有簽章請求的請求內容中。匿名請求不包含此鍵。您可以在此條件鍵中指定以下類型的主體：
  - IAM 角色
  - IAM 使用者
  - AWS STS 聯合身分使用者主體
  - AWS 帳戶 根使用者
- 資料類型 – ARN

  AWS 建議您在比較 ARN 時使用 ARN 運算子，而非字串運算子。 ARNs
- 值類型 - 單一值
- 範例值以下清單顯示您可以在 `aws:PrincipalArn` 條件索引鍵中指定的不同類型主體傳回的請求內容值：
  - IAM 角色 – 請求內容包含條件鍵 `aws:PrincipalArn` 的下列值。請勿將擔任角色工作階段 ARN 指定為此條件鍵的值。如需有關擔任角色工作階段主體的詳細資訊，請參閱 角色工作階段主體。

    ```
    arn:aws:iam::123456789012:role/role-name
    ```
  - IAM 使用者 – 請求內容包含條件鍵 `aws:PrincipalArn` 的下列值。

    ```
    arn:aws:iam::123456789012:user/user-name
    ```
  - AWS STS 聯合身分使用者主體 – 請求內容包含條件索引鍵 的下列值`aws:PrincipalArn`。

    ```
    arn:aws:sts::123456789012:federated-user/user-name
    ```
  - AWS 帳戶 根使用者 – 請求內容包含條件索引鍵 的下列值`aws:PrincipalArn`。將根使用者 ARN 指定為 `aws:PrincipalArn` 條件鍵的值時，僅會限制 AWS 帳戶根使用者的許可。這不同

於在資源型政策的主體元素中指定根使用者 ARN，後者將授權委派給 AWS 帳戶。如需在資源型政策的主體元素中指定根使用者 ARN 的相關資訊，請參閱 AWS 帳戶 主體。

```
arn:aws:iam::123456789012:root
```

您可以在 AWS Organizations 服務控制政策 (SCPs) aws:PrincipalArn中將根使用者 ARN 指定為條件索引鍵的值。SCP 是一種組織政策，用於管理您組織中的許可，並且僅會影響組織中的成員帳戶。SCP 會限制 IAM 使用者和成員帳戶中 IAM 使用者和角色的許可，包括成員帳戶的根使用者。如需有關 SCP 對許可之影響的詳細資訊，請參閱 AWS Organizations User Guide 中的 SCP effects on permissions。

aws:PrincipalAccount

使用此鍵來將請求主體所屬的帳戶與您在政策中指定的帳戶識別碼進行比較。對於匿名請求，請求內容會返回 anonymous。

- 可用性 – 此金鑰會包含在所有請求的請求內容中，包括匿名請求。
- 資料類型 – 字串
- 值類型 - 單一值

以下範例中，除了帳號為 123456789012 的主體之外，一律拒絕存取。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:resource"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
```

```
          ]
        }
      }
    }
  ]
}
```

aws:PrincipalOrgPaths

使用此金鑰來比較提出請求之委託人的 AWS Organizations 路徑與政策中的路徑。該主體可以是 IAM 使用者、IAM 角色、 AWS STS 聯合身分使用者主體或 AWS 帳戶根使用者。在政策中，這項條件鍵會確保申請者是在指定組織根或 AWS Organizations中組織單位 (OU) 內的帳戶成員。 AWS Organizations 路徑是 AWS Organizations 實體結構的文字表示法。如需有關使用和了解路徑的詳細資訊，請參閱 了解 AWS Organizations 實體路徑。

- 可用性 – 只有在主體是組織的成員時，請求內容中才會包含此鍵。匿名請求不包含此鍵。
- 資料類型 – 字串 (清單)
- 值類型 - 多重值

> **Note**
>
> 組織 ID 是全域唯一，但 OU ID 和根 ID 只有在組織內是唯一。這表示沒有兩個組織共用相同的組織 ID。不過，另一個組織的 OU 或根可能與您的 ID 相同。我們建議您在指定 OU 或根時，一律包含組織 ID。

例如，以下條件會針對帳戶中的主體傳回 `true`。這些帳戶是直接連接到 `ou-ab12-22222222` OU，但並非其子 OU。

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

以下條件會針對帳戶中的主體傳回 `true`。該帳戶是直接連接到 OU 或其任何的子 OU。當您包含萬用字元時，您必須使用 `StringLike` 條件運算子。

```
"Condition" : { "ForAnyValue:StringLike" : {
```

```
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

對於直接連接到任何子 OU 但不直接連接到父 OU 的帳戶中的主體，以下條件將傳回 `true`。先前的條件適用於 OU 或任何子系。以下條件僅適用於子系 (以及這些子系的任何子系)。

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

以下條件會允許存取 `o-a1b2c3d4e5` 組織中的任何主體，無論其父 OU 為何。

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` 是多重值條件鍵。多重值金鑰在請求內容中可以擁有多個值。當您搭配 `ForAnyValue` 條件運算子使用多個值時，主體的路徑必須符合政策列出的其中一個路徑。如需有關多重值條件鍵的詳細資訊，請參閱 用於多值內容索引鍵的集運算子。

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

aws:PrincipalOrgID

使用此金鑰來比較 AWS Organizations 請求主體所屬組織的識別符與政策中指定的識別符。

- 可用性 – 只有在主體是組織的成員時，請求內容中才會包含此鍵。匿名請求不包含此鍵。
- 資料類型 – 字串
- 值類型 - 單一值

此全域鍵提供了列出組織中所有 AWS 帳戶的所有帳戶 ID 的替代方法。您可以使用此條件鍵來簡化在資源型政策中指定 Principal 元素。您可以在條件元素中指定組織 ID。當您新增和移除帳戶時，包含 aws:PrincipalOrgID 鍵的政策會自動包含正確的帳戶，不需要手動更新。

例如，下列 Amazon S3 儲存貯體政策允許 o-xxxxxxxxxxx 組織中任何帳戶的成員將物件新增至 amzn-s3-demo-bucket 儲存貯體。

JSON

```
{
 "Version":"2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
   "Condition": {"StringEquals":
     {"aws:PrincipalOrgID":"o-xxxxxxxxxxx"}
   }
 }
}
```

> **Note**
>
> 此全域條件也適用於 AWS 組織的管理帳戶。此政策可防止指定組織以外的所有主體存取 Amazon S3 儲存貯體。這包括與您的內部資源互動的任何 AWS 服務，例如 AWS CloudTrail 將日誌資料傳送到 Amazon S3 儲存貯體。若要了解如何安全地授予 AWS 服務的存取權，請參閱 aws:PrincipalIsAWSService。

如需 的詳細資訊 AWS Organizations，請參閱什麼是 AWS Organizations？ AWS Organizations 《使用者指南》中的 。

aws:PrincipalTag/tag-key

使用此鍵來將連接至提出請求主體的標籤，與您在政策中所指定的標籤進行比較。若主體連接的標籤超過一個，請求內容會針對每個連接的標籤鍵包含一個 aws:PrincipalTag 鍵。

- 可用性 – 若主體搭配連接標籤使用 IAM 使用者，此鍵會包含在請求內容中。其會針對主體，使用具備連接標籤或工作階段標籤的 IAM 角色包含在其中。匿名請求不包含此鍵。
- 資料類型 – 字串
- 值類型 - 單一值

您可以將自訂屬性以鍵值對的形式新增至使用者或角色。如需有關 IAM 標籤的詳細資訊，請參閱 AWS Identity and Access Management 資源的標籤。您可以使用 `aws:PrincipalTag` 對 AWS 主體進行控制存取。

此範例會示範如何建立身分型政策，允許具有 **department=hr** 標籤的使用者管理 IAM 使用者、群組或角色。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:Get*",
        "iam:List*",
        "iam:Generate*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

aws:PrincipalIsAWSService

使用此金鑰來檢查 AWS 服務主體是否直接呼叫您的資源。例如： AWS CloudTrail 使用服務主體 `cloudtrail.amazonaws.com` 將日誌寫入至 Amazon S3 儲存貯體。當服務使用服務主體對您的

資源執行直接動作時，請求內容鍵會設定為 true (真)。服務代表 IAM 主體使用主體的憑證來提出請求時，內容鍵設定為 false。若服務使用服務角色或服務連結角色來代表主體進行呼叫，則也會設定為 false。

- 可用性 – 對於所有使用 AWS 憑證的已簽署 API 請求，此鍵會出現在請求內容中。匿名請求不包含此鍵。
- 資料類型 – 布林值
- 值類型 - 單一值

您可以使用此條件金鑰來限制對信任身分和預期網路位置的存取，同時安全地授予對 AWS 服務的存取。

在下列 Amazon S3 儲存貯體政策範例中，存取儲存貯體會受到限制，除非請求來自 vpc-111bbb22 或來自服務主體，例如 CloudTrail。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

在下列影片中，進一步了解如何在政策中使用 aws:PrincipalIsAWSService 條件鍵。

安全地將存取權授予授權使用者、預期的網路位置 AWS 和服務。

aws:PrincipalServiceName

使用此鍵將政策中的服務主體名稱，以及向您的資源提出請求的服務主體進行比較。您可以使用這個鍵來檢查這個呼叫是否由特定的服務主體進行比較。當服務主體直接向您的資源要求時，aws:PrincipalServiceName 鍵會包含服務主體的名稱。例如， AWS CloudTrail 服務主體名稱為 cloudtrail.amazonaws.com。

- 可用性 – 當 AWS 服務委託人進行呼叫時，此金鑰會出現在請求中。此鍵在任何其他情況下都不存在，包括以下情況：
  - 若服務使用服務角色或服務連結角色來代表主體進行呼叫。
  - 服務使用 IAM 主體的憑證代表主體提出請求。
  - 如果呼叫是由 IAM 主體直接進行。
  - 如果呼叫由匿名請求者發出。
- 資料類型 – 字串
- 值類型 - 單一值

您可以使用此條件金鑰來限制對信任身分和預期網路位置的存取，同時安全地授予對 AWS 服務的存取。

在下列 Amazon S3 儲存貯體政策範例中，存取儲存貯體會受到限制，除非請求來自 vpc-111bbb22 或來自服務主體，例如 CloudTrail。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
```

```
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
        }
      }
    }
  ]
}
```

aws:PrincipalServiceNamesList

此鍵提供屬於該服務的所有服務主體名稱清單。這是進階條件鍵。您可以用其來限制服務僅從特定區域存取您的資源。有些服務可能會建立區域服務主體，以指出特定區域內服務的特定執行個體。您可以將資源的存取限制為服務的特定執行個體。當服務主體直接向您的資源要求時，`aws:PrincipalServiceNamesList` 包含與服務區域執行個體相關聯的所有服務主體名稱的無序清單。

- 可用性 – 當 AWS 服務委託人發出呼叫時，此金鑰會出現在請求中。此鍵在任何其他情況下都不存在，包括以下情況：
  - 若服務使用服務角色或服務連結角色來代表主體進行呼叫。
  - 服務使用 IAM 主體的憑證代表主體提出請求。
  - 如果呼叫是由 IAM 主體直接進行。
  - 如果呼叫由匿名請求者發出。
- 資料類型 – 字串 (清單)
- 值類型 - 多重值

`aws:PrincipalServiceNamesList` 是多重值條件鍵。多值鍵在請求內容中可以擁有多個值。使用此鍵時，您必須使用 `ForAnyValue` 或 `ForAllValues` 設定運算子搭配字串條件運算子。如需有關多重值條件鍵的詳細資訊，請參閱 用於多值內容索引鍵的集運算子。

aws:PrincipalType

使用此鍵來將發出請求的主體類型與您在政策中所指定的主體類型進行比較。如需詳細資訊，請參閱如何指定主體。如需 `principal` 鍵值的具體範例，請參閱 主體索引鍵值。

- 可用性 – 此鍵會包含在所有請求的請求內容中，包括匿名請求。
- 資料類型 – 字串
- 值類型 - 單一值

aws:userid

使用此鍵來將申請者的主體識別碼與您在政策中所指定的 ID 進行比較。針對 IAM 使用者，請求內容值是使用者 ID。針對 IAM 角色，此值的格式可能會不同。如需針對不同主體資訊顯示方式的詳細資訊，請參閱 如何指定主體。如需 `principal` 鍵值的具體範例，請參閱 主體索引鍵值。

- 可用性 – 此鍵會包含在所有請求的請求內容中，包括匿名請求。
- 資料類型 – 字串
- 值類型 - 單一值

aws:username

使用此鍵將申請者的使用者名稱與您在政策中所指定的使用者名稱進行比較。如需針對不同主體資訊顯示方式的詳細資訊，請參閱 如何指定主體。如需 `principal` 鍵值的具體範例，請參閱 主體索引鍵值。

- 可用性 – 此鍵一律會包含在 IAM 使用者的請求內容中。使用 AWS 帳戶根使用者 或 IAM 角色提出的匿名請求和請求不包含此金鑰。使用 IAM Identity Center 憑證提出的請求不會在內容中包含此鍵。
- 資料類型 – 字串
- 值類型 - 單一值

角色工作階段的屬性

使用下列條件索引鍵來比較產生工作階段時角色工作階段的屬性。只有在主體使用角色工作階段或聯合身分使用者主體憑證發出請求時，這些條件索引鍵方才可用。這些條件索引鍵的值會內嵌在角色的工作階段權杖中。

角色是一種主體。您也可以使用 主體的屬性 區段中的條件索引鍵，在角色提出請求時評估角色的屬性。

aws:AssumedRoot

使用此金鑰來檢查是否使用 AssumeRoot 提出請求。`AssumeRoot` 會傳回特權根使用者工作階段的短期憑證，可以用來對組織中的成員帳戶採取特權動作。如需詳細資訊，請參閱集中管理成員帳戶的根存取權。

- 可用性：只有在主體使用 AssumeRoot 的憑證發出請求時，請求內容中才會包含此金鑰。

- 資料類型 – 布林值
- 值類型 - 單一值

在下列範例中，當用作服務控制政策時， 拒絕使用 AWS Organizations 成員帳戶中根使用者的長期登入資料。不過，該政策不會拒絕 `AssumeRoot` 工作階段執行 `AssumeRoot` 工作階段允許的動作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement":[
       {
          "Effect":"Deny",
          "Action":"*",
          "Resource": "*",
          "Condition":{
             "ArnLike":{
                "aws:PrincipalArn":[
                   "arn:aws:iam::*:root"
                ]
             },
             "Null":{
                "aws:AssumedRoot":"true"
             }
          }
       }
    ]
 }
```

aws:FederatedProvider

使用此鍵來將主體的發行身分提供者 (IdP) 與您在政策中所指定的 IdP 進行比較。這表示使用 `AssumeRoleWithWebIdentity` AWS STS 操作擔任的 IAM 角色。使用產生角色工作階段的暫時憑證提出要求時，要求內容會識別驗證原始聯合身分的 IdP。

- 可用性 – 此金鑰存在於使用 OpenID Connect (OIDC) 提供者擔任的角色工作階段中，以及使用 OIDC 提供者呼叫 `AssumeRoleWithWebIdentity` 時的角色信任政策中。
- 資料類型 – 字串*
- 值類型 - 單一值

* 資料類型取決於 IdP：

- 如果您使用的是內建 AWS IdP，例如 Amazon Cognito，則金鑰值會是字串。索引鍵值可能看起來像：`cognito-identity.amazonaws.com`。
- 如果您使用的 IdP 不是內建的 AWS，例如 GitHub或 Amazon EKS，則金鑰值將為 ARN。索引鍵值可能看起來像：`arn:aws:iam::111122223333:oidc-provider/oidc.eks.region.amazonaws.com/id/OIDC_Provider_ID`。

如需有關外部 IdP 和 `AssumeRoleWithWebIdentity` 的詳細資訊，請參閱常用案例。如需詳細資訊，請參閱角色工作階段主體。

aws:TokenIssueTime

使用此鍵來將發行暫時性安全憑證的日期和時間與您在政策中所指定的日期和時間進行比較。

- 可用性 – 只有在主體使用暫時憑證發出請求時，請求內容中才會包含此鍵。金鑰不存在於使用存取金鑰提出的 AWS CLI API AWS 或 AWS SDK 請求中。
- 資料類型 – 日期
- 值類型 - 單一值

若要了解哪些服務支援使用暫時憑證，請參閱AWS 使用 IAM 的 服務。

aws:SignInSessionArn

使用此金鑰來比較 AWS 您的登入OAuth工作階段與您在政策中指定的工作階段。

當您使用 OAuth型流程，例如 AWS CLI 登入 (`aws login`) 或 AWS MCP 伺服器時， AWS 登入會在發行的登入資料中包含登入工作階段 ARN。 AWS 登入會將該 ARN 傳播到使用OAuth存取字符提出的後續請求。使用此金鑰，您可以將 API 活動與原始登入工作階段建立關聯，並將 IAM 政策套用至個別工作階段。

對於互動式OAuth流程，重新整理字符會決定登入工作階段 ARN 生命週期。在重新整理期間發出的新存取字符會繼續攜帶相同的登入工作階段 ARN。

對於非互動式OAuth流程，存取字符會決定登入工作階段 ARN 生命週期。每個新的存取字符都包含新的登入工作階段 ARN。

- 可用性 – 當請求來自 AWS 登入OAuth工作階段時，此金鑰會包含在請求內容中。此金鑰不適用於主控台工作階段。

- 資料類型 – ARN
- 值類型 - 單一值

當您比較 ARN 時，請使用 ARNs 運算子而非字串運算子。

下列政策會拒絕與特定 AWS 登入工作階段相關聯的所有動作，但來自其他作用中工作階段的請求會繼續：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenySpecificSignInSession",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:SignInSessionArn": "arn:aws:signin:us-east-1:111122223333:session/
session-id"
        }
      }
    }
  ]
}
```

aws:MultiFactorAuthAge

使用此鍵來將請求主體獲得授權使用 MFA 以來的秒數，與您在政策中所指定的數字進行比較。如需有關 MFA 的詳細資訊，請參閱 AWS IAM 中的多重要素驗證。

> **Important**
>
> 此條件金鑰不適用於使用存取金鑰簽署 CLI、 AWS API 或 AWS SDK AWS 請求的聯合身分或請求。若要進一步了解如何使用臨時安全憑證將 MFA 保護新增至 API 操作，請參閱 透過 MFA 實現安全的 API 存取。
>
> 若要檢查 MFA 是否用於驗證 IAM 聯合身分，您可以將身分提供者的身分驗證方法 AWS 作為工作階段標籤傳遞至 。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。若要針對 IAM Identity Center 身分強制執行 MFA，您可以啟用存取控制的屬性，以使用身分提供者的身分驗證方法將 SAML 聲明宣告傳遞至 IAM Identity Center。

- 可用性：只有在主體使用臨時安全憑證發出請求時，請求內容中才會包含此金鑰。含有 MFA 條件的政策可連接到：
  - IAM 使用者或群組
  - 資源，例如 Amazon S3 儲存貯體、Amazon SQS 佇列或者 Amazon SNS 主題
  - 可由使用者擔任的 IAM 角色的信任政策
- 資料類型 – 數值
- 值類型 - 單一值

aws:MultiFactorAuthPresent

使用此金鑰來檢查是否使用了多重要素驗證 (MFA) 來驗證發出請求的臨時安全憑證。

> **Important**
>
> 此條件金鑰不適用於使用存取金鑰簽署 CLI、 AWS API 或 AWS SDK AWS 請求的聯合身分或請求。若要進一步了解如何使用臨時安全憑證將 MFA 保護新增至 API 操作，請參閱 透過 MFA 實現安全的 API 存取。
>
> 若要檢查 MFA 是否用於驗證 IAM 聯合身分，您可以將身分提供者的身分驗證方法 AWS 作為工作階段標籤傳遞至 。如需詳細資訊，請參閱在 中傳遞工作階段標籤 AWS STS。若要針對 IAM Identity Center 身分強制執行 MFA，您可以啟用存取控制的屬性，以使用身分提供者的身分驗證方法將 SAML 聲明宣告傳遞至 IAM Identity Center。

- 可用性 – 只有在主體使用暫時憑證發出請求時，請求內容中才會包含此鍵。含有 MFA 條件的政策可連接到：
  - IAM 使用者或群組
  - 資源，例如 Amazon S3 儲存貯體、Amazon SQS 佇列或者 Amazon SNS 主題
  - 可由使用者擔任的 IAM 角色的信任政策
- 資料類型 – 布林值
- 值類型 - 單一值

臨時憑證可用來驗證擁有 AssumeRole 或 GetSessionToken 臨時權杖的 IAM 角色和 IAM 使用者，以及 AWS 管理主控台使用者。

IAM 使用者存取金鑰是長期憑證，但在某些情況下， 會代表 IAM 使用者 AWS 建立臨時憑證以執行操作。在這些情況下，aws:MultiFactorAuthPresent 鍵會出現在請求中，並設為 false 值。在兩種常見的情況下可能會發生此情況：

- 中的 IAM 使用者 AWS 管理主控台 不小心使用臨時登入資料。使用者會使用他們的使用者名稱和密碼 (長期憑證) 登入主控台。不過，主控台會在背景中代表使用者產生臨時憑證。
- 如果 IAM 使用者呼叫 AWS 服務，服務會重複使用使用者的登入資料，對不同的服務提出另一個請求。例如，呼叫 Athena 存取 Amazon S3 儲存貯體時，或使用 CloudFormation 建立 Amazon EC2 執行個體時。對於後續請求， AWS 會使用臨時登入資料。

若要了解哪些服務支援使用暫時憑證，請參閱AWS 使用 IAM 的 服務。

aws:MultiFactorAuthPresent 鍵在使用長期憑證 (例如使用者存取金鑰對) 呼叫 API 或 CLI 命令時，便不會出現。因此，我們建議您在檢查此鍵時使用條件運算子的 ...IfExists 版本。

請務必了解，下列 Condition 元素不是檢查是否使用 MFA 針對請求進行身分驗證的可靠方法。

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

這個 Deny 效果、Bool 元素與 false 值的組合會拒絕可使用 MFA 驗證但卻未驗證的請求。這僅適用於支援使用 MFA 的暫時憑證。此陳述式不會拒絕存取使用長期憑證所提出的請求，或是使用 MFA 進行身分驗證的請求。請小心使用本範例，因為其邏輯十分複雜，而且未測試是否實際使用 MFA 身分驗證。

此外，請不要使用 Deny 效果、Null 元素與 true 的組合，因為其行為相同，而邏輯更為複雜。

建議的組合

我們建議您使用 BoolIfExists 運算子來檢查是否使用 MFA 驗證請求。

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

這個 Deny、BoolIfExists 與 false 的組合會拒絕未使用 MFA 驗證的請求。特別地是，它會拒絕來自未包含 MFA 之暫時憑證的請求。它也會拒絕使用長期登入資料提出的請求，例如使用存取金鑰進行的 AWS CLI 或 AWS API 操作。*IfExists 運算子會檢查 aws:MultiFactorAuthPresent 鍵

是否存在以及它是否可能存在，如其存在所指出。當您想要拒絕任何未使用 MFA 驗證的請求時，請使用此項目。這更安全，但可能會破壞任何使用存取金鑰來存取 或 API 的程式碼 AWS CLI 或 AWS 指令碼。

替代組合

您也可以使用 BoolIfExists運算子來允許使用長期憑證提出的 MFA 驗證請求和 AWS CLI 或 AWS API 請求。

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

此條件會比對鍵是否存在，或者鍵是否不存在。這個 Allow、BoolIfExists 與 true 的組合允許使用 MFA 驗證的請求，或允許無法使用 MFA 驗證的請求。這表示請求者使用其長期存取金鑰時，允許 AWS CLI AWS API 和 AWS SDK 操作。此組合不允許來自暫時憑證，且能夠包含 MFA 的請求但未這麼做的請求。

當您使用 IAM 主控台視覺化編輯器建立政策並選擇 MFA required (需要 MFA) 時，即會套用此組合。此設定需要 MFA 來存取主控台，但允許不使用 MFA 進行程式設計存取。

或者，您可以使用 Bool 運算子，僅在使用 MFA 驗證時允許程式設計和主控台請求。

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

這個 Allow、Bool 與 true 的組合只允許 MFA 驗證的請求。這僅適用於支援使用 MFA 的暫時憑證。此陳述式不允許存取使用長期存取鍵所提出的請求，或是使用暫時憑證但未使用 MFA 所提出的請求。

請不要使用與下列內容類似的政策建構來檢查是否存在 MFA 鍵：

```
#####   WARNING: USE WITH CAUTION   #####

"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

這個 Allow 效果、Null 元素與 false 值的組合只允許可使用 MFA 驗證的請求，不論是否實際驗證請求都一樣。這會允許所有使用暫時憑證所提出的請求，並拒絕長期憑證的存取。請小心使用本範例，因為未測試是否實際使用 MFA 身分驗證。

aws:ChatbotSourceArn

使用此金鑰，將主體設定的來源聊天組態 ARN 與您在頻道組態相關聯之 IAM 角色政策中指定的聊天組態 ARN 進行比較。可以根據 Amazon Q Developer 在聊天應用程式中啟動的擔任角色工作階段來授權請求。

- 可用性 – 每當擔任角色工作階段時，聊天應用程式中的 Amazon Q Developer 都會在請求內容中包含此索引鍵。金鑰值是聊天組態 ARN，例如當您從聊天頻道執行 AWS CLI 命令時。
- 資料類型 – ARN
- 值類型 - 單一值
- 範例值 – `arn:aws::chatbot::123456789021:chat-configuration/slack-channel/private_channel`

以下政策拒絕 Amazon S3 對來自 Slack 頻道的所有請求的指定儲存貯體提出請求。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleS3Deny",
            "Effect": "Deny",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnLike": {
                      "aws:ChatbotSourceArn": "arn:aws:chatbot::*:chat-
configuration/slack-channel/*"
                }
            }
        }
    ]
}
```

aws:Ec2InstanceSourceVpc

此鍵可識別要接收 Amazon EC2 IAM 角色憑證的 VPC。您可以在具有 aws:SourceVPC 全域鍵的政策中使用此鍵，以檢查是否從與接收憑證之 VPC (aws:Ec2InstanceSourceVpc) 相符的 VPC (aws:SourceVPC) 進行呼叫。

- 可用性 – 每當請求者使用 Amazon EC2 角色憑證簽署請求時，請求內容中會包含此鍵。它可用於 IAM 政策、服務控制政策、VPC 端點政策和資源政策。
- 資料類型 – 字串
- 值類型 - 單一值

此鍵可與 VPC 識別符值搭配使用，但是當與 aws:SourceVpc 內容鍵一起用作變數時最有用。只有在請求者使用 VPC 端點提出請求時，請求內容中才會包含 aws:SourceVpc 內容鍵。搭配使用 aws:Ec2InstanceSourceVpc 與 aws:SourceVpc 可讓您更廣泛地使用 aws:Ec2InstanceSourceVpc，因為其會比較通常一起變更的值。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
```

```
}
```

在上面的範例中，如果 aws:SourceVpc 值不等於 aws:Ec2InstanceSourceVpc 值，則會拒絕存取。政策聲明僅限於透過測試 ec2:SourceInstanceARN 條件鍵是否存在，作為 Amazon EC2 執行個體角色使用的角色。

此政策使用 aws:ViaAWSService AWS 允許 在代表 Amazon EC2 執行個體角色提出請求時授權請求。例如，當您從 Amazon EC2 執行個體向加密的 Amazon S3 儲存貯體提出請求時，Amazon S3 AWS KMS 會代表您呼叫 。提出請求時，某些金鑰不存在 AWS KMS。

aws:Ec2InstanceSourcePrivateIPv4

此鍵可識別接收 Amazon EC2 IAM 角色憑證之主要彈性網路介面的私有 IPv4 地址。您必須將此條件金鑰與其配套金鑰搭配使用，aws:Ec2InstanceSourceVpc以確保您具有 VPC ID 和來源私有 IP 的全域唯一組合。將此金鑰與 搭配使用aws:Ec2InstanceSourceVpc，以確保從交付憑證的相同私有 IP 地址提出請求。

- 可用性 – 每當請求者使用 Amazon EC2 角色憑證簽署請求時，請求內容中會包含此鍵。它可用於 IAM 政策、服務控制政策、VPC 端點政策和資源政策。
- 資料類型 – IP 位址
- 值類型 - 單一值

> **Important**
>
> 此鍵不應該單獨在 Allow 陳述式中使用。私有 IP 地址根據定義不是全域唯一的。每次使用 aws:Ec2InstanceSourcePrivateIPv4 鍵指定可從中使用 Amazon EC2 執行個體憑證的 VPC 時，都應該使用 aws:Ec2InstanceSourceVpc 鍵。

下列範例是服務控制政策 (SCP)，拒絕存取所有資源，除非請求透過與 角色登入資料相同的 VPC 中的 VPC 端點抵達。在此範例中，aws:Ec2InstanceSourcePrivateIPv4 會根據來源 IP 將憑證來源限制為特定執行個體。

JSON

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

```
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        }
    ]
}
```

aws:SourceIdentity

使用此鍵來將主體所設定的來源身分與您在政策中所指定的來源身分進行比較。

- 可用性 – 當使用任何 AWS STS 擔任角色 CLI 命令或 AWS STS `AssumeRole` API 操作擔任角色時，設定來源身分後，此金鑰會包含在請求內容中。

- 資料類型 – 字串
- 值類型 - 單一值

您可以在政策中使用此金鑰，允許 AWS 在擔任角色時設定來源身分的委託人在 中執行動作。AWS CloudTrail 中會顯示角色指定來源身分的活動。這可讓管理員更輕鬆地判斷角色執行動作的對象或對象 AWS。

與 sts:RoleSessionName 不同，在設定來源身分之後，就無法變更值。這會出現在由角色所採取的所有動作的請求內容中。當您使用工作階段憑證來擔任另一個角色時，此值仍然存在於後續角色工作階段。從另一個角色取得及擔任角色稱為角色鏈結。

當委託人最初設定來源身分，同時使用任何 AWS STS 擔任角色 CLI 命令或 AWS STS AssumeRole API 操作擔任角色時， sts:SourceIdentity金鑰會出現在請求中。針對具有來源身分集的角色工作階段所採取的任何動作，aws:SourceIdentity 鍵會出現在請求中。

下列在帳戶 111122223333 中的 CriticalRole 角色信任政策包含針對 aws:SourceIdentity 的條件，可防止沒有設定為 Saanvi 或 Diego 的來源身分的主體擔任角色。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

若要進一步了解有關使用來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。

ec2:RoleDelivery

使用此金鑰來比較已簽署請求中的執行個體中繼資料服務版本與 Amazon EC2 的 IAM 角色憑證。執行個體中繼資料服務會依據任何指定請求 (PUT 或 GET 標頭) 來區分 IMDSv1 及 IMDSv2 請求，這些對 IMDSv2 而言是唯一的內容，會出現在該請求中。

- 可用性：每當 Amazon EC2 執行個體建立角色工作階段時，請求內容中會包含此鍵。
- 資料類型 – 數值
- 值類型 - 單一值
- 範例值 – 1.0、2.0

您可以在每個執行個體上設定執行個體中繼資料服務 (IMDS)，此類本機程式碼或使用者必須使用 IMDSv2。當您指定必須使用該 IMDSv2 時，IMDSv1 則無法繼續運作。

- 執行個體中繼資料服務第 1 版 (IMDSv1) – 請求/回應方法
- 執行個體中繼資料服務第 2 版 (IMDSv2) – 工作階段導向方法

如需如何設定執行個體以使用 IMDSv2 的資訊，請參閱設定執行個體中繼資料選項。

在下列範例中，如果請求內容中的 ec2:RoleDelivery 值為 1.0 (IMDSv1)，則會拒絕存取。一般情況下，可套用此政策陳述式，如果 Amazon EC2 角色憑證未簽署該請求，則不具任何作用。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
```

```
        }
    ]
}
```

如需詳細資訊，請參閱使用執行個體中繼資料的範例政策。

ec2:SourceInstanceArn

使用此金鑰來比較產生角色工作階段之執行個體的 ARN。

- 可用性：每當 Amazon EC2 執行個體建立角色工作階段時，請求內容中會包含此鍵。
- 資料類型 – ARN
- 值類型 - 單一值
- 範例值 – arn：aws：ec2：us-west-2：111111111111：instance/instance-id

如需政策範例，請參閱允許特定執行個體檢視其他 AWS 服務中的資源。

glue:RoleAssumedBy

AWS Glue 服務會為每個 AWS API 請求設定此條件金鑰，其中 AWS Glue 代表客戶使用服務角色提出請求（不是透過任務或開發人員端點，而是直接透過 AWS Glue 服務）。使用此金鑰來驗證對 AWS 資源的呼叫是否來自 AWS Glue 服務。

- 可用性 – 當 代表客戶使用服務角色 AWS Glue 提出請求時，此金鑰會包含在請求內容中。
- 資料類型 – 字串
- 值類型 - 單一值
- 範例值 – 此金鑰始終設定為 `glue.amazonaws.com`。

下列範例新增條件，以允許 AWS Glue 服務從 Amazon S3 儲存貯體取得物件。

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
```

```
    }
}
```

glue:CredentialIssuingService

AWS Glue 服務會使用來自任務或開發人員端點的服務角色，為每個 AWS API 請求設定此金鑰。使用此金鑰來驗證對 AWS 資源的呼叫是否來自 AWS Glue 任務或開發人員端點。

- 可用性 – 當 從任務或開發人員端點 AWS Glue 發出請求時，此金鑰會包含在請求內容中。
- 資料類型 – 字串
- 值類型 - 單一值
- 範例值 – 此金鑰始終設定為 `glue.amazonaws.com`。

下列範例會新增條件，此條件會連接到 AWS Glue 任務所使用的 IAM 角色。這可確保根據角色工作階段是否用於 AWS Glue 任務執行期環境，允許/拒絕特定動作。

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

codebuild:BuildArn

此金鑰可識別已交付 IAM 角色登入資料的 CodeBuild 組建 ARN。使用此金鑰來驗證對 AWS 資源的呼叫是否來自特定的 CodeBuild 組建。

> **Note**
>
> 的完整值`codebuild:BuildArn`無法事先得知，因為它包含動態產生的組建 ID。

- 可用性 – 每當 CodeBuild 擔任的角色提出請求時，請求內容中都會包含此金鑰。
- 資料類型 – ARN

- 值類型 - 單一值
- 範例值 – arn：aws：codebuild：us-east-1：123456789012：build/MyBuildProject：12345678-1234-1234-1234-123456789012

下列範例允許特定 CodeBuild 組建`s3:GetObject`存取指定的儲存貯體。

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnLike": {
            "codebuild:BuildArn": "arn:aws:codebuild:us-east-1:123456789012:build/
MyBuildProject:*"
        }
    }
}
```

codebuild:ProjectArn

此金鑰可識別已將 IAM 角色登入資料交付至 CodeBuild 組建的 CodeBuild 專案 ARN。使用此金鑰來驗證對 AWS 資源的呼叫是否來自特定的 CodeBuild 專案。

- 可用性 – 每當請求是由 CodeBuild 擔任的角色提出時，請求內容中都會包含此金鑰。
- 資料類型 – ARN
- 值類型 - 單一值
- 範例值 – arn：aws：codebuild：us-east-1：123456789012：project/MyBuildProject

下列範例允許來自特定 CodeBuild 專案的任何組建`s3:GetObject`存取指定的儲存貯體。

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnEquals": {
            "codebuild:ProjectArn": "arn:aws:codebuild:us-east-1:123456789012:project/
MyBuildProject"
        }
```

```
    }
}
```

lambda:SourceFunctionArn

使用此金鑰來識別接收 IAM 角色憑證的 Lambda 函數 ARN。Lambda 服務會針對來自函數執行環境的每個 AWS API 請求設定此金鑰。使用此金鑰來驗證對 AWS 資源的呼叫是否來自特定 Lambda 函數的程式碼。Lambda 也會為來自執行環境外部的某些請求設定此金鑰，例如將日誌寫入到 CloudWatch 並將追蹤傳送至 X-Ray。

- 可用性 – 每當叫用 Lambda 函數程式碼時，此金鑰會包含在請求內容中。
- 資料類型 – ARN
- 值類型 - 單一值
- 範例值 – arn:aws:lambda:us-east-1:123456789012:function:TestFunction

下列範例允許一個特定 Lambda 函數對指定的儲存貯體進行 `s3:PutObject` 存取。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
}
```

如需詳細資訊，請參閱 AWS Lambda Developer Guide 中的 Working with Lambda execution environment credentials。

ssm:SourceInstanceArn

使用此金鑰來識別交付 IAM 角色登入資料的 AWS Systems Manager 受管執行個體 ARN。當請求來自具有與 Amazon EC2 執行個體設定檔關聯之 IAM 角色的受管執行個體時，則此條件索引鍵不存在。

- 可用性：每當角色憑證交付至 AWS Systems Manager 受管執行個體時，此金鑰會包含在請求內容中。
- 資料類型 – ARN
- 值類型 - 單一值
- 範例值 – arn：aws：ec2：us-west-2：111111111111：instance/instance-id

identitystore:UserId

使用此金鑰來比較已簽署請求中的 IAM Identity Center 人力資源身分與政策中指定的身分。

- 可用性：當請求的發起人是 IAM Identity Center 中的使用者時，會包含此金鑰。
- 資料類型 – 字串
- 值類型 - 單一值
- 範例值 – 94482488-3041-7026-18f3-be45837cd0e4

您可以使用、 AWS API 或 AWS SDK 向 UserId API 提出請求 AWS CLI，在 IAM Identity Center 中找到使用者的 UserId。 GetUserId

網路的屬性

使用下列條件索引鍵，將請求來源或傳遞通過的網路詳細資訊與您在政策中指定的網路屬性進行比較。

aws:SourceIp

使用此鍵來將申請者的 IP 地址和您在政策中所指定的 IP 地址進行比較。`aws:SourceIp` 條件鍵僅可用於公有 IP 地址範圍。

- 可用性 – 此鍵會包含在請求內容中，但當申請者使用 VPC 端點提出請求時則除外。
- 資料類型 – IP 位址

- 值類型 - 單一值

aws:SourceIp 條件鍵可用在政策中，只允許主體在指定的 IP 範圍內提出請求。

> **Note**
>
> aws:SourceIp 同時支援 IPv4 和 IPv6 IP 地址或者 IP 地址範圍。如需支援 IPv6 AWS 服務的清單，請參閱《Amazon VPC 使用者指南》中的AWS 服務 支援 IPv6 的 。

例如，您可以將以下身分型政策連接至 IAM 角色。如果使用者從指定的 IPv4 地址範圍進行呼叫，則此政策允許使用者將物件放入 amzn-s3-demo-bucket3 Amazon S3 儲存貯體。此政策也允許 AWS 服務使用 轉送存取工作階段 代表您執行此操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        }
    ]
}
```

如果您需要限制來自同時支援 IPv4 和 IPv6 定址的網路存取，您可以在 IAM 政策條件中包含 IPv4 和 IPv6 地址或 IP 地址範圍。如果使用者從指定的 IPv4 或 IPv6 地址範圍進行呼叫，則以下身分型政策允許使用者將物件放入 amzn-s3-demo-bucket3 Amazon S3 儲存貯體。在 IAM 政策中包含 IPv6 地址範圍之前，請確認 AWS 服務 您正在使用的 支援 IPv6。如需支援 IPv6 AWS 服務 的清單，請參閱《Amazon VPC 使用者指南》中的AWS 服務 支援 IPv6 的 。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

如果請求來自使用 Amazon VPC 端點的主機，則 `aws:SourceIp` 鍵不可用。建議您改為使用 VPC 專屬鍵，例如 aws:VpcSourceIp。如需有關使用 VPC 端點的詳細資訊，請參閱 AWS PrivateLink 指南中的 VPC 端點和 VPC 端點服務的身分與存取管理。

> **Note**
>
> AWS 服務 代表您 AWS 服務 呼叫其他 (service-to-service呼叫） 時，會修訂特定網路的授權內容。如果您的政策搭配 `Deny`陳述式使用此條件金鑰， AWS 服務 原則可能會意外遭到封鎖。為在滿足安全需求的同時讓 AWS 服務 可以正常運作，請透過新增值為 `false` 的 `aws:PrincipalIsAWSService` 條件索引鍵，將服務主體從 `Deny` 陳述式中排除。

aws:SourceVpc

使用此金鑰來檢查請求是否通過 VPC 端點所連接的 VPC。在政策中，您可以使用此鍵，只允許存取特定 VPC。如需詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的限制特定 VPC 的存取。

- 可用性 – 只有在申請者使用 VPC 端點提出請求時，請求內容中才會包含此鍵。
- 資料類型 – 字串
- 值類型 - 單一值

在政策中，您可以使用此金鑰來允許或限制對特定 VPC 端點的存取。

例如，您可以將下列身分型政策連接至 IAM 角色以拒絕 `PutObject` Amazon S3 `amzn-s3-demo-bucket3` 儲存貯體，除非請求是從指定的 VPC ID 提出 AWS 服務 ，或使用轉送存取工作階段 (FAS) 代表該角色提出請求。與 aws:SourceIp 不同，您必須使用 aws:ViaAWSService 或 aws:CalledVia 來允許 FAS 請求，因為不會保留初始請求的來源 VPC。

> **Note**
>
> 此政策不允許任何動作。將此政策與允許特定動作的其他政策結合使用。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "PutObjectIfNotVPCID",
      "Effect": "Deny",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-1234567890abcdef0"
        },
        "Bool": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

> **Note**
>
> AWS 建議您使用 aws:SourceVpc aws:SourceVpcArn，而不是您要鎖定的服務是否aws:SourceVpcArn支援 。如需支援的 服務清單，請參閱 aws：SourceVpcArn。

> **Note**
>
> AWS 服務 代表您 AWS 服務 呼叫其他 (service-to-service呼叫） 時，會修訂特定網路的授權內容。如果您的政策搭配 Deny陳述式使用此條件金鑰， AWS 服務 原則可能會意外遭到封鎖。為在滿足安全需求的同時讓 AWS 服務 可以正常運作，請透過新增值為 false 的 aws:PrincipalIsAWSService 條件索引鍵，將服務主體從 Deny 陳述式中排除。

aws:SourceVpcArn

使用此金鑰來驗證使用 VPC 端點傳遞請求的 VPC ARN。此金鑰會傳回 VPC 端點所連接的 VPC ARN。

- 可用性 – 透過 VPC 端點提出請求時，此金鑰會包含在受支援服務的請求內容中。若請求透過公有服務端點發出，則不會包含此鍵。下列服務支援此鍵：
  - AWS App Runner（ 字首：apprunner)
  - AWS Application Discovery Service（ 字首：discovery)
  - Amazon Athena (字首：athena)
  - AWS Cloud Map（ 字首：servicediscovery)
  - Amazon CloudWatch Application Insights (字首：applicationinsights)
  - AWS CloudFormation（ 字首：cloudformation)
  - Amazon Comprehend Medical (字首：comprehendmedical)
  - AWS Compute Optimizer（ 字首：compute-optimizer)
  - Amazon； Elastic Container Registry（ 字首：ecr)
  - Amazon Elastic Container Service（ 字首：ecs)
  - Amazon Kinesis Analytics（ 字首：kinesisanalytics)
  - Amazon Route 53（ 字首：route53)
  - AWS DataSync（ 字首：datasync)
  - Amazon Elastic Block Store (字首：ebs)

- Amazon EventBridge 排程器 (字首：scheduler)
- Amazon Data Firehose (字首：firehose)
- AWS HealthImaging （字首：medical-imaging)
- AWS HealthLake （字首：healthlake)
- AWS HealthOmics （字首：omics)
- AWS Identity and Access Management (iam:PassRole動作除外）（字首：iam)
- AWS IoT FleetWise （字首：iotfleetwise)
- AWS IoT Wireless （字首：iotwireless)
- AWS Key Management Service （字首：kms)
- AWS Lambda （字首：lambda)
- AWS Payment Cryptography （字首：payment-cryptography)
- Amazon Polly (字首：polly)
- AWS 私有憑證授權單位 （字首：acm-pca)
- AWS 資源回收筒 （字首：rbin)
- Amazon Rekognition (字首：rekognition)
- Service Quotas (字首：servicequotas)
- Amazon Simple Storage Service (字首：s3)
- AWS Storage Gateway （字首：storagegateway)
- AWS Systems Manager Incident Manager 聯絡人 （字首：ssm-contacts)
- Amazon Textract (字首：textract)
- Amazon Transcribe (字首：transcribe)
- AWS Transfer Family （字首：transfer)

- 資料類型 – ARN

  AWS 建議您在比較 ARN 時使用 ARN 運算子，而非字串運算子。 ARNs
- 值類型 - 單一值
- 範例值 – arn:aws:ec2:us-east-1:123456789012:vpc/vpc-0e9801d129EXAMPLE

以下是拒絕從 VPC 外部任何位置存取 amzn-s3-demo-bucket及其物件的儲存貯體政策範例vpc-1a2b3c4d。

```
{
```

```
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPC-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
       "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket",
                    "arn:aws:s3:::amzn-s3-demo-bucket/*"],
       "Condition": {
         "ArnNotEquals": {
           "aws:SourceVpcArn": "arn:aws:ec2:us-east-1:*:vpc/vpc-1a2b3c4d"
         }
       }
     }
   ]
}
```

> **Note**
>
> AWS 服務 代表您 AWS 服務 呼叫其他 (service-to-service呼叫） 時，會修訂特定網路的授權內容。如果您的政策搭配 Deny陳述式使用此條件金鑰， AWS 服務 原則可能會意外遭到封鎖。為在滿足安全需求的同時讓 AWS 服務 可以正常運作，請透過新增值為 `false` 的 `aws:PrincipalIsAWSService` 條件索引鍵，將服務主體從 Deny 陳述式中排除。

aws:SourceVpce

使用此鍵來將請求的 VPC 端點識別碼與您在政策中所指定的端點 ID 進行比較。

- 可用性 – 只有在申請者使用 VPC 端點提出請求時，請求內容中才會包含此鍵。
- 資料類型 – 字串
- 值類型 - 單一值

在政策中，您可以使用此鍵來將存取限制在特定 VPC 端點。如需詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的限制特定 VPC 的存取。與使用 類似aws:SourceVpc，您必須使用 aws:ViaAWSService或 aws:CalledVia 來允許 AWS 服務 使用轉送存取工作階段 (FAS) 提出的請求。這是因為不會保留初始請求的來源 VPC 端點。

> **Note**
>
> AWS 服務 代表您 AWS 服務 呼叫其他 (service-to-service呼叫） 時，會修訂特定網路的授權內容。如果您的政策搭配 Deny陳述式使用此條件金鑰， AWS 服務 原則可能會意外遭到封鎖。為在滿足安全需求的同時讓 AWS 服務 可以正常運作，請透過新增值為 `false` 的 `aws:PrincipalIsAWSService` 條件索引鍵，將服務主體從 Deny 陳述式中排除。

aws:VpceAccount

使用此金鑰將擁有 VPC 端點 AWS 的帳戶 ID 與您在政策中指定的帳戶 ID 進行比較。此條件索引鍵可確保請求來自特定帳戶擁有的 VPC 端點，協助您建立網路周邊控制。

- 可用性 – 若請求透過 VPC 端點發出，請求內容中會包含此鍵。若請求透過公有服務端點發出，則不會包含此鍵。

  下列服務支援此鍵：

  - AWS App Runner（字首：`apprunner`)
  - AWS Application Discovery Service（字首：`discovery`)
  - Amazon Athena (字首：`athena`)
  - AWS Cloud Map（字首：`servicediscovery`)
  - Amazon CloudWatch Application Insights (字首：`applicationinsights`)
  - AWS CloudFormation（字首：`cloudformation`)
  - Amazon Comprehend Medical (字首：`comprehendmedical`)
  - AWS Compute Optimizer（字首：`compute-optimizer`)
  - Amazon； Elastic Container Registry（字首：`ecr`)
  - Amazon Elastic Container Service（字首：`ecs`)
  - Amazon Kinesis Analytics（字首：`kinesisanalytics`)
  - Amazon Route 53（字首：`route53`)
  - AWS DataSync（字首：`datasync`)
  - Amazon DynamoDB（字首：`dynamodb`)
  - Amazon Elastic Block Store (字首：`ebs`)
  - Amazon EventBridge 排程器 (字首：`scheduler`)
  - Amazon Data Firehose (字首：`firehose`)

- Amazon GameLift（字首：gamelift）
- AWS HealthImaging（字首：medical-imaging）
- AWS HealthLake（字首：healthlake）
- AWS HealthOmics（字首：omics）
- AWS Identity and Access Management (iam:PassRole動作除外）（字首：iam）
- AWS IoT FleetWise（字首：iotfleetwise）
- AWS IoT Wireless（字首：iotwireless）
- AWS Key Management Service（字首：kms）
- AWS Lambda（字首：lambda）
- AWS Payment Cryptography（字首：payment-cryptography）
- Amazon Polly (字首：polly)
- AWS 私有憑證授權單位（字首：acm-pca）
- AWS 資源回收筒（字首：rbin）
- Amazon Rekognition (字首：rekognition)
- Service Quotas (字首：servicequotas)
- Amazon Simple Storage Service (字首：s3)
- AWS Shield（字首：shield）
- AWS Storage Gateway（字首：storagegateway）
- AWS Systems Manager Incident Manager 聯絡人（字首：ssm-contacts）
- Amazon Textract (字首：textract)
- Amazon Transcribe (字首：transcribe)
- AWS Transfer Family（字首：transfer）

- 資料類型 – 字串
- 值類型 - 單一值
- 範例值 – 123456789012

您可以使用此條件索引鍵來限制存取資源，使得請求必須透過帳戶擁有的 VPC 端點送達。下列 Amazon S3 儲存貯體政策範例允許在請求來自指定帳戶擁有的 VPC 端點時存取：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessToSpecificVpceAccountOnly",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/RoleName"
            },
            "Action": "s3:GetObject",
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
            "Condition": {
                "StringEquals": {
                    "aws:VpceAccount": "111122223333"
                }
            }
        }
    ]
}
```

> **Note**
>
> 選取的服務集目前支援此條件金鑰 AWS 。將此鍵與不受支援的服務搭配使用，可能會導致意外的授權結果。請務必將條件索引鍵的範圍限定為政策中受支援的服務。

有些 AWS 服務會在代表您時，從其網路存取您的資源。如果您使用這類服務，則需要編輯上述政策範例，以允許 AWS 服務從網路外部存取您的資源。如需有關根據請求來源強制執行存取控制時需要考量之存取模式的詳細資訊，請參閱使用資料周邊建立許可防護機制。

aws:VpceOrgID

使用此金鑰來比較 中 AWS Organizations 擁有 VPC 端點的組織識別符，而該 VPC 端點具有您在政策中指定的識別符。此條件索引鍵提供了最具可擴展性的網路周邊控制方法，可自動包含組織內帳戶擁有的所有 VPC 端點。

- 可用性 – 當透過 VPC 端點提出請求，且 VPC 端點擁有者帳戶是 AWS 組織的成員時，此金鑰會包含在請求內容中。如果透過其他網路路徑發出請求，或 VPC 端點擁有者帳戶不屬於組織，則不會包含此索引鍵。

  下列服務支援此鍵：

  - AWS App Runner（字首：apprunner）
  - AWS Application Discovery Service（字首：discovery）
  - Amazon Athena (字首：athena)
  - AWS B2B 資料交換（字首：b2bi）
  - AWS Cloud Map（字首：servicediscovery）
  - AWS CloudHSM（字首：cloudhsmv2）
  - Amazon CloudWatch Application Insights (字首：applicationinsights)
  - AWS CloudFormation（字首：cloudformation）
  - Amazon Cognito（字首：cognito-identity）
  - Amazon Comprehend Medical (字首：comprehendmedical)
  - AWS Compute Optimizer（字首：compute-optimizer）
  - AWS Database Migration Service（字首：dms）
  - AWS Directory Service Data（字首：ds-data）
  - Amazon DynamoDB（字首：dynamodb）
  - Amazon； Elastic Container Registry（字首：ecr）
  - Amazon Elastic Container Service（字首：ecs）
  - Amazon Kinesis Data Streams（字首：kinesis）
  - Amazon Kinesis Analytics（字首：kinesisanalytics）
  - Amazon Q Business（字首：qbusiness）
  - Amazon Route 53（字首：route53）
  - AWS DataSync（字首：datasync）
  - Amazon Elastic Block Store (字首：ebs)
  - Amazon EventBridge 排程器 (字首：scheduler)
  - Amazon Data Firehose (字首：firehose)
  - Amazon GameLift（字首：gamelift）
  - AWS HealthImaging（字首：medical-imaging）

- AWS HealthLake（字首：healthlake）
- AWS HealthOmics（字首：omics）
- AWS Identity and Access Management (iam:PassRole動作除外）（字首：iam）
- AWS Identity Store（字首：identitystore）
- AWS IoT FleetWise（字首：iotfleetwise）
- AWS IoT TwinMaker（字首：iottwinmaker）
- AWS IoT Wireless（字首：iotwireless）
- Amazon Keyspaces（適用於 Apache Cassandra）（字首：cassandra）
- AWS Key Management Service（字首：kms）
- AWS Lambda（字首：lambda）
- AWS Network Firewall（字首：network-firewall）
- AWS Payment Cryptography（字首：payment-cryptography）
- Amazon Pinpoint SMS 和語音服務（字首：sms-voice）
- Amazon Polly (字首：polly)
- AWS 價格表（字首：pricing）
- AWS 私有憑證授權單位（字首：acm-pca）
- AWS 資源回收筒（字首：rbin）
- Amazon Rekognition (字首：rekognition)
- Service Quotas (字首：servicequotas)
- Amazon Simple Email Service（字首：ses）
- AWS Shield（字首：shield）
- Amazon Simple Storage Service (字首：s3)
- Amazon Simple Queue Service（字首：sqs）
- AWS Storage Gateway（字首：storagegateway）
- AWS Systems Manager Incident Manager 聯絡人（字首：ssm-contacts）
- Amazon Textract (字首：textract)
- Amazon Transcribe (字首：transcribe)
- AWS Transfer Family（字首：transfer）
- Amazon WorkMail（字首：workmail）

- 資料類型 – 字串

- 值類型 - 單一值
- 範例值 – o-a1b2c3d4e5

下列資源控制政策範例拒絕存取您的 Amazon S3 和資源， AWS Key Management Service 除非請求是透過指定組織擁有的 VPC 端點，或來自代表您行事 AWS 的服務網路。有些組織可能需要進一步編輯此政策，以滿足其組織的需求，例如允許第三方合作夥伴存取。如需有關根據請求來源強制執行存取控制時需要考量之存取模式的詳細資訊，請參閱使用資料周邊建立許可防護機制。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceNetworkPerimeterVpceOrgID",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*",
        "kms:*"
      ],
      "Resource": "*",
      "Condition": {
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false",
          "aws:ViaAWSService": "false"
        },
        "StringNotEqualsIfExists": {
            "aws:VpceOrgID": "o-abcdef0123",
            "aws:PrincipalTag/network-perimeter-exception": "true"
        }
      }
    }
  ]
}
```

> **Note**
>
> 選取的服務集目前支援此條件金鑰 AWS 。將此鍵與不受支援的服務搭配使用，可能會導致意外的授權結果。請務必將條件索引鍵的範圍限定為政策中受支援的服務。

> **Note**
>
> AWS 服務 代表您 AWS 服務 呼叫其他 (service-to-service呼叫） 時，會修訂特定網路的授權內容。如果您的政策搭配 Deny陳述式使用此條件金鑰， AWS 服務 原則可能會意外遭到封鎖。為在滿足安全需求的同時讓 AWS 服務 可以正常運作，請透過新增值為 `false` 的 `aws:PrincipalIsAWSService` 條件索引鍵，將服務主體從 Deny 陳述式中排除。

aws:VpceOrgPaths

使用此金鑰，將發出請求的 VPC 端點 AWS Organizations 路徑與您在政策中指定的路徑進行比較。借助此條件索引鍵，您可以在組織單位層級實作網路周邊控制，當您在指定的組織單位內新增端點時，系統會隨 VPC 端點用量自動擴展。

- 可用性 – 如果透過 VPC 端點發出請求，且 VPC 端點擁有者帳戶是組織成員，請求內容中會包含此索引鍵。如果透過其他網路路徑發出請求，或 VPC 端點擁有者帳戶不屬於組織，則不會包含此索引鍵。

  下列服務支援此鍵：

  - AWS App Runner（字首：`apprunner`)
  - AWS Application Discovery Service（字首：`discovery`)
  - Amazon Athena (字首：`athena`)
  - AWS Cloud Map（字首：`servicediscovery`)
  - Amazon CloudWatch Application Insights (字首：`applicationinsights`)
  - AWS CloudFormation（字首：`cloudformation`)
  - Amazon Comprehend Medical (字首：`comprehendmedical`)
  - AWS Compute Optimizer（字首：`compute-optimizer`)
  - Amazon； Elastic Container Registry（字首：`ecr`)
  - Amazon Elastic Container Service（字首：`ecs`)
  - Amazon Kinesis Analytics（字首：`kinesisanalytics`)

- Amazon Route 53（字首：route53）
- AWS DataSync（字首：datasync）
- Amazon DynamoDB（字首：dynamodb）
- Amazon Elastic Block Store (字首：ebs)
- Amazon EventBridge 排程器 (字首：scheduler)
- Amazon Data Firehose (字首：firehose)
- Amazon GameLift（字首：gamelift）
- AWS HealthImaging（字首：medical-imaging）
- AWS HealthLake（字首：healthlake）
- AWS HealthOmics（字首：omics）
- AWS Identity and Access Management (`iam:PassRole`動作除外）（字首：iam）
- AWS IoT FleetWise（字首：iotfleetwise）
- AWS IoT Wireless（字首：iotwireless）
- AWS Key Management Service（字首：kms）
- AWS Lambda（字首：lambda）
- AWS Payment Cryptography（字首：payment-cryptography）
- Amazon Polly (字首：polly)
- AWS 私有憑證授權單位（字首：acm-pca）
- AWS 資源回收筒（字首：rbin）
- Amazon Rekognition (字首：rekognition)
- Service Quotas (字首：servicequotas)
- AWS Secrets Manager（字首：secretsmanager）
- AWS Security Token Service（字首：sts）
- Amazon Simple Storage Service (字首：s3)
- AWS Shield（字首：shield）
- AWS Storage Gateway（字首：storagegateway）
- AWS Systems Manager Incident Manager 聯絡人（字首：ssm-contacts）
- Amazon Textract (字首：textract)
- Amazon Transcribe (字首：transcribe)
- AWS Transfer Family（字首：transfer）

- 資料類型 – 字串 (清單)
- 值類型 - 多重值
- 範例值 – o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/

由於 aws:VpceOrgPaths 是多值條件索引鍵，因此使用此鍵時，您必須搭配字串條件運算子使用 ForAnyValue 或 ForAllValues 集運算子。下列 Amazon S3 儲存貯體政策範例僅在請求透過特定組織單位中帳戶擁有的 VPC 端點送達時允許存取：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessFromSpecificOrgPaths",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/RoleName"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:VpceOrgPaths": [
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/*",
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-33333333/*"
          ]
        }
      }
    }
  ]
}
```

> **Note**
>
> 選取的服務集目前支援此條件金鑰 AWS 。將此鍵與不受支援的服務搭配使用，可能會導致意外的授權結果。請務必將條件索引鍵的範圍限定為政策中受支援的服務。

有些 AWS 服務會在代表您時，從其網路存取您的資源。如果您使用這類服務，則需要編輯上述政策範例，以允許 AWS 服務從網路外部存取您的資源。如需有關根據請求來源強制執行存取控制時需要考量之存取模式的詳細資訊，請參閱使用資料周邊建立許可防護機制。

aws:VpcSourceIp

使用此鍵來將提出請求的 IP 地址與您在政策中所指定的 IP 地址進行比較。在政策中，只有在請求來自指定 IP 地址並且透過 VPC 端點時，此鍵才會相符。

- 可用性 – 只有在請求是使用 VPC 端點提出時，請求內容中才會包含此鍵。
- 資料類型 – IP 位址
- 值類型 - 單一值

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用端點政策控制 VPC 端點的存取。與使用 類似aws:SourceVpc，您必須使用 aws:ViaAWSService或 aws:CalledVia 來允許 AWS 服務 使用轉送存取工作階段 (FAS) 提出的請求。這是因為使用 VPC 端點提出的初始請求的來源 IP 並不會保留在 FAS 請求中。

> **Note**
>
> `aws:VpcSourceIp` 同時支援 IPv4 和 IPv6 IP 地址或者 IP 地址範圍。如需支援 IPv6 AWS 服務 的清單，請參閱《Amazon VPC 使用者指南》中的AWS 服務 支援 IPv6 的 。
> 一律使用 `aws:VpcSourceIp`條件金鑰搭配 `aws:SourceVpc`、 `aws:SourceVpce`或 `aws:SourceVpcArn`條件金鑰。如果不這樣做，政策可能會允許來自使用相同或重疊 IP CIDR 之非預期 VPC 的 API 呼叫。這可能是因為來自兩個不相關 VPCs IP CIDRs 可能相同或重疊。

> **Note**
>
> AWS 服務 代表您 AWS 服務 呼叫其他 (service-to-service呼叫） 時，會修訂特定網路的授權內容。如果您的政策搭配 `Deny`陳述式使用此條件金鑰， AWS 服務 原則可能會意外遭到封鎖。為在滿足安全需求的同時讓 AWS 服務 可以正常運作，請透過新增值為 `false` 的 `aws:PrincipalIsAWSService` 條件索引鍵，將服務主體從 `Deny` 陳述式中排除。

資源的屬性

使用下列條件索引鍵，將請求目標的資源詳細資訊與您在政策中指定的資源屬性進行比較。

aws:ResourceAccount

使用此鍵來比較請求的資源所有者的 AWS 帳戶 ID 與政策中的資源帳戶。然後，根據擁有該資源的帳戶來允許或拒絕對該資源的存取。

- 可用性 – 此鍵一律會包含在大部分服務動作的請求內容中。下列動作不支援此鍵：
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store – 所有動作
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents` – 如果該事件匯流排在 2023 年 3 月 2 日之前設定為跨帳戶 EventBridge 目標，則 EventBridge `PutEvents` 會呼叫另一個帳戶中的事件匯流排。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的授予許可以便允許來自其他 AWS 帳戶的事件。
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - Amazon OpenSearch Service
    - `es:AcceptInboundConnection`

  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
  - AWS Security Hub CSPM
    - `securityhub:AcceptAdministratorInvitation`
- 資料類型 – 字串
- 值類型 - 單一值

> **Note**
>
> 如需上述不受支援動作的其他考量事項，請參閱資料周邊政策範例儲存庫。

此金鑰等於帳戶 AWS 帳戶 ID，其中包含請求中評估的資源。

對於您帳戶中的大部分資源，ARN 包含該資源的擁有者帳戶 ID。對於某些資源 (例如 Amazon S3 儲存貯體)，資源 ARN 不包含帳戶 ID。以下兩個範例顯示在 ARN 中具有帳戶 ID 的資源與沒有帳戶 ID 的 Amazon S3 ARN 之間的差異：

- `arn:aws:iam::123456789012:role/AWSExampleRole` – 在帳戶 123456789012 中建立和擁有的 IAM 角色。
- `arn:aws:s3:::amzn-s3-demo-bucket2` – 在帳戶 111122223333 中建立和擁有 Amazon S3 儲存貯體，不會顯示在 ARN 中。

使用 AWS 主控台、API 或 CLI 來尋找您的所有資源和對應的 ARNs。

您撰寫的政策會根據資源擁有者的帳戶 ID 拒絕資源許可。例如，如果指定資源不屬於指定帳戶，下列身分型政策會拒絕存取這些資源。

若要使用此政策，請將斜體預留位置文字取代為您的帳戶資訊。

> **Important**
>
> 此政策不允許任何動作。相反，它使用 Deny 效果，明確拒絕存取陳述式中列出的不屬於所列帳戶的所有資源。將此政策與允許存取特定資源的其他政策結合使用。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

除非指定的 AWS 帳戶 擁有資源，否則此政策會拒絕存取特定 AWS 服務的所有資源。

> **Note**
>
> 有些 AWS 服務 需要存取在另一個 中託管的 AWS 擁有資源 AWS 帳戶。在身分型政策中使用 `aws:ResourceAccount`，可能會影響您的身分存取這些資源。

某些 AWS 服務，例如 AWS Data Exchange，依賴存取 外部的資源 AWS 帳戶 進行正常操作。如果在政策中使用元素 `aws:ResourceAccount`，請包含其他陳述式來建立這些服務的豁免。範例政策 AWS：拒絕存取您帳戶外部的 Amazon S3 資源，但 AWS Data Exchange 示範如何在定義服務所擁有資源的例外狀況時，根據資源帳戶拒絕存取。

使用此政策範例作為建立您自己的自訂政策的範本。如需詳細資訊，請參閱您的服務文件。

aws:ResourceOrgPaths

使用此金鑰來比較存取資源的 AWS Organizations 路徑與政策中的路徑。在政策中，此條件索引鍵會確保資源屬於指定組織根帳戶或 AWS Organizations中組織單位內的帳戶成員。 AWS Organizations 路徑是 Organizations 實體結構的文字表示法。如需有關使用和了解路徑的詳細資訊，請參閱 了解 AWS Organizations 實體路徑。

- 可用性 – 只有在擁有資源的帳戶是組織成員時，請求內容中才會包含此鍵。此全域條件鍵不支援下列動作：
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store – 所有動作
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents` – 如果該事件匯流排在 2023 年 3 月 2 日之前設定為跨帳戶 EventBridge 目標，則 EventBridge `PutEvents` 會呼叫另一個帳戶中的事件匯流排。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的授予許可以便允許來自其他 AWS 帳戶的事件。
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - Amazon OpenSearch Service
    - `es:AcceptInboundConnection`

  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
  - AWS Security Hub CSPM
    - `securityhub:AcceptAdministratorInvitation`
- 資料類型 – 字串 (清單)
- 值類型 - 多重值

> **Note**
>
> 如需上述不受支援動作的其他考量事項，請參閱資料周邊政策範例儲存庫。

`aws:ResourceOrgPaths` 是多重值條件鍵。多值鍵在請求內容中可以擁有多個值。使用此鍵時，您必須使用 `ForAnyValue` 或 `ForAllValues` 設定運算子搭配字串條件運算子。如需有關多重值條件鍵的詳細資訊，請參閱 用於多值內容索引鍵的集運算子。

例如，下列條件會對屬於組織 `o-a1b2c3d4e5` 的資源傳回 `True`。當您包含萬用字元時，您必須使用 StringLike 條件運算子。

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

針對具有 OU ID `ou-ab12-11111111` 的資源，下列條件會傳回 `True`。該條件會比對連接至 OU ou-ab12-11111111 或任何子 OU 的帳戶所擁有的資源。

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

下列條件會對直接連接至 OU ID `ou-ab12-22222222` 但未連接至子 OU 的帳戶所擁有的資源傳回 `True`。下列範例會使用 StringEquals 條件運算子來指定 OU ID 的精確比對要求，而非萬用字元比對。

```
"Condition": { "ForAnyValue:StringEquals" : {
```

```
    "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

> **Note**
>
> 有些 AWS 服務 需要存取在另一個 中託管的 AWS 擁有資源 AWS 帳戶。在身分型政策中使用 `aws:ResourceOrgPaths`，可能會影響您的身分存取這些資源。

某些 AWS 服務，例如 AWS Data Exchange，依賴存取 外部的資源 AWS 帳戶 進行正常操作。如果在政策中使用 `aws:ResourceOrgPaths` 鍵，請包含其他陳述式來建立這些服務的豁免。範例政策 AWS：拒絕存取您帳戶外部的 Amazon S3 資源，但 AWS Data Exchange 示範如何在定義服務所擁有資源的例外狀況時，根據資源帳戶拒絕存取。您可以建立類似的政策，使用 `aws:ResourceOrgPaths` 鍵來限制組織單位 (OU) 內資源的存取，同時考量服務擁有的資源。

使用此政策範例作為建立您自己的自訂政策的範本。如需詳細資訊，請參閱您的服務文件。

aws:ResourceOrgID

使用此金鑰來比較所 AWS Organizations 請求資源所屬組織的識別符與政策中指定的識別符。

- 可用性 – 只有在擁有資源的帳戶是組織成員時，請求內容中才會包含此鍵。此全域條件鍵不支援下列動作：
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store – 所有動作
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`

    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents` – 如果該事件匯流排在 2023 年 3 月 2 日之前設定為跨帳戶 EventBridge 目標，則 EventBridge `PutEvents` 會呼叫另一個帳戶中的事件匯流排。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的授予許可以便允許來自其他 AWS 帳戶的事件。
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - Amazon OpenSearch Service
    - `es:AcceptInboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
  - AWS Security Hub CSPM
    - `securityhub:AcceptAdministratorInvitation`
- 資料類型 – 字串
- 值類型 - 單一值

> **Note**
>
> 如需上述不受支援動作的其他考量事項，請參閱資料周邊政策範例儲存庫。

此全域鍵會傳回指定請求的資源組織 ID。它可讓您建立規則，這些規則適用於身分型政策的 `Resource` 元素中指定的組織中的所有資源。您可以在條件元素中指定組織 ID。當您新增和移除帳戶時，包含 `aws:ResourceOrgID` 鍵的政策會自動包含正確的帳戶，您無需手動更新它。

例如，下列政策可避免主體將物件新增至 `policy-genius-dev` 資源，除非 Amazon S3 資源屬於與提出請求的主體相同的組織。

> **Important**
>
> 此政策不允許任何動作。相反，它使用 Deny 效果，明確拒絕存取陳述式中列出的不屬於所列帳戶的所有資源。將此政策與允許存取特定資源的其他政策結合使用。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
        "Effect": "Deny",
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::policy-genius-dev/*",
        "Condition": {
            "StringNotEquals": {
                "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
            }
        }
    }
}
```

> **Note**
>
> 有些 AWS 服務 需要存取在另一個 中託管的 AWS 擁有資源 AWS 帳戶。在身分型政策中使用 aws:ResourceOrgID，可能會影響您的身分存取這些資源。

某些 AWS 服務，例如 AWS Data Exchange，依賴存取 外部的資源 AWS 帳戶 進行正常操作。如果在政策中使用 aws:ResourceOrgID 鍵，請包含其他陳述式來建立這些服務的豁免。範例政策 AWS：拒絕存取您帳戶外部的 Amazon S3 資源，但 AWS Data Exchange 示範如何在定義服務所擁有資源的例外狀況時，根據資源帳戶拒絕存取。您可以建立類似的政策，使用 aws:ResourceOrgID 鍵來限制組織內資源的存取，同時考量服務擁有的資源。

使用此政策範例作為建立您自己的自訂政策的範本。如需詳細資訊，請參閱您的服務文件。

在下列影片中，進一步了解如何在政策中使用 aws:ResourceOrgID 條件鍵。

確保身分和網路只能用來存取受信任的資源。

aws:ResourceTag/tag-key

使用此鍵來將您在政策中所指定的標籤鍵值對與連接到資源的鍵值對進行比較。例如，您可以要求只在資源擁有連接標籤鍵 "Dept" 和值 "Marketing" 時才允許資源的存取。如需詳細資訊，請參閱控制對 AWS 資源的存取。

- 可用性 – 當所請求資源已連接標籤時，此鍵包含在請求內容中，或者包含在使用所連接標籤建立資源的請求中。只有在資源支援以標籤為基礎的授權時，才會傳回此鍵。每個標籤鍵值對都會有一個內容鍵。
- 資料類型 – 字串
- 值類型 - 單一值

此內容鍵的格式為 "aws:ResourceTag/*tag-key*":"*tag-value*"，其中 *tag-key* 和 *tag-value* 是標籤鍵值組。標籤索引鍵不區分大小寫。這表示如果您在政策的條件元素中指定 "aws:ResourceTag/TagKey1": "Value1"，則該條件會符合名為 TagKey1 或 tagkey1 的資源標籤鍵 (但不會同時符合兩者)。這些標籤索引鍵值對中的值需區分大小寫。這表示如果您在政策的條件元素中指定 "aws:ResourceTag/TagKey1": "Production"，該條件會符合名為 Production 的資源標籤值 (但不會符合 production 或 PRODUCTION)。

如需使用 aws:ResourceTag 鍵來控制 IAM 資源存取的範例，請參閱 控制對 AWS 資源的存取。

如需使用 aws:ResourceTag金鑰控制對其他 AWS 資源之存取的範例，請參閱 使用標籤控制對 AWS 資源的存取。

如需有關使用屬性型存取控制 (ABAC) 之 aws:ResourceTag 條件鍵的教程，請參閱 IAM 教學課程：定義根據標籤存取 AWS 資源的許可。

請求的屬性

使用下列條件索引鍵，將請求本身和請求內容的詳細資訊與您在政策中指定的請求屬性進行比較。

aws:CalledVia

使用此鍵可將政策中的服務與代表 IAM 主體 (使用者或角色) 提出請求的服務進行比較。當委託人向 AWS 服務提出請求時，該服務可能會使用委託人的登入資料向其他 服務提出後續請求。如果使用轉寄存取工作階段 (FAS) 發出請求，此索引鍵的值會設定為服務主體的值。aws:CalledVia 鍵包含代表主體提出請求的鏈結中，每個服務的排序清單。

如需詳細資訊，請參閱轉送存取工作階段。

- 可用性 – 支援 aws:CalledVia 的服務使用 IAM 主體的憑證向其他服務提出請求時，此鍵即出現在請求中。若服務使用服務角色或服務連結角色來代表主體進行呼叫，則此金鑰不存在。若主體直接進行呼叫，此鍵亦不存在。
- 資料類型 – 字串 (清單)
- 值類型 - 多重值

若要在政策中使用 aws:CalledVia 條件金鑰，您必須提供服務委託人，以允許或拒絕 AWS 服務請求。例如，您可以使用 從 Amazon DynamoDB 資料表 AWS CloudFormation 讀取和寫入。DynamoDB 接著會使用由 AWS Key Management Service (AWS KMS) 提供的加密。

若要在任何服務使用主體憑證提出請求時，允許或拒絕存取，請使用 aws:ViaAWSService 條件鍵。該條件索引鍵支援 AWS 服務。

aws:CalledVia 鍵是多值鍵。但是，您無法在條件中使用此鍵強制排序。使用上面的例子，使用者 1 向 CloudFormation發出請求，其中 CloudFormation會呼叫 DynamoDB，而 DynamoDB 呼叫 AWS KMS。這是三個獨立的請求。對 的最終呼叫是由使用者 1 透過 執行 CloudFormation ，然後 AWS KMS 由 DynamoDB 執行。
使用 aws：CalledVia 的範例。

在此情況下，請求環境中的 aws:CalledVia 鍵會依該排序包括 cloudformation.amazonaws.com 和 dynamodb.amazonaws.com。如果您只關心透過 DynamoDB 在請求鏈中的某處進行呼叫，您可以在政策中使用此條件索引鍵。

例如，下列政策允許管理名為 的 AWS KMS 金鑰my-example-key，但前提是 DynamoDB 是其中一個請求服務。ForAnyValue:StringEquals 條件運算子確保 DynamoDB 為呼叫服務之一。若主體直接呼叫 AWS KMS ，則條件會傳回 false，且此政策不允許該請求。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
```

```
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": [
                        "dynamodb.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

如果您想強制執行鏈結中進行第一個或最後一個呼叫的服務，可以使用 aws:CalledViaFirst 和 aws:CalledViaLast 鍵。例如，下列政策允許管理my-example-key名為 的金鑰 AWS KMS。只有在鏈結中包含多個請求時，才允許這些 AWS KMS 操作。第一個請求必須透過 提出 CloudFormation ，最後一個則透過 DynamoDB 提出。如果其他服務在鏈結中間發出請求，則仍允許該操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
}
```

當服務使用 IAM 主體憑證來呼叫另一個服務時，aws:CalledViaFirst 和 aws:CalledViaLast 鍵即會出現在請求中。它們會指出在請求鏈中進行呼叫的第一個和最後一個服務。例如，假設 CloudFormation 呼叫另一個名為 的服務X Service，它會呼叫 DynamoDB，然後呼叫 AWS KMS。對 的最終呼叫 AWS KMS 是透過 CloudFormation執行，然後User 1是 X Service，然後是 DynamoDB。它首先透過 呼叫 CloudFormation ，最後透過 DynamoDB 呼叫。
使用 aws：CalledViaFirst 和 aws：CalledViaLast 的範例。

aws:CalledViaFirst

使用此鍵可將政策中的服務與代表 IAM 主體 (使用者或角色) 提出請求的第一個服務進行比較。如需詳細資訊，請參閱 aws:CalledVia。

- 可用性 – 服務使用 IAM 主體的憑證向不同服務提出至少一個其他請求時，此鍵即出現在請求中。若服務使用服務角色或服務連結角色來代表主體進行呼叫，則此金鑰不存在。若主體直接進行呼叫，此鍵亦不存在。
- 資料類型 – 字串
- 值類型 - 單一值

aws:CalledViaLast

使用此鍵可將政策中的服務與代表 IAM 主體 (使用者或角色) 提出請求的最後一個服務進行比較。如需詳細資訊，請參閱 aws:CalledVia。

- 可用性 – 服務使用 IAM 主體的憑證向不同服務提出至少一個其他請求時，此鍵即出現在請求中。若服務使用服務角色或服務連結角色來代表主體進行呼叫，則此金鑰不存在。若主體直接進行呼叫，此鍵亦不存在。
- 資料類型 – 字串
- 值類型 - 單一值

aws:ViaAWSService

使用此金鑰來檢查 是否使用轉送存取工作階段 (FAS) 代表您向其他 服務 AWS 服務 提出請求。

當服務使用轉寄存取工作階段代表原始 IAM 主體發出請求時，請求內容索引鍵會傳回 `true`。主體直接進行呼叫時，請求環境鍵也會傳回 `false`。

- 可用性 – 此金鑰一律會包含在請求內容中。
- 資料類型 – 布林值
- 值類型 - 單一值

aws:ViaCustomerDomain

使用此金鑰來檢查委託人是否使用從暫時存取委派 (TAD) 取得的登入資料提出請求。當合作夥伴或廠商透過 TAD 字符交換取得臨時憑證時，我們會將此金鑰設定為請求合作夥伴的帳戶識別符。

在 AWS Identity and Access Management (IAM) 政策中使用此條件金鑰，將存取控制範圍限定為建立臨時存取委派的工作階段，而不會影響您自己的工作階段。例如，您可以使用此金鑰僅撤銷委派型工作階段，同時保留直接主控台存取。

- 可用性 – 當委託人使用透過 TAD 字符交換取得的登入資料時，此金鑰會出現在請求內容中。
- 資料類型 – 字串
- 值類型 - 單一值

下列範例顯示從特定合作夥伴帳戶透過暫時存取委派取得之工作階段的拒絕政策。此政策僅適用於指定時間之前發行的工作階段。此範例使用 `Deny` 陳述`Resource`式中 `Action`和 的萬用字元 (`*`)，這是從特定工作階段撤銷所有存取的標準模式。若要僅撤銷特定動作，請將萬用字元取代為適當的動作清單。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "DateLessThan": {
                    "aws:TokenIssueTime": "2026-07-30T01:00:37.819Z"
                },
                "StringEquals": {
```

```
                    "aws:ViaCustomerDomain": "123456789012"
                }
            }
        }
    ]
}
```

aws:CalledViaAWSMCP

使用此金鑰來比較政策中的服務與代表 IAM 主體（使用者或角色）提出請求的 AWS MCP 服務。當委託人向 AWS MCP 服務提出請求時，該服務會使用委託人的登入資料向其他服務提出後續請求。使用 AWS MCP 服務提出請求時，會以服務主體的值設定此金鑰。`aws:CalledViaAWSMCP` 金鑰包含代表委託人提出請求的 MCP 服務的服務委託人名稱。

- 可用性 – 當 AWS MCP 服務使用 IAM 主體的登入資料向 AWS 服務提出請求時，此金鑰會出現在請求中。若主體直接進行呼叫，此鍵亦不存在。
- 資料類型 – 字串
- 值類型 - 單一值

您可以使用此條件金鑰，根據啟動請求的特定 MCP 伺服器來允許或拒絕存取。例如，以下政策拒絕透過特定 MCP 伺服器啟動敏感刪除操作：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaSpecificMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaAWSMCP": "aws-mcp.amazonaws.com"
                }
            }
        }
    ]
```

```
}
```

aws:ViaAWSMCPService

使用此金鑰來檢查 AWS MCP 服務是否使用轉送存取工作階段 (FAS) AWS 代表您向其他服務提出請求。當 AWS MCP 服務代表原始 IAM 主體將請求轉送至 AWS 服務`true`時，請求內容金鑰會傳回。主體直接進行呼叫時，請求環境鍵也會傳回 `false`。

- 可用性 – 當 AWS MCP 伺服器代表 IAM 主體向下游 AWS 服務提出請求時，此金鑰會包含在請求內容中。
- 資料類型 – 布林值
- 值類型 - 單一值

您可以使用此金鑰來限制透過 MCP 伺服器執行的特定動作。例如，當透過任何 AWS MCP 伺服器啟動時，下列政策會拒絕敏感的刪除操作：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "aws:ViaAWSMCPService": "true"
                }
            }
        }
    ]
}
```

aws:CurrentTime

使用此鍵來將請求的日期和時間及您在政策中所指定的日期與時間進行比較。如要檢視使用此條件鍵的範例政策，請參閱 AWS：允許根據日期和時間進行存取。

- 可用性 – 此金鑰一律會包含在請求內容中。
- 資料類型 – 日期
- 值類型 - 單一值

aws:EpochTime

使用此鍵來將請求的日期和時間 (epoch 或 Unix 時間) 與您在政策中所指定的日期與時間進行比較。此鍵也接受自 1970 年 1 月 1 日以來的秒數。

- 可用性 – 此金鑰一律會包含在請求內容中。
- 資料類型 – 日期、數字
- 值類型 - 單一值

aws:referer

使用此鍵來將在用戶端瀏覽器中推薦請求的人員，與您在政策中所指定的推薦者進行比較。`aws:referer` 請求內容的值會由呼叫者在 HTTP 標頭中提供。當您選取網頁上的連結時，`Referer` 標頭會包含在 Web 瀏覽器請求中。`Referer` 標頭包含選取連結的網頁 URL。

- 可用性 – 只有在從瀏覽器中的網頁 URL 連結叫用對 AWS 資源的請求時，請求內容才會包含此金鑰。程式設計請求不會包含此鍵，因為它不會使用瀏覽器連結來存取 AWS 資源。
- 資料類型 – 字串
- 值類型 - 單一值

例如，您可以直接使用 URL 或使用直接 API 呼叫來存取 Amazon S3 物件。如需詳細資訊，請參閱直接使用 Web 瀏覽器的 Amazon S3 API 操作。當您從存在於網頁中的 URL 存取 Amazon S3 物件時，會在 `aws:referer` 中使用來源網頁的 URL。當您在瀏覽器中輸入 URL 來存取 Amazon S3 物件時，`aws:referer` 不存在。當您直接呼叫 API 時，`aws:referer` 也不存在。您可以使用政策中的 `aws:referer` 條件鍵來允許特定參照者提出的請求，例如公司網域中網頁上的連結。

> **Warning**
>
> 應該小心使用此鍵。包含公開已知推薦者標頭值相當危險。未授權方可以使用修改的或自訂瀏覽器來提供他們選擇的任何 `aws:referer` 值。因此， `aws:referer` 不應用於防止未經授權方提出直接 AWS 請求。它僅用於允許客戶保護其數位內容 (例如存放在 Amazon S3 中的內容)，使其不被未經授權的第三方網站參考。

aws:RequestedRegion

使用此金鑰來比較請求中呼叫 AWS 的區域與您在政策中指定的區域。您可以使用此全域條件鍵來控制可接受請求的區域。若要檢視每個服務的 AWS 區域，請參閱 中的服務端點和配額Amazon Web Services 一般參考。

- 可用性 – 此鍵一律會包含在請求內容中。
- 資料類型 – 字串
- 值類型 - 單一值

有些全域服務 (例如 IAM) 只會有單一端點。因為此端點實際位於美國東部 (維吉尼亞北部) 區域，所以一律會對 us-east-1 區域提出 IAM 呼叫。例如，如果您建立的政策拒絕存取所有服務 (如果請求的區域不是 us-west-2)，則 IAM 呼叫一律會失敗。若要檢視如何解決此問題的範例，請參閱帶拒絕的 NotAction。

> **Note**
>
> `aws:RequestedRegion` 條件鍵可讓您控制哪個端點叫用的服務，但不控制操作的影響。有些服務具有跨區域影響。
> 例如，Amazon S3 具有延及其他區域的 API 操作。
>
> - 您可以在一個區域 (受 `s3:PutBucketReplication` 條件鍵影響) 中叫用`aws:RequestedRegion`，但其他區域會根據複寫組態設定而受到影響。
> - 您可以調用 `s3:CreateBucket` 以在其他區域建立儲存貯體，並使用 `s3:LocationConstraint` 條件鍵控制適用的區域。

您可以使用此內容金鑰來限制存取指定區域集中 AWS 的服務。例如，下列政策可讓使用者在 AWS 管理主控台中檢視所有 Amazon EC2 執行個體。不過，只允許他們對愛爾蘭 (eu-west-1)、倫敦 (eu-west-2) 或巴黎 (eu-west-3) 的執行個體進行變更。

JSON

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

```
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
                "ec2:Run*",
                "ec2:Start*",
                "ec2:Stop*",
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

aws:RequestTag/tag-key

使用此鍵來將請求中傳遞的標籤鍵/值對與您在政策中所指定的標籤對進行比較。例如，您可以檢查請求是否包含標籤鍵 "Dept" 並且其具有值 "Accounting"。如需詳細資訊，請參閱在 AWS 請求期間控制存取。

- 可用性 – 在請求中傳遞標籤鍵/值對時，鍵會包含在請求內容中。在請求中傳遞多個標籤時，每個標籤鍵值對都會有一個內容鍵。
- 資料類型 – 字串
- 值類型 - 單一值

此內容鍵的格式為 "aws:RequestTag/*tag-key*":"*tag-value*"，其中 *tag-key* 和 *tag-value* 是標籤鍵值組。標籤索引鍵不區分大小寫。這表示如果您在政策的條件元素中指定 "aws:RequestTag/TagKey1": "Value1"，則該條件會符合名為 TagKey1 或 tagkey1 的要求標籤鍵 (但不會同時符合兩者)。這些標籤索引鍵值對中的值需區分大小寫。這表示如果您在政策的條件元素中指定 "aws:RequestTag/TagKey1": "Production"，該條件會符合名為 Production 的請求標籤值 (但不會符合 production 或 PRODUCTION)。

此範例顯示雖然鍵為單一值，但如果鍵不同，您仍然可以在請求中使用多個鍵/值對。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2::111122223333:instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

aws:TagKeys

使用此鍵來將請求中的標籤鍵與您在政策中所指定的鍵進行比較。當使用政策來控制使用標籤的存取時，建議您使用 aws:TagKeys 條件鍵來定義允許的標籤鍵。如需範例政策和詳細資訊，請參閱 the section called “根據標籤索引鍵控制存取權限”。

- 可用性 – 如果操作支援在請求中傳遞標籤，則此鍵會包含在請求內容中。
- 資料類型 – 字串 (清單)
- 值類型 - 多重值

此內容鍵的格式為 "aws:TagKeys":"*tag-key*"，其中 *tag-key* 是沒有值的標籤鍵清單 (例如，["Dept","Cost-Center"])。

由於您可以在請求中包含多個標籤鍵值對，因此請求內容可能會是多值請求。在這種情況下，您必須使用 ForAllValues 或 ForAnyValue 設定運算子。如需詳細資訊，請參閱用於多值內容索引鍵的集運算子。

有些服務支援標記搭配資源操作，如建立、修改或刪除資源。若要允許標記和操作作為單一呼叫，您必須建立一個政策，同時包括標記動作和資源修改動作。然後，您可以使用 aws:TagKeys 條件鍵以強制執行在請求中使用特定的標籤鍵。例如，若要在某人建立 Amazon EC2 快照時限制標籤，您必須在政策中包含 ec2:CreateSnapshot 建立動作和 ec2:CreateTags 標記動作。若要檢視適用於此案例 (使用 aws:TagKeys) 的政策，請參閱《Amazon EC2 使用者指南》中的使用標籤建立快照。

aws:SecureTransport

使用此金鑰來檢查請求是否使用 TLS 進行傳送。請求內容會傳回 true 或 false。在政策中，您可以允許只有在使用 TLS 傳送請求時，才能進行特定動作。

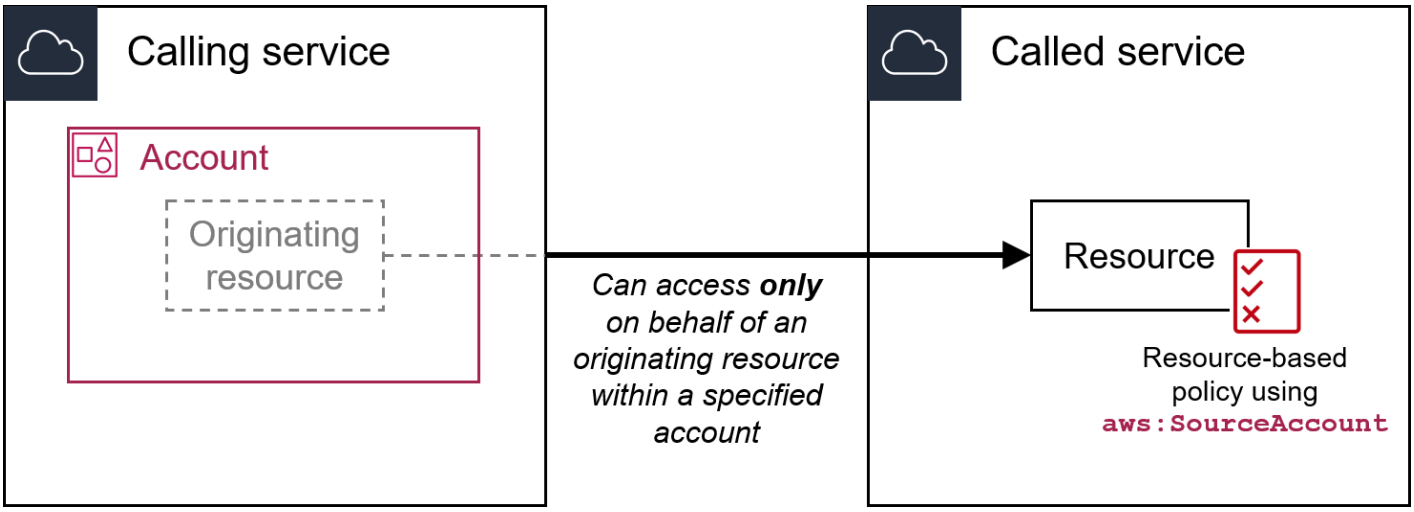
- 可用性 – 此鍵一律會包含在請求內容中。
- 資料類型 – 布林值
- 值類型 - 單一值

aws:SourceAccount

使用此金鑰來比較提出service-to-service請求之資源的帳戶 ID 與您在政策中指定的帳戶 ID，但僅限於服務 AWS 委託人提出請求時。

- 可用性 – 只有當 AWS 服務主體代表組態觸發服務對服務要求的資源直接呼叫您的資源時，才會將此鍵包含在請求內容中。發出呼叫服務會將原始資源的帳戶 ID 傳遞給被呼叫的服務。

> **Note**
>
> 此金鑰提供統一機制，以跨 AWS 服務強制執行跨服務混淆代理控制。不過，並非所有服務整合都需要使用此全域條件索引鍵。如需緩解跨服務混淆代理人風險之服務特定機制的詳細資訊，請參閱 AWS 服務 您使用的 文件。



- 資料類型 – 字串
- 值類型 - 單一值

您可以使用此條件金鑰，以協助確保呼叫服務只有在請求來自特定帳戶時，才能存取您的資源。例如，您可以連接下列資源控制政策 (RCP)，以拒絕服務主體對 Amazon S3 儲存貯體的請求，除非它們由指定帳戶中的資源所觸發。此政策只會對具有 `aws:SourceAccount` 金鑰 (`"Null": {"aws:SourceAccount": "false"}`) 的服務主體 (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) 提出的請求套用控制項，因此不需要使用此金鑰和主體呼叫的服務整合不會受到影響。如果請求內容中存在 `aws:SourceAccount` 金鑰，則 `Null` 條件將評估為 `true`，導致強制執行 `aws:SourceAccount` 金鑰。

JSON

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

```
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceAccount": "111122223333"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

在委託人為 AWS 服務 委託人的資源型政策中，使用 金鑰來限制授予服務的許可。例如，當 Amazon S3 儲存貯體設定為傳送通知至 Amazon SNS 主題時，Amazon S3 服務會叫用所有已設定事件的 `sns:Publish` API 操作。在允許 `sns:Publish` 操作的主題政策中，將條件鍵的值設定為 Amazon S3 儲存貯體的帳戶 ID。

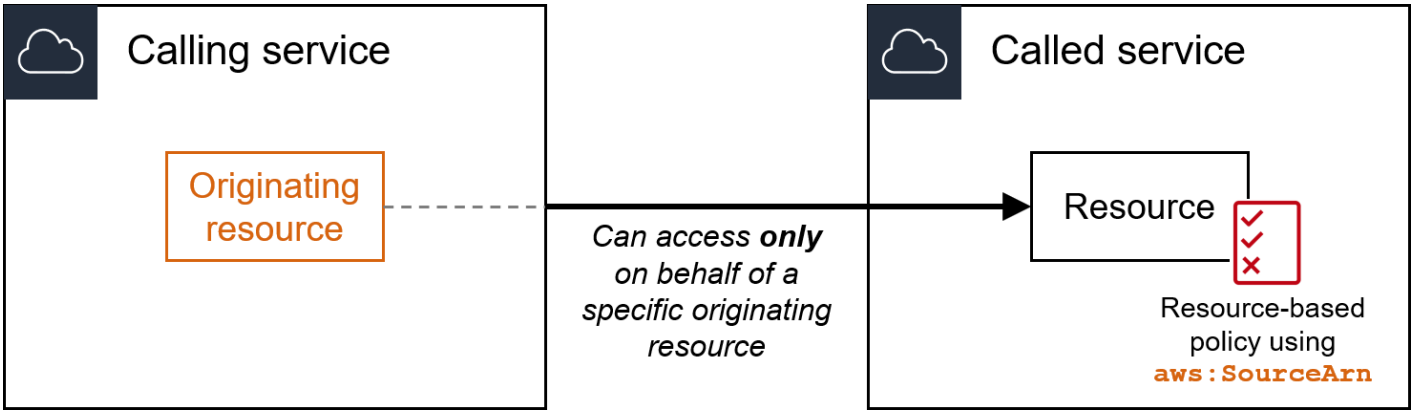
aws:SourceArn

使用此鍵，可將提出服務對服務請求之資源的 Amazon Resource Name (ARN) 與您在政策中所指定的 ARN 進行比較，但僅限於請求是由 AWS 服務主體提出的情況。如果來源的 ARN 包括帳戶 ID 時，則不需要搭配 `aws:SourceAccount` 使用 `aws:SourceArn`。

此鍵不適用提出請求的主體 ARN。請改用 `aws:PrincipalArn`。

- 可用性 – 只有當 AWS 服務主體代表組態觸發服務對服務要求的資源直接呼叫您的資源時，才會將此鍵包含在請求內容中。發出呼叫的服務會將原始資源的 ARN 傳遞給被呼叫的服務。

> **Note**
>
> 此金鑰提供統一機制，以跨 AWS 服務強制執行跨服務混淆代理控制。不過，並非所有服務整合都需要使用此全域條件索引鍵。如需緩解跨服務混淆代理人風險之服務特定機制的詳細資訊，請參閱 AWS 服務 您使用的 文件。



- 資料類型 – ARN

  AWS 建議您在比較 ARN 時使用 ARN 運算子，而非字串運算子。 ARNs
- 值類型 - 單一值

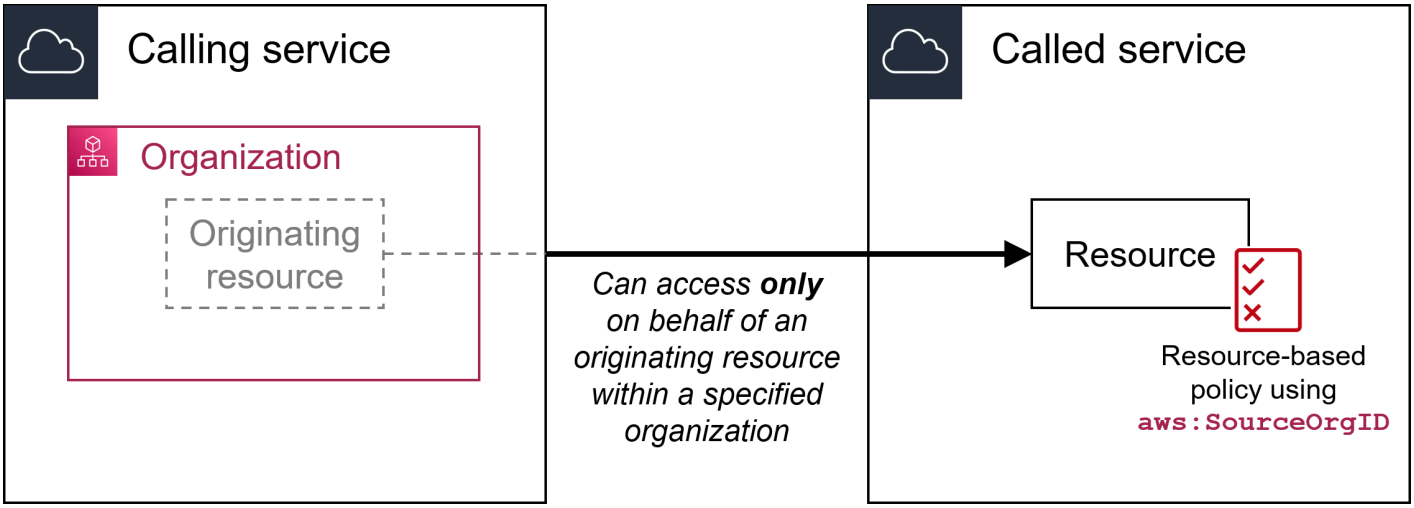
您可以使用此條件金鑰，以協助確保呼叫服務只有在請求來自特定資源時，才能存取您的資源。使用以資源為基礎的政策搭配 AWS 服務 委託人做為 時`Principal`，請將此條件索引鍵的值設定為您要限制存取之資源的 ARN。例如，當 Amazon S3 儲存貯體設定為傳送通知至 Amazon SNS 主題時，Amazon S3 服務會叫用所有已設定事件的 `sns:Publish` API 操作。在允許 `sns:Publish` 操作的主題政策中，將條件鍵的值設定為 Amazon S3 儲存貯體的 ARN。如需有關何時在資源型政策中使用此條件索引鍵的建議，請參閱您正在使用的 AWS 服務 文件。

aws:SourceOrgID

使用此金鑰來比較提出service-to-service請求之資源的組織 ID 與您在政策中指定的組織 ID，但僅限於服務 AWS 委託人提出請求時。當您對 AWS Organizations中的組織新增和移除帳戶時，包含 `aws:SourceOrgID` 鍵的政策會自動包含正確的帳戶，您無需手動更新政策。

- 可用性 – 只有當 AWS 服務主體代表組織成員帳戶所擁有的資源直接呼叫您的資源時，此鍵才會包含在請求內容中。發出呼叫的服務會將原始資源的組織 ID 傳遞給被呼叫的服務。

> **Note**
>
> 此金鑰提供統一機制，以跨 AWS 服務強制執行跨服務混淆代理控制。不過，並非所有服務整合都需要使用此全域條件索引鍵。如需緩解跨服務混淆代理人風險之服務特定機制的詳細資訊，請參閱 AWS 服務 您使用的 文件。



- 資料類型 – 字串
- 值類型 - 單一值

您可以使用此條件金鑰，以協助確保呼叫服務只有在請求來自特定組織時，才能存取您的資源。例如，您可以連接下列資源控制政策 (RCP)，以拒絕服務主體對 Amazon S3 儲存貯體的請求，除非它們是由指定 AWS 組織中的資源觸發。此政策只會對具有 `aws:SourceAccount` 金鑰 (`"Null": {"aws:SourceAccount": "false"}`) 的服務主體 (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) 提出的請求套用控制項，因此不需要使用該金鑰和主體呼叫的服務整合不會受到影響。如果請求內容中存在 `aws:SourceAccount` 金鑰，則 `Null` 條件將評估為 `true`，導致強制執行 `aws:SourceOrgID` 金鑰。我們在 `Null` 條件運算子中使用 `aws:SourceAccount` 而非 `aws:SourceOrgID`，因此如果請求來自不屬於組織的帳戶，則控制仍然適用。

JSON

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

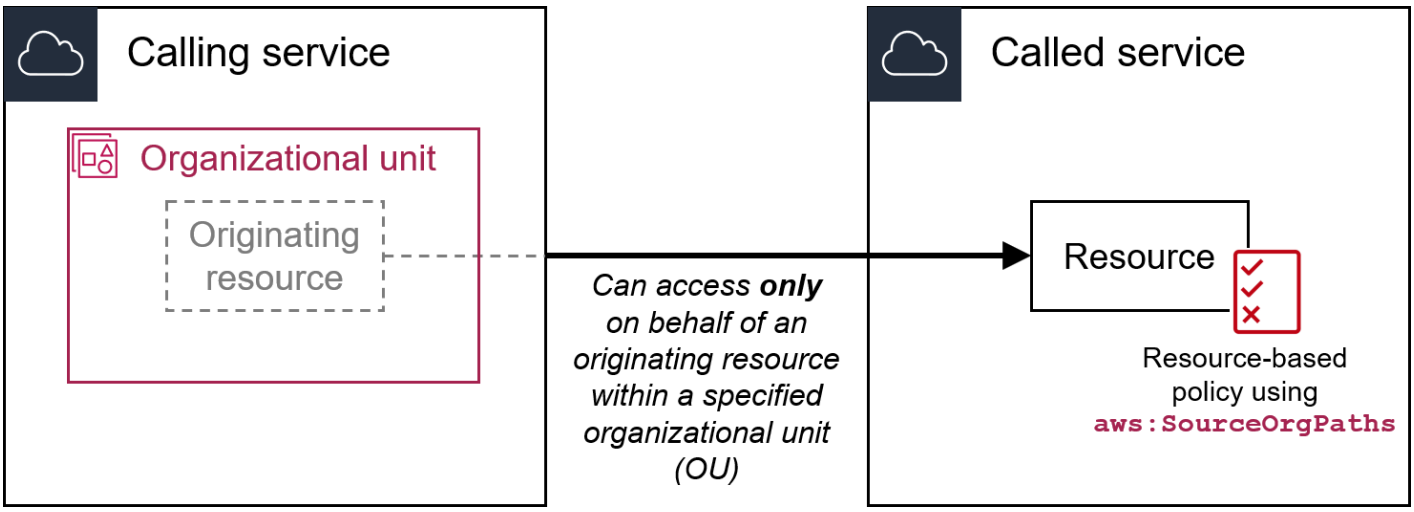
```
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-xxxxxxxxxx"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

aws:SourceOrgPaths

使用此金鑰來比較提出service-to-service請求的資源 AWS Organizations 路徑與您在政策中指定的組織路徑，但僅限於服務 AWS 主體提出請求時。 AWS Organizations 路徑是 AWS Organizations 實體結構的文字表示法。如需有關使用和了解路徑的詳細資訊，請參閱了解 AWS Organizations 實體路徑。

- 可用性 – 只有當 AWS 服務主體代表組織成員帳戶所擁有的資源直接呼叫您的資源時，此鍵才會包含在請求內容中。發出呼叫的服務會將原始資源的組織路徑傳遞給被呼叫的服務。

> **Note**
>
> 此金鑰提供統一機制，以跨 AWS 服務強制執行跨服務混淆代理控制。不過，並非所有服務整合都需要使用此全域條件索引鍵。如需緩解跨服務混淆代理人風險之服務特定機制的詳細資訊，請參閱 AWS 服務 您使用的 文件。



- 資料類型 – 字串 (清單)
- 值類型 - 多重值

使用此條件金鑰，可協助確保呼叫服務只有在請求來自特定的組織單位 (OU) 時，才能存取您的資源 AWS Organizations。

與 aws:SourceOrgID 類似，為了協助防止對不需要使用此索引鍵的服務整合造成影響，請使用 Null 條件運算子與 aws:SourceAccount 條件索引鍵，以便在請求來自不屬於組織的帳戶時仍可套用控制項。

```
{
      "Condition": {
        "ForAllValues:StringNotLikeIfExists": {
            "aws:SourceOrgPaths": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-
ab12-22222222/"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
}
```

aws:SourceOrgPaths 是多重值條件鍵。多值鍵在請求內容中可以擁有多個值。使用此鍵時，您必須使用 ForAnyValue 或 ForAllValues 設定運算子搭配字串條件運算子。如需有關多重值條件鍵的詳細資訊，請參閱 用於多值內容索引鍵的集運算子。

aws:UserAgent

使用此鍵來將申請者的用戶端應用程式與您在政策中指定的應用程式進行比較。

- 可用性 – 此鍵一律會包含在請求內容中。
- 資料類型 – 字串
- 值類型 - 單一值

> **Warning**
>
> 應該小心使用此鍵。由於aws:UserAgent 值由 HTTP 標頭中的呼叫者提供，未授權方可以使用修改的或自訂瀏覽器來提供他們選擇的任何 aws:UserAgent 值。因此， aws:UserAgent 不應用於防止未經授權方提出直接 AWS 請求。您可以使用它僅允許特定用戶端應用程式，而且只在測試政策之後才能使用。

aws:IsMcpServiceAction

使用此金鑰來驗證正在授權的動作是否為 MCP 服務動作。此金鑰不是指 MCP 服務對其他服務 AWS 採取的動作。

- 可用性 – 此金鑰包含在請求內容中，且只有在 MCP 服務授權 MCP 服務動作時，才會設為 True。
- 資料類型 – 布林值
- 值類型 - 單一值

其他跨服務條件鍵

AWS STS 支援 OIDC 聯合的 SAML 型聯合條件金鑰和跨服務條件金鑰。當使用 OIDC 或 SAML 聯合身分的使用者在其他 服務中執行 AWS 操作時，即可使用這些金鑰。

IAM 和 AWS STS 條件內容索引鍵

您可以使用 JSON 政策中的 Condition元素來測試包含在所有請求請求內容中的金鑰值 AWS 。這些鍵提供有關請求本身或請求參考的資源的資訊。在允許使用者請求的動作之前，您可以檢查鍵是否有

指定的值。這可讓您可以在 JSON 政策陳述式符合或不符合傳入請求時進行精細控制。如需有關如何 JSON 政策中使用 `Condition` 元素的資訊，請參閱 IAM JSON 政策元素：Condition。

本主題說明 IAM 服務（含 `iam:` 字首）和 AWS Security Token Service (AWS STS) 服務（含 `sts:`字首）定義和提供的金鑰。其他數個 AWS 服務也提供與該服務定義的動作和資源相關的服務特定金鑰。如需詳細資訊，請參閱 AWS 服務的動作、資源和條件金鑰。支援條件鍵的服務文件通常包含其他資訊。例如，如需可在 Amazon S3 資源政策中使用之金鑰的資訊，請參閱 Amazon Simple Storage Service 使用者指南中的 Amazon S3 政策鍵。

主題

- IAM 的可用鍵
- AWS OIDC 聯合的可用金鑰
- SAML AWS STS 型聯合的可用金鑰
- 跨服務 SAML AWS STS 型聯合內容索引鍵
- 的可用金鑰 AWS STS

IAM 的可用鍵

您可以在控制存取 IAM 資源的政策中使用以下條件鍵：

iam：AccountPropertyNamespaces

使用字串運算子。

指定在請求中讀取或修改的帳戶屬性命名空間。帳戶屬性會組織成命名空間（例如 `RoleManager`)。當您呼叫 `PutAccountProperties`或 時`GetAccountProperties`，使用此條件索引鍵來限制主體可存取的屬性命名空間。

iam:AssociatedResourceArn

使用 ARN 運算子。

指定將在目標服務與此角色建立關聯之資源的 ARN。資源通常屬於主體者傳遞角色的目標服務。有時候，資源也可能屬於第三個服務。例如，您可以將角色傳遞給他們在 Amazon EC2 執行個體上使用的 Amazon EC2 Auto Scaling。在這種情況下，條件可能會符合 Amazon EC2 執行個體的 ARN。

此條件鍵僅適用於政策中的 PassRole 動作。不能用來限制任何其他動作。

> **Important**
>
> 在政策中使用 `iam:AssociatedResourceArn` 條件來限制 PassRole 動作時，如果政策旨在定義 AddRoleToInstanceProfile 動作的存取權，則需特別考量。在這種情況下，您無法在 EC2 執行個體 ARN 中指定區域或執行個體 ID。ARN 值必須為 `arn:aws:ec2:*:CallerAccountId:instance/*`。使用任何其他 ARN 值可能會導致非預期的評估結果。

在身分型政策中使用此條件索引鍵以允許實體傳遞角色，但前提是該角色與指定的資源相關聯。例如，您可以允許 IAM 使用者或角色將任何角色傳遞給 Amazon EC2 服務，以便與 AWS 帳戶中的執行個體搭配使用。不允許 IAM 使用者或角色將角色傳遞給其他服務。

```
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "iam:PassedToService": "ec2.amazonaws.com"
        },
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:*:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> AWS 支援 iam：PassedToService 的 服務也支援此條件金鑰。

iam:AWSServiceName

使用字串運算子。

指定此角色所連接的 AWS 服務。

`CreateServiceLinkedRole` API 操作支援此條件索引鍵。

> **Tip**
>
> 如需哪些服務支援使用服務連結角色的資訊，請參閱 AWS 使用 IAM 的 服務，並尋找 Service-Linked Role (服務連結角色) 欄中顯示 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。

在此範例中，若服務名稱為 access-analyzer.amazonaws.com，您可允許實體使用 `CreateServiceLinkedRole` API 操作建立服務連結角色。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "*",
       "Condition": {
         "StringLike": {
           "iam:AWSServiceName": "access-analyzer.amazonaws.com"
         }
       }
     }]
 }
```

iam:FIDO-certification

使用字串運算子。

在註冊 FIDO 安全金鑰時檢查 MFA 裝置 FIDO 認證等級。從 FIDO Alliance Metadata Service (MDS) 中擷取裝置認證。如果 FIDO 安全金鑰的認證狀態或等級發生變更，除非裝置已取消註冊並再次註冊以取得更新的認證資訊，否則其將不會更新。

L1、L1plus、L2、L2plus、L3、L3plus 的可能值

在此範例中，您註冊安全金鑰並擷取您裝置的 FIDO Level 1 plus 認證。

JSON

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

```
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-2-certification

使用字串運算子。

在註冊 FIDO 安全金鑰時檢查 MFA 裝置 FIPS-140-2 驗證認證等級。從 FIDO Alliance Metadata Service (MDS) 中擷取裝置認證。如果 FIDO 安全金鑰的認證狀態或等級發生變更，除非裝置已取消註冊並再次註冊以取得更新的認證資訊，否則其將不會更新。

L1、L2、L3、L4 的可能值

在此範例中，您註冊安全金鑰並擷取您裝置的 FIPS-140-2 Level 2 認證。

JSON

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

```
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-3-certification

使用字串運算子。

在註冊 FIDO 安全金鑰時檢查 MFA 裝置 FIPS-140-3 驗證認證等級。從 FIDO Alliance Metadata Service (MDS) 中擷取裝置認證。如果 FIDO 安全金鑰的認證狀態或等級發生變更，除非裝置已取消註冊並再次註冊以取得更新的認證資訊，否則其將不會更新。

L1、L2、L3、L4 的可能值

在此範例中，您註冊安全金鑰並擷取裝置的 FIPS-140-3 Level 3 認證。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
                }
            }
        }
    ]

}
```

iam:OrganizationsPolicyId

使用字串運算子。

檢查具有指定 AWS Organizations ID 的政策是否符合請求中使用的政策。如要檢視使用此條件鍵的範例 IAM 政策，請參閱 IAM：檢視 AWS Organizations 政策的服務上次存取資訊。

iam:PassedToService

使用字串運算子。

指定可以將角色傳遞到的服務的服務主體。此條件鍵僅適用於政策中的 PassRole 動作。不能用來限制任何其他動作。

當您在政策中使用此條件鍵時，請使用服務主體指定服務。服務主體是可以在政策的 `Principal` 元素中指定的服務名稱。這是通常的格式：`SERVICE_NAME_URL.amazonaws.com`。

您可以使用 `iam:PassedToService` 限制使用者，以便他們只能將角色傳遞給特定服務。例如，使用者可能會建立服務角色，它信任 CloudWatch 代他們將日誌資料寫入 Amazon S3 儲存貯體。然後，使用者必須將許可政策和信任政策連接到新的服務角色。在這種情況下，信任政策必須在 `cloudwatch.amazonaws.com` 元素中指定 `Principal`。若要檢視允許使用者將該角色傳遞至 CloudWatch 的政策，請參閱 IAM：將 IAM 角色傳遞至特定 AWS 服務。

透過使用此條件金鑰，您可以確保使用者僅為您指定的服務建立服務角色。例如，如果具有上述政策的使用者嘗試為 Amazon EC2 建立服務角色，操作將會失敗。發生失敗的原因是使用者沒有將角色傳遞至 Amazon EC2 的許可。

有時您會將角色傳遞給服務，然後將角色傳遞給不同服務。`iam:PassedToService` 僅包含擔任角色的最終服務，而不是傳遞角色的中繼服務。

> **Note**
>
> 有些服務不支援此條件鍵。

iam:PermissionsBoundary

使用 ARN 運算子。

檢查指定的政策連接為 IAM 主體資源上的許可界限。如需詳細資訊，請參閱IAM 實體的許可界限

iam:PolicyARN

使用 ARN 運算子。

在涉及受管政策的請求中檢查託管政策的 Amazon Resource Name (ARN) 。如需詳細資訊，請參閱控制對政策的存取。

iam：RoleTemplateARN

使用 ARN 運算子。

指定用於建立或管理角色的角色範本 ARN。當 AWS 服務從角色範本取得角色時，角色範本 ARN 會包含在 `CreateRole`、`TagRole`、、`PutRolePolicyAttachRolePolicy`、`PutRolePermissionsBoundary`和 的授權檢查中做為內容索引鍵`GetRole`。

iam:RegisterSecurityKey

使用字串運算子。

檢查 MFA 裝置啟用的目前狀態。

可能值為 `Create` 或 `Activate`。

在此範例中，您註冊安全金鑰並擷取裝置的 FIPS-140-3 Level 1 認證。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]

 }
```

iam:ResourceTag/***key-name***

使用字串運算子。

檢查連接至身分資源 (使用者或角色) 的標籤是否符合指定的鍵名稱和值。

> **Note**
>
> IAM 和 同時 AWS STS 支援 `iam:ResourceTag` IAM 條件金鑰和`aws:ResourceTag`全域條件金鑰。

您可以將自訂屬性以鍵/值組的形式新增至 IAM 資源。如需有關 IAM 資源標籤的詳細資訊，請參閱 the section called “IAM 資源的標籤”。您可以使用 ResourceTag 控制對 AWS 資源的存取，包括 IAM 資源。但是，由於 IAM 不支援群組的標籤，因此您無法使用標籤來控制群組的存取。

此範例會示範如何建立身分型政策，允許刪除具有 **status=terminated** 標籤的使用者。若要使用此政策，請將範例政策中的########取代為您自己的資訊。然後，遵循建立政策或編輯政策中的指示進行操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status":
 "terminated"}}
    }]
}
```

iam:ServiceSpecificCredentialAgeDays

適用於數字運算子。

此條件索引鍵會根據服務特定憑證的過期設定限制憑證的建立，便於您控制可建立之服務特定憑證的存留期上限 (以天為單位)。

天的有效範圍為 1 至 36600 (最短 1 天，最長 36600 天)。

CreateServiceSpecificCredential API 操作支援此條件索引鍵。

在此範例中，您僅允許使用者在服務特定憑證於 90 天內過期時，為 Amazon Bedrock 服務建立服務特定憑證。

JSON

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

```
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                },
                "NumericLessThanEquals": {
                    "iam:ServiceSpecificCredentialAgeDays": "90"
                }
            }
        }
    ]
}
```

iam:ServiceSpecificCredentialServiceName

使用字串運算子。

指定管理 AWS 服務特定登入資料時可以使用哪些服務。此條件金鑰可讓您限制管理 AWS 服務特定登入資料時允許哪些服務。

下列 API 操作支援此條件索引鍵：

- CreateServiceSpecificCredential
- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential
- UpdateServiceSpecificCredential

下列服務支援服務特定憑證，其確切值格式如下：

- bedrock.amazonaws.com
- cassandra.amazonaws.com
- codecommit.amazonaws.com

在此範例中，您僅允許使用者使用 CreateServiceSpecificCredential API 操作，為 Amazon Bedrock 服務建立服務特定憑證。

JSON

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                }
            }
        }
    ]
}
```

iam：DelegationDuration

適用於數字運算子。

根據委派請求中請求的暫時存取持續時間篩選存取權。

產品提供者可以使用此條件金鑰來控制他們在傳送給客戶的委派請求中允許的最長持續時間。持續時間以秒為單位指定，並決定臨時憑證在客戶釋出交換字符後保持有效的時間長度。這有助於產品提供者根據其使用案例強制執行有關存取持續時間限制的內部政策。

CreateDelegationRequest API 操作支援此條件索引鍵。

在此範例中，只有在請求的持續時間為 7200 秒 (2 小時） 或更短時，您才允許建立委派請求。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "NumericLessThanEquals": {
                    "iam:DelegationDuration": "7200"
                }
            }
        }
    ]
```

```
}
```

iam：NotificationChannel

使用 ARN 運算子。

根據為接收委派請求通知指定的 Amazon SNS 主題 ARN 篩選存取權。

產品提供者可以使用此條件金鑰來限制哪些 SNS 主題可用於 CreateDelegationRequest API 呼叫中的委派請求通知。產品提供者必須指定 SNS 主題，才能接收狀態變更通知和交換字符。這可確保通知只會傳送到產品提供者組織內已核准的管道。

`CreateDelegationRequest` API 操作支援此條件索引鍵。

在此範例中，只有在委派請求使用特定 SNS 主題進行通知時，您才允許建立委派請求。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:NotificationChannel": "arn:aws:sns:us-
east-1:123456789012:delegation-notifications"
                }
            }
        }
    ]
}
```

iam：TemplateArn

使用 ARN 運算子。

根據用於在委派請求中定義許可的政策範本 ARN 篩選存取權。

產品提供者可以使用此條件金鑰來控制可在 CreateDelegationRequest API 呼叫中使用的政策範本。政策範本定義產品提供者在客戶帳戶中請求的暫時許可。這可讓產品提供者限制在建立委派請求時可以使用哪些已註冊的政策範本。

CreateDelegationRequest API 操作支援此條件索引鍵。

在此範例中，只有當委派請求使用來自特定合作夥伴網域的政策範本時，您才允許建立委派請求。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "ArnLike": {
                    "iam:TemplateArn": "arn:aws:iam::*:delegation-template/
partner_*"
                }
            }
        }
    ]
}
```

iam：DelegationRequestOwner

使用 ARN 運算子。

根據擁有委派請求的 AWS 身分或委託人篩選存取權。

客戶可以使用此條件金鑰來控制誰可以根據擁有權對委派請求執行動作。委派請求的擁有者是客戶帳戶中啟動或接收委派請求的 AWS 身分或委託人。

下列 API 操作支援此條件索引鍵：

- GetDelegationRequest
- AcceptDelegationRequest
- RejectDelegationRequest
- SendDelegationToken
- ListDelegationRequests
- UpdateDelegationRequest

在此範例中，您允許使用者僅管理他們擁有的委派請求。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetDelegationRequest",
                "iam:AcceptDelegationRequest",
                "iam:RejectDelegationRequest",
                "iam:SendDelegationToken",
                "iam:UpdateDelegationRequest",
                "iam:ListDelegationRequests"
            ],
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:DelegationRequestOwner": "${aws:PrincipalArn}"
                }
            }
        }
    ]
}
```

AWS OIDC 聯合的可用金鑰

您可以使用 OIDC 聯合，將暫時性安全登入資料提供給已透過 OpenID Connect 相容身分提供者 (IdP) 驗證的使用者，提供給 AWS 帳戶中的 IAM OpenID Connect (OIDC) 身分提供者。此類身分提供者範例包括 GitHub、Amazon Cognito、Login with Amazon 以及 Google。可能會使用來自您自己的 IdP 的身分字符和存取字符，以及授予 Amazon Elastic Kubernetes Service 工作負載的服務帳戶字符。

您可以使用 AWS OIDC 條件內容金鑰來撰寫政策，將聯合主體的存取權限制為與特定提供者、應用程式或使用者相關聯的資源。這些鍵通常用於角色的信任政策。使用 OIDC 提供者的名稱 (`token.actions.githubusercontent.com`) 後跟宣告 (`:aud`): **`token.actions.githubusercontent.com:aud`** 來定義條件索引鍵。

某些 OIDC 聯合條件索引鍵可用於角色工作階段，以授權資源存取。如果此值在工作階段中可用欄中為是，您可以在政策中使用這些條件索引鍵來定義允許使用者在其他 AWS 服務中存取哪些 。當工作階段中無法使用宣告時，OIDC 條件內容索引鍵僅用於初始 AssumeRoleWithWebIdentity 身分驗證的角色信任政策。

選取您的 IdP，查看如何將 IdP 中的宣告對應至 AWS中的 IAM 條件內容索引鍵。適用於 GitHub 和 Google 的索引鍵的詳細資訊位於預設索引標籤下。

Default

預設會列出標準 OIDC 宣告及其對應至 AWS STS 條件內容索引鍵的方式 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。如果您的 IdP 未列在索引標籤選項中，可使用此映射。

GitHub Actions 工作流程和 Google 是在其 OIDC JWT ID 權杖中使用預設實作的 IdP 的一些範例。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
amr	amr	是
aud	azp 如果未設定 azp 的值，則 aud 條件索引鍵會映射至 aud 宣告。	是
email	email	否
oaud	aud	否
sub	sub	是

如需有關搭配使用條件內容索引鍵與 GitHub 的詳細資訊，請參閱 設定 GitHub OIDC 身分提供者的角色。如需有關 Google aud 和 azp 欄位的詳細資訊，請參閱 Google Identity Platform OpenID Connect 指南。

amr

使用字串運算子。鍵是多值的，這表示您使用條件設定運算子在政策中對其進行測試。

範例：`token.actions.githubusercontent.com:amr`

身分驗證方法參考包含使用者的登入資訊。鍵可以包含以下值：

- 如果使用者未經身分驗證，則鍵只包含 `unauthenticated`。
- 如果使用者已通過身分驗證，則金鑰會包含值 `authenticated` 以及呼叫中所使用的登入提供者的名稱 (`accounts.google.com`)。

aud

使用字串運算子。

範例：

- accounts.google.com:aud
- token.actions.githubusercontent.com:aud

使用 aud 條件索引鍵來驗證對象是否與您在政策中所指定的相符。可以將 aud 金鑰與 sub 金鑰用於相同的身分提供者。

在下列權杖欄位中設定此條件索引鍵：

- 未設定 aud 欄位時，您應用程式的 OAuth 2.0 Google 用戶端 ID 的 azp。設定 azp 欄位時，aud 欄位與 accounts.google.com:oaud 條件索引鍵相符。
- 設定 azp 欄位時為 azp。這可能發生於混合式應用程式，其中，Web 應用程式和 Android 應用程式具有不同的 OAuth 2.0 Google 用戶端 ID，但共用相同的 Google API 專案。

當您使用 accounts.google.com:aud 條件鍵撰寫政策時，您必須知道應用程式是否為設定 azp 欄位的混合式應用程式。

azp Field Not Set

下列範例政策適用於未設定 azp 欄位的非混合式應用程式。在這種情況下，Google ID 權杖 aud 欄位值符合 accounts.google.com:aud 和 accounts.google.com:oaud 條件鍵值。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
```

```
            }
        }
    ]
}
```

azp Field Set

下列範例政策適用於已設定 azp 欄位的混合式應用程式。在這種情況下，Google ID 權杖 aud 欄位值只符合 accounts.google.com:oaud 條件鍵值。azp 欄位值符合 accounts.google.com:aud 條件鍵值。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

email

使用字串運算子。

範例：accounts.google.com:email

此條件索引鍵會驗證使用者的電子郵件地址。此宣告的值對於此帳戶可能不是唯一的，而且可能會隨著時間而變更，因此您不應該使用此值做為主要識別符來驗證您的使用者記錄。

oaud

使用字串運算子。

範例：accounts.google.com:oaud

此金鑰會指定預期使用此 ID 權杖的其他對象 (aud)。必須是您應用程式的其中一個 OAuth 2.0 用戶端 ID。

sub

使用字串運算子。

範例：

- accounts.google.com:sub
- token.actions.githubusercontent.com:sub

使用這些金鑰來驗證主旨是否與您在政策中所指定的相符。您可以搭配相同身分提供者的 sub 鍵來使用 aud 鍵。

在下列角色信任政策中，sub條件索引鍵會將角色限制為名為 的 GitHub 分支demo。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub": "repo:org-name/repo-
name:ref:refs/heads/demo"
        }
      }
    }
  ]
}
```

Amazon Cognito

此索引標籤說明 Amazon Cognito 如何將 OIDC 宣告映射至 中的 AWS STS 條件內容索引鍵 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

對於 Amazon Cognito 使用的角色，金鑰的定義方式使用宣告後的 `cognito-identity.amazonaws.com`。

如需有關身分池宣告映射的詳細資訊，請參閱 Amazon Cognito Developer Guide 中的 Default provider mappings。如需有關使用者集區宣告映射的詳細資訊，請參閱Amazon Cognito Developer Guide 中的 Using the ID token。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
amr	amr	是
aud	aud	是
oaud	aud	否
sub	sub	是

amr

使用字串運算子。鍵是多值的，這表示您使用條件設定運算子在政策中對其進行測試。

範例 – `cognito-identity.amazonaws.com:amr`

身分驗證方法參考包含使用者的登入資訊。鍵可以包含以下值：

- 如果使用者未經身分驗證，則鍵只包含 `unauthenticated`。
- 如果使用者已通過身分驗證，則金鑰會包含值 `authenticated` 以及呼叫中所使用的登入提供者的名稱 (`cognito-identity.amazonaws.com`)。

舉例來說，Amazon Cognito 角色的信任政策中的以下條件會測試使用者是否未經身分驗證。

```
"Condition": {
  "StringEquals":
```

```
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

aud

使用字串運算子。

範例 – cognito-identity.amazonaws.com:aud

對您的使用者進行身分驗證的使用者集區應用程式用戶端。Amazon Cognito 會在存取權杖 client_id 宣告中呈現相同的值。

oaud

使用字串運算子。

範例 – cognito-identity.amazonaws.com:oaud

對您的使用者進行身分驗證的使用者集區應用程式用戶端。Amazon Cognito 會在存取權杖 client_id 宣告中呈現相同的值。

sub

使用字串運算子。

範例 – cognito-identity.amazonaws.com:sub

已驗證使用者的唯一識別碼 (UUID) 或主體。使用者名稱在您的使用者集區中可能不是唯一的。子宣告是識別特定使用者的最佳方法。您可以搭配相同身分提供者的 sub 鍵來使用 aud 鍵。

```
"Condition": {
         "StringEquals": {
            "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
            "cognito-identity.amazonaws.com:sub": [
               "us-east-1:12345678-1234-1234-1234-123456790ab",
               "us-east-1:98765432-1234-1234-1243-123456790ab"
            ]
         }
      }
```

Login with Amazon

此索引標籤說明 Login with Amazon 如何將 OIDC 宣告對應至其中 AWS STS 的條件內容金鑰 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
app_id	應用程式 ID	是
sub	使用者 ID	是
user_id	使用者 ID	是

app_id

使用字串運算子。

範例 – `www.amazon.com:app_id`

此金鑰會指定與其他身分提供者所使用之 `aud` 欄位相符的對象內容。

sub

使用字串運算子。

範例 – `www.amazon.com:sub`

此金鑰會確認使用者 ID 與您在政策中所指定的相符。您可以搭配相同身分提供者的 `sub` 鍵來使用 `aud` 鍵。

user_id

使用字串運算子。

範例 – `www.amazon.com:user_id`

此金鑰會指定與其他身分提供者使用之 `aud` 欄位相符的對象內容。您可以將 `user_id` 金鑰與 `id` 金鑰用於相同身分提供者。

Facebook

此索引標籤說明 Facebook 如何將 OIDC 宣告對應至 中的 AWS STS 條件內容索引鍵 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
app_id	應用程式 ID	是
id	id	是

app_id

使用字串運算子。

範例 – `graph.facebook.com:app_id`

此金鑰會確認對象內容與其他身分提供者所使用之 aud 欄位相符。

id

使用字串運算子。

範例 – `graph.facebook.com:id`

此金鑰已確認應用程式 (或網站) ID 與您在政策中所指定的相符。

GitHub

此索引標籤說明 GitHub Actions 如何將 OIDC 宣告映射至 中的 AWS STS 條件內容索引鍵 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
演員	演員	否
actor_id	actor_id	否
job_workflow_ref	job_workflow_ref	否

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
repository	repository	否
repository_id	repository_id	否
repository_owner_id	repository_owner_id	否
工作流程	工作流程	否
參考	參考	否
環境	環境	否
enterprise_id	enterprise_id	否

演員

使用字串運算子。

範例 – `token.actions.githubusercontent.com:actor`

此金鑰可識別啟動工作流程執行的個人帳戶。使用此項目來限制特定演員的存取。

actor_id

使用字串運算子。

範例 – `token.actions.githubusercontent.com:actor_id`

此金鑰會驗證啟動工作流程執行之個人帳戶的 ID。演員 IDs由 GitHub 產生，而且不可變。

job_workflow_ref

使用字串運算子。

範例 – `token.actions.githubusercontent.com:job_workflow_ref`

此金鑰包含使用可重複使用工作流程之任務可重複使用工作流程的參考路徑。使用此項目來限制對特定工作流程的存取，並確保只有核准的工作流程才能擔任角色。

儲存庫

使用字串運算子。

範例 – `token.actions.githubusercontent.com:repository`

此金鑰可識別工作流程執行所在的儲存庫。使用此項目來限制對特定 GitHub 儲存庫的存取。

repository_id

使用字串運算子。

範例 – `token.actions.githubusercontent.com:repository_id`

此金鑰會驗證工作流程執行所在之儲存庫的 ID。儲存庫 IDs 不可變，即使已重新命名儲存庫也不會變更。

repository_owner_id

使用字串運算子。

範例 – `token.actions.githubusercontent.com:repository_owner_id`

此金鑰會驗證儲存庫存放所在組織的 ID。儲存庫擁有者 ID 是穩定且唯一的識別碼，不會變更。

工作流程

使用字串運算子。

範例 – `token.actions.githubusercontent.com:workflow`

此金鑰包含工作流程的名稱。使用此項目來限制對儲存庫中特定工作流程的存取。

參考

使用字串運算子。

範例 – `token.actions.githubusercontent.com:ref`

此金鑰可識別觸發工作流程執行的 git ref（分支或標籤）。使用此選項根據特定分支限制存取，例如僅允許 `main`或 `production`分支。

環境

使用字串運算子。

範例 – `token.actions.githubusercontent.com:environment`

此金鑰包含任務所使用的環境名稱。使用此選項實作環境型存取控制，例如開發、預備和生產環境的個別許可。

> **Note**
>
> 如果您的信任政策中包含環境宣告，則必須在 GitHub 工作流程中設定並提供環境。

enterprise_id

使用字串運算子。

範例 – `token.actions.githubusercontent.com:enterprise_id`

此金鑰會驗證包含工作流程執行所在儲存庫的企業 ID。使用此選項可確保存取權僅限於 GitHub Enterprise 組織內的儲存庫。

> **使用不可變的識別符，而非名稱**
>
> 在 GitHub 上，儲存庫、組織和使用者名稱可能會變更。可透過重新命名或刪除來釋放的名稱，可由不同的 帳戶申請。僅依賴可變名稱型宣告 （ 例如 `repository`或 `actor`) 的政策可以授予對意外身分的存取權。為了協助防範這種情況，請在您的角色信任政策或資源控制政策中使用 GitHub 提供的不可變識別符`actor_id`，例如 `repository_owner_id`、`repository_id`或 。

下列範例信任政策使用 GitHub OIDC 權杖中的自訂宣告來限制對角色的存取。

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
            "StringEquals": {
                "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
                "token.actions.githubusercontent.com:repository_owner_id": "123456",
                "token.actions.githubusercontent.com:repository_id": "1296269",
```

```
                "token.actions.githubusercontent.com:actor_id": "1",
                "token.actions.githubusercontent.com:ref": "refs/heads/main",
                "token.actions.githubusercontent.com:enterprise_id": "345"
              }
          }
      }
  ]
}
```

Google

此索引標籤說明 Google 如何將 OIDC 宣告映射至 中的 AWS STS 條件內容索引鍵 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
google/organization_number	google：organizatio n_number	否

google/organization_number

適用於數字運算子。

範例 – `accounts.google.com:google/organization_number`

此金鑰會驗證權杖是否代表屬於特定 Google Cloud 或 Google Workspace 組織的 Google 身分。使用此選項來限制特定組織的使用者存取，確保只有您組織的身分才能擔任該角色。

下列範例信任政策使用 `google/organization_number`宣告來限制對角色的存取。

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "accounts.google.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
```

```
          "Condition": {
             "NumericEquals": {
                 "accounts.google.com:google/organization_number": "123456"
                }
            }
         }
    ]
}
```

CircleCI

此索引標籤說明 CircleCI 如何將 OIDC 宣告映射至 中的 AWS STS 條件內容索引鍵 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
oidc.circleci.com/project-id	oidc.circleci.com/project-id	否

oidc.circleci.com/project-id

使用字串運算子。

範例 – `circleci-issuer-url:oidc.circleci.com/project-id`

此金鑰可識別正在執行任務的 CircleCI 專案。其值是包含唯一識別 CircleCI 專案之 UUID 的字串。使用此項目來限制對特定 CircleCI 專案的存取。

下列範例信任政策使用 `oidc.circleci.com/project-id`宣告來限制對角色的存取。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/oidc.circleci.com/
org/12345"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
```

```
        "StringEquals": {
          "oidc.circleci.com/org/12345:aud": "sts.amazonaws.com",
          "oidc.circleci.com/org/12345:oidc.circleci.com/project-id": "76543210-
ba98-fedc-3210-edcba0987654"
        }
      }
    }
  ]
}
```

Oracle Cloud Infrastructure (OCI)

此索引標籤說明 Oracle Cloud Infrastructure 如何將 OIDC 宣告映射至 中的 AWS STS 條件內容索引鍵 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
rpst_id	rpst_id	否

rpst_id

使用字串運算子。

範例 – `oci-issuer-url:rpst_id`

此金鑰可唯一識別 OCI 中的資源主體。使用此選項來限制對特定 OCI 資源主體的存取。rpst_id（資源主體工作階段字符 ID) 為 OCI 資源型身分驗證提供穩定的識別符。

下列範例信任政策使用 `rpst_id`宣告來限制對角色的存取。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/idcs-
abc123ef5678901234abcd.identity.oraclecloud.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
```

```
      "Condition": {
        "StringEquals": {
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:aud":
 "sts.amazonaws.com",
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:rpst_id": "your-
rpst-id"
        }
      }
    }
  ]
}
```

GitLab.com

此索引標籤說明 GitLab.com CI/CD 如何將 OIDC 宣告映射至 中的 AWS STS 條件內容索引鍵。AWS您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

> **Note**
>
> 只有在使用 GitLab.com 多租用戶 SaaS 執行個體做為 OIDC 身分提供者時，才支援這些自訂條件金鑰。您無法將這些條件金鑰與 GitLab 專用或 GitLab 自我管理執行個體搭配使用。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
namespace_id	namespace_id	否
project_id	project_id	否
user_id	user_id	否
user_login	user_login	否
user_email	user_email	否
user_access_level	user_access_level	否
ref_protected	ref_protected	否
pipeline_source	pipeline_source	否

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
runner_environment	runner_environment	否

namespace_id

使用字串運算子。

範例 – `gitlab.com:namespace_id`

此金鑰可識別執行 CI/CD 任務之專案的命名空間（群組）ID。命名空間 ID 是穩定的唯一識別符，在重新命名群組時不會變更。

project_id

使用字串運算子。

範例 – `gitlab.com:project_id`

此金鑰可識別執行 CI/CD 任務的專案 ID。專案 ID 是穩定的唯一識別符，在重新命名或移動專案時不會變更。

user_id

使用字串運算子。

範例 – `gitlab.com:user_id`

此金鑰可識別執行 CI/CD 任務之使用者的 ID。使用者 ID 是穩定且唯一的識別符。

user_login

使用字串運算子。

範例 – `gitlab.com:user_login`

此金鑰可識別執行 CI/CD 任務之使用者的使用者名稱。使用者名稱可以變更；若要穩定識別符，請`user_id`改用 。

user_email

使用字串運算子。

範例 – `gitlab.com:user_email`

此金鑰可識別執行 CI/CD 任務之使用者的電子郵件地址。此宣告的值可能會隨著時間而變更；如需穩定的識別符，請user_id改用 。

user_access_level

使用字串運算子。

範例 – gitlab.com:user_access_level

此金鑰可識別專案內使用者的存取層級（例如 maintainer、developer、)owner。

ref_protected

使用字串運算子。

範例 – gitlab.com:ref_protected

此金鑰指出觸發任務的 Git ref 是否受到保護。如果受保護，true則值為 ，false否則為 。

pipeline_source

使用字串運算子。

範例 – gitlab.com:pipeline_source

此金鑰可識別觸發管道的來源（例如 push、、webschedule、api、)merge_request_event。

runner_environment

使用字串運算子。

範例 – gitlab.com:runner_environment

此金鑰可識別 CI/CD 任務使用的執行器類型。

> **Important**
>
> 在 GitLab.com SaaS 上，已刪除的群組或專案路徑先前可由不同的使用者重新建立，這可能會導致僅依賴路徑型宣告（例如 sub) 的政策授予對意外身分的存取權。自 2026 年 6 月 1 日起，GitLab 已部署平台端更新，以防止針對先前發行權杖的專案路徑發行新的 CI ID 權杖，進而降低此風險。作為另一層保護， AWS GitLab 建議在您的角色信任政策或資源控制政策中包含穩定、唯一識別符的條件project_id，例如 namespace_id 或 。

下列範例信任政策將 namespace_id宣告與 搭配使用sub，以限制對特定 GitLab.com 專案和分支的存取。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/gitlab.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "gitlab.com:sub": "project_path:mygroup/
myproject:ref_type:branch:ref:main",
          "gitlab.com:namespace_id": "12345"
        }
      }
    }
  ]
}
```

Buildkite

此索引標籤說明 Buildkite 如何將 OIDC 宣告映射至 中的 AWS STS 條件內容索引鍵 AWS。您可以使用這些金鑰來控制角色的存取。為此，請將 AWS STS 條件索引鍵與 IdP JWT 宣告欄中的值進行比較。

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
organization_slug	organization_slug	否
organization_id	organization_id	否
pipeline_slug	pipeline_slug	否
pipeline_id	pipeline_id	否
cluster_name	cluster_name	否
cluster_id	cluster_id	否

AWS STS 條件索引鍵	IdP JWT 宣告	在工作階段中可用
build_branch	build_branch	否

organization_slug

使用字串運算子。

範例 – `agent.buildkite.com:organization_slug`

此金鑰可識別執行任務的 Buildkite 組織。使用此項目來限制對特定組織的存取。

organization_id

使用字串運算子。

範例 – `agent.buildkite.com:organization_id`

此金鑰會驗證任務執行所在之 Buildkite 組織的 ID。組織 IDs由 Buildkite 產生，不可變。

pipeline_slug

使用字串運算子。

範例 – `agent.buildkite.com:pipeline_slug`

此金鑰可識別任務執行所在的管道。使用此項目來限制對特定 Buildkite 管道的存取。

pipeline_id

使用字串運算子。

範例 – `agent.buildkite.com:pipeline_id`

此金鑰會驗證任務執行所在管道的 ID。管道 IDs由 Buildkite 產生且不可變。

cluster_name

使用字串運算子。

範例 – `agent.buildkite.com:cluster_name`

此金鑰可識別正在執行任務的叢集。使用此項目來限制對特定 Buildkite 叢集的存取。

cluster_id

使用字串運算子。

範例 – `agent.buildkite.com:cluster_id`

此金鑰會驗證任務執行所在叢集的 ID。叢集 IDs是由 Buildkite 產生，而且不可變。

build_branch

使用字串運算子。

範例 – `agent.buildkite.com:build_branch`

此金鑰可識別觸發建置的 git 分支。使用此選項可根據特定分支來限制存取，例如僅允許 `main`或 `production`分支。

> **使用不可變的識別符，而不是 slug**
>
> 在 Buildkite 上，刪除的組織或管道槽可以由不同的組織或管道立即重新註冊。Buildkite 會從相同的發行者 () 發出這些 slug 的 OIDC 權杖`agent.buildkite.com`。因此，僅依賴 slug 型宣告 （ 例如 `organization_slug`或 `pipeline_slug`) 的政策可以授予意外身分的存取權。為了協助防範這種情況，請在您的角色信任政策或資源控制政策中使用穩定、不可變的識別符`pipeline_id`，例如 `organization_id`或 。

下列範例信任政策使用 Buildkite OIDC 權杖中的自訂宣告來限制對角色的存取。

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "arn:aws:iam::123456789012:oidc-provider/
agent.buildkite.com"
        },
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {
                "agent.buildkite.com:aud": "sts.amazonaws.com",
                "agent.buildkite.com:organization_id": "0191e7a0-1234-7abc-
def0-123456789abc",
                "agent.buildkite.com:pipeline_id": "0191e7a1-5678-7abc-
def0-123456789abc",
                "agent.buildkite.com:build_branch": "main"
```

```
            }
        }
    }
    ]
}
```

關於 OIDC 聯合的詳細資訊

- Amazon Cognito 使用者指南
- OIDC 聯合身分

SAML AWS STS 型聯合的可用金鑰

如果您使用以 SAML 為基礎的聯合使用 AWS Security Token Service (AWS STS)，您可以在政策中包含其他條件金鑰。

SAML 角色信任政策

在角色的信任政策中，您可以包含以下鍵，這些鍵可幫助您確定是否允許呼叫者擔任該角色。除了 `saml:doc` 外，所有值均來自 SAML 聲明。當您建立或編輯具有條件的政策時，在 IAM 主控台視覺化編輯器中可使用清單中的所有項目。標示為 `[]` 的項目可以具有一個值，該值是指定類型的清單。

saml:aud

使用字串運算子。

顯示 SAML 聲明的端點 URL。此鍵值來自聲明中的 `SAML Recipient` 欄位，而不是 `Audience` 欄位。

saml:commonName[]

使用字串運算子。

這是 `commonName` 屬性。

saml:cn[]

使用字串運算子。

這是 `eduOrg` 屬性。

saml:doc

使用字串運算子。

這代表用來擔任該角色的主體。格式為 *account-ID*/*provider-friendly-name*，例如 123456789012/SAMLProviderName。account-ID 值是指擁有 SAML 提供者的帳戶。

saml:edupersonaffiliation[]

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonassurance[]

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonentitlement[]

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonnickname[]

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonorgdn

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonorgunitdn[]

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonprimaryaffiliation

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonprimaryorgunitdn

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonprincipalname

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersonscopedaffiliation[]

使用字串運算子。

這是 eduPerson 屬性。

saml:edupersontargetedid[]

使用字串運算子。

這是 eduPerson 屬性。

saml:eduorghomepageuri[]

使用字串運算子。

這是 eduOrg 屬性。

saml:eduorgidentityauthnpolicyuri[]

使用字串運算子。

這是 eduOrg 屬性。

saml:eduorglegalname[]

使用字串運算子。

這是 eduOrg 屬性。

saml:eduorgsuperioruri[]

使用字串運算子。

這是 eduOrg 屬性。

saml:eduorgwhitepagesuri[]

使用字串運算子。

這是 eduOrg 屬性。

saml:givenName[]

使用字串運算子。

這是 givenName 屬性。

saml:iss

使用字串運算子。

發行者，由 URN 代表。

saml:mail[]

使用字串運算子。

這是 mail 屬性。

saml:name[]

使用字串運算子。

這是 name 屬性。

saml:namequalifier

使用字串運算子。

以 SAML 提供者易記名稱為基礎的雜湊值。該值是下列值的串連，依照順序且以 '/' 字元區隔：

1. Issuer 回應值 (saml:iss)
2. AWS 帳戶 ID
3. IAM 中 SAML 提供者的易記名稱 (ARN 的最後一個部分)

帳戶 ID 與 SAML 提供者的易記名稱的串聯可作為鍵 saml:doc 供 IAM 政策使用。如需詳細資訊，請參閱 單獨辨識以 SAML 為基礎的聯合身分中的使用者。

saml:organizationStatus[]

使用字串運算子。

這是 organizationStatus 屬性。

saml:primaryGroupSID[]

使用字串運算子。

這是 primaryGroupSID 屬性。

saml:sub

使用字串運算子。

這是該陳述的主題，其中包含單獨辨識組織中某個使用者的值 (例如 _cbb88bf52c2510eabe00c1642d4643f41430fe25e3)。

saml:sub_type

使用字串運算子。

此鍵可以具有值 persistent、transient 或者由 SAML 聲明中使用的 Format 與 Subject 元素的完整 NameID URI 組成。persistent 的值表示 saml:sub 中的值對於工作階段之間的使用者是相同的。如果值為 transient，則使用者在每個工作階段中擁有不同的 saml:sub 值。如需 NameID 元素的 Format 屬性的詳細資訊，請參閱 為身分驗證回應設定 SAML 聲明。

saml:surname[]

使用字串運算子。

這是 surnameuid 屬性。

saml:uid[]

使用字串運算子。

這是 uid 屬性。

saml:x500UniqueIdentifier[]

使用字串運算子。

這是 x500UniqueIdentifier 屬性。

如需 eduPerson 和 eduOrg 屬性的一般資訊，請參閱 REFEDS Wiki 網站。如需 eduPerson 屬性的清單，請參閱 eduPerson 物件類別規格 (201602)。

類型為清單的條件鍵可包括多個值。若要在清單值的政策中建立條件，可以使用設定運算子 (ForAllValues、ForAnyValue)。例如，若要允許關係為「教職員」或「員工」 (但不是「學生」) 的任何使用者，您可以使用如下條件：

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

跨服務 SAML AWS STS 型聯合內容索引鍵

某些以 SAML 為基礎的聯合條件金鑰可用於後續請求，以授權其他 服務和`AssumeRole`呼叫中的 AWS 操作。以下是下列條件索引鍵，當聯合身分主體擔任另一個角色時，可用於角色信任政策，也可以用於其他 AWS 服務的資源政策，以授權聯合身分主體存取資源。如需使用這些金鑰的詳細資訊，請參閱關於以 SAML 2.0 為基礎的聯合。

選取條件鍵以查看描述。

- saml:namequalifier
- saml:sub
- saml:sub_type

> **Note**
>
> 在初始外部身分提供者 (IdP) 驗證回應之後，沒有其他以 SAML 為基礎的聯合條件鍵可供使用。

的可用金鑰 AWS STS

對於使用 AWS Security Token Service (AWS STS) 操作擔任的角色，您可以在 IAM 角色信任政策中使用下列條件金鑰。

saml:sub

使用字串運算子。

這是該陳述的主題，其中包含單獨辨識組織中某個使用者的值 (例如 `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`)。

sts:AWSServiceName

使用字串運算子。

使用此鍵來指定可以使用持有人權杖的服務。當您在政策中使用此條件鍵時，請使用服務主體指定服務。服務主體是可以在政策的 `Principal` 元素中指定的服務名稱。例如，`codeartifact.amazonaws.com`是 AWS CodeArtifact 服務主體。

可用性 – 此鍵會呈現在取得持有人權杖的請求中。您無法直接呼叫 AWS STS 以取得承載字符。當您在其他服務中執行某些操作時，服務會代表您請求持有人權杖。

有些 AWS 服務需要您取得 AWS STS 服務承載字符的許可，才能以程式設計方式存取其資源。例如， AWS CodeArtifact 需要主體使用持有人權杖來執行某些操作。`aws codeartifact get-authorization-token` 命令會傳回一個持有人權杖。然後，您可以使用承載字符來執行 AWS CodeArtifact 操作。如需有關持有人權杖的詳細資訊，請參閱服務持有人權杖。

您可以使用此條件鍵來允許主體取得持有人權杖，以與特定服務搭配使用。

sts:DurationSeconds

適用於數字運算子。

使用此金鑰指定主體從 GetWebIdentityToken API 取得 AWS AWS STS 承載字符或 JSON Web 字符時可使用的持續時間 （以秒為單位）。

可用性 – 此金鑰存在於從 GetWebIdentityToken API 取得承載字符或 JSON Web 字符的請求中。您無法直接呼叫 AWS STS 以取得承載字符。當您在其他服務中執行某些操作時，服務會代表您請求持有人權杖。金鑰不適用於 AWS STS 擔任角色操作。

有些 AWS 服務需要您取得 AWS STS 服務承載字符的許可，才能以程式設計方式存取其資源。例如， AWS CodeArtifact 需要主體使用持有人權杖來執行某些操作。`aws codeartifact get-authorization-token` 命令會傳回一個持有人權杖。然後，您可以使用承載字符來執行 AWS CodeArtifact 操作。如需有關持有人權杖的詳細資訊，請參閱服務持有人權杖。

sts：IdentityTokenAudience

使用字串運算子。

使用此金鑰指定 IAM 主體可以使用 GetWebIdentityToken API 請求 JSON Web Token JWTs) 的對象。當 IAM 政策中存在此條件金鑰時，IAM 主體只能為政策中指定的對象請求權杖。外部服務會驗證 JSON Web 權杖中的對象（「aud」）宣告，以確保權杖適用於他們。

可用性 – 此金鑰存在於 GetWebIdentityToken API 的請求中，該 API 用於取得 JSON Web Token (JWTs) 以使用外部服務進行身分驗證。

當您在政策中使用此條件索引鍵時，請指定符合預期收件人識別符的對象值 （例如 https://api.example.com)。

下列範例政策允許委託人請求指定外部服務的字符：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api2.example.com",
                        "https://api1.example.com"
                    ]
                }
            }
        }
    ]
}
```

sts：SigningAlgorithm

使用字串運算子。

使用此金鑰指定密碼編譯演算法， AWS AWS STS 用來簽署 GetWebIdentityToken API 所產生的 JSON Web Token JWTs)。當您在政策中使用此條件金鑰時，請指定 ES384 (ECDSA 搭配 P-384 曲線和 SHA-384) 或 RS256 (RSA 搭配 SHA-256)。

可用性 – 此金鑰存在於 GetWebIdentityToken API 的請求中，該 API 用於取得 JSON Web Token (JWTs) 以使用外部服務進行身分驗證。

您可以使用此條件金鑰，強制執行 IAM 主體使用與您的安全需求或整合的外部服務相容的簽署演算法來請求權杖。ES384 提供最佳的安全性和效能，而 RS256 提供與不支援 ECDSA 的系統更廣泛的相容性。

下列範例政策要求主體使用 ES384 簽署演算法：

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

```
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

sts:ExternalId

使用字串運算子。

使用此鍵來請求主體在擔任 IAM 角色時提供特定的識別碼。

可用性 – 當委託人在使用 AWS CLI 或 AWS API 擔任角色時提供外部 ID 時，此金鑰會出現在請求中。

當您在其他帳戶擔任角色時，可能需要此唯一識別符。若帳戶管理員 (該角色所屬的帳戶) 提供給您外部 ID，請將該數值填入 `ExternalId` 參數。該值可為任何字串，例如密碼短語或帳號。外部 ID 的主要功能是解決並防止「混淆代理人」問題。如需有關外部 ID 和混淆代理人問題的詳細資訊，請參閱 存取第三方 AWS 帳戶 擁有的。

`ExternalId` 值必須最少為 2 個字元，最多為 1,224 個字元。該值必須為英數字元，且不包含空格。也可以包含下列符號：加號 (+)、等號 (=)、逗號 (,)、句號 (.)、小老鼠 (@)、冒號 (:)、正斜線 (/) 和連字號 (-)。

sts:RequestContext/context-key

使用字串運算子。

使用此鍵，可將內嵌在請求中所傳遞的受信任權杖發行者簽署之內容聲明的工作階段內容鍵/值對，與角色信任政策中指定的內容鍵/值進行比較。

可用性 – 當使用 AWS STS AssumeRole API 操作擔任角色時，`ProvidedContexts`在請求參數中提供內容聲明時，請求中會出現此金鑰。

此內容鍵會格式化為 `"sts:RequestContext/context-key":"context-value"`，其中位 `context-key` 和 `context-value` 為內容鍵/值對。如果請求中傳遞簽署內容聲明中內嵌了多個內容鍵，則每一組鍵/值對都有一個內容鍵。您必須為角色信任政策中的 `sts:SetContext` 動作授予許可，才能允許主體在產生的工作階段權杖中設定內容鍵。若要進一步了解可與此索引鍵搭配

使用的受支援 IAM Identity Center 內容索引鍵，請參閱 AWS IAM Identity Center User Guide 中的 AWS STS condition keys for IAM Identity Center。

您可以在角色信任政策中使用此鍵，以便在使用者或其屬性擔任角色時，能根據使用者或其屬性實行精細的存取控制。擔任角色後，活動會出現在 AdditionalEventData 屬性內的 AWS CloudTrail 日誌中，其中包含由擔任角色請求中的內容提供者所設定的工作階段內容索引鍵/值對。當不同的主體使用角色時，這可讓系統管理員更容易區分角色工作階段。鍵/值對是由指定的內容提供者設定，而不是由 AWS CloudTrail 或 設定 AWS STS。這可讓內容提供者控制 CloudTrail 日誌和工作階段資訊所包含的內容。

sts:RequestContextProviders

使用 ARN 運算子。

使用此鍵，可將請求中的內容提供者 ARN 與角色信任政策中指定的內容提供者 ARN 進行比較。

可用性 – 當使用 AWS STS AssumeRole API 操作擔任角色時，ProvidedContexts在請求參數中提供內容聲明時，請求中會出現此金鑰。

下列範例條件會檢查請求中傳遞的內容提供者 ARN 是否符合角色信任政策條件中指定的 ARN。建議您新增帶有 ForAllValues 的 Null 檢查，防止遺失內容索引鍵或具有空值的內容索引鍵評估為 true。如需詳細資訊，請參閱用於檢查條件索引鍵是否存在的條件運算子。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Action": "sts:SetContext",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
      "ForAllValues:ArnEquals": {
        "sts:RequestContextProviders": [
          "arn:aws:iam::aws:contextProvider/IdentityCenter"
        ]
      },
      "Null": {
        "sts:RequestContextProviders": "false"
      }
    }
  }
```

```
}
```

sts：RoleAuthorizedByIdp

適用於布林運算子。

使用此金鑰來驗證身分提供者 (IdP) 透過 OIDC 權杖中的`https://aws.amazon.com/roles`宣告明確授權請求的角色。角色宣告包含允許字符擔任的 IAM 角色 ARNs。 AWS STS 支援宣告值的兩種格式：

JSON 陣列 （ 建議 ）：

```
"https://aws.amazon.com/roles": [
  "arn:aws:iam::111122223333:role/RoleA",
  "arn:aws:iam::111122223333:role/RoleB"
]
```

分號分隔字串：

```
"https://aws.amazon.com/roles": "arn:aws:iam::111122223333:role/
RoleA;arn:aws:iam::111122223333:role/RoleB"
```

可用性 – 此金鑰存在於使用 `AssumeRoleWithWebIdentity`操作的請求中。值是指`true`字符包含`https://aws.amazon.com/roles`宣告，且請求的角色 ARN 符合宣告中的值。當字符不包含宣告`false`時，值為 。由於 金鑰一律存在於`AssumeRoleWithWebIdentity`請求中，請使用 `Bool`條件運算子而非 `BoolIfExists`。

當字符包含宣告，但請求的角色 ARN 與宣告中的值不相符時，`AssumeRoleWithWebIdentity`呼叫會傳回`InvalidIdentityToken`錯誤。

您可以在角色信任政策、資源控制政策 (RCPs) 和 VPC 端點政策中使用此條件索引鍵，要求 IdP 在擔任角色之前授權該角色。

下列角色信任政策`AssumeRoleWithWebIdentity`僅在 IdP 授權權杖中請求的角色時允許 。

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

```
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": { "idp.example.com:aud": "my-app-id" },
                "Bool": { "sts:RoleAuthorizedByIdp": "true" }
            }
        }
    ]
}
```

sts:RoleSessionName

使用字串運算子。

使用此鍵，比較使用政策中指定的值擔任角色時，主體指定的工作階段名稱。

可用性 – 當委託人使用 AWS 管理主控台、任何擔任角色的 CLI 命令或任何 AWS STS `AssumeRole` API 操作擔任角色時，此金鑰會出現在請求中。

您可以在角色信任政策中使用此鍵，請求您的使用者在擔任角色時提供特定的工作階段名稱。例如，您可以請求 IAM 使用者將自己的使用者名稱指定為其工作階段名稱。在 IAM 使用者擔任角色之後，活動會出現在 AWS CloudTrail 日誌中，其工作階段名稱與其使用者名稱相符。當不同的主體使用角色時，這可讓系統管理員更容易區分角色工作階段。

下列角色信任政策會請求 111122223333 帳戶中的 IAM 使用者在擔任角色時，提供其 IAM 使用者名稱作為工作階段名稱。使用條件鍵中的 `aws:username` 條件變數強制執行此需求。此政策允許 IAM 使用者擔任政策所連接的角色。此政策不允許使用暫時憑證的任何人擔任該角色，因為 `username` 變數僅供 IAM 使用者使用。

> **Important**
>
> 您可以使用任何單一值條件鍵作為變數。您無法使用多重值條件鍵做為變數。

JSON

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

```
    "Statement": [
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "prefix-
${aws:username}"}
            }
        }
    ]
}
```

當管理員檢視 動作的 AWS CloudTrail 日誌時，他們可以將工作階段名稱與其帳戶中的使用者名稱進行比較。在下列範例中，名為 matjac 的使用者使用名為 MateoRole 的角色執行作業。然後，管理員可以聯絡 Mateo Jackson，他擁有名為 matjac 的使用者。

```
    "assumedRoleUser": {
        "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
        "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
    }
```

如果您允許使用角色進行跨帳戶存取，則某個帳戶中的使用者可以在另一個帳戶中擔任角色。CloudTrail 中列出的假設角色使用者 ARN 包含角色存在的帳號。它不包括擔任角色的使用者帳戶。使用者僅在帳戶中是唯一的。因此，建議您只使用這個方法來檢查所管理帳戶中使用者假設之角色的 CloudTrail 日誌。您的使用者可能會在多個帳戶中使用相同的使用者名稱。

sts:SourceIdentity

使用字串運算子。

使用此鍵，比較使用政策中指定的值擔任角色時，主體指定的來源身分。

可用性 – 當委託人在使用任何 AWS STS 擔任角色 CLI 命令或 AWS STS AssumeRole API 操作時提供來源身分時，此金鑰會出現在請求中。

您可以在角色信任政策中使用此鍵，請求您的使用者在擔任角色時設定特定的來源身分。例如，您可以要求人力或聯合身分來指定來源身分的值。您可以將身分提供者 (IdP) 設定為使用與使用者相關聯的其中一個屬性，例如使用者名稱或電子郵件作為來源身分。然後，IdP 會在傳送的聲明或宣告中，將來源身分做為屬性傳遞 AWS。來源身分屬性的值可識別擔任該角色的使用者或應用程式。

在使用者擔任該角色之後，活動會出現在具有已設定來源身分值的 AWS CloudTrail 日誌中。這可讓管理員更輕鬆地判斷角色在其中執行動作的對象或對象 AWS。您必須對 sts:SetSourceIdentity 動作授予許可，允許身分以設定來源身分。

與 sts:RoleSessionName 不同，在設定來源身分之後，就無法變更值。這會出現在由角色以來源身分所採取的所有動作的請求內容中。當您使用工作階段憑證來擔任另一個角色時，此值仍然存在於後續角色工作階段。從另一個角色取得及擔任角色稱為角色鏈結。

您可以使用 aws:SourceIdentity全域條件金鑰，根據後續請求中的來源身分值進一步控制對 AWS 資源的存取。

下列角色信任政策允許 IAM 使用者 AdminUser 擔任帳戶 111122223333 中的角色。它也會對 AdminUser 許可授予來設定來源身分，只要來源身分識別設定為 DiegoRamirez。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
}
```

若要進一步了解有關使用來源身分的詳細資訊，請參閱 監控並控制使用擔任角色所採取的動作。

sts:TaskPolicyArn

使用 ARN 運算子。

使用此金鑰來比較 sts:AssumeRoot 請求中的政策 ARN 與政策中指定的政策 ARN。

可用性：當您使用 sts:AssumeRoot 提出請求時，此金鑰會出現在請求中。

管理員可以在 IAM 政策中使用此條件索引鍵，限制管理帳戶或委派管理員帳戶內的特定角色或使用者在擔任根憑證時執行特定動作。如需詳細資訊，請參閱在 AWS Organizations 成員帳戶上執行特權任務。

sts:TransitiveTagKeys

使用字串運算子。

使用此鍵，可將請求中的可轉移工作階段標籤鍵與政策中指定的內容進行比較。

可用性 – 當您使用臨時安全憑證提出請求時，此鍵會呈現在請求中。這些包括使用任何 assume-role 操作或 `GetFederationToken` 操作所建立的憑證。

當您使用暫時安全憑證提出請求時，請求內容會包含 `aws:PrincipalTag` 內容鍵。此鍵包含工作階段標籤的清單、可轉移工作階段標籤和角色標籤。可轉移工作階段標籤是在您使用工作階段憑證擔任另一個角色時，保存到所有後續工作階段的標籤。從另一個角色取得及擔任角色稱為角色鏈結。

您可以在政策中使用此條件鍵，要求在擔任角色或與使用者聯合身分時將特定工作階段標籤設為可轉移。

AWS 服務的動作、資源和條件索引鍵

每個 AWS 服務都可以定義動作、資源和條件內容索引鍵，以用於 IAM 政策。如需 AWS 服務及其動作、資源和條件內容索引鍵的清單，請參閱服務授權參考中的動作、資源和條件索引鍵。

進一步了解有關 IAM 的資源

IAM 是豐富的產品，您會找到許多資源來協助您進一步了解 IAM 如何協助您保護 AWS 帳戶 和 資源。

主題

- 身分
- 憑證 (密碼、存取金鑰和 MFA 裝置)
- 許可和政策
- 聯合與委派
- IAM 和其他 AWS 產品
- 一般安全實務
- 一般資源

身分

請諮詢這些資源以建立、管理和使用身分。

- 管理 AM Identity Center 中的身分 - 有關在 IAM Identity Center 中建立使用者和群組的程序資訊。
- IAM 身分 - 深入討論使用者、群組和角色。

憑證 (密碼、存取金鑰和 MFA 裝置)

檢閱下列指南，為您的 和 IAM 使用者管理密碼、存取金鑰 AWS 帳戶 和 MFA 裝置。

- 中的使用者密碼 AWS – 描述管理您的帳戶中 IAM 使用者密碼的選項。
- 管理 IAM 使用者的存取金鑰 - 描述存取金鑰的運作方式，以及您可以如何使用它們對 AWS發出程式化呼叫。我們建議您先考慮其他更安全的方法來替代存取金鑰。如需詳細資訊，請參閱 AWS 一般參考 指南中的長期存取金鑰的考量事項和替代方案。
- AWS IAM 中的多重要素驗證 – 描述如何設定您的帳戶和 IAM 使用者，要求在登入前允許在裝置上產生密碼和一次性使用程式碼。(有時稱為雙重身分驗證)。

如需有關您用於存取 Amazon Web Services 的憑證類型的一般資訊，請參閱 AWS 一般參考 指南中的 AWS 安全憑證。

許可和政策

了解 IAM 政策的內部工作原理，並找到有關授予許可的最佳方式的秘訣：

- 中的政策和許可 AWS Identity and Access Management – 介紹用於定義許可的政策語言。說明如何將許可連接到使用者或群組，或者對於某些 AWS 產品，如何連接到資源本身。
- IAM JSON 政策元素參考 – 提供每個政策語言元素的描述和範例。
- IAM 政策驗證 – 尋找用於 JSON 政策驗證的資源。
- 以身分為基礎的 IAM 政策範例 – 顯示各種 AWS 產品中常見任務的政策範例。
- AWS 政策產生器 – 透過從清單中選擇產品和動作來建立自訂政策。
- IAM 政策模擬器 – 測試政策是否允許或拒絕特定請求 AWS。

聯合與委派

對於在其他位置經過身分驗證 （ 登入 ） AWS 帳戶 的使用者，您可以授予 中 資源的存取權。這些可以是另一個 AWS 帳戶 （ 稱為委派) 中的 IAM 使用者、使用組織的登入程序進行身分驗證的使用者，或來自網際網路身分提供者的使用者，例如 Login with Amazon、Facebook、Google 或任何其他 OpenID Connect (OIDC) 相容身分提供者。在這些情況下，使用者會取得臨時安全登入資料來存取 AWS 資源。

- IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權 – 指導您透過授予另一個 AWS 帳戶中的 IAM 使用者的跨帳戶存取權。
- 暫時性憑證的常見案例 – 說明使用者在 外部進行身分驗證 AWS 後可以聯合到 的方式 AWS。

IAM 和其他 AWS 產品

大多數 AWS 產品都與 IAM 整合，因此您可以使用 IAM 功能來協助保護對這些產品中資源的存取。下列資源討論一些最熱門 AWS 產品的 IAM 和安全性。如需有關使用 IAM 的產品的完整清單，包括指向每個產品的詳細資訊的連結，請參閱 AWS 使用 IAM 的 服務。

搭配使用 IAM 與 Amazon EC2

- 控制對 Amazon EC2 資源的存取 – 描述如何使用 IAM 功能來允許使用者管理 Amazon EC2 執行個體、磁碟區等。

- 使用執行個體設定檔 – 說明如何使用 IAM 角色安全地為在 Amazon EC2 執行個體上執行且需要存取其他 AWS 產品的應用程式提供登入資料。

搭配使用 IAM 與 Amazon S3

- 管理 Amazon S3 資源的存取許可 – 討論儲存貯體和物件的 Amazon S3 安全模型，其中包含 IAM 政策。
- 撰寫 IAM 政策：授予存取 Amazon S3 儲存貯體中的使用者特定資料夾 – 討論如何讓使用者保護 Amazon S3 中自己的資料夾 (有關 Amazon S3 和 IAM 的詳細文章，請選擇部落格文章標題下方的 S3 標籤)。

搭配使用 IAM 與 Amazon RDS

- 使用 AWS Identity and Access Management (IAM) 管理對 Amazon RDS 資源的存取 – 說明如何使用 IAM 控制對資料庫執行個體、資料庫快照等的存取。
- RDS 資源層級許可的入門 – 描述如何使用 IAM 控制存取特定 Amazon RDS 執行個體。

搭配使用 IAM 與 Amazon DynamoDB

- 使用 IAM 控制存取 DynamoDB 資源 – 描述如何使用 IAM 來允許使用者管理 DynamoDB 表格和索引。
- 以下影片 (8:55) 解釋如何為個別 DynamoDB 資料庫項目或屬性 (或兩者) 提供存取控制。

  DynamoDB 的精細存取控制入門

一般安全實務

尋找保護 和 資源的最佳方式的專家秘訣 AWS 帳戶 和指導：

- 安全性、身分、&、合規的最佳實務 – 尋找如何管理 AWS 帳戶 和 產品安全性的資源，包括安全架構、IAM 的使用、加密和資料安全性等建議。
- Identity and Access Management – AWS Well-Architected Framework 可協助您了解在雲端設計和執行工作負載的關鍵概念、設計原則和架構最佳實務。
- IAM 中的安全最佳實務 – 提供有關如何使用 IAM 協助保護您的 AWS 帳戶 和資源的建議。

- AWS CloudTrail 使用者指南 – 使用 AWS CloudTrail 追蹤對 發出的 API 呼叫歷史記錄， AWS 並將該資訊存放在日誌檔案中。這有助於您判斷哪些使用者和帳戶存取帳戶中的資源，何時進行呼叫，請求了哪些動作等等。

一般資源

探索下列資源，進一步了解 IAM 和 AWS。

- IAM 的產品資訊 – 有關 AWS Identity and Access Management 產品的一般資訊。
- AWS re:Post for AWS Identity and Access Management – 造訪 AWS re:Post 以與 AWS 社群討論 IAM 相關的技術問題。

- 課程與研討會 – 連結至以角色為基礎的特殊課程，以及自主進度實驗室，以協助強化您的 AWS 技能並取得實際經驗。
- AWS 開發人員中心 – 探索教學課程、下載工具，並了解 AWS 開發人員事件。
- AWS 開發人員工具 – 開發人員工具、SDKs、IDE 工具組和命令列工具的連結，用於開發和管理 AWS 應用程式。
- 入門資源中心 – 了解如何設定您的 AWS 帳戶、加入 AWS 社群，以及啟動您的第一個應用程式。
- 實用的教學課程 - 按照逐步教學課程在 AWS上啟動第一個應用程式。
- AWS 白皮書 – 技術 AWS 白皮書的完整清單連結，涵蓋架構、安全和經濟等主題，並由 AWS 解決方案架構師或其他技術專家撰寫。
- AWS 支援 中心 – 建立和管理 AWS 支援 案例的中樞。也包含其他實用資源的連結，例如論壇、技術 FAQs、服務運作狀態和 AWS Trusted Advisor。
- 支援 – 相關資訊的主要網頁 支援，一對一 one-on-one 的快速回應支援管道，可協助您在雲端中建置和執行應用程式。
- 聯絡我們 – 查詢有關 AWS 帳單、帳戶、事件、濫用與其他問題的中心聯絡點。
- AWS 網站條款 – 有關我們的著作權和商標、您的帳戶、授權和網站存取，以及其他主題的詳細資訊。

使用 HTTP 查詢請求呼叫 IAM API

目錄

- 端點
- 必要的 HTTPS
- 簽署 IAM API 請求

您可以使用查詢 API 以程式設計方式存取 IAM AWS STS 和服務。查詢 API 請求為 HTTPS 請求，其一定含有 `Action` 參數，以指示所要執行的動作。IAM 和 AWS STS 支援所有動作的 GET 和 POST 請求。也就是說，API 不會要求您在某些動作上使用 GET，在其他動作上使用 POST。但是，GET 請求受到 URL 的限制大小；雖然此限制取決於瀏覽器，但一般限制為 2048 個位元組。因此，對於需要較大流量的查詢 API 請求，必須使用 POST 請求。

回應為 XML 文件。如需回應的詳細資訊，請參閱 IAM API 參考或 AWS Security Token Service API 參考中的個別動作頁面。

> **Tip**
>
> 您可以使用其中一個 AWS SDKs，而不是直接呼叫 IAM 或 AWS STS API 操作。 AWS SDKs 由各種程式設計語言和平台 (Java、Ruby、.NET、iOS、Android 等） 的程式庫和範本程式碼組成。SDKs提供便捷的方式來建立 IAM 和 的程式設計存取 AWS。例如，開發套件會負責的工作諸如以密碼演算法簽署請求 (參閱下面)、管理錯誤以及自動重試請求。如需有關 AWS SDKs 的資訊，包括如何下載和安裝它們，請參閱適用於 Amazon Web Services 的工具頁面。

如需 API 動作和錯誤的詳細資訊，請參閱IAM API 參考或AWS Security Token Service API 參考。

端點

IAM 和 AWS STS 每個 都有一個全域端點：

- (IAM)https://iam.amazonaws.com
- (AWS STS) https://sts.amazonaws.com

> **Important**
>
> AWS STS 除了全域端點之外， 也支援將請求傳送至區域端點。 AWS 建議使用區域端點而非全域端點，以減少延遲、建置備援並提高工作階段字符有效性。在區域中使用 AWS STS 之前，必須先在該區域中為 AWS 帳戶啟用 STS。如需為 啟用其他區域的詳細資訊 AWS STS，請參閱 在 AWS STS 中管理 AWS 區域。

如需所有 服務的 AWS 端點和區域的詳細資訊，請參閱《》中的服務端點和配額AWS 一般參考。

必要的 HTTPS

由於查詢 API 會傳回安全憑證等敏感資訊，必須對所有 API 請求使用 HTTPS。

簽署 IAM API 請求

請求必須使用存取金鑰 ID 和私密存取金鑰簽署。強烈建議您不要將 AWS 帳戶根使用者 登入資料用於 IAM 的日常工作。您可以使用 IAM 使用者的登入資料，也可以使用 AWS STS 產生臨時安全登入資料。

若要簽署您的 API 請求，建議使用 AWS Signature 第 4 版。如需有關使用簽章第 4 版的詳細資訊，請參閱 https://docs.aws.amazon.com/general/latest/gr/signature-version-4.html 一般參考 中的AWS 簽章第 4 版簽署程序。

如果需要使用簽章第 2 版，則 AWS 一般參考中提供簽章第 2 版的使用資訊。

如需詳細資訊，請參閱下列內容：

- AWS 安全登入資料。提供有關用於存取之登入資料的一般資訊 AWS。
- IAM 中的安全最佳實務。 提供使用 IAM 服務的建議清單，以協助保護您的 AWS 資源。
- IAM 中的暫時安全憑證. 描述如何建立和使用暫時性安全憑證。

IAM 的文件歷史紀錄

下表說明 IAM 的主要文件更新。

變更	描述	日期
job_workflow_ref 接受為 GitHub Actions OIDC 供應商的替代範圍宣告	對於 GitHub Actions OIDC 提供者，IAM 現在接受sub宣告或job_workflow_ref 宣告做為角色信任政策中的身分提供者控制項。	2026 年 5 月 12 日
服務特定憑證 API 的 IAM 條件內容索引鍵	IAM 現在支援兩個新的條件內容索引鍵，用於控制對服務特定憑證 API 的存取：iam:ServiceSpecificCredentialAgeDays 和 iam:ServiceSpecificCredentialServiceName 。這些金鑰可讓您根據過期設定和允許的服務，限制服務特定登入資料的建立和管理 AWS 。 • iam:ServiceSpecificCredentialAgeDays • iam:ServiceSpecificCredentialServiceName	2025 年 9 月 4 日
網路周邊控制的 VPC 端點條件索引鍵	IAM 現在支援三個新的 AWS 全域條件內容金鑰，用於實作可擴展的網路周邊控制：aws:VpceAccount 、 aws:VpceOrgID 和 aws:VpceOrgPaths 。這三個索引鍵有	2025 年 8 月 28 日

	助於確保請求透過特定帳戶、組織或組織單位擁有的 VPC 端點傳遞，並隨著 VPC 端點用量自動擴展，而無需在建立新端點時更新政策。如需詳細資訊，請參閱使用資料周界建立許可防護機制。	
IAM SAML 聯合 CloudFormation 教學課程	IAM 新增了使用 CloudFormation建立 SAML 身分提供者 (IdP) 和聯合身分角色的新教學課程。 • 使用 CloudFormation建立 SAML IdP • 使用 建立 SAML 聯合角色 CloudFormation • 使用 建立 SAML IdP 和聯合角色 CloudFormation	2025 年 8 月 25 日
共用 OIDC 提供者的其他共用 OIDC 提供者控制項	IAM 現在將 Amazon Cognito、Azure Sentinel、Pulumi Cloud 和 Vercel 全域端點新增至共用 OIDC 身分提供者清單中，這些身分提供者要求對 JSON Web 權杖 (JWT) 的特定宣告進行明確評估。	2025 年 8 月 1 日
IAM Access Analyzer 新增了內部存取分析器	IAM Access Analyzer 可協助您識別組織或帳戶內的哪些主體有權存取所選業務關鍵型資源。內部存取分析器透過確保指定的資源僅可由 AWS 組織或帳戶內的預期主體存取，來支援實作最低權限原則。	2025 年 6 月 16 日

共用 OIDC 提供者的身分提供者控制項	IAM 現在要求針對已辨識的共用 OIDC 身分提供者明確評估 JSON Web 權杖 (JWT) 中的特定宣告。此安全控制可確保只有來自預期組織的授權身分可以擔任角色和存取 AWS 資源。	2025 年 6 月 6 日
IAMUserChangePassword ：新增許可	IAM 向 IAMUserChangePassword 新增了許可，以允許路徑中指定的使用者。	2025 年 5 月 28 日
AccessAnalyzerServiceRolePolicy ：新增許可	IAM Access Analyzer 已將 `iam:GetAccountAuthorizationDetails` 新增至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2025 年 5 月 12 日
AccessAnalyzerServiceRolePolicy ：新增許可	IAM Access Analyzer 已將 Amazon S3 目錄儲存貯體存取點新增至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2025 年 3 月 31 日
IAM 雙堆疊端點支援	IAM 現在提供更完善的雙堆疊端點支援，讓用戶端能夠使用 IPv4 或 IPv6 位址與 IAM 通訊。	2025 年 3 月 20 日
政策評估邏輯更新	政策評估邏輯的增強型文件，包括改善流程圖，以及更清楚地說明 如何 AWS 評估政策以決定是否允許或拒絕請求。	2025 年 1 月 30 日
IAMDeleteRootUserCredentials – 移除了許可	IAM 從受管政策中移除了 `iam:DeleteVirtualMFADevice` 許可。	2025 年 1 月 7 日

AccessAnalyzerServiceRolePolicy：新增許可	IAM Access Analyzer 新增了許可支援，可將 Amazon ECR 帳戶設定和登錄檔政策的相關資訊擷取至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2024 年 12 月 10 日
IAM Access Analyzer 新增了存取組態	IAM Access Analyzer 新增了設定分析器的支援，以變更產生調查結果的 AWS 帳戶、IAM 使用者和角色的範圍。	2024 年 11 月 14 日
集中管理成員帳戶的根存取權	您現在可以透過集中式根存取權，管理 AWS Organizations 中成員帳戶間的特權根使用者憑證。使用 集中保護 AWS 帳戶 受管 的根使用者憑證 AWS Organizations ，以移除和防止根使用者憑證大規模復原和存取。 • 根使用者最佳實務	2024 年 11 月 14 日
AWS 受管政策更新 - 新政策	IAM 新增了兩個新政策，以設定您在集中管理組織中成員帳戶的根使用者存取後可以啟動的特權根使用者工作階段的許可範圍。 • IAMAuditRootUserCredentials • IAMCreateRootUserPassword • IAMDeleteRootUserCredentials • S3UnlockBucketPolicy • SQSUnlockQueuePolicy	2024 年 11 月 14 日

支援 AWS Organizations 資源控制政策 RCPs)	使用 AWS Organizations 資源控制政策 (RCP) 來定義組織或組織單位 (OU) 中帳戶內資源的最大許可。RCP 會限制身分型和資源型政策可以授予組織內帳戶資源的許可。	2024 年 11 月 13 日
AccessAnalyzerServiceRolePolicy：新增許可	IAM Access Analyzer 新增了許可支援，可將 IAM 使用者和角色標籤的相關資訊擷取至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2024 年 10 月 29 日
SAML 加密支援增強功能	IAM SAML 提供者中 SAML 加密支援的增強型文件，其中包含經過改進的疑難排解指引和服務相容性說明。	2024 年 6 月 5 日
AccessAnalyzerServiceRolePolicy：新增許可	IAM Access Analyzer 新增了許可支援，可將 IAM 使用者和角色政策的相關資訊擷取至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2024 年 5 月 30 日
對 SAML 身分提供者的加密支援	IAM SAML 提供者現在支援來自外部 IdP 的 SAML 回應中的加密聲明。若要了解加密如何與 IAM SAML 聯合身分運作，請參閱使用 SAML 型聯合身分進行 API 存取。	2024 年 2 月 4 日
AccessAnalyzerServiceRolePolicy：新增許可	IAM Access Analyzer 新增了許可支援，可將 Amazon EC2 快照的封鎖公開存取的目前狀態擷取至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2024 年 1 月 23 日

AccessAnalyzerServiceRolePolicy：新增許可	IAM Access Analyzer 已將 DynamoDB 串流和資料表新增至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2024 年 1 月 11 日
AccessAnalyzerServiceRolePolicy：新增許可	IAM Access Analyzer 已將 Amazon S3 目錄儲存貯體新增至 AccessAnalyzerServiceRolePolicy 的服務層級許可。	2023 年 12 月 1 日
IAMAccessAnalyzerReadOnlyAccess：新增許可	IAM Access Analyzer 為 IAMAccessAnalyzerReadOnlyAccess 新增了許可，可用來檢查政策更新是否授予其他存取權。 IAM Access Analyzer 需要此許可，才能對您的政策執行政策檢查。	2023 年 11 月 26 日
IAM Access Analyzer 新增了未使用的存取權分析器	IAM Access Analyzer 簡化了檢查未使用的存取權，引導您達到最低權限。IAM Access Analyzer 會持續分析您的帳戶，以識別未使用的存取權，並建立包含調查結果的集中式儀表板。	2023 年 11 月 26 日
IAM Access Analyzer 新增了自訂政策檢查	IAM Access Analyzer 現在提供自訂政策檢查，可在部署之前驗證 IAM 政策是否符合您的安全標準。	2023 年 11 月 26 日

AccessAnalyzerServiceRolePolicy：新增許可	IAM Access Analyzer 為 AccessAnalyzerServiceRolePolicy 服務層級許可新增了 IAM 動作，以支援下列動作： • 列出政策的實體 • 產生上次存取的服務詳細資訊 • 列出存取金鑰資訊	2023 年 11 月 26 日
對 60 多項其他服務和動作的最近存取動作資訊和政策產生支援	IAM 現在支援最近存取的動作資訊，並針對 60 多項其他服務產生包含動作層級資訊的政策，以及可取得最近存取資訊的動作清單。	2023 年 11 月 1 日
對 140 多項服務的最近存取的動作資訊支援	IAM 現在會為 140 多項服務提供最近存取的動作資訊，以及可取得最近存取資訊的動作清單。	2023 年 9 月 14 日
支援根使用者和 IAM 使用者使用多台多重要素驗證 (MFA) 裝置	現在，您最多可以為每個使用者新增最多八台 MFA 裝置，包括 FIDO 安全金鑰、使用虛擬驗證器應用程式的以時間為基礎的一次性密碼 (TOTP)，或者硬體 TOTP 權杖。	2022 年 11 月 16 日

IAM Access Analyzer 支援新的資源類型	IAM Access Analyzer 已新增對下列資源類型的支援： • Amazon EBS 磁碟區快照 • Amazon ECR 儲存庫 • Amazon EFS 檔案系統 • Amazon RDS 資料庫快照 • Amazon RDS 資料庫叢集快照 • Amazon SNS 主題	2022 年 10 月 25 日
U2F 棄用和 WebAuthn/FIDO 更新	已移除提及 U2F 作為 MFA 選項的內容，並新增了有關 WebAuthn、FIDO2 和 FIDO 安全金鑰的資訊。	2022 年 5 月 31 日
IAM 中彈性的更新	新增有關當某個事件中斷 AWS 區域之間的通訊時，維持對 IAM 憑證進行存取的資訊。	2022 年 5 月 16 日
資源的新全域條件金鑰	您現在可以根據 中包含您資源的帳戶、組織單位 (OU) 或組織來控制對 資源 AWS Organizations 的存取。您可以在 IAM 政策中使用 `aws:ResourceAccount` 、`aws:ResourceOrgID` 以及 `aws:ResourceOrgPaths` 全域條件金鑰。	2022 年 4 月 27 日

使用 AWS SDKs IAM 程式碼範例	新增程式碼範例，示範如何搭配 AWS 軟體開發套件 (SDK) 使用 IAM。這些範例分為程式碼摘錄 (示範如何呼叫個別服務函數) 和範例 (示範如何透過呼叫相同服務中的多個函數來完成特定任務)。	2022 年 4 月 7 日
更新政策評估邏輯流程圖	更新政策評估邏輯流程圖，以及決定是否允許或拒絕帳戶中的請求區段中的相關文字。	2021 年 11 月 17 日
更新安全最佳實務	新增了有關建立系統管理使用者而非使用根使用者憑證的資訊，移除了使用 IAM 群組指派許可給 IAM 使用者的最佳實務，以及澄清何時使用受管政策而非內嵌政策。	2021 年 10 月 5 日
更新資源型政策的政策評估邏輯主題	新增有關資源型政策和相同帳戶中不同主體類型影響的資訊。	2021 年 10 月 5 日
更新單一值和多重值條件金鑰	現在會更詳細地說明單一值和多重值條件金鑰之間的差異。值類型新增至每個 AWS 全域條件內容金鑰。	2021 年 9 月 30 日
IAM Access Analyzer 支援 Amazon S3 多區域存取點	IAM Access Analyzer 可識別允許公有和跨帳戶存取的 Amazon S3 儲存貯體，包括使用 Amazon S3 多區域存取點。	2021 年 9 月 2 日
AWS 受管政策更新 - 更新現有政策	IAM Access Analyzer 已更新現有的 AWS 受管政策。	2021 年 9 月 2 日

支援更多服務用於動作層級政策產生	IAM Access Analyzer 可以為其他服務產生具有動作層級存取活動資訊的 IAM 政策 AWS 。	2021 年 8 月 24 日
產生跨帳戶追蹤的 IAM 政策	您現在可以使用 IAM Access Analyzer，使用不同帳戶中的 AWS CloudTrail 線索，例如集中式 AWS Organizations 線索，根據存取活動產生精細的政策。	2021 年 8 月 18 日
其他 IAM Access Analyzer 政策檢查	IAM Access Analyzer 透過新增對 IAM 政策中包含的條件進行驗證的新政策檢查，來延伸政策驗證。這些檢查會分析政策陳述式中的條件區塊，並報告安全性警告、錯誤和建議，以及可採取行動的建議。 IAM Access Analyzer新增了下列政策檢查： • 錯誤 — 無效的服務主體格式 • 錯誤 – 條件中缺少標籤鍵 • 安全性警告 - 使用服務不支援的標籤條件金鑰拒絕 NotAction • 安全性警告 - 使用服務不支援的標籤條件金鑰拒絕 • 安全性警告 - 遺失配對的條件金鑰 • 建議 - 使用服務不支援的標籤條件金鑰允許 NotAction • 建議 - 使用服務不支援的標籤條件金鑰允許	2021 年 6 月 29 日

對更多服務提供最近存取動作支援	您現在可以在 IAM 主控台中檢視關於 IAM 主體上次針對下列服務使用動作的最近存取動作資訊：Amazon EC2、IAM、Lambda 和 Amazon S3 管理動作。您也可以使用 AWS CLI 或 AWS API 來擷取資料報告。您可以使用此資訊來辨別不必要的許可，以便您能夠微調您的 IAM 政策以更完善地遵循最低權限的原則。	2021 年 4 月 19 日
監控並控制使用擔任角色所採取的動作	管理員可以將 IAM 角色設定為要求身分傳遞來源身分，其在 AWS CloudTrail中登入。檢閱來源身分資訊可協助管理員判斷曾使用擔任角色工作階段執行動作者。	2021 年 4 月 13 日
根據存取活動產生 IAM 政策	您現在可以使用 IAM Access Analyzer 根據您在 AWS CloudTrail中的存取活動產生精細政策。	2021 年 4 月 7 日
IAM Access Analyzer 政策檢查	IAM Access Analyzer 現在會在政策編寫期間提供超過 100 項政策檢查及可採取行動的建議。	2021 年 3 月 16 日
擴充的政策驗證選項	擴展政策驗證可在 IAM 主控台、 AWS API 中使用，並在 AWS CLI IAM Access Analyzer 中使用政策檢查來協助您撰寫安全且實用的 JSON 政策。	2021 年 3 月 15 日

標記 IAM 資源	現在，您可以使用標籤鍵值對標籤其他 IAM 資源。	2021 年 2 月 11 日
IAM 使用者的預設密碼政策	如果您未為 設定自訂密碼政策 AWS 帳戶，IAM 使用者密碼現在必須符合預設 AWS 密碼政策。	2020 年 11 月 18 日
AWS 服務的動作、資源和條件索引鍵頁面已移動	每個 AWS 服務都可以定義動作、資源和條件內容索引鍵，以用於 IAM 政策。您現在可以在 AWS 服務授權參考中找到服務及其動作、資源和條件內容索引鍵的清單。	2020 年 11 月 16 日
IAM 使用者的較長角色工作階段持續時間	IAM 使用者在 中切換角色時，現在可以有較長的角色工作階段持續時間 AWS 管理主控台，從而減少工作階段過期造成的中斷。使用者會被授予針對角色設定的工作階段持續時間上限，或 IAM 使用者工作階段中的剩餘時間，以較少者為準。	2020 年 7 月 24 日
使用服務配額 (Service Quotas) 來請求 IAM 實體的快速增加	您可以使用 Service Quotas 主控台，為可調整的 IAM 配額請求增加配額。現在，在 Service Quotas 中有些增加會自動核准，且在幾分鐘內您的帳戶便可使用。較大的請求會提交至 AWS 支援。	2020 年 6 月 25 日

IAM 中的最近存取資訊現在包括 Amazon S3 管理動作	除了服務最近存取資訊外，您現在還可以在 IAM 主控台中檢視 IAM 主體上次使用 Amazon S3 動作的相關資訊。您也可以使用 AWS CLI 或 AWS API 來擷取資料報告。此報告包含主體上次嘗試存取的允許服務和動作，以及何時存取的相關資訊。您可以使用此資訊來辨別不必要的許可，以便您能夠微調您的 IAM 政策以更完善地遵循最低權限的原則。	2020 年 6 月 3 日
新增安全性章節	安全章節可協助您了解如何設定 IAM 和 AWS STS 以符合您的安全與合規目標。您也會了解如何使用其他 AWS 服務來協助監控並保護 IAM 資源。	2020 年 4 月 29 日
sts:RoleSessionName	您現在可以撰寫根據主體擔任角色時所指定的工作階段名稱而授予許可的政策。	2020 年 4 月 21 日
AWS 登入頁面更新	當您登入主 AWS 登入頁面時，您現在可以選擇以 AWS 帳戶根使用者 或 IAM 使用者身分登入。當您這麼做時，頁面上的標籤會指出您是否應該提供根使用者電子郵件地址或您的 IAM 使用者資訊。本文件包含更新的螢幕擷取畫面，以協助您了解 AWS 登入頁面。	2020 年 3 月 4 日

aws:ViaAWSService 和 aws:CalledVia 條件金鑰	您現在可以寫入政策，限制服務是否能代表 IAM 主體 (使用者或角色) 提出請求。主體向 AWS 服務提出請求時，該服務可能會使用主體的憑證，向其他服務提出後續請求。若任何服務使用主體憑證提出請求，請使用 aws:ViaAWSService 條件金鑰來匹配。若特定服務使用主體憑證提出請求，請使用 aws:CalledVia 條件金鑰來匹配。	2020 年 2 月 20 日
政策模擬器新增了許可界限的支援	您現在可以使用 IAM 政策模擬器，測試許可界限對 IAM 實體的效果。	2020 年 1 月 23 日
跨帳戶政策評估	您現在可以了解 如何 AWS 評估跨帳戶存取的政策。當信任帳戶中的資源包含以資源為基礎的政策，而允許其他帳戶中的主體存取該資源時，就會發生這種情況。兩個帳戶中都必須允許此請求。	2020 年 1 月 2 日
工作階段標籤	現在當您在 AWS STS中擔任角色或聯合使用者時，可以包含標籤。當您執行 AssumeRole 或 GetFederationToken 操作時，您可以將工作階段標籤傳遞為屬性。當您執行 AssumeRoleWithSAML 或 AssumeRoleWithWebIdentity 操作時，您可以將屬性從公司身分傳遞至 AWS。	2019 年 11 月 22 日

控制 AWS 帳戶 中 群組的存取 AWS Organizations	您現在可以在 IAM 政策 AWS Organizations 中參考來自 的組織單位 (OUs)。如果您使用 AWS Organizations 將帳戶組織為 OUs，您可以在授予 資源存取權之前，要求委託人屬於特定 OU。委託人包括 AWS 帳戶根使用者 IAM 使用者和 IAM 角色。若要執行此作業，請在政策的 `aws:PrincipalOrgPaths` 條件金鑰中指定 OU 路徑。	2019 年 11 月 20 日
上次使用的角色	您現在可以檢視上次使用角色的日期、時間和區域。此資訊也可協助您識別帳戶中未使用的角色。您可以使用 AWS 管理主控台、 AWS CLI 和 AWS API 來檢視上次使用角色時的相關資訊。	2019 年 11 月 19 日
更新 Global condition context keys (全域條件內容索引鍵) 頁面	您現在可以了解每個全域條件金鑰包含在請求內容中的時機。您也可以使用頁面目錄 (TOC) 更輕易地導覽至每個金鑰。頁面上的資訊可協助您撰寫更精確的政策。例如，若您的員工搭配 IAM 角色使用聯合，建議您使用 `aws:userId` 金鑰而非 `aws:userName` 金鑰。`aws:userName` 金鑰僅適用於 IAM 使用者而非角色。	2019 年 10 月 6 日

中的 ABAC AWS	了解屬性型存取控制 (ABAC) 如何在 AWS 中使用標籤，以及它與傳統 AWS 授權模型的比較。使用 ABAC 教學來了解如何建立和測試政策，允許具備主體標籤的 IAM 角色存取具備相符標籤的資源。此策略允許個人僅檢視或編輯其任務所需的 AWS 資源。	2019 年 10 月 3 日
AWS STS GetAccessKeyInfo 操作	您可以檢閱程式碼中的 AWS 存取金鑰，以判斷金鑰是否來自您擁有的帳戶。您可以使用 `aws sts get-access-key-info` AWS CLI 命令或 `GetAccessKeyInfo` AWS API 操作傳遞存取金鑰 ID。	2019 年 7 月 24 日
在 IAM 中檢視 AWS Organizations 服務上次存取的資訊	您現在可以在 IAM 主控台的 AWS Organizations區段中檢視 AWS Organizations 實體或政策的服務上次存取資訊。您也可以使用 AWS CLI 或 AWS API 來擷取資料報告。此資料包含 AWS Organizations 帳戶主體上次嘗試存取的允許服務以及存取時間的相關資訊。您可以使用此資訊來識別不必要的許可，以便精簡您的 AWS Organizations 政策，以更好地遵守最低權限原則。	2019 年 6 月 20 日
使用受管政策做為工作階段政策	當您擔任角色時，您現在可以傳遞最多 10 個受管政策 ARN。這可讓您限制該角色臨時憑證的許可。	2019 年 5 月 7 日

AWS STS 全域端點工作階段字符的區域相容性	您現在可以選擇是否使用版本 1 或版本 2 的全域端點權杖。第 1 版字符僅在預設可用的 AWS 區域中有效。這些權杖不適用於手動啟用的區域，例如亞太區域 (香港)。版本 2 權杖在所有區域都有效。不過，版本 2 權杖較長且可能會影響暫時存放權杖的系統。	2019 年 4 月 26 日
允許啟用和停用 AWS 區域	您現在可以建立可讓管理員啟用和停用亞太區域 (香港) (ap-east-1)的政策。	2019 年 4 月 24 日
IAM 使用者的「我的安全憑證」頁面	IAM 使用者現在可以在 My Security Credentials (我的安全憑證) 頁面上管理其所有憑證。 AWS 管理主控台 此頁面會顯示帳戶資訊，例如帳戶 ID 和正式使用者 ID。使用者也可以檢視和編輯其密碼、存取金鑰、X.509 憑證、SSH 金鑰和 Git 憑證。	2019 年 1 月 24 日
存取分析器 API	您現在可以使用 AWS CLI 和 AWS API 來檢視服務上次存取的資訊。	2018 年 12 月 7 日
標記 IAM 使用者和角色	您現在可以使用 IAM 標籤，以使用標籤鍵值組將自訂屬性新增到身分 (IAM 使用者或角色)。您也可以使用標籤來控制身分對資源的存取權或控制哪些標籤可以連接到身分。	2018 年 11 月 14 日

U2F 安全金鑰	您現在可以在登入 AWS 管理主控台時使用 U2F 安全金鑰的多重要素驗證 (MFA) 選項。	2018 年 9 月 25 日
支援 Amazon VPC 端點	您現在可以在美國西部（奧勒岡）AWS STS 區域建立 VPC 和 之間的私有連線。	2018 年 7 月 31 日
許可界限	新功能可讓您更輕鬆地授與信任的員工管理 IAM 許可的能力，無需授與完整的 IAM 管理存取許可。	2018 年 7 月 12 日
aws:PrincipalOrgID	新條件金鑰透過指定 IAM 主體 AWS 的組織，提供更簡單的方法來控制對 AWS 資源的存取。	2018 年 5 月 17 日
aws:RequestedRegion	新的條件金鑰可讓您更輕鬆地使用 IAM 政策來控制對 AWS 區域的存取。	2018 年 4 月 25 日
增加 IAM 角色的工作階段持續時間	IAM 角色現在可以有 12 小時的工作階段持續時間。	2018 年 3 月 28 日
更新角色建立工作流程	新工作流程改善建立信任關係及連接許可至角色的處理程序。	2017 年 9 月 8 日
AWS 帳戶 登入程序	更新的 AWS 登入體驗允許根使用者和 IAM 使用者使用 AWS 管理主控台首頁上的登入主控台連結。	2017 年 8 月 25 日
範例 IAM 政策	文件更新推出超過 30 個範例政策。	2017 年 8 月 2 日

IAM 最佳實務	資訊新增到 IAM 主控台的Users (使用者) 區段，可更輕鬆遵循 IAM 最佳實務。	2017 年 7 月 5 日
Auto Scaling 資源	資源層級許可能夠控制 Auto Scaling 資源的存取和許可。	2017 年 5 月 16 日
Amazon RDS for MySQL 和 Amazon Aurora 資料庫	資料庫管理員可以將資料庫使用者與 IAM 使用者和角色建立關聯，進而管理使用者從單一位置存取所有 AWS 資源的權限。	2017 年 4 月 24 日
服務連結角色	服務連結角色提供更輕鬆、更安全的方式來將許可委派給 AWS 服務。	2017 年 4 月 19 日
政策摘要	新的政策摘要讓您更輕鬆了解 IAM 政策中的許可。	2017 年 3 月 23 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。