aws

使用者指南

Amazon Verified Permissions

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

Amazon Verified Permissions: 使用者指南

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

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

Table of Contents

什麼是 Amazon Verified Permissions？ 1
驗證許可中的授權 1
Cedar 政策語言 1
Verified Permissions 的優點 2
加速應用程式開發 2
更安全的應用程式 2
最終使用者功能 2
相關服務 2
存取已驗證的許可 3
Verified Permissions 的定價 4
政策存放區的入門 6
先決條件 7
步驟 1：建立 PhotoFlash 政策存放區 8
步驟 2：建立政策 9
步驟 3：測試政策存放區 9
步驟 4：清除資源 11
設計授權模型 12
沒有單一正確的模型 13
傳回錯誤 13
專注於資源 13
考慮多租戶 14
比較共用政策存放區和每個租用戶政策存放區 16
如何選擇 17
政策存放區 18
建立政策存放區 18
使用 Rust 建立政策存放區 25
API 連結政策存放區 30
運作方式 32
考量事項 33
新增 ABAC 34
移至生產環境 35
故障診斷 37
刪除政策存放區 40
政策存放區結構描述 42

編輯結構描述 44
政策驗證模式 46
政策 48
建立靜態政策 48
編輯靜態政策 50
..... 52
評估範例內容 54
測試政策 59
政策範例 61
使用括號表示法來參考字符屬性 62
使用點表示法來參考屬性 62
反映 Amazon Cognito ID 字符屬性 63
反映 OIDC ID 字符屬性 63
反映 Amazon Cognito 存取字符屬性 64
反映 OIDC 存取權杖屬性 64
政策範本和範本連結政策 65
建立政策範本 65
建立範本連結政策 66
編輯政策範本 68
範本連結政策範例 70
PhotoFlash 範例 70
DigitalPetStore 範例 71
TinyToDo 範例 72
身分來源 73
使用 Amazon Cognito 身分來源 74
使用 OIDC 身分來源 76
用戶端和對象驗證 77
JWTs的用戶端授權 78
建立身分來源 80
Amazon Cognito 身分來源 80
OIDC 身分來源 82
編輯身分來源 85
Amazon Cognito 使用者集區身分來源 85
OpenID Connect (OIDC) 身分來源 87
將字符映射至結構描述 88
映射 ID 字符 89

映射存取權杖 93
Amazon Cognito 冒號分隔宣告的替代表示法 97
結構描述映射須知 98
整合 102
使用 Express 102
先決條件 103
設定整合 103
設定授權 104
實作授權中介軟體 106
測試整合 107
故障診斷 107
後續步驟 108
授權請求 109
API 操作 109
測試模型 110
與應用程式整合 112
安全 115
資料保護 115
資料加密 116
身分與存取管理 117
目標對象 117
使用身分驗證 118
使用政策管理存取權 120
Amazon Verified Permissions 如何使用 IAM 122
IAM Verified Permissions 的 政策 127
身分型政策範例 130
AWS 受管政策 132
故障診斷 135
法規遵循驗證 137
恢復能力 138
監控 139
CloudTrail 日誌 139
CloudTrail 中的已驗證許可資訊 139
了解 Verified Permissions 日誌檔案項目 140
使用 AWS CloudFormation 158
已驗證的許可和 AWS CloudFormation 範本 158

AWS CDK 建構 158
進一步了解 AWS CloudFormation 159
使用 AWS PrivateLink 160
考量事項 160
建立介面端點 160
建立端點政策 160
配額 162
資源的配額 162
範本連結政策大小範例 163
階層的配額 165
每秒操作的配額 166
術語和概念 169
授權模型 169
授權請求 170
授權回應 170
已考量的政策 170
內容資料 170
決定政策 170
實體資料 170
許可、授權和主體 171
政策強制執行 171
政策存放區 171
滿意的政策 171
Cedar 的差異 171
命名空間定義 171
政策範本支援 172
結構描述支援 172
動作群組定義 172
實體格式 172
長度和大小限制 177
Cedar v4 常見問答集 179
升級的目前狀態為何？ 179
我需要現在執行任何動作嗎？ 179
升級主控台是否會影響授權服務？ 179
Cedar v3 和 Cedar v4 中的重大變更是什麼？ 179
何時完成 Cedar v4 的升級？ 180

文件歷史紀錄 181
..... clxxxiii

什麼是 Amazon Verified Permissions？

Amazon Verified Permissions 是一種可擴展的精細許可管理和授權服務，適用於您建置的自訂應用程式。Verified Permissions 可讓您的開發人員透過外部化授權並集中管理政策，更快速地建置安全的應用程式。Verified Permissions 使用 Cedar 政策語言來定義精細的許可，以保護您應用程式的資源。

如需使用 Verified Permissions 設定政策決策點 (PDP) 的指引和範例，請參閱 AWS 方案指引中的使用 Amazon Verified Permissions 實作 PDP。

主題

- 驗證許可中的授權
- Cedar 政策語言
- Verified Permissions 的優點
- 相關服務
- 存取已驗證的許可
- Verified Permissions 的定價

驗證許可中的授權

Verified Permissions 透過驗證是否允許委託人在應用程式中的指定內容中對資源執行動作，來提供授權。Verified Permissions 假設委託人先前已透過其他方式識別和驗證，例如使用 OpenID Connect 等通訊協定、Amazon Cognito 等託管提供者，或其他身分驗證解決方案。Verified Permissions 與主體的管理位置以及驗證方式無關。

Verified Permissions 是一種服務，可讓客戶在 中建立、維護和測試政策 AWS Management Console、以程式設計方式使用 Verified Permissions APIs，或透過基礎設施做為程式碼解決方案 AWS CloudFormation。許可是以 Cedar 政策語言表示。用戶端應用程式會呼叫授權 APIs 來評估與服務一起存放的 Cedar 政策，並提供是否允許 動作的存取決策。

Cedar 政策語言

Verified Permissions 中的授權政策是使用 Cedar 政策語言撰寫。Cedar 是一種開放原始碼語言，用於撰寫授權政策，並根據這些政策做出授權決策。當您建立應用程式時，您需要確保只有授權的委託人、人類使用者或機器可以存取應用程式，並且只能執行他們獲得授權執行的操作。使用 Cedar，您可以

將商業邏輯與授權邏輯分離。在應用程式的程式碼中，您會在呼叫 Cedar 授權引擎時對操作提出請求前，詢問「是否授權此請求？」。然後，如果決策為「允許」，應用程式可以執行請求的操作，如果決策為「拒絕」，則傳回錯誤訊息。

Verified Permissions 目前使用 Cedar 2.4 版。

如需 Cedar 的詳細資訊，請參閱下列內容：

- Cedar 政策語言參考指南
- Cedar GitHub 儲存庫

Verified Permissions 的優點

加速應用程式開發

將授權從商業邏輯解耦，以加速應用程式開發。

Verified Permissions 提供與熱門開發架構的整合，讓您以最少的程式碼變更，更輕鬆地在應用程式中實作授權。這些整合可讓您專注於核心業務邏輯，同時 Verified Permissions 會處理授權決策。

- Express.js – 以中介軟體為基礎的整合，可讓您保護 Express 應用程式中的 API 端點，而無需修改現有的路由處理常式。如需詳細資訊，請參閱the section called “使用 Express”。

更安全的應用程式

Verified Permissions 可讓開發人員建置更安全的應用程式。

最終使用者功能

Verified Permissions 可讓您為許可管理提供更豐富的最終使用者功能。

相關服務

- Amazon Cognito – Amazon Cognito 是適用於 Web 和行動應用程式的身分平台。是一種使用者目錄、身分驗證伺服器，以及 OAuth 2.0 存取權杖和 AWS 憑證的授權服務。建立政策存放區時，您可以選擇從 Amazon Cognito 使用者集區建置委託人和群組。如需詳細資訊，請參閱 Amazon Cognito 開發人員指南。

- Amazon API Gateway – Amazon API Gateway 是一種 AWS 服務，用於建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。當您建立政策存放區時，您可以選擇從 API Gateway 中的 API 建置動作和資源。如需 API Gateway 的詳細資訊，請參閱 API Gateway 開發人員指南。
- AWS IAM Identity Center – 透過 IAM Identity Center，您可以管理人力資源身分的登入安全性，也稱為人力資源使用者。IAM Identity Center 提供一個位置，您可以在其中建立或連接人力資源使用者，並集中管理其所有 AWS 帳戶 和應用程式的存取權。如需詳細資訊，請參閱「AWS IAM Identity Center 使用者指南」。

存取已驗證的許可

您可以透過下列任何方式使用 Amazon Verified Permissions。

AWS Management Console

主控台是以瀏覽器為基礎的界面，用於管理 Verified Permissions 和資源 AWS 。如需透過主控台存取已驗證許可的詳細資訊，請參閱AWS 登入 《 使用者指南》中的如何登入 AWS。

- Amazon Verified Permissions 主控台

AWS 命令列工具

您可以使用 AWS 命令列工具在系統的命令列發出命令，以執行 Verified Permissions 和 AWS 任務。使用命令列可以比主控台更快，也更便利。若您想要建構執行 AWS 任務的指令碼，命令列工具也非常實用。

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

- 《 AWS CLI 命令參考》中的已驗證許可
- 中的 Amazon Verified 許可 AWS Tools for Windows PowerShell

AWS SDKs

AWS 提供SDKs（ 軟體開發套件 ），其中包含適用於各種程式設計語言和平台 (Java、Python、Ruby、.NET、iOS、Android 等 ） 的程式庫和範本程式碼。SDKs提供便捷的方式來建立對 Verified Permissions 和 的程式設計存取 AWS。例如，開發套件會負責的工作諸如以密碼演算法簽署請求、管理錯誤以及自動重試請求。

若要進一步了解和 AWS SDKs，請參閱適用於 的工具 Amazon Web Services。

以下是各種 AWS SDKs 中已驗證許可資源的文件連結。

- 適用於 .NET 的 AWS SDK
- 適用於 C++ 的 AWS SDK
- 適用於 Go 的 AWS SDK
- 適用於 Java 的 AWS SDK
- 適用於 JavaScript 的 AWS SDK
- 適用於 PHP 的 AWS SDK
- AWS SDK for Python (Boto)
- 適用於 Ruby 的 AWS SDK
- 適用於 Rust 的 AWS SDK

AWS CDK 建構

AWS Cloud Development Kit (AWS CDK) 是開放原始碼軟體開發架構，用於在程式碼中定義雲端基礎設施並透過其佈建 AWS CloudFormation。建構或可重複使用的雲端元件可用於建立 AWS CloudFormation 範本。然後，您可以使用這些範本來部署雲端基礎設施。

若要進一步了解並下載 AWS CDK，請參閱AWS 雲端開發套件。

以下是 Verified Permissions AWS CDK 資源的文件連結，例如 constructs。

- Amazon Verified Permissions L2 CDK 建構

已驗證的許可 API

您可以使用 Verified Permissions API 以 AWS 程式設計方式存取 Verified Permissions，這可讓您直接向 服務發出 HTTPS 請求。當您使用 API 時，必須包含使用您的登入資料來數位簽署請求的程式碼。

- Amazon Verified Permissions API 參考指南

Verified Permissions 的定價

Verified Permissions 根據應用程式對 Verified Permissions 每月提出的授權請求數量，提供分層定價。政策管理動作也會根據應用程式對 Verified Permissions 每月提出的 cURL （用戶端 URL) 政策 API 請求數量定價。

如需 Verified Permissions 費用和價格的完整清單，請參閱 Amazon Verified Permissions 定價。

若要查看您的帳單，請前往 AWS 帳單與成本管理 主控台中的帳單與成本管理儀表板。您的帳單內含用量報告的連結，可提供帳單的詳細資訊。若要進一步了解 AWS 帳戶 帳單，請參閱AWS Billing 《使用者指南》。

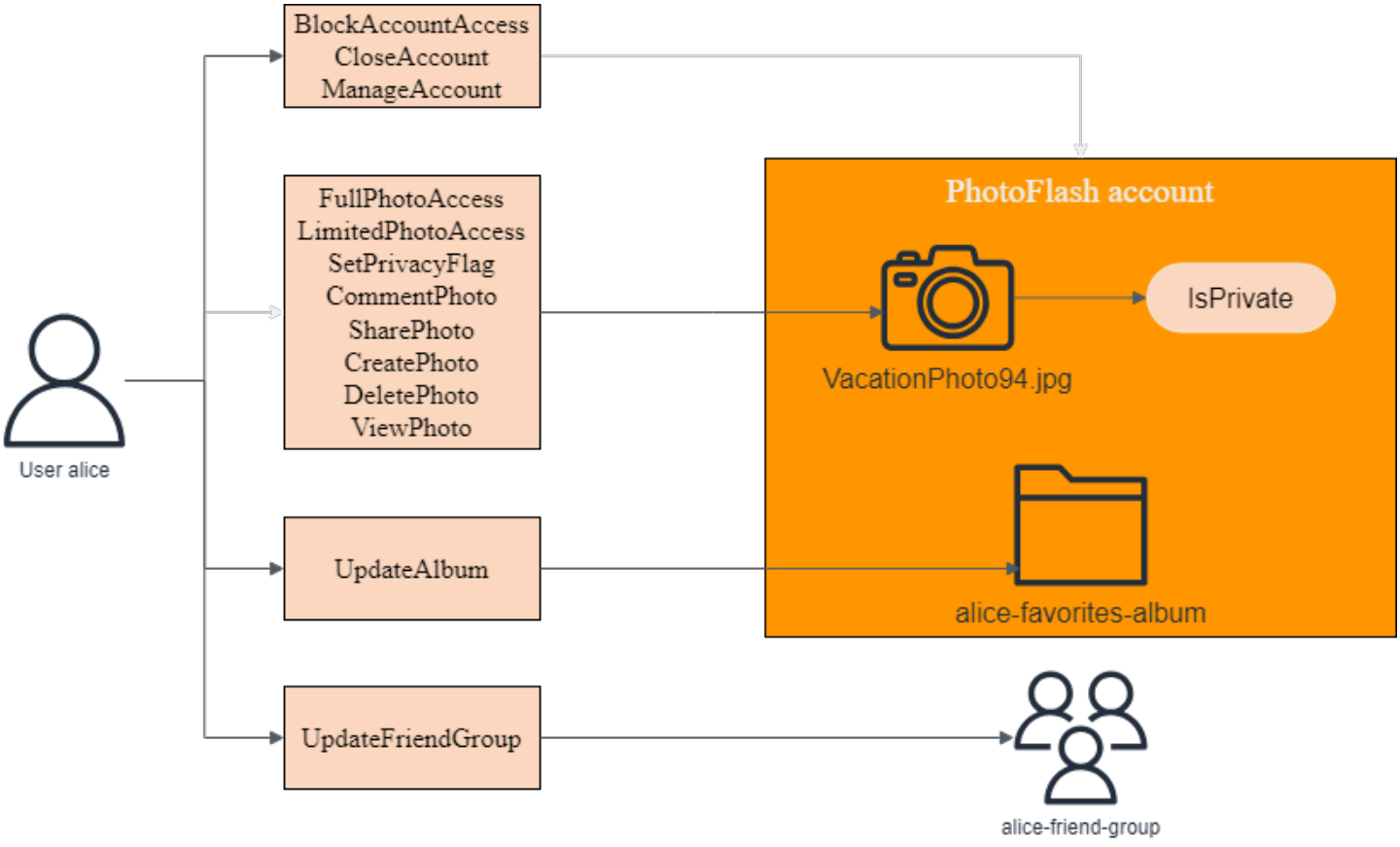
如果您對 AWS 帳單、帳戶和事件有任何疑問，請聯絡 支援 。

建立您的第一個 Amazon Verified Permissions 政策存放區

在本教學課程中，假設您是相片共享應用程式的開發人員，而且您正在尋找方法來控制應用程式使用者可以執行的動作。您想要控制誰可以新增、刪除或檢視相片和相簿。您也想要控制使用者可以對其帳戶採取哪些動作。他們可以管理自己的帳戶，朋友的帳戶呢？ 若要控制這些動作，您可以建立根據使用者身分允許或禁止這些動作的政策。Verified Permissions 提供政策存放區或容器來存放這些政策。

在本教學課程中，我們將逐步解說如何使用 Amazon Verified Permissions 主控台建立範例政策存放區。主控台提供幾個範例政策存放區選項，我們將建立 PhotoFlash 政策存放區。此政策存放區允許 使用者等主體對相片或相簿等資源執行共用等動作。

下圖說明委託人 與 之間的關係`User::alice`，以及她可以對各種資源採取的動作，也就是她的 PhotoFlash 帳戶、 `VactionPhoto94.jpg` 檔案`alice-favorites-album`、相簿 和使用者群組 `alice-friend-group`。



現在您已了解 PhotoFlash 政策存放區，讓我們建立並探索政策存放區。

先決條件

註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

   註冊程序的一部分包括接聽電話或文字訊息，並在電話鍵盤上輸入驗證碼。

   當您註冊 時 AWS 帳戶，AWS 帳戶根使用者會建立 。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行需要根使用者存取權的任務。

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

建立具有管理存取權的使用者

註冊 後 AWS 帳戶，請保護 AWS 帳戶根使用者、啟用 AWS IAM Identity Center和建立 管理使用者，以免將根使用者用於日常任務。

保護您的 AWS 帳戶根使用者

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

   如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的以根使用者身分登入。
2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

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

建立具有管理存取權的使用者

1. 啟用 IAM Identity Center。

如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的啟用 AWS IAM Identity Center。

2. 在 IAM Identity Center 中，將管理存取權授予使用者。

   如需使用 IAM Identity Center 目錄 做為身分來源的教學課程，請參閱AWS IAM Identity Center 《 使用者指南》中的使用預設值設定使用者存取權 IAM Identity Center 目錄。

以具有管理存取權的使用者身分登入

- 若要使用您的 IAM Identity Center 使用者簽署，請使用建立 IAM Identity Center 使用者時傳送至您電子郵件地址的簽署 URL。

  如需使用 IAM Identity Center 使用者登入的說明，請參閱AWS 登入 《 使用者指南》中的登入 AWS 存取入口網站。

指派存取權給其他使用者

1. 在 IAM Identity Center 中，建立一個許可集來遵循套用最低權限的最佳實務。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的建立許可集。

2. 將使用者指派至群組，然後對該群組指派單一登入存取權。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的新增群組。

步驟 1：建立 PhotoFlash 政策存放區

在下列程序中，您將使用 AWS 主控台建立 PhotoFlash 政策存放區。

建立 PhotoFlash 政策存放區

1. 在 Verified Permissions 主控台中，選擇建立新政策存放區。
2. 針對開始選項，從範例政策存放區選擇開始。
3. 針對範例專案，選擇 PhotoFlash。
4. 選擇建立政策存放區。

看到「已建立和設定的政策存放區」訊息後，請選擇移至概觀來探索您的政策存放區。

步驟 2：建立政策

當您建立政策存放區時，已建立預設政策，允許使用者完全控制自己的帳戶。這是一個有用的政策，但為了我們的目的，讓我們建立更嚴格的政策來探索 Verified Permissions 的細微差別。如果您記得我們在教學課程稍早看到的圖表，我們有委託人 `User::alice`，他們可以`UpdateAlbum`在資源 上執行動作 `alice-favorites-album`。讓我們新增允許 Alice 和僅限 Alice 管理此相簿的政策。

建立政策

1. 在 Verified Permissions 主控台中，選擇您在步驟 1 中建立的政策存放區。
2. 在導覽中，選擇政策。
3. 選擇建立政策，然後選擇建立靜態政策。
4. 針對政策效果，選擇允許。
5. 對於主體範圍，選擇特定主體，然後對於指定實體類型，選擇 PhotoFlash：：User，對於指定實體識別符，輸入 **`alice`**。
6. 在資源範圍中，選擇特定資源，然後在指定實體類型中，選擇 PhotoFlash：：Album，然後在指定實體識別符中，輸入 **`alice-favorites-album`**。
7. 針對動作範圍，選擇特定的動作集，然後針對此政策應套用的動作，選取 UpdateAlbum (UpdateAlbum)。 UpdateAlbum
8. 選擇下一步。
9. 在詳細資訊下，針對政策描述 - 選擇性輸入 **`Policy allowing alice to update alice-favorites-album.`**。
10. 選擇 Create policy (建立政策)

現在您已建立政策，您可以在 Verified Permissions 主控台中測試該政策。

步驟 3：測試政策存放區

建立政策存放區和政策之後，您可以使用 Verified Permissions 測試工作台執行模擬授權請求來測試它們。

測試政策存放區政策

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇測試工作台。

3. 選擇視覺化模式。
4. 對於委託人，請執行下列動作：
   a. 針對主體採取動作，選擇 PhotoFlash：：User，並針對指定實體識別符，輸入 **`alice`**。
   b. 在屬性下，針對帳戶：實體，請確定已選取 PhotoFlash：：Account 實體，並針對指定實體識別符，輸入 **`alice-account`**。
5. 在資源下，針對主體執行動作的資源，選擇 PhotoFlash：：Album 資源類型，然後在指定實體識別符下，輸入 **`alice-favorites-album`**。
6. 針對動作，從有效動作清單中選擇 PhotoFlash：：Action：："UpdateAlbum"。
7. 在頁面頂端，選擇執行授權請求，以模擬範例政策存放區中 Cedar 政策的授權請求。測試工作台應該會顯示決策：允許 指出我們的政策如預期般運作。

下表提供您可以使用 Verified Permissions 測試台測試的委託人、資源和動作的其他值。資料表包含以 PhotoFlash 範例政策存放區隨附的靜態政策，以及您在步驟 2 中建立的政策為基礎的授權請求決策。

委託人值	委託人帳戶：實體值	資源值	資源父值	Action	授權決策
PhotoFlash：：User | Bub	PhotoFlash：：Account | alice-account	PhotoFlash：：Album | alice-favorites-album	N/A	PhotoFlash：：Action：："UpdateAlbum"	拒絕
PhotoFlash：：User | alice	PhotoFlash：：Account | alice-account	PhotoFlash：：Photo | photo.jpeg	PhotoFlash：：Account | bob-account	PhotoFlash：：Action：："ViewPhoto"	拒絕
PhotoFlash：：User | alice	PhotoFlash：：Account | alice-account	PhotoFlash：：Photo | photo.jpeg	PhotoFlash：：Account | alice-account	PhotoFlash：：Action：："ViewPhoto"	允許
PhotoFlash：：User | alice	PhotoFlash：：Accoun	PhotoFlash：：Photo	PhotoFlash：：Album	PhotoFlash：：Action	拒絕

委託人值	委託人帳戶：實體值	資源值	資源父值	Action	授權決策
	t | alice-account	| bob-photo.jpeg	| Bob-Vacation-Album	：："DeletePhoto"	

步驟 4：清除資源

完成探索政策存放區後，請將其刪除。

刪除政策存放區

1. 在 Verified Permissions 主控台中，選擇您在步驟 1 中建立的政策存放區。
2. 在導覽中，選擇設定。
3. 在刪除政策存放區下，選擇刪除此政策存放區。
4. 在刪除此政策存放區？對話方塊中，輸入刪除，然後選擇刪除。

設計授權模型的最佳實務

當您準備在軟體應用程式內使用 Amazon Verified Permissions 服務時，第一步是立即跳到撰寫政策陳述式可能很困難。這類似於在完全決定應用程式應該怎麼做之前，透過撰寫 SQL 陳述式或 API 規格開始開發應用程式的其他部分。反之，您應該從使用者體驗開始。然後，從該體驗向後工作，以達成實作方法。

當您執行這項工作時，您會發現自己會提出問題，例如：

- 我的資源有哪些？ 如何組織它們？ 例如，檔案是否位於 資料夾內？
- 資源的組織是否在許可模型中扮演某種角色？
- 主體可以對每個資源執行哪些動作？
- 主體如何取得這些許可？
- 您希望最終使用者從預先定義的許可中選擇「Admin」、「Operator」或「ReadOnly」，還是他們應該建立臨時政策陳述式？ 還是兩者？
- 角色是全域或範圍？ 例如，單一租用戶中的「運算子」是否受到限制，或者「運算子」是否表示整個應用程式的運算子？
- 轉譯使用者體驗需要哪些類型的查詢？ 例如，是否需要列出委託人可存取的所有資源，以轉譯該使用者的首頁？
- 使用者可以不小心將自己鎖定在自己的資源之外嗎？ 是否需要避免？

此練習的最終結果稱為授權模型；它定義了主體、資源、動作，以及它們如何相互關聯。產生此模型不需要 Cedar 或 Verified Permissions 服務的獨特知識。相反地，它是使用者體驗設計練習的首要任務，就像任何其他練習一樣，並且可以在成品中呈現，例如界面模型、邏輯圖，以及許可如何影響使用者在產品中可以執行的操作的整體描述。Cedar 的設計具有足夠的彈性，可滿足模型中的客戶，而不是強制模型不自然地彎曲以符合 Cedar 的實作。因此，清楚了解所需的使用者體驗是實現最佳模型的最佳方式。

為了協助回答問題並獲得最佳模型，請執行下列動作：

- 在 Cedar 政策語言參考指南中檢閱 Cedar 設計模式。
- 請考慮 Cedar 政策語言參考指南中的最佳實務。
- 請考慮此頁面中包含的最佳實務。

最佳實務

- 沒有正式的「正確」模型
- 傳回 403 禁止的錯誤，而不是 404 找不到錯誤
- 專注於 API 操作以外的資源
- 多租戶考量

沒有正式的「正確」模型

當您設計授權模型時，沒有單一且唯一的正確答案。不同的應用程式可以有效地使用不同的授權模型來實現類似的概念，這是可以的。例如，請考慮電腦檔案系統的表示法。當您在類似 Unix 的作業系統中建立檔案時，它不會自動繼承父資料夾的許可。相反地，在許多其他作業系統和大多數線上檔案共用服務中，檔案確實繼承了其父資料夾的許可。根據應用程式最佳化的情況，這兩個選項都是有效的。

授權解決方案的正確性並非絕對，但應該根據它如何提供客戶所需的體驗，以及它是否以預期的方式保護其資源來檢視。如果您的授權模型交付此項目，則表示成功。

因此，以所需的使用者體驗開始設計是建立有效授權模型最有幫助的先決條件。

傳回 403 禁止的錯誤，而不是 404 找不到錯誤

最好將 403 禁止的錯誤傳回給包含 實體的請求，特別是 資源，該請求未對應至任何政策，而不是 404 找不到錯誤。這可提供最高層級的安全性，因為您不會公開實體是否存在，只是請求不符合政策存放區中任何政策的政策條件。

專注於 API 操作以外的資源

在大多數應用程式中，許可是以支援的資源為模型。例如，檔案共用應用程式可能代表許可，做為可在檔案或資料夾上執行的動作。這是很好、簡單的模型，可抽象基礎實作和後端 API 操作。

相反地，其他類型的應用程式，特別是 Web 服務，經常設計 API 操作本身的許可。例如，如果 Web 服務提供名為 的 APIcreateThing()，授權模型可能會定義對應的許可，或在名為 的 Cedar `action`中定義 `createThing`。這可在許多情況下運作，並讓您輕鬆了解許可。若要叫用 `createThing`操作，您需要 `createThing`動作許可。看起來很簡單，對吧？

您會發現 Verified Permissions 主控台中的入門程序包含直接從 API 建置資源和動作的選項。這是有用的基準：您的政策存放區與其授權的 API 之間的直接映射。

不過，當您進一步開發模型時，此以 API 為中心的方法可能不適合具有非常精細授權模型的應用程式，因為 APIs 只是您客戶真正嘗試保護的代理：基礎資料和資源。如果多個 APIs控制對相同資源的存取，管理員可能很難推斷這些資源的路徑，並相應地管理存取。

例如，請考慮包含組織成員的使用者目錄。使用者可以組織成群組，而其中一個安全目標是禁止未經授權方探索群組成員資格。管理此使用者目錄的服務提供兩種 API 操作：

- `listMembersOfGroup`
- `listGroupMembershipsForUser`

客戶可以使用其中一個操作來探索群組成員資格。因此，許可管理員必須記住協調對這兩個操作的存取。如果您稍後選擇新增 API 操作以解決其他使用案例，這會更複雜，例如：

- `isUserInGroups`（新的 API，可快速測試使用者是否屬於一或多個群組）

從安全角度來看，此 API 開啟第三個路徑來探索群組成員資格，中斷管理員精心製作的許可。

我們建議您專注於基礎資料和資源及其關聯操作。將此方法套用到群組成員資格範例，將導致抽象許可，例如 `viewGroupMembership`，這三個 API 操作都必須參考。

API 名稱	許可
`listMembersOfGroup`	需要 群組的`viewGroupMembership` 許可
`listGroupMembershipsForUser`	需要 使用者的`viewGroupMembership` 許可
`isUserInGroups`	需要 使用者的`viewGroupMembership` 許可

透過定義這一項許可，管理員成功控制對探索群組成員資格的存取，無論現在還是永遠。做為權衡，每個 API 操作現在都必須記錄可能需要的數個許可，而且管理員在製作許可時必須參考此文件。必要時，這可能是有效的權衡，以滿足您的安全需求。

多租戶考量

您可能想要開發供多個客戶使用的應用程式 - 取用您應用程式的企業，或租用戶 - 並將其與 Amazon Verified Permissions 整合。在您開發授權模型之前，請開發多租戶策略。您可以在一個共用政策存放

區中管理客戶的政策，或為每個租用戶政策存放區指派每個政策。如需詳細資訊，請參閱 AWS 規範指南中的 Amazon Verified Permissions 多租戶設計考量事項。

1. 一個共用政策存放區

   所有租戶共用單一政策存放區。應用程式會將所有授權請求傳送至共用政策存放區。

2. 每個租戶政策存放區

   每個租戶都有一個專用政策存放區。應用程式會根據提出請求的租戶，查詢不同的政策存放區以取得授權決策。

這兩種策略都不會對您的 AWS 帳單產生很大的影響。那麼，您應該如何設計您的方法？ 以下是可能對您的 Verified Permissions 多租用戶授權策略有所貢獻的常見條件。

租用戶政策隔離

將每個租戶的政策與其他租戶隔離對於保護租戶資料至關重要。當每個租用戶都有自己的政策存放區時，他們都有自己的隔離政策集。

授權流程

您可以在請求中，使用每個租用戶的政策存放區，識別提出授權請求的租用戶。使用共用政策存放區時，所有請求都會使用相同的政策存放區 ID。

範本和結構描述管理

當您的應用程式有多個政策存放區時，您的政策範本和政策存放區結構描述會在每個政策存放區中新增設計和維護開銷層級。

全球政策管理

您可能想要將一些全域政策套用至每個租戶。管理全球政策的開銷層級，會因共用政策存放區模型和每個租用戶政策存放區模型而有所不同。

租戶離職

有些租戶將為您的結構描述和其案例特定的政策提供元素。當租用戶不再與組織處於作用中狀態，而且您想要移除其資料時，工作量會隨著與其他租用戶的隔離程度而有所不同。

服務資源配額

Verified Permissions 具有資源和請求率配額，可能會影響您的多租用戶決策。如需配額的詳細資訊，請參閱 資源的配額。

比較共用政策存放區和每個租用戶政策存放區

每個考量需要自己的時間和資源投入程度，才能用於共用政策存放區模型和每個租戶政策存放區模型。

考量事項	共用政策存放區中的工作層級	每個租用戶政策存放區的工作量層級
租用戶政策隔離	中。Must include tenant identifiers in policies and authorization requests.	低。 Isolation is default behavior. Tenant-specific policies are inaccessible to other tenants.
授權流程	低。 All queries target one policy store.	中。 Must maintain mappings between each tenant and their policy store ID.
範本和結構描述管理	低。 Must make one schema work for all tenants.	高。 Schemas and templates might be less complex individually, but changes require more coordination and complexity.
全球政策管理	低。 All policies are global and can be centrally updated.	高。 You must add global policies to each policy store in onboarding. Replicate global policy updates between many policy stores.
租戶離職	高。 Must identify and delete only tenant-specific policies.	低。 Delete the policy store.
服務資源配額	高。 Tenants share resource quotas that affect policy stores like schema size, policy size per resource, and identity sources per policy store.	低。 Each tenant has dedicated resource quotas.

如何選擇

每個多租戶應用程式都不同。在做出架構決策之前，請仔細比較這兩種方法及其考量。

如果您的應用程式不需要租用戶特定的政策，並使用單一身分來源，則所有租用戶的一個共用政策存放區可能是最有效的解決方案。這會導致更簡單的授權流程和全球政策管理。使用一個共用政策存放區讓租用戶離職需要的精力較少，因為應用程式不需要刪除租用戶特定的政策。

但是，如果您的應用程式需要許多租戶特定的政策，或使用多個身分來源，則每個租戶政策存放區可能最有效。您可以使用將每個租戶許可授予每個政策存放區的 IAM 政策來控制對租戶政策的存取。離職租戶涉及刪除其政策存放區；在shared-policy-store環境中，您必須尋找和刪除租戶特定的政策。

Amazon Verified Permissions 政策存放區

政策存放區是政策和政策範本的容器。在每個政策存放區中，您可以建立用於驗證新增至政策存放區的政策的結構描述。此外，您可以開啟政策驗證。如果您將政策新增至已啟用政策驗證的政策存放區，則會針對結構描述驗證政策中定義的實體類型、常見類型和動作，並拒絕無效的政策。

刪除保護可防止意外刪除政策存放區。透過 建立的所有新政策存放區都會啟用刪除保護 AWS Management Console。相反地，它會針對透過 API 或 SDK 呼叫建立的所有政策存放區停用。

我們建議為每個應用程式建立一個政策存放區，或針對多租用戶應用程式為每個租用戶建立一個政策存放區。提出授權請求時，您必須指定政策存放區。

我們建議您將命名空間用於政策存放區中的 Cedar 實體，以防止模棱兩可的情況。命名空間是 類型的字串字首，以一對冒號 (`::`) 分隔為分隔符號。例如 `MyApplicationNamespace::exampleType`。Verified Permissions 支援每個政策存放區一個命名空間。當您使用多個類似的應用程式時，這些命名空間有助於讓物件保持筆直。例如，在多租用戶應用程式中，使用命名空間將租用戶的名稱附加到結構描述中定義的類型，將使它們與其他租用戶使用的類似對應項目不同。查看授權請求的日誌時，您可以輕鬆識別處理授權請求的租戶。如需詳細資訊，請參閱《 Cedar 政策語言參考指南》中的命名空間。

主題

- 建立已驗證許可政策存放區
- API 連結政策存放區
- 刪除政策存放區

建立已驗證許可政策存放區

您可以使用下列方法建立政策存放區：

- 遵循引導式設定 – 建立第一個政策之前，您將定義具有有效動作和主體類型的資源類型。
- 使用 API Gateway 和身分來源設定 – 使用身分提供者 (IdP) 登入的使用者，以及 Amazon API Gateway API 的動作和資源實體，來定義您的主體實體。如果您希望應用程式授權具有使用者群組成員資格或其他屬性的 API 請求，建議您使用此選項。
- 從範例政策存放區開始 – 選擇預先定義的範例專案政策存放區。如果您了解 Verified Permissions 並想要檢視和測試範例政策，建議您使用此選項。

- 建立空的政策存放區 – 您將自行定義結構描述和所有存取政策。如果您已經熟悉設定政策存放區，建議您使用此選項。

Guided setup

使用引導式設定組態方法建立政策存放區

引導式設定精靈會引導您完成建立政策存放區的第一次反覆運算的程序。您將為第一個資源類型建立結構描述、描述適用於該資源類型的動作，以及您授予許可的委託人類型。然後，您將建立您的第一個政策。完成此精靈後，您就可以將 新增至您的政策存放區、擴展結構描述來描述其他資源和主體類型，以及建立其他政策和範本。

1. 在驗證許可主控台中，選取建立新政策存放區。
2. 在開始選項區段中，選擇引導式設定。
3. 輸入政策存放區描述。此文字可以是適合您組織的任何內容，做為目前政策存放區的 函數的易用參考，例如 Weather Update Web 應用程式。
4. 在詳細資訊區段中，輸入結構描述的命名空間。如需命名空間的詳細資訊，請參閱命名空間定義。
5. 選擇 Next (下一步)。
6. 在資源類型視窗中，輸入資源類型的名稱。例如， `currentTemperature` 可以是天氣更新 Web 應用程式的資源。
7. （選用）選擇新增屬性以新增資源屬性。輸入屬性名稱，並為資源的每個屬性選擇屬性類型。選擇是否需要每個屬性。例如， `temperatureFormat`可以是 `currentTemperature` 資源的屬性，可以是華氏或攝氏。若要移除為資源類型新增的屬性，請選擇屬性旁的移除。
8. 在動作欄位中，輸入要授權給指定資源類型的動作。若要為資源類型新增其他動作，請選擇新增動作。例如， `viewTemperature` 可能是天氣更新 Web 應用程式中的動作。若要移除為資源類型新增的動作，請選擇動作旁的移除。
9. 在主體類型名稱欄位中，輸入將針對資源類型使用指定動作的主體類型名稱。根據預設，使用者會新增至此欄位，但可以取代。
10. 選擇 Next (下一步)。
11. 在主體類型視窗中，選擇主體類型的身分來源。
    - 如果您的 Verified Permissions 應用程式將直接提供委託人的 ID 和屬性，請選擇自訂。選擇新增屬性以新增主體屬性。已驗證的許可在針對結構描述驗證政策時使用指定的屬性值。若要移除已針對委託人類型新增的屬性，請選擇屬性旁的移除。

- 如果委託人的 ID 和屬性會從 Amazon Cognito Cognito 產生的 ID 或存取字符提供，請選擇 Cognito 使用者集區。選擇 Connect 使用者集區。選取 AWS 區域，然後輸入要連線之 Amazon Cognito 使用者集區的使用者集區 ID。選擇連線。如需詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的使用 Amazon Verified Permissions 授權。 Amazon Cognito
- 如果委託人的 ID 和屬性將從外部 OIDC 提供者產生的 ID 和/或存取字符中擷取，請選擇外部 OIDC 提供者，並新增提供者和字符詳細資訊。

12. 選擇 Next (下一步)。
13. 在政策詳細資訊區段中，輸入第一個 Cedar 政策的選用政策描述。
14. 在主體範圍欄位中，選擇將從政策授予許可的主體。
    - 選擇特定委託人，將政策套用至特定委託人。在委託人中選擇允許採取動作的委託人欄位，然後輸入委託人的實體識別碼。例如， `user-id` 可以是天氣更新 Web 應用程式中的實體識別符。

      > **Note**
      >
      > 如果您使用的是 Amazon Cognito，實體識別符必須格式化為 `<userpool-id>|<sub>`。

    - 選擇所有主體，將政策套用至政策存放區中的所有主體。
15. 在資源範圍欄位中，選擇指定委託人將有權執行的資源。
    - 選擇特定資源，將政策套用至特定資源。在資源中選擇此政策應套用至 欄位的資源，並輸入資源的實體識別符。例如， `temperature-id` 可以是天氣更新 Web 應用程式中的實體識別符。
    - 選擇所有資源，將政策套用至政策存放區中的所有資源。
16. 在動作範圍欄位中，選擇指定主體將獲授權執行的動作。
    - 選擇特定動作集，將政策套用至特定動作。選取 Action(s) 此政策應套用至 欄位的動作旁的核取方塊。
    - 選擇所有動作，將政策套用至政策存放區中的所有動作。
17. 在政策預覽區段中檢閱政策。選擇建立政策存放區。

Set up with API Gateway and an identity source

使用設定 API Gateway 和身分來源組態方法建立政策存放區

API Gateway 選項會使用 Verified Permissions 政策來保護 APIs，這些政策旨在從使用者群組或角色進行授權決策。此選項會建置政策存放區，以使用身分來源群組和具有 Lambda 授權方的 API 來測試授權。

IdP 中的使用者及其群組會成為您的主體 (ID 字符） 或您的內容 （ 存取字符 ）。API Gateway API 中的方法和路徑會成為政策授權的動作。您的應用程式會成為 資源。由於此工作流程，驗證許可會建立政策存放區、Lambda 函數和 API Lambda 授權方。完成此工作流程後，您必須將 Lambda 授權方指派給您的 API。

1. 在驗證許可主控台中，選取建立新政策存放區。
2. 在開始選項區段中，選擇使用 API Gateway 和身分來源設定，然後選取下一步。
3. 在匯入資源和動作步驟的 API 下，選擇將做為政策存放區資源和動作模型的 API。
   a. 從 API 中設定的階段中選擇部署階段，然後選取匯入 API。如需 API 階段的詳細資訊，請參閱《Amazon API Gateway 開發人員指南》中的設定 REST API 的階段。
   b. 預覽匯入資源和動作的映射。
   c. 若要更新資源或動作，請在 API Gateway 主控台中修改您的 API 路徑或方法，然後選取匯入 API 以查看更新。
   d. 當您滿意您的選擇時，請選擇下一步。
4. 在身分來源中，選擇身分提供者類型。您可以選擇 Amazon Cognito 使用者集區或 OpenID Connect (OIDC) IdP 類型。
5. 如果您選擇 Amazon Cognito：
   a. 選擇與您政策存放區相同 AWS 區域 和 AWS 帳戶 的使用者集區。
   b. 選擇權杖類型以傳遞至您要提交以進行授權的 API。任一種字符類型都包含使用者群組，這是此 API 連結授權模型的基礎。
   c. 在應用程式用戶端驗證下，您可以將政策存放區的範圍限制為多租戶使用者集區中 Amazon Cognito 應用程式用戶端的子集。若要要求該使用者使用使用者集區中的一或多個指定應用程式用戶端進行身分驗證，請選取僅接受具有預期應用程式用戶端 IDs字符。若要接受使用使用者集區進行身分驗證的任何使用者，請選取不驗證應用程式用戶端 IDs。
   d. 選擇 Next (下一步)。
6. 如果您選擇外部 OIDC 提供者：

a. 在發行者 URL 中，輸入 OIDC 發行者的 URL。這是提供服務端點，可提供授權伺服器、簽署金鑰，以及有關提供者的其他資訊，例如 `https://auth.example.com`。您的發行者 URL 必須在 託管 OIDC 探索文件`/.well-known/openid-configuration`。

b. 在權杖類型中，選擇您希望應用程式提交以進行授權的 OIDC JWT 類型。如需詳細資訊，請參閱將身分提供者字符映射至結構描述。

c. （選用）在權杖宣告中 - 選用，選擇新增權杖宣告，輸入權杖的名稱，然後選擇值類型。

d. 在使用者和群組字符宣告中，執行下列動作：

i. 在身分來源的字符中輸入使用者宣告名稱。這是來自您的 ID 或存取權杖`sub`的宣告，該憑證會保留要評估之實體的唯一識別符。來自已連線 OIDC IdP 的身分會映射至政策存放區中的使用者類型。

ii. 在身分來源的字符中輸入群組宣告名稱。這是您的 ID 或存取權杖`groups`中通常包含使用者群組清單的宣告。您的政策存放區會根據群組成員資格來授權請求。

e. 在對象驗證中，選擇`Add value`並新增您希望政策存放區在授權請求中接受的值。

f. 選擇 Next (下一步)。

7. 如果您選擇 Amazon Cognito，驗證許可會查詢您的使用者集區是否有群組。對於 OIDC 供應商，手動輸入群組名稱。將動作指派給群組步驟會為您的政策存放區建立政策，以允許群組成員執行動作。

a. 選擇或新增您要包含在政策中的群組。

b. 將動作指派給您選取的每個群組。

c. 選擇 Next (下一步)。

8. 在部署應用程式整合中，選擇是否要稍後手動連接 Lambda 授權方，還是希望 Verified Permissions 現在為您執行，並檢閱 Verified Permissions 建立政策存放區和 Lambda 授權方時將採取的步驟。

9. 當您準備好建立新資源時，請選擇建立政策存放區。

10. 在瀏覽器中保持政策存放區狀態步驟開啟，以監控 Verified Permissions 建立資源的進度。

11. 一段時間後，通常大約一個小時，或者當部署 Lambda 授權方步驟顯示成功時，如果您選擇手動連接授權方，請設定授權方。

已驗證的許可將在您的 API 中建立 Lambda 函數和 Lambda 授權方。選擇開啟 API 以導覽至您的 API。

若要了解如何指派 Lambda 授權方，請參閱《Amazon API Gateway API Gateway 開發人員指南》中的使用 API Gateway Lambda 授權方。

a. 導覽至 API 的授權方，並記下驗證許可建立的授權方名稱。

b. 導覽至 資源，並在 API 中選取最上層方法。

c. 選取方法請求設定下的編輯。

d. 將授權方設定為您先前記下的授權方名稱。

e. 展開 HTTP 請求標頭，輸入名稱或 AUTHORIZATION，然後選取必要。

f. 部署 API 階段。

g. 儲存您的變更。

12. 使用您在選擇身分來源步驟中選取的字符類型的使用者集區字符來測試您的授權方。如需使用者集區登入和擷取權杖的詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的使用者集區身分驗證流程。
13. 在請求 API 的AUTHORIZATION標頭中使用使用者集區字符再次測試身分驗證。
14. 檢查新的政策存放區。新增和精簡政策。

Sample policy store

使用範例政策存放區組態方法建立政策存放區

1. 在開始選項區段中，選擇範例政策存放區。
2. 在範例專案區段中，選擇要使用的範例已驗證許可應用程式類型。

   - PhotoFlash 是面向客戶的 Web 應用程式範例，可讓使用者與好友共用個別相片和相簿。使用者可以針對允許檢視、評論和重新共用其相片的人員設定精細的許可。帳戶擁有者也可以建立朋友群組，並將相片整理成相簿。
   - DigitalPetStore 是範例應用程式，任何人都可以註冊並成為客戶。客戶可以新增寵物以供銷售、搜尋寵物和下訂單。新增寵物的客戶會記錄為寵物擁有者。寵物擁有者可以更新寵物的詳細資訊、上傳寵物影像或刪除寵物清單。已下訂單的客戶會記錄為訂單擁有者。訂單擁有者可以取得訂單的詳細資訊或取消訂單。寵物商店管理員具有管理存取權。

> **Note**
>
> DigitalPetStore 範例政策存放區不包含政策範本。PhotoFlash 和 TinyTodo 範例政策存放區包含政策範本。

- TinyTodo 是一種範例應用程式，可讓使用者建立任務和任務清單。清單擁有者可以管理和共用其清單，並指定誰可以檢視或編輯其清單。

3. 根據您選擇的範例專案，系統會自動產生範例政策存放區的結構描述命名空間。
4. 選擇建立政策存放區。

   您的政策存放區會建立您所選範例政策存放區的政策和結構描述。如需您可以為範例政策存放區建立的範本連結政策的詳細資訊，請參閱 Amazon Verified Permissions 範本連結政策範例。

Empty policy store

使用空白政策存放區組態方法建立政策存放區

1. 在開始選項區段中，選擇空白政策存放區。
2. 選擇建立政策存放區。

建立空白政策存放區時沒有結構描述，這表示政策不會驗證。如需更新政策存放區結構描述的詳細資訊，請參閱Amazon Verified Permissions 政策存放區結構描述。

如需為政策存放區建立政策的詳細資訊，請參閱 建立 Amazon Verified Permissions 靜態政策和 建立 Amazon Verified Permissions 範本連結政策。

AWS CLI

使用 建立空的政策存放區 AWS CLI。

您可以使用 `create-policy-store`操作來建立政策存放區。

> **Note**
>
> 您使用 建立的政策存放區 AWS CLI 是空的。
>
> - 若要新增結構描述，請參閱 Amazon Verified Permissions 政策存放區結構描述。
> - 若要新增政策，請參閱 建立 Amazon Verified Permissions 靜態政策。

- 若要新增政策範本，請參閱 建立 Amazon Verified Permissions 政策範本。

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

AWS SDKs

您可以使用 CreatePolicyStore API 建立政策存放區。如需詳細資訊，請參閱《Amazon Verified Permissions API 參考指南》中的 CreatePolicyStore。

使用 SDK 在 AWS Rust 中實作 Amazon Verified Permissions

本主題提供使用 AWS SDK 在 Rust 中實作 Amazon Verified Permissions 的實際範例。此範例示範如何開發授權模型，以測試使用者是否可以檢視相片。範例程式碼使用 SDK AWS for Rust 中的 aws-sdk-verifiedpermissions 木箱，該木箱提供一組強大的工具來與 AWS 服務互動。

先決條件

開始之前，請確定您已在系統上設定 AWS CLI，而且您已熟悉 Rust。

- 如需安裝 的指示 AWS CLI，請參閱 AWS CLI 安裝指南。
- 如需設定 的指示 AWS CLI，請參閱 中的設定 和 AWS CLI組態和登入資料檔案設定。 AWS CLI
- 如需 Rust 的詳細資訊，請參閱 rust-lang.org：// 和AWS 適用於 Rust 的 SDK 開發人員指南。

準備好您的環境後，讓我們來探索如何在 Rust 中實作 Verified Permissions。

測試範例程式碼

範例程式碼會執行下列動作：

- 設定要與 通訊的 SDK 用戶端 AWS

- 建立政策存放區
- 透過新增結構描述來定義政策存放區的結構
- 新增政策以檢查授權請求
- 傳送測試授權請求，以確認一切設定正確

測試範本程式碼

1. 建立 Rust 專案。
2. `main.rs` 使用下列程式碼取代 中的任何現有程式碼：

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

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

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

```
}

//Function that creates a policy in the policy store in the client
async fn create_policy(client: &Client, ps_id: &str,
 policy_definition:&PolicyDefinition) -> CreatePolicyOutput {
    let create_policy =
 client.create_policy().definition(policy_definition.clone()).policy_store_id(ps_id).send()
    return create_policy.unwrap();
}

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

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

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

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

//Add schema to policy store
    let schema= r#"{
        "PhotoFlash": {
            "actions": {
                "ViewPhoto": {
                    "appliesTo": {
                        "context": {
```

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

//Create policy
    let policy_text = r#"
```

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

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

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

3. 在終端機`cargo run`中輸入 以執行程式碼。

如果程式碼正確執行，終端機會顯示 ，`Decision: Allow`後面接著決定政策的政策 ID。這表示您已成功建立政策存放區，並使用適用於 Rust 的 AWS SDK 進行測試。

清除資源

完成探索政策存放區後，請將其刪除。

刪除政策存放區

您可以使用 `delete-policy-store`操作來刪除政策存放區，*PSEXAMPLEabcdefg111111*將 取代為要刪除的政策存放區 ID。

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

如果成功，此命令不會產生輸出。

API 連結政策存放區

常見的使用案例是使用 Amazon Verified Permissions 來授權使用者存取 Amazon API Gateway 上託管 APIs API。 Amazon API Gateway 使用 AWS 主控台中的精靈，您可以為在 Amazon Cognito 或任何 OIDC 身分提供者 (IdP) 中管理的使用者建立角色型存取政策，並部署 AWS Lambda 呼叫已驗證許可來評估這些政策的授權方。

若要完成精靈，請在建立新政策存放區時，選擇使用 API Gateway 和身分提供者設定，然後依照步驟進行。

API 連結的政策存放區已建立，並佈建授權請求的授權模型和資源。政策存放區具有身分來源和 Lambda 授權方，可將 API Gateway 連線至已驗證的許可。政策存放區建立後，您可以根據使用者的群組成員資格來授權 API 請求。例如，已驗證許可只能將存取權授予群組成員的使用者`Directors`。

隨著應用程式的成長，您可以使用 Cedar 政策語言，使用使用者屬性和 OAuth 2.0 範圍實作精細的授權。例如，已驗證許可只能將存取權授予網域 中具有 `email` 屬性的使用者`mycompany.co.uk`。

設定 API 的授權模型後，您剩餘的責任是驗證使用者，並在應用程式中產生 API 請求，以及維護您的政策存放區。

若要查看示範，請參閱 Amazon Web Services YouTube 頻道上的 Amazon Verified Permissions - Quick Start Overview 和示範。

主題

- Verified Permissions 如何授權 API 請求

- API 連結政策存放區的考量事項
- 新增屬性型存取控制 (ABAC)
- 使用 移至生產環境 AWS CloudFormation
- 對 API 連結政策存放區進行故障診斷

> **Important**
>
> 您在 Verified Permissions 主控台中使用 API Gateway 設定和身分來源選項建立的政策存放區不適用於立即部署到生產環境。使用初始政策存放區，完成授權模型並將政策存放區資源匯出至 CloudFormation。使用 AWS 雲端開發套件 (CDK)，以程式設計方式將已驗證的許可部署至生產環境。如需詳細資訊，請參閱使用 移至生產環境 AWS CloudFormation。

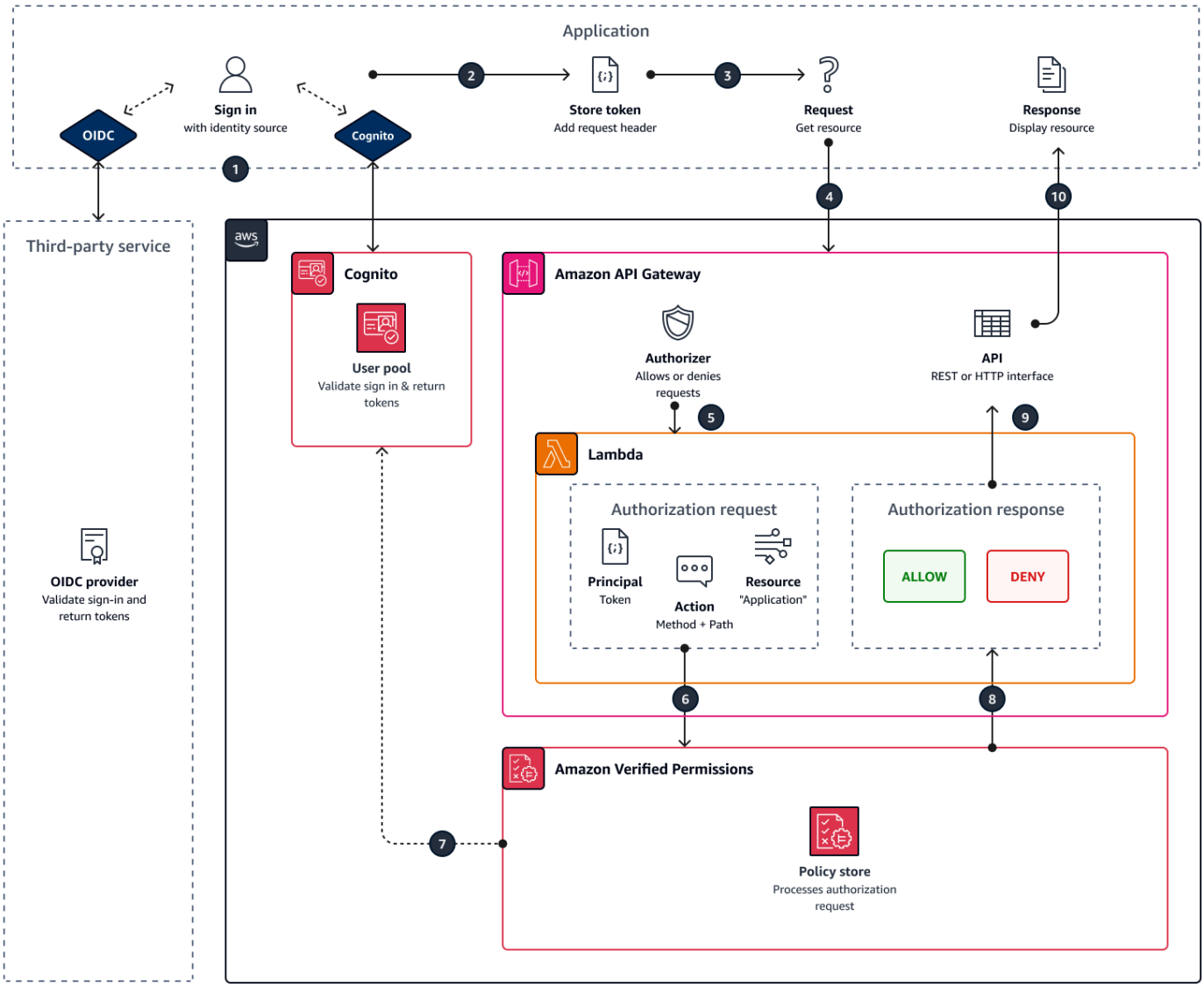
在連結至 API 和身分來源的政策存放區中，您的應用程式會在向 API 提出請求時，在授權標頭中顯示使用者集區權杖。政策存放區的身分來源提供驗證許可的字符驗證。權杖會使用 IsAuthorizedWithToken API 在授權請求`principal`中形成 。Verified Permissions 會建置您使用者群組成員資格的政策，如身分 (ID) 和存取權杖中的群組宣告所示，例如`cognito:groups`使用者集區。您的 API 會在 Lambda 授權方中處理您應用程式的權杖，並將其提交給驗證許可以進行授權決策。當您的 API 從 Lambda 授權方收到授權決策時，它會將請求傳遞到您的資料來源或拒絕請求。

身分來源和 API Gateway 授權與驗證許可的元件

- 驗證和分組使用者的 Amazon Cognito 使用者集區或 OIDC IdP。使用者的字符會填入群組成員資格，以及 Verified Permissions 在政策存放區中評估的主體或內容。
- API Gateway REST API。Verified Permissions 會從 API 路徑和 API 方法定義動作，例如 `MyAPI::Action::get /photo`。
- 適用於 API 的 Lambda 函數和 Lambda 授權方。Lambda 函數會從使用者集區取得承載符記、請求驗證許可的授權，並將決策傳回給 API Gateway。使用 API Gateway 和身分來源工作流程設定 會自動為您建立此 Lambda 授權方。
- Verified Permissions 政策存放區。政策存放區身分來源是您的 Amazon Cognito 使用者集區或 OIDC 提供者群組。政策存放區結構描述會反映 API 的組態，而政策會將使用者群組連結至允許的 API 動作。
- 使用 IdP 驗證使用者並將字符附加至 API 請求的應用程式。

Verified Permissions 如何授權 API 請求

當您建立新的政策存放區，並選取使用 API Gateway 和身分來源設定選項時，驗證許可會建立政策存放區結構描述和政策。結構描述和政策反映了 API 動作，以及您想要授權 採取動作的使用者群組。已驗證的許可也會建立 Lambda 函數和授權方。



1. 您的使用者透過 Amazon Cognito 或其他 OIDC IdP 使用您的應用程式登入。IdP 會發出 ID 和存取權杖與使用者的資訊。
2. 您的應用程式會存放 JWTs。如需詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的搭配使用者集區使用權杖。
3. 您的使用者請求您的應用程式必須從外部 API 擷取的資料。

4. 您的應用程式會從 API Gateway 中的 REST API 請求資料。它會附加 ID 或存取權杖做為請求標頭。
5. 如果您的 API 具有授權決策的快取，則會傳回先前的回應。如果停用快取或 API 目前沒有快取，API Gateway 會將請求參數傳遞給以字符為基礎的 Lambda 授權方。
6. Lambda 函數會使用 IsAuthorizedWithToken API 將授權請求傳送至 Verified Permissions 政策存放區。Lambda 函數會傳遞授權決策的元素：
   a. 使用者的權杖做為委託人。
   b. API 方法與 API 路徑結合，例如 `GetPhoto`作為 動作。
   c. `Application` 做為 資源的 字詞。
7. 已驗證的許可會驗證權杖。如需如何驗證 Amazon Cognito 權杖的詳細資訊，請參閱《Amazon Amazon Cognito授權 Amazon 驗證許可。
8. Verified Permissions 會根據政策存放區中的政策評估授權請求，並傳回授權決策。
9. Lambda 授權方會將 `Allow`或 `Deny`回應傳回 API Gateway。
10. API 會傳回資料或回應`ACCESS_DENIED`給您的應用程式。您的應用程式會處理並顯示 API 請求的結果。

API 連結政策存放區的考量事項

當您在 Verified Permissions 主控台中建置 API 連結政策存放區時，您正在為最終生產部署建立測試。移至生產環境之前，請為您的 API 和使用者集區建立固定組態。請考慮下列因素：

API Gateway 快取回應

在 API 連結政策存放區中，驗證許可會建立授權快取 TTL 為 120 秒的 Lambda 授權方。您可以調整此值或關閉授權方中的快取。在啟用快取的 授權方中，您的授權方每次都會傳回相同的回應，直到 TTL 過期為止。這可以將使用者集區權杖的有效生命週期延長至等於所請求階段快取 TTL 的持續時間。

Amazon Cognito 群組可以重複使用

Amazon Verified Permissions 會從使用者的 ID 或存取權杖中的`cognito:groups`宣告，判斷使用者集區使用者的群組成員資格。此宣告的值是使用者所屬之使用者集區群組的易記名稱陣列。您無法將使用者集區群組與唯一識別符建立關聯。

您刪除的使用者群組，並以與相同群組相同的名稱重新建立到您的政策存放區。當您從使用者集區刪除群組時，請從您的政策存放區刪除群組的所有參考。

API 衍生命名空間和結構描述是point-in-time

驗證許可會在某個時間點擷取您的 API：它只會在您建立政策存放區時查詢您的 API。當 API 的結構描述或名稱變更時，您必須更新政策存放區和 Lambda 授權方，或建立新的 API 連結政策存放區。Verified Permissions 會從 API 的名稱衍生政策存放區命名空間。

Lambda 函數沒有 VPC 組態

Verified Permissions 為 API 授權方建立的 Lambda 函數會在預設 VPC 中啟動。根據預設。限制網路存取私有 VPCs APIs 無法與 Lambda 函數通訊，該函數授權使用 Verified Permissions 存取請求。

驗證許可在 CloudFormation 中部署授權方資源

若要建立 API 連結政策存放區，您必須登入 Verified Permissions 主控台的高度權限 AWS 主體。此使用者會部署 AWS CloudFormation 堆疊，以跨數個 建立資源 AWS 服務。此主體必須具有在 Verified Permissions IAM、Lambda 和 API Gateway 中新增和修改資源的許可。最佳實務是，請勿與組織中的其他管理員共用這些登入資料。

如需 Verified Permissions 建立之資源的概觀，使用 移至生產環境 AWS CloudFormation請參閱 。

新增屬性型存取控制 (ABAC)

具有 IdP 的典型身分驗證工作階段會傳回 ID 和存取權杖。您可以在應用程式請求中將這些字符類型做為承載字符傳遞至您的 API。根據您的選擇，當您建立政策存放區時，已驗證許可會預期兩種字符類型之一。這兩種類型都包含有關使用者群組成員資格的資訊。如需 Amazon Cognito 中字符類型的詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的搭配使用者集區使用字符。

建立政策存放區之後，您可以新增和延伸政策。例如，您可以在將新群組新增至使用者集區時，將新群組新增至政策。由於您的政策存放區已了解您的使用者集區在字符中呈現群組的方式，因此您可以允許具有新政策的任何新群組執行一組動作。

您可能也想要根據使用者屬性，將政策評估的群組型模型擴展為更精確的模型。使用者集區字符包含其他使用者資訊，有助於授權決策。

ID 字符

ID 字符代表使用者的屬性，並具有高度精細的存取控制。若要評估電子郵件地址、電話號碼或自訂屬性，例如部門和經理，請評估 ID 字符。

存取權杖

存取權杖代表使用者具有 OAuth 2.0 範圍的許可。若要新增授權層或設定其他資源的請求，請評估存取權杖。例如，您可以驗證使用者是否位於適當的群組中，並攜帶一個像`PetStore.read`一般授權存取 API 的範圍。使用者集區可以新增自訂範圍到具有資源伺服器的字符，以及在執行時間使用字符自訂。

如需在 ID 和存取權杖中處理宣告的範例政策將身分提供者字符映射至結構描述，請參閱 。

使用 移至生產環境 AWS CloudFormation

API 連結政策存放區是快速建置 API Gateway API 授權模型的一種方式。它們旨在做為您應用程式授權元件的測試環境。建立測試政策存放區之後，請花時間完善政策、結構描述和 Lambda 授權方。

您可以調整 API 的架構，需要對政策存放區結構描述和政策進行同等調整。API 連結政策存放區不會從 API 架構自動更新其結構描述 – 驗證許可只會在您建立政策存放區時輪詢 API。如果您的 API 有足夠變更，您可能需要使用新的政策存放區重複此程序。

當您的應用程式和授權模型準備好部署到生產環境時，請整合您開發的 API 連結政策存放區與自動化程序。最佳實務是，建議您將政策存放區結構描述和政策匯出到您可以部署到其他 AWS 帳戶 和 的 AWS CloudFormation 範本 AWS 區域。

API 連結政策存放區程序的結果是初始政策存放區和 Lambda 授權方。Lambda 授權方有數個相依資源。驗證許可會在自動產生的 CloudFormation 堆疊中部署這些資源。若要部署到生產環境，您必須將政策存放區和 Lambda 授權方資源收集到範本中。API 連結政策存放區是由下列資源組成：

1. AWS::VerifiedPermissions::PolicyStore： 將您的結構描述複製到 `SchemaDefinition` 物件。逸出"字元為 \"。
2. AWS::VerifiedPermissions::IdentitySource：從您的測試政策存放區複製 GetIdentitySource 輸出的值，並視需要修改。
3. 一或多個 AWS::VerifiedPermissions::Policy： 將您的政策陳述式複製到 `Definition` 物件。逸出"字元為 \"。
4. AWS：：Lambda：：Function、AWS：IAM：：Role、AWS：IAM：Policy、AWS：：ApiGateway：：Authorizer、AWS::Lambda::Permission

下列範本是範例政策存放區。您可以將 Lambda 授權方資源從現有堆疊附加至此範本。

```
{
```

```
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "MyExamplePolicyStore": {
            "Type": "AWS::VerifiedPermissions::PolicyStore",
            "Properties": {
                "ValidationSettings": {
                    "Mode": "STRICT"
                },
                "Description": "ApiGateway: PetStore/test",
                "Schema": {
                    "CedarJson": "{\"PetStore\":{\"actions\":{\"get /pets\":
{\"appliesTo\":{\"principalTypes\":[\"User\"],\"resourceTypes\":[\"Application\"],
\"context\":{\"type\":\"Record\",\"attributes\":{}}}},\"get /\":{\"appliesTo\":
{\"principalTypes\":[\"User\"],\"resourceTypes\":[\"Application\"],\"context\":{\"type
\":\"Record\",\"attributes\":{}}}},\"get /pets/{petId}\":{\"appliesTo\":{\"context
\":{\"type\":\"Record\",\"attributes\":{}},\"resourceTypes\":[\"Application\"],
\"principalTypes\":[\"User\"]}},\"post /pets\":{\"appliesTo\":{\"principalTypes\":
[\"User\"],\"resourceTypes\":[\"Application\"],\"context\":{\"type\":\"Record\",
\"attributes\":{}}}}},\"entityTypes\":{\"Application\":{\"shape\":{\"type\":\"Record\",
\"attributes\":{}}},\"User\":{\"memberOfTypes\":[\"UserGroup\"],\"shape\":{\"attributes
\":{},\"type\":\"Record\"}},\"UserGroup\":{\"shape\":{\"type\":\"Record\",\"attributes
\":{}}}}}}"
                }
            }
        },
        "MyExamplePolicy": {
            "Type": "AWS::VerifiedPermissions::Policy",
            "Properties": {
                "Definition": {
                    "Static": {
                        "Description": "Policy defining permissions for testgroup
 cognito group",
                        "Statement": "permit(\nprincipal in PetStore::UserGroup::
\"us-east-1_EXAMPLE|testgroup\",\naction in [\n  PetStore::Action::\"get /\",
\n  PetStore::Action::\"post /pets\",\n  PetStore::Action::\"get /pets\",\n
 PetStore::Action::\"get /pets/{petId}\"\n],\nresource);"
                    }
                },
                "PolicyStoreId": {
                    "Ref": "MyExamplePolicyStore"
                }
            },
            "DependsOn": [
                "MyExamplePolicyStore"
```

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

對 API 連結政策存放區進行故障診斷

當您建置 Amazon Verified Permissions API 連結政策存放區時，請使用此處的資訊來協助您診斷和修正常見問題。

主題

- 我更新了我的政策，但授權決策未變更
- 我將 Lambda 授權方連接到我的 API，但它不會產生授權請求
- 我收到非預期的授權決定，並想要檢閱授權邏輯
- 我想要從我的 Lambda 授權方尋找日誌
- 我的 Lambda 授權方不存在

- 我的 API 位於私有 VPC 中，無法叫用授權方
- 我想要在授權模型中處理其他使用者屬性
- 我想要新增動作、動作內容屬性或資源屬性

我更新了我的政策，但授權決策未變更

根據預設，驗證許可會設定 Lambda 授權方快取授權決策 120 秒。請在兩分鐘後再試一次，或停用授權方上的快取。如需詳細資訊，請參閱《Amazon API Gateway API Gateway 開發人員指南》中的啟用 API 快取以增強回應能力。

我將 Lambda 授權方連接到我的 API，但它不會產生授權請求

若要開始處理請求，您必須部署您連接授權方的 API 階段。如需詳細資訊，請參閱《Amazon API Gateway 開發人員指南》中的部署 REST API。 Amazon API Gateway

我收到非預期的授權決定，並想要檢閱授權邏輯

API 連結政策存放區程序會為您的授權方建立 Lambda 函數。Verified Permissions 會自動將授權決策的邏輯建置到授權方函數中。您可以在建立政策存放區後返回，以檢閱和更新函數中的邏輯。

若要從 AWS CloudFormation 主控台尋找 Lambda 函數，請選擇新政策存放區概觀頁面上的檢查部署按鈕。

您也可以在 AWS Lambda 主控台中找到您的 函數。導覽至政策存放區 AWS 區域 中的 主控台，並搜尋字首為 的函數名稱`AVPAuthorizerLambda`。如果您已建立多個 API 連結政策存放區，請使用函數的上次修改時間來建立它們與政策存放區建立的關聯。

我想要從我的 Lambda 授權方尋找日誌

Lambda 函數會收集指標，並在 Amazon CloudWatch 中記錄其調用結果。若要檢閱您的日誌，請在 Lambda 主控台中尋找您的函數，然後選擇監控索引標籤。選取檢視 CloudWatch 日誌，並檢閱日誌群組中的項目。

如需 Lambda 函數日誌的詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的搭配 使用 Amazon CloudWatch Logs AWS Lambda。

我的 Lambda 授權方不存在

完成 API 連結政策存放區的設定後，您必須將 Lambda 授權方連接至您的 API。如果您在 API Gateway 主控台中找不到授權方，則政策存放區的其他資源可能已失敗或尚未部署。API 連結政策會將這些資源部署在 AWS CloudFormation 堆疊中。

已驗證許可會在建立程序結束時顯示具有標籤的連結 檢查部署。如果您已離開此畫面，請前往 CloudFormation 主控台，並搜尋最近堆疊中以 開頭的名稱`AVPAuthorizer-<policy store ID>`。CloudFormation 在堆疊部署的輸出中提供寶貴的疑難排解資訊。

如需對 CloudFormation 堆疊進行故障診斷的說明，請參閱AWS CloudFormation 《 使用者指南》中的對 CloudFormation 進行故障診斷。

我的 API 位於私有 VPC 中，無法叫用授權方

驗證許可不支援透過 VPC 端點存取 Lambda 授權方。您必須在 API 和做為授權方的 Lambda 函數之間開啟網路路徑。

我想要在授權模型中處理其他使用者屬性

API 連結政策存放區程序會從使用者字符中的群組宣告衍生驗證許可政策。若要更新您的授權模型以考慮其他使用者屬性，請將這些屬性整合到您的政策中。

您可以將 Amazon Cognito 使用者集區的許多 ID 和存取權杖宣告對應至 Verified Permissions 政策陳述式。例如，大多數使用者在其 ID 字符中都有`email`宣告。如需有關將宣告從您的身分來源新增至政策的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

我想要新增動作、動作內容屬性或資源屬性

API 連結政策存放區及其建立的 Lambda 授權方是point-in-time資源。它們會在建立時反映您的 API 狀態。政策存放區結構描述不會將任何內容屬性指派給動作，也不會將任何屬性或父項指派給預設`Application`資源。

當您將動作 — 路徑和方法 — 新增至您的 API 時，您必須更新政策存放區，才能了解新的動作。您也必須更新您的 Lambda 授權方，以處理新動作的授權請求。您可以重新開始新的政策存放區，也可以更新現有的政策存放區。

若要更新現有的政策存放區，請尋找您的 函數。檢查自動產生的函數中的邏輯，並更新它以處理新的動作、屬性或內容。然後，編輯您的結構描述以包含新的動作和屬性。

刪除政策存放區

您可以使用 AWS Management Console 或 刪除 Amazon Verified Permissions 政策存放區 AWS CLI。刪除政策存放區會永久刪除政策存放區中的結構描述和任何政策。

刪除保護可防止意外刪除政策存放區。透過 建立的所有新政策存放區都會啟用刪除保護 AWS Management Console。相反地，它會針對透過 API 或 SDK 呼叫建立的所有政策存放區停用。

您可能因為下列原因而想要刪除政策存放區：

- 您已達到指定區域中可用政策存放區的配額。如需詳細資訊，請參閱資源的配額。
- 您不再支援多租戶應用程式中的租戶，因此不再需要該政策存放區。

AWS Management Console

刪除政策存放區

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側的導覽窗格中，選擇 Settings (設定)。
3. 選擇刪除此政策存放區。
4. 在文字方塊delete中輸入 ，然後選擇刪除。

> **Note**
>
> 如果啟用刪除保護，您必須先停用它，才能選擇刪除。若要停用，請選取停用刪除保護。

AWS CLI

刪除政策存放區

您可以使用 delete-policy-store操作來刪除政策存放區，*PSEXAMPLEabcdefg111111*以您要刪除的政策存放區 ID 取代 。

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

如果成功，此命令不會產生輸出。

> **Note**
>
> 如果此政策存放區已啟用刪除保護，您必須先執行 `update-policy-store`操作並停用刪除保護。

```
aws verifiedpermissions update-policy-store \
    --deletion-protection "DISABLED" \
    --policy-store-id PSEXAMPLEabcdefg111111
```

Amazon Verified Permissions 政策存放區結構描述

結構描述是應用程式所支援實體類型的結構宣告，以及應用程式在授權請求中可能提供的動作。若要查看 Verified Permissions 和 Cedar 如何處理結構描述之間的差異，請參閱 結構描述支援。

如需詳細資訊，請參閱《 Cedar 政策語言參考指南》中的 Cedar 結構描述格式。

> **Note**
>
> 在 Verified Permissions 中使用結構描述是選用的，但強烈建議用於生產軟體。當您建立新的政策時，Verified Permissions 可以使用結構描述來驗證範圍和條件中參考的實體和屬性，以避免可能導致混淆系統行為的政策錯誤。如果您啟用政策驗證，則所有新政策必須符合結構描述。

AWS Management Console

建立結構描述

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇結構描述。
3. 選擇建立結構描述。

AWS CLI

若要提交新的結構描述，或使用 覆寫現有的結構描述 AWS CLI。

您可以執行類似下列範例的 AWS CLI 命令來建立政策存放區。

請考慮包含下列 Cedar 內容的結構描述：

```
{
    "MySampleNamespace": {
        "actions": {
            "remoteAccess": {
                "appliesTo": {
                    "principalTypes": [ "Employee" ]
                }
            }
```

```
        },
        "entityTypes": {
            "Employee": {
                "shape": {
                    "type": "Record",
                    "attributes": {
                        "jobLevel": {"type": "Long"},
                        "name": {"type": "String"}
                    }
                }
            }
        }
    }
}
```

您必須先將 JSON 逸出至單一行字串，並在 JSON 前面加上其資料類型的宣告：`cedarJson`。下列範例使用下列`schema.json`檔案內容，其中包含 JSON 結構描述的逸出版本。

> **Note**
>
> 此處的範例為折線包裝以提供可讀性。您必須在單行中擁有整個檔案，命令才能接受它。

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

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

AWS SDKs

您可以使用 PutSchema API 建立政策存放區。如需詳細資訊，請參閱《Amazon Verified Permissions API 參考指南》中的 PutSchema。

編輯政策存放區結構描述

當您在 Amazon Verified Permissions 主控台中選取結構描述時，會顯示組成結構描述的實體類型和動作。您可以在視覺化模式或 JSON 模式中檢視編輯結構描述。視覺化模式可讓您使用各種精靈新增新類型和動作，以更新結構描述。使用 JSON 模式，您可以直接在 JSON 編輯器中開始更新結構描述的 JSON 程式碼。

Visual Mode

視覺化結構描述編輯器從一系列圖表開始，說明結構描述中實體之間的關係。選擇展開以最大化圖表的檢視。有兩個可用的圖表：

- 動作圖表 – 動作圖表檢視列出您在政策存放區中設定的委託人類型、他們有資格執行的動作，以及他們有資格執行動作的資源。實體之間的行表示您能夠建立允許委託人對資源採取動作的政策。如果您的動作圖表未指出兩個實體之間的關係，您必須在它們之間建立該關係，才能允許或拒絕政策中的關係。選取實體以查看屬性概觀，並向下切入以檢視完整詳細資訊。選擇依此 【動作 | 資源類型 | 委託人類型】 篩選，以查看檢視中僅具有其自身連線的實體。
- 實體類型圖表 – 實體類型圖表著重於委託人和資源之間的關係。當您想要了解結構描述中的複雜巢狀父系關係時，請檢閱此圖表。將滑鼠游標暫留在實體上，以深入了解其擁有的父系關係。

  在圖表下，列出結構描述中實體類型和動作的檢視。當您想要立即檢視特定動作或實體類型的詳細資訊時，清單檢視非常有用。選取任何實體以檢視詳細資訊。

在視覺化模式中編輯已驗證許可結構描述

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇結構描述。
3. 選擇視覺化模式。檢閱實體關係圖表，並規劃您要對結構描述進行的變更。您可以選擇依一個實體篩選，以檢查其與其他實體的個別連線。
4. 選擇編輯結構描述。
5. 在詳細資訊區段中，輸入結構描述的命名空間。
6. 在實體類型區段中，選擇新增實體類型。

7. 輸入實體的名稱。
8. （選用）選擇新增父實體，以新增新實體所屬的父實體。若要移除已新增至實體的父項，請選擇父項名稱旁的移除。
9. 選擇新增屬性，將屬性新增至實體。輸入屬性名稱，並為實體的每個屬性選擇屬性類型。Verified Permissions 在針對結構描述驗證政策時使用指定的屬性值。選取每個屬性是否為必要。若要移除已新增至實體的屬性，請選擇屬性旁的移除。
10. 選擇新增實體類型，將實體新增至結構描述。
11. 在動作區段中，選擇新增動作。
12. 輸入動作的名稱。
13. （選用）選擇新增資源以新增套用動作的資源類型。若要移除已新增至動作的資源類型，請選擇資源類型名稱旁的移除。
14. （選用）選擇新增主體以新增套用動作的主體類型。若要移除已新增至動作的委託人類型，請選擇委託人類型名稱旁的移除。
15. 選擇新增屬性，以新增可新增至授權請求中動作內容的屬性。輸入屬性名稱，並為每個屬性選擇屬性類型。Verified Permissions 在針對結構描述驗證政策時使用指定的屬性值。選取每個屬性是否為必要。若要移除已新增至動作的屬性，請選擇屬性旁的移除。
16. 選擇新增動作。
17. 將所有實體類型和動作新增至結構描述後，選擇儲存變更。

JSON mode

進行更新時，您會注意到 JSON 編輯器會根據 JSON 語法驗證您的程式碼，並在編輯時識別錯誤和警告，讓您更輕鬆地快速找到問題。此外，您不需要擔心 JSON 的格式，只需在進行更新後選擇格式化 JSON，格式就會更新以符合預期的 JSON 格式。

在 JSON 模式下編輯已驗證許可結構描述

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇結構描述。
3. 選擇 JSON 模式，然後選擇編輯結構描述。
4. 在內容欄位中輸入 JSON 結構描述的內容。在解決所有語法錯誤之前，您無法儲存結構描述的更新。您可以選擇格式化 JSON，以使用建議的間距和縮排來格式化結構描述的 JSON 語法。
5. 選擇儲存變更。

啟用 Amazon Verified Permissions 政策驗證模式

您可以在 Verified Permissions 中設定政策驗證模式，以控制政策變更是否針對政策存放區中的結構描述進行驗證。

> **Important**
>
> 當您開啟政策驗證時，所有建立或更新政策或政策範本的嘗試都會根據政策存放區中的結構描述進行驗證。如果驗證失敗，已驗證許可會拒絕請求嘗試。因此，我們建議您在開發應用程式時關閉驗證，並開啟它進行測試，並在應用程式處於生產狀態時將其保持開啟。

AWS Management Console

設定政策存放區的政策驗證模式

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 選擇設定。
3. 在政策驗證模式區段中，選擇修改。
4. 執行以下任意一項：
   - 若要啟用政策驗證並強制執行所有政策變更都必須針對您的結構描述進行驗證，請選擇嚴格（建議）選項按鈕。
   - 若要關閉政策變更的政策驗證，請選擇關閉選項按鈕。輸入 `confirm` 以確認政策的更新將不再針對您的結構描述進行驗證。
5. 選擇 Save changes (儲存變更)。

AWS CLI

設定政策存放區的驗證模式

您可以使用 UpdatePolicyStore 操作，並為 ValidationSettings 參數指定不同的值，來變更政策存放區的驗證模式。

```
$ aws verifiedpermissions update-policy-store \
    --validation-settings "mode=OFF",
    --policy-store-id PSEXAMPLEabcdefg111111
```

```
{
    "createdDate": "2023-05-17T18:36:10.134448+00:00",
    "lastUpdatedDate": "2023-05-17T18:36:10.134448+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "validationSettings": {
        "Mode": "OFF"
    }
}
```

如需詳細資訊，請參閱 Cedar 政策語言參考指南中的政策驗證。

Amazon Verified Permissions 政策

政策是允許或禁止委託人對資源採取一或多個動作的陳述式。每個政策都會獨立於其他政策進行評估。如需如何建構和評估 Cedar 政策的詳細資訊，請參閱《Cedar 政策語言參考指南》中的針對結構描述的 Cedar 政策驗證。

Important

當您撰寫參考委託人、資源和動作的 Cedar 政策時，您可以定義用於每個元素的唯一識別符。我們強烈建議您遵循下列最佳實務：

- 對所有主體和資源識別符使用通用的唯一識別符 (UUIDs)。

  例如，如果使用者jane離開公司，而您稍後讓其他人使用名稱 jane，則該新使用者會自動存取仍然參考 的政策授予的所有內容User::"jane"。Cedar 無法區分新使用者和舊使用者。這同時適用於主體和資源識別符。一律使用保證唯一且永遠不會重複使用的識別符，以確保您不會因為政策中存在舊識別符而意外授予存取權。

  如果您為實體使用 UUID，我們建議您使用 // 評論指標和實體的「易記」名稱來遵循它。這有助於讓您的政策更容易理解。例如：主體 == Role： ："a1b2c3d4-e5f6-a1b2-c3d4-EXAMPLE11111"， // 管理員
- 請勿在主體或資源的唯一識別符中包含個人識別、機密或敏感資訊。這些識別符包含在 AWS CloudTrail 線索中共用的日誌項目中。

主題

- 建立 Amazon Verified Permissions 靜態政策
- 編輯 Amazon Verified Permissions 靜態政策
- 新增內容
- 使用 Amazon Verified Permissions 測試工作台
- Amazon Verified Permissions 範例政策

建立 Amazon Verified Permissions 靜態政策

您可以為主體建立靜態政策，以允許或禁止他們對應用程式的指定資源執行指定的動作。靜態政策包含principal和 的特定值，resource並準備好用於授權決策。

AWS Management Console

建立靜態政策

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側的導覽窗格中，選擇 Policies (政策)。
3. 選擇建立政策，然後選擇建立靜態政策。

> **Note**
>
> 如果您有想要使用的政策陳述式，請跳到步驟 8，並將政策貼到下一頁的政策區段。

4. 在政策效果區段中，選擇當請求符合政策時，政策是否允許或禁止。如果您選擇允許，政策會允許委託人對資源執行動作。相反地，如果您選擇禁止，政策不允許委託人對資源執行動作。
5. 在主體範圍欄位中，選擇政策將套用的主體範圍。
   - 選擇特定委託人，將政策套用至特定委託人。指定將允許或禁止採取政策中指定動作之主體的實體類型和識別符。
   - 選擇委託人群組，將政策套用至一組委託人。在主體群組欄位中輸入主體群組名稱。
   - 選擇所有委託人，將政策套用到政策存放區中的所有委託人。
6. 在資源範圍欄位中，選擇政策將套用的資源範圍。
   - 選擇特定資源，將政策套用至特定資源。指定政策應套用之資源的實體類型和識別符。
   - 選擇資源群組，將政策套用至資源群組。在資源群組欄位中輸入資源群組名稱。
   - 選擇所有資源，將政策套用至政策存放區中的所有資源。
7. 在動作範圍區段中，選擇政策將套用的資源範圍。
   - 選擇特定的動作集，將政策套用至一組動作。選取要套用政策之動作旁的核取方塊。
   - 選擇所有動作，將政策套用至政策存放區中的所有動作。
8. 選擇下一步。
9. 在政策區段中，檢閱您的 Cedar 政策。您可以選擇格式化，以使用建議的間距和縮排來格式化政策的語法。如需詳細資訊，請參閱《Cedar 政策語言參考指南》中的 Cedar 中的基本政策建構。
10. 在詳細資訊區段中，輸入政策的選用描述。
11. 選擇建立政策。

AWS CLI

建立靜態政策

您可以使用 CreatePolicy 操作建立靜態政策。下列範例會建立簡單的靜態政策。

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

編輯 Amazon Verified Permissions 靜態政策

您可以在政策存放區中編輯現有的靜態政策。您只能直接更新靜態政策。若要變更範本連結政策，您必須更新政策範本。如需詳細資訊，請參閱編輯 Amazon Verified Permissions 政策範本。

您可以變更靜態政策的下列元素：

- 政策`action`參考的 。
- 條件子句，例如 `when`和 `unless`。

您無法變更靜態政策的下列元素。若要變更任何這些元素，您需要刪除並重新建立政策。

- 從靜態政策到範本連結政策的政策。
- 來自 `permit`或 的靜態政策效果`forbid`。
- 靜態政策`principal`參考的 。
- 靜態政策`resource`參考的 。

AWS Management Console

編輯靜態政策

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側的導覽窗格中，選擇 Policies (政策)。
3. 選擇要編輯的靜態政策旁的選項按鈕，然後選擇編輯。
4. 在政策內文區段中，更新靜態政策的 `action`或 條件子句。您無法更新政策的 或 `principalresource`政策效果。
5. 選擇更新政策。

> **Note**
>
> 如果在政策存放區中啟用政策驗證，則更新靜態政策會導致 Verified Permissions 根據政策存放區中的結構描述驗證政策。如果更新的靜態政策未通過驗證，操作會失敗，而且不會儲存更新。

AWS CLI

編輯靜態政策

您可以使用 UpdatePolicy 操作編輯靜態政策。下列範例會編輯簡單的靜態政策。

此範例使用 檔案`definition.txt`來包含政策定義。

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

下列命令參考該檔案。

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

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

新增內容

內容是與政策決策相關的資訊，但不屬於委託人、動作或資源的身分。存取權杖宣告是內容。您可能只想允許來自一組來源 IP 地址的動作，或只有在您的使用者已使用 MFA 登入時。您的應用程式可以存取此內容式工作階段資料，且必須將其填入授權請求。Verified Permissions 授權請求中的內容資料必須在 `contextMap`元素中採用 JSON 格式。

說明此內容的範例來自範例政策存放區。若要遵循，請在您的測試環境中建立 DigitalPetStore 範例政策存放區。

下列內容物件會根據範例 DigitalPetStore 政策存放區，宣告應用程式的其中一個 Cedar 資料類型。

```
"context": {
  "contextMap": {
    "AccountCodes": {
      "set": [
        {
          "long": 111122223333
        },
        {
          "long": 444455556666
        },
        {
          "long": 123456789012
```

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

授權內容中的資料類型

Boolean

二進位`true`或`false`值。在此範例中，`true`的布林值`MfaAuthenticated`表示客戶在請求檢視其順序之前已執行多重要素驗證。

設定

內容元素的集合。集合成員可以是所有相同的類型，如本範例所示，也可以是不同類型的類型，包括巢狀集合。在此範例中，客戶與 3 個不同的帳戶相關聯。

字串

字母、數字或符號的序列，以"字元括住。在此範例中，`UserAgent`字串代表客戶用來請求檢視訂單的瀏覽器。

Long

整數。在此範例中， `RequestedOrderCount`表示此請求是客戶要求檢視四個過去訂單所產生的批次的一部分。

記錄

屬性的集合。您必須在請求內容中宣告這些屬性。具有結構描述的政策存放區必須在結構描述中包含此實體和實體的屬性。在此範例中，`NetworkInfo`記錄包含使用者原始 IP、用戶端決定的 IP 地理位置，以及傳輸中加密的相關資訊。

EntityIdentifier

在請求的 `entities`元素中宣告的實體和屬性參考。在此範例中，使用者順序已由員工 核准Bob。

若要在 DigitalPetStore 應用程式中測試此範例內容，您必須更新請求 `entities`、您的政策存放區結構描述，以及描述客戶角色 - 取得訂單的靜態政策。

修改 DigitalPetStore 以接受授權內容

DigitalPetStore 一開始不是非常複雜的政策存放區。它不包含任何預先設定的政策或內容屬性，以支援我們呈現的內容。若要使用此內容資訊評估範例授權請求，請對政策存放區和授權請求進行下列修改。如需使用存取權杖資訊做為內容的內容範例，請參閱 映射存取權杖。

Schema

將下列更新套用至您的政策存放區結構描述，以支援新的內容屬性。在 `GetOrder`中`actions`更新，如下所示。

```
"GetOrder": {
  "memberOf": [],
  "appliesTo": {
    "resourceTypes": [
      "Order"
```

```
    ],
    "context": {
      "type": "Record",
      "attributes": {
        "AccountCodes": {
          "type": "Set",
          "required": true,
          "element": {
            "type": "Long"
          }
        },
        "approvedBy": {
          "name": "User",
          "required": true,
          "type": "Entity"
        },
        "MfaAuthorized": {
          "type": "Boolean",
          "required": true
        },
        "NetworkInfo": {
          "type": "NetworkInfo",
          "required": true
        },
        "RequestedOrderCount": {
          "type": "Long",
          "required": true
        },
        "UserAgent": {
          "required": true,
          "type": "String"
        }
      }
    },
    "principalTypes": [
      "User"
    ]
  }
}
```

若要參考請求內容NetworkInfo中名為 的record資料類型，請在 之前的結構描述中新增以下內容，以在結構描述中建立 commonType 建構actions。commonType 建構是一組共用屬性，您可以套用至不同的實體。

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

Policy

下列政策會設定必須由每個提供的內容元素滿足的條件。它以現有的靜態政策為基礎，並描述客戶角色 - 取得訂單。此政策最初只需要提出請求的委託人是資源的擁有者。

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

我們現在要求擷取訂單的請求，必須符合我們新增至請求的其他內容條件。

1. 使用者必須已使用 MFA 登入。
2. 使用者的 Web 瀏覽器User-Agent必須包含字串 My UserAgent。
3. 使用者必須請求檢視 4 個或更少的訂單。
4. 其中一個使用者的帳戶代碼必須是 111122223333。
5. 使用者的 IP 地址必須源自美國，且必須位於加密的工作階段中，且其 IP 地址必須以 開頭192.0.2.。
6. 員工必須已核准其訂單。在授權請求的 entities元素中，我們將宣告具有 角色Bob的使用者Employee。

Request body

使用適當的結構描述和政策設定政策存放區之後，您可以將此授權請求呈現給 Verified Permissions API 操作 IsAuthorized。請注意，客entities群包含 的定義Bob，即角色為 的使用者Employee。

```
{
  "principal": {
    "entityType": "DigitalPetStore::User",
    "entityId": "Alice"
  },
  "action": {
    "actionType": "DigitalPetStore::Action",
    "actionId": "GetOrder"
  },
  "resource": {
    "entityType": "DigitalPetStore::Order",
    "entityId": "1234"
  },
  "context": {
    "contextMap": {
      "AccountCodes": {
        "set": [
          {"long": 111122223333},
          {"long": 444455556666},
          {"long": 123456789012}
        ]
      },
      "approvedBy": {
        "entityIdentifier": {
          "entityId": "Bob",
```

```
          "entityType": "DigitalPetStore::User"
        }
      },
      "MfaAuthorized": {
        "boolean": true
      },
      "NetworkInfo": {
        "record": {
          "Country": {"string": "United States of America"},
          "IPAddress": {"string": "192.0.2.178"},
          "SSL": {"boolean": true}
        }
      },
      "RequestedOrderCount":{
        "long": 4
      },
      "UserAgent": {
        "string": "My UserAgent 1.12"
      }
    }
  },
  "entities": {
    "entityList": [
      {
        "identifier": {
          "entityType": "DigitalPetStore::User",
          "entityId": "Alice"
        },
        "attributes": {
          "memberId": {
            "string": "801b87f2-1a5c-40b3-b580-eacad506d4e6"
          }
        },
        "parents": [
          {
            "entityType": "DigitalPetStore::Role",
            "entityId": "Customer"
          }
        ]
      },
      {
        "identifier": {
          "entityType": "DigitalPetStore::User",
          "entityId": "Bob"
```

```
        },
        "attributes": {
          "memberId": {
            "string": "49d9b81e-735d-429c-989d-93bec0bcfd8b"
          }
        },
        "parents": [
          {
            "entityType": "DigitalPetStore::Role",
            "entityId": "Employee"
          }
        ]
      },
      {
        "identifier": {
          "entityType": "DigitalPetStore::Order",
          "entityId": "1234"
        },
        "attributes": {
          "owner": {
            "entityIdentifier": {
              "entityType": "DigitalPetStore::User",
              "entityId": "Alice"
            }
          }
        },
        "parents": []
      }
    ]
  },
  "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

使用 Amazon Verified Permissions 測試工作台

使用 Verified Permissions 測試工作台，透過對 Verified Permissions 政策執行授權請求來測試和疑難排解。測試台會使用您指定的參數來判斷政策存放區中的 Cedar 政策是否會授權請求。您可以在測試授權請求時，在視覺化模式和 JSON 模式之間切換。如需如何建構和評估 Cedar 政策的詳細資訊，請參閱《Cedar 政策語言參考指南》中的 Cedar 中的基本政策建構。

> **Note**
>
> 當您使用 Verified Permissions 提出授權請求時，您可以在其他實體區段中提供委託人和資源的清單做為請求的一部分。不過，您無法包含動作的詳細資訊。它們必須在結構描述中指定或從請求推斷。您無法在其他實體區段中放置動作。

如需測試工作台的視覺化概觀和示範，請參閱 AWS YouTube 頻道上的 Amazon Verified Permissions - Policy Creation and Testing (Primer Series #3)。

Visual mode

> **Note**
>
> 您必須在政策存放區中定義結構描述，才能使用測試台的視覺化模式。

在視覺化模式中測試政策

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇測試工作台。
3. 選擇視覺化模式。
4. 在主體區段中，從結構描述中的主體類型中選擇主體採取動作。在文字方塊中輸入委託人的識別符。
5. （選用）選擇新增父項，為指定的委託人新增父項實體。若要移除已新增至主體的父系，請選擇父系名稱旁的移除。
6. 為指定主體的每個屬性指定屬性值。測試台使用模擬授權請求中指定的屬性值。
7. 在資源區段中，選擇委託人正在處理的資源。在文字方塊中輸入資源的識別符。
8. （選用）選擇新增父項以新增指定資源的父項實體。若要移除已新增至資源的父項，請選擇父項名稱旁的移除。
9. 為指定資源的每個屬性指定屬性值。測試台使用模擬授權請求中指定的屬性值。
10. 在動作區段中，從指定委託人和資源的有效動作清單中選擇委託人正在採取的動作。
11. 為指定動作的每個屬性指定屬性值。測試台使用模擬授權請求中指定的屬性值。
12. （選用）在其他實體區段中，選擇新增實體以新增要評估授權決策的實體。
13. 從下拉式清單中選擇實體識別符，然後輸入實體識別符。

14. （選用）選擇新增父項以新增指定實體的父項實體。若要移除已新增至實體的父項，請選擇父項名稱旁的移除。
15. 為指定實體的每個屬性指定屬性值。測試台使用模擬授權請求中指定的屬性值。
16. 選擇確認，將實體新增至測試台。
17. 選擇執行授權請求，以模擬政策存放區中 Cedar 政策的授權請求。測試工作台會顯示允許或拒絕請求的決定，以及有關所滿足政策或評估期間發生錯誤的資訊。

JSON mode

在 JSON 模式中測試政策

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇測試工作台。
3. 選擇 JSON 模式。
4. 在請求詳細資訊區段中，如果您已定義結構描述，請從結構描述中的委託人類型中選擇委託人採取動作。在文字方塊中輸入委託人的識別符。

   如果您沒有定義結構描述，請在委託人採取動作文字方塊中輸入委託人。
5. 如果您已定義結構描述，請從結構描述中的資源類型中選擇資源。在文字方塊中輸入資源的識別符。

   如果您沒有定義結構描述，請在資源文字方塊中輸入資源。
6. 如果您已定義結構描述，請從指定委託人和資源的有效動作清單中選擇動作。

   如果您沒有定義結構描述，請在動作文字方塊中輸入動作。
7. 在內容欄位中輸入要模擬的請求內容。請求內容是可用於授權決策的其他資訊。
8. 在實體欄位中，輸入要評估授權決策的實體階層及其屬性。
9. 選擇執行授權請求，以模擬政策存放區中 Cedar 政策的授權請求。測試工作台會顯示允許或拒絕請求的決定，以及有關所滿足政策或評估期間發生錯誤的資訊。

Amazon Verified Permissions 範例政策

此處包含的一些政策範例是基本的 Cedar 政策範例，有些則是 Verified Permissions-specific。基本政策連結至 Cedar 政策語言參考指南，並包含在其中。如需 Cedar 政策語法的詳細資訊，請參閱《Cedar 政策語言參考指南》中的 Cedar 中的基本政策建構。

政策範例

- 允許存取個別實體
- 允許存取實體群組
- 允許存取任何實體
- 允許存取實體的屬性 (ABAC)
- 拒絕存取
- 使用括號表示法來參考字符屬性
- 使用點表示法來參考屬性
- 反映 Amazon Cognito ID 字符屬性
- 反映 OIDC ID 字符屬性
- 反映 Amazon Cognito 存取字符屬性
- 反映 OIDC 存取權杖屬性

使用括號表示法來參考字符屬性

以下範例示範如何建立使用括號表示法參考字符屬性的政策。

如需在 Verified Permissions 中的政策中使用字符屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

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

使用點表示法來參考屬性

以下範例示範如何建立使用點表示法來參考屬性的政策。

如需在 Verified Permissions 中的政策中使用字符屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

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

反映 Amazon Cognito ID 字符屬性

以下範例示範如何從 Amazon Cognito 建立政策參考 ID 字符屬性。

如需在 Verified Permissions 中的政策中使用字符屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

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

反映 OIDC ID 字符屬性

以下範例示範如何從 OIDC 供應商建立政策參考 ID 字符屬性。

如需在 Verified Permissions 中的政策中使用字符屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

```
permit (
    principal in MyCorp::UserGroup::"MyOIDCProvider|MyUserGroup",
    action,
    resource
) when {
```

```
    principal.email_verified == true && principal.email == "alice@example.com" &&
    principal.phone_number_verified == true && principal.phone_number like "+1206*"
};
```

反映 Amazon Cognito 存取字符屬性

以下範例示範如何建立政策，參考來自 Amazon Cognito 的存取權杖屬性。

如需在 Verified Permissions 中的政策中使用字符屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

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

反映 OIDC 存取權杖屬性

以下範例示範如何建立政策，參考來自 OIDC 供應商的存取權杖屬性。

如需在 Verified Permissions 中的政策中使用字符屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

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

Amazon Verified Permissions 政策範本和範本連結政策

在已驗證的許可中，政策範本是具有 `principal`、 `resource`或兩者預留位置的政策。無法單獨使用政策範本來處理授權請求。若要處理授權請求，必須根據政策範本建立範本連結政策。政策範本允許定義政策一次，然後與多個主體和資源搭配使用。政策範本的更新會反映在使用該範本的所有政策中。如需詳細資訊，請參閱《 Cedar 政策語言參考指南》中的 Cedar 政策範本。

例如，下列政策範本為使用政策範本的主體和資源提供 `Edit`、 和 `ReadComment`許可。

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

如果您要`Editor`根據此範本建立名為 的政策，當委託人被指定為特定資源的編輯器時，您的應用程式會建立政策，提供許可給委託人讀取、編輯和評論資源。

與靜態政策不同，範本連結政策是動態的。以上述範例為例，如果您要從政策範本中移除`Comment`動作，任何連結至該範本或根據該範本的政策都會隨之更新，且政策中指定的主體將無法再對對應的資源進行評論。

如需更多範本連結政策範例，請參閱 Amazon Verified Permissions 範本連結政策範例。

建立 Amazon Verified Permissions 政策範本

您可以使用 AWS Management Console、 AWS CLI或 AWS SDKs，在已驗證的許可中建立政策範本。政策範本允許定義政策一次，然後與多個主體和資源搭配使用。建立政策範本之後，您可以建立範本連結政策，以搭配特定主體和資源使用政策範本。如需詳細資訊，請參閱建立 Amazon Verified Permissions 範本連結政策。

AWS Management Console

建立政策範本

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇政策範本。
3. 選擇建立政策範本。

4. 在詳細資訊區段中，輸入政策範本描述。
5. 在政策範本內文區段中，使用預留位置?principal?resource並允許根據此範本建立的政策自訂其授予的許可。您可以選擇格式化，以使用建議的間距和縮排來格式化政策範本的語法。
6. 選擇建立政策範本。

AWS CLI

建立政策範本

您可以使用 CreatePolicyTemplate 操作來建立政策範本。下列範例會建立具有委託人預留位置的政策範本。

檔案template1.txt包含下列項目。

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

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

建立 Amazon Verified Permissions 範本連結政策

您可以使用 、 或 AWS SDKs，建立以政策範本為基礎的範本連結政策 AWS Management Console AWS CLI或政策。範本連結政策會與其政策範本保持連結。如果您變更政策範本中的政策陳述式，則任何連結至該範本的政策都會針對從該時刻開始的所有授權決策，自動使用新的陳述式。

如需範本連結政策範例，請參閱 Amazon Verified Permissions 範本連結政策範例。

AWS Management Console

透過執行個體化政策範本來建立範本連結政策

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側的導覽窗格中，選擇 Policies (政策)。
3. 選擇建立政策，然後選擇建立範本連結政策。
4. 選擇要使用的政策範本旁的選項按鈕，然後選擇下一步。
5. 輸入要用於此範本連結政策特定執行個體的主體和資源。指定的值會顯示在政策陳述式預覽欄位中。

   > **Note**
   >
   > 主體和資源值的格式必須與靜態政策相同。例如，若要指定委託人的`AdminUsers`群組，請輸入 `Group::"AdminUsers"`。如果您輸入 `AdminUsers`，則會顯示驗證錯誤。

6. 選擇建立範本連結政策。

   新的範本連結政策會顯示在政策下。

AWS CLI

透過執行個體化政策範本來建立範本連結政策

您可以建立參考現有政策範本的範本連結政策，並指定範本使用的任何預留位置值。

下列範例會建立範本連結政策，該政策使用具有下列陳述式的範本：

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

它也會使用下列`definition.txt`檔案來提供 `definition` 參數的值：

```
{
    "templateLinked": {
        "policyTemplateId": "PTEXAMPLEabcdefg111111",
```

```
        "principal": {
            "entityType": "PhotoFlash::User",
            "entityId": "alice"
        }
    }
}
```

輸出會顯示資源，而資源是從範本取得，而主體則是從定義參數取得

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

編輯 Amazon Verified Permissions 政策範本

您可以使用 、 或 AWS SDKs 在已驗證許可中編輯 AWS Management Console AWS CLI或更新政策範本。編輯政策範本會自動更新連結至或根據範本的政策，因此在編輯政策範本時要小心，並確保您不會意外引入中斷應用程式的變更。

您可以變更政策範本的下列元素：

- 政策範本`action`參考的
- 條件子句，例如 `when`和 `unless`

您無法變更政策範本的下列元素。若要變更任何這些元素，您需要刪除並重新建立政策範本。

- 來自 `permit`或 的政策範本效果 `forbid`
- 政策範本`principal`參考的
- 政策範本`resource`參考的

AWS Management Console

編輯您的政策範本

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇政策範本。主控台會顯示您在目前政策存放區中建立的所有政策範本。
3. 選擇政策範本旁的選項按鈕，以顯示政策範本的詳細資訊，例如建立、更新和政策範本內容的時間。
4. 選擇編輯以編輯您的政策範本。視需要更新政策描述和政策內文，然後選擇更新政策範本。
5. 您可以選擇政策範本旁的選項按鈕，然後選擇刪除，以刪除政策範本。選擇確定以確認刪除政策範本。

AWS CLI

編輯政策範本

您可以使用 UpdatePolicy 操作來建立靜態政策。下列範例會將其政策內文取代為 檔案中定義的新政策，以更新指定的政策範本。

檔案 的內容`template1.txt`：

```
permit(
    principal in ?principal,
    action == Action::"view",
    resource in ?resource)
when {
    principal has department && principal.department == "research"
};
```

```
$ aws verifiedpermissions update-policy-template \
    --policy-template-id PTEXAMPLEabcdefg111111 \
    --description "My updated template description" \
    --statement file://template1.txt \
```

```
    --policy-store-id PSEXAMPLEabcdefg111111
{
    "createdDate": "2023-05-17T18:58:48.795411+00:00",
    "lastUpdatedDate": "2023-05-17T19:18:48.870209+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111"
}
```

Amazon Verified Permissions 範本連結政策範例

當您使用範例政策存放區方法在已驗證許可中建立政策存放區時，您的政策存放區會使用您所選範例專案的預先定義政策、政策範本和結構描述來建立。下列 Verified Permissions 範本連結政策範例可與範例政策存放區及其個別政策、政策範本和結構描述搭配使用。

PhotoFlash 範例

下列範例示範如何建立使用政策範本的範本連結政策 授予非私有共用相片的有限存取權，以與個別使用者和相片分享。

> **Note**
>
> Cedar 政策語言會將實體視為in本身。因此，`principal in User::"Alice"` 等同於 `principal == User::"Alice"`。

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

下列範例示範如何建立使用政策範本的範本連結政策 授予與個別使用者和相簿的非私有共用相片的有限存取權。

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

下列範例示範如何建立使用政策範本的範本連結政策 授予非私有共用相片的有限存取權，其中包含好友群組和個別相片。

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

下列範例示範如何建立使用政策範本的範本連結政策，授予非私有共享相片的有限存取權，其中包含好友群組和相簿。

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

下列範例示範如何建立範本連結政策，該政策使用政策範本 授予非私有共用相片的完整存取權，其中包含好友群組和個別相片。

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

下列範例示範如何建立使用政策範本的範本連結政策 從 帳戶封鎖使用者。

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

DigitalPetStore 範例

DigitalPetStore 範例政策存放區不包含任何政策範本。您可以在建立 DigitalPetStore 範例政策存放區後，選擇左側導覽窗格中的政策，以檢視政策存放區中包含的政策。

TinyToDo 範例

下列範例示範如何建立範本連結政策，該政策使用政策範本，為個別使用者和任務清單提供瀏覽者存取權。

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

下列範例示範如何建立範本連結政策，該政策使用政策範本，為個別使用者和任務清單提供編輯器存取權。

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

搭配身分提供者使用 Amazon Verified Permissions

身分來源是 Amazon Verified Permissions 中外部身分提供者 (IdP) 的表示法。身分來源提供來自使用與您政策存放區具有信任關係的 IdP 驗證的使用者的資訊。當您的應用程式從身分來源使用字符提出授權請求時，您的政策存放區可以對使用者屬性和存取許可做出授權決策。您可以新增 Amazon Cognito 使用者集區或自訂 OpenID Connect (OIDC) IdP 做為您的身分來源。

您可以搭配 Verified Permissions 使用 OpenID Connect (OIDC) 身分提供者 (IdPs)。您的應用程式可以使用 OIDC 相容身分提供者產生的 JSON Web 字符 (JWTs) 產生授權請求。權杖中的使用者身分會對應至主體 ID。使用 ID 權杖時，已驗證的許可會將屬性宣告對應至主體屬性。使用存取字符，這些宣告會映射到內容。使用這兩種字符類型，您可以將類似 的宣告對應`groups`至委託人群組，並建置評估角色型存取控制 (RBAC) 的政策。

> **Note**
>
> Verified Permissions 根據來自 IdP 權杖的資訊做出授權決策，但不會以任何方式直接與 IdP 互動。

如需使用 Amazon Cognito 使用者集區或 OIDC 身分提供者為 Amazon API Gateway REST APIs 建置授權邏輯step-by-step演練，請參閱使用 Amazon Verified Permissions 搭配 Amazon Cognito 授權 API Gateway APIs，或在安全部落格上自攜身分提供者。 Amazon Cognito AWS

主題

- 使用 Amazon Cognito 身分來源
- 使用 OIDC 身分來源
- 用戶端和對象驗證
- JWTs的用戶端授權
- 建立 Amazon Verified Permissions 身分來源
- 編輯 Amazon Verified Permissions 身分來源
- 將身分提供者字符映射至結構描述

使用 Amazon Cognito 身分來源

已驗證的許可與 Amazon Cognito 使用者集區密切相關。Amazon Cognito JWTs具有可預測的結構。Verified Permissions 會辨識此結構，並從其中包含的資訊中取得最大利益。例如，您可以使用 ID 字符或存取字符來實作角色型存取控制 (RBAC) 授權模型。

新的 Amazon Cognito 使用者集區身分來源需要下列資訊：

- AWS 區域。
- 使用者集區 ID。
- 您要與身分來源建立關聯的主體實體類型，例如 `MyCorp::User`。
- 您要與身分來源建立關聯的主體群組實體類型，例如 `MyCorp::UserGroup`。
- 使用者集區的用戶端 IDs，您想要授權 向政策存放區發出請求。

由於 Verified Permissions 僅適用於相同 中的 Amazon Cognito 使用者集區 AWS 帳戶，因此您無法在另一個帳戶中指定身分來源。Verified Permissions 會將實體字首 - 在對使用者集區主體採取行動的政策中必須參考的身分來源識別符 - 設定為使用者集區的 ID，例如 `us-west-2_EXAMPLE`。在此情況下，您會參考該使用者集區中的使用者，並將 ID `a1b2c3d4-5678-90ab-cdef-EXAMPLE22222`為 `us-west-2_EXAMPLE|a1b2c3d4-5678-90ab-cdef-EXAMPLE22222`

使用者集區字符宣告可以包含屬性、範圍、群組、用戶端 IDs和自訂資料。Amazon Cognito JWTs能夠包含各種資訊，有助於在 Verified Permissions 中做出授權決策。其中包含：

1. 具有`cognito:`字首的使用者名稱和群組宣告
2. 使用 自訂使用者屬性 `custom: prefix`
3. 在執行時間新增的自訂宣告
4. OIDC 標準宣告，例如 `sub`和 `email`

我們會詳細說明這些宣告，以及如何在 的 Verified Permissions 政策中管理這些宣告將身分提供者字符映射至結構描述。

> **Important**
>
> 雖然您可以在過期前撤銷 Amazon Cognito 權杖，但 JWTs被視為無狀態資源，這些資源會與簽章和有效性獨立。符合 JSON Web 權杖 RFC 7519 的服務預期會遠端驗證權杖，而且不需

要向發行者驗證權杖。這表示 Verified Permissions 可以根據已撤銷的字符授予存取權，或為稍後刪除的使用者授予存取權。若要降低此風險，我們建議您建立最短有效期間的權杖，並在想要移除繼續使用者工作階段的授權時撤銷重新整理權杖。如需詳細資訊，請參閱使用字符撤銷結束使用者工作階段

下列範例示範如何建立政策，以參考與委託人相關聯的一些 Amazon Cognito 使用者集區宣告。

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

下列範例示範如何建立 政策，以參考 Cognito 使用者集區中的使用者主體。請注意，主體 ID 採用 的形式"<userpool-id>|<sub>"。

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

Verified Permissions 中使用者集區身分來源的 Cedar 政策針對包含英數字元和底線 () 以外的字元的宣告名稱使用特殊語法_。這包括包含:字元的使用者集區字首宣告，例如 cognito:username和 custom:department。若要撰寫參考 cognito:username或 custom:department宣告的政策條件principal["custom:department"]，請分別將它們寫入 principal["cognito:username"]和 。

> **Note**
>
> 如果權杖包含具有 cognito:或 custom:字首的宣告，以及具有常值 cognito或 的宣告名稱custom，IsAuthorizedWithToken 的授權請求將使用 失敗ValidationException。

如需映射宣告的詳細資訊，請參閱將 ID 字符映射至結構描述。如需 Amazon Cognito 使用者授權的詳細資訊，請參閱《Amazon Amazon Cognito開發人員指南》中的使用 Amazon 驗證許可授權。

使用 OIDC 身分來源

您也可以將任何合規的 OpenID Connect (OIDC) IdP 設定為政策存放區的身分來源。OIDC 提供者類似於 Amazon Cognito 使用者集區：它們會產生 JWTs作為身分驗證產品。若要新增 OIDC 提供者，您必須提供發行者 URL

新的 OIDC 身分來源需要下列資訊：

- 發行者 URL。驗證的許可必須能夠在此 URL 上探索`.well-known/openid-configuration`端點。
- 不包含萬用字元的 CNAME 記錄。例如， `a.example.com` 無法映射到 `*.example.net`。相反地， `*.example.com` 無法映射到 `a.example.net`。
- 您想要在授權請求中使用的字符類型。在這種情況下，您選擇了身分字符。
- 您要與身分來源建立關聯的使用者實體類型，例如 `MyCorp::User`。
- 您要與身分來源建立關聯的群組實體類型，例如 `MyCorp::UserGroup`。
- ID 字符範例，或 ID 字符中宣告的定義。
- 您要套用至使用者和群組實體 IDs字首。在 CLI 和 API 中，您可以選擇此字首。在您使用 API Gateway 設定和身分提供者或引導式設定選項建立的政策存放區中，驗證許可會指派發行者名稱減 的字首`https://`，例如 `MyCorp::User::"auth.example.com|a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"`。

如需使用 API 操作來授權來自 OIDC 來源請求的詳細資訊，請參閱 授權可用的 API 操作。

以下範例示範如何建立政策，允許會計部門員工存取年底報告、進行機密分類，而且不在衛星辦公室。已驗證的許可會從委託人的 ID 權杖中的宣告衍生這些屬性。

請注意，在主體中參考群組時，您必須使用 `in` 運算子才能正確評估政策。

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

```
};
```

用戶端和對象驗證

當您將身分來源新增至政策存放區時，已驗證許可具有組態選項，可驗證 ID 和存取權杖是否如預期般使用。此驗證會在處理 `IsAuthorizedWithToken`和 `BatchIsAuthorizedWithToken` API 請求時發生。ID 和存取字符，以及 Amazon Cognito 和 OIDC 身分來源的行為有所不同。使用 Amazon Cognito 使用者集區提供者，驗證許可可以驗證 ID 和存取權杖中的用戶端 ID。使用 OIDC 提供者時，已驗證許可可以在 ID 字符中驗證用戶端 ID，並在存取字符中驗證對象。

用戶端 ID 是與您的應用程式使用的身分提供者執行個體相關聯的識別符，例如 `1example23456789`。對象是與存取權杖的預期相依方或目的地相關聯的 URL 路徑，例如 `https://mytoken.example.com`。使用存取權杖時，`aud`宣告一律與對象相關聯。

Verified Permissions 會執行身分來源對象和用戶端驗證，如下所示：

Amazon Cognito

Amazon Cognito ID 權杖具有包含應用程式用戶端 ID 的`aud`宣告。存取權杖的`client_id`宣告也包含應用程式用戶端 ID。

當您在身分來源中輸入一或多個用戶端應用程式驗證值時，驗證許可會將此應用程式用戶端 IDs 清單與 ID 字符`aud`宣告或存取字符`client_id`宣告進行比較。Verified Permissions 不會驗證 Amazon Cognito 身分來源的相依對象 URL。

OIDC

OIDC ID 字符具有包含用戶端 IDs 的`aud`宣告，例如 `1example23456789`。

OIDC Access 字符具有包含字符受眾 URL 的`aud`宣告，例如 `https://myapplication.example.com`，以及包含用戶端 IDs的`client_id`宣告，例如 `1example23456789`。

設定您的政策存放區時，請輸入一或多個值，以供您的政策存放區用來驗證權杖的對象之對象的對象驗證。

- ID 字符 – Verified Permissions 透過檢查`aud`宣告中至少有一個用戶端 IDs成員符合對象驗證值來驗證用戶端 ID。
- 存取權杖 – Verified Permissions 透過檢查`aud`宣告中的 URL 是否符合對象驗證值來驗證對象。如果不存在`aud`宣告，可以使用 `cid`或 `client_id`宣告來驗證對象。請洽詢您的身分提供者，了解正確的受眾聲明和格式。

JWTs的用戶端授權

您可能想要在應用程式中處理 JSON Web 字符，並將其宣告傳遞給驗證許可，而無需使用政策存放區身分來源。您可以從 JSON Web 權杖 (JWT) 擷取實體屬性，並將其剖析為驗證許可。

此範例示範如何使用 JWT.1 從應用程式呼叫 Verified Permissions。

```
async function authorizeUsingJwtToken(jwtToken) {

    const payload = await verifier.verify(jwtToken);

    let principalEntity = {
        entityType: "PhotoFlash::User", // the application needs to fill in the
 relevant user type
        entityId: payload["sub"], // the application need to use the claim that
 represents the user-id
    };
    let resourceEntity = {
        entityType: "PhotoFlash::Photo", //the application needs to fill in the
 relevant resource type
        entityId: "jane_photo_123.jpg", // the application needs to fill in the
 relevant resource id
    };
    let action = {
        actionType: "PhotoFlash::Action", //the application needs to fill in the
 relevant action id
        actionId: "GetPhoto", //the application needs to fill in the relevant action
 type
    };
    let entities = {
        entityList: [],
    };
    entities.entityList.push(...getUserEntitiesFromToken(payload));
    let policyStoreId = "PSEXAMPLEabcdefg111111"; // set your own policy store id

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

```
        .promise();

    return authResult;

}

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

  });

  let entityItem = {
    attributes: attributes,
    identifier: {
      entityType: "PhotoFlash::User",
```

```
      entityId: payload["sub"], // the application needs to use the claim that
  represents the user-id
     }
   };
   return [entityItem];
}
```

1 此程式碼範例使用 aws-jwt-verify 程式庫來驗證由 OIDC 相容 IdPs 簽署的 JWTs。

建立 Amazon Verified Permissions 身分來源

下列程序會將身分來源新增至現有的政策存放區。新增身分來源之後，您必須將屬性新增至結構描述。

您也可以在 Verified Permissions 主控台中建立新的政策存放區時建立身分來源。在此程序中，您可以將身分來源字符中的宣告自動匯入實體屬性。選擇引導式設定或使用 API Gateway 和身分提供者設定選項。這些選項也會建立初始政策。

> **Note**
>
> 在您建立政策存放區之前，左側導覽窗格中無法使用身分來源。您建立的身分來源與目前的政策存放區相關聯。

當您在 中使用 create-identity-source 建立身分來源， AWS CLI 或在 Verified Permissions API 中建立 CreateIdentitySource 時，您可以捨棄主體實體類型。不過，空白實體類型會建立實體類型為 的身分來源`AWS::Cognito`。此實體名稱與政策存放區結構描述不相容。若要將 Amazon Cognito 身分與您的政策存放區結構描述整合，您必須將主體實體類型設定為支援的政策存放區實體。

主題

- Amazon Cognito 身分來源
- OIDC 身分來源

Amazon Cognito 身分來源

AWS Management Console

建立 Amazon Cognito 使用者集區身分來源

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。

2. 在左側導覽窗格中，選擇身分來源。
3. 選擇建立身分來源。
4. 在 Cognito 使用者集區詳細資訊中，選取 AWS 區域 並輸入身分來源的使用者集區 ID。
5. 在主體組態中，針對主體類型，從此來源選擇主體的實體類型。來自已連線 Amazon Cognito 使用者集區的身分將對應至選取的主體類型。
6. 在群組組態中，如果您想要對應使用者集區`cognito:groups`宣告，請選取使用 Cognito 群組。選擇屬於委託人類型的父系實體類型。
7. 在用戶端應用程式驗證中，選擇是否要驗證用戶端應用程式 IDs。
   - 若要驗證用戶端應用程式 IDs，請選擇僅接受具有相符用戶端應用程式 IDs字符。選擇為每個要驗證的用戶端應用程式 ID 新增用戶端應用程式 ID。若要移除已新增的用戶端應用程式 ID，請選擇用戶端應用程式 ID 旁的移除。
   - 如果您不想驗證用戶端應用程式 IDs請選擇不要驗證用戶端應用程式 IDs。
8. 選擇建立身分來源。

如果您的政策存放區有結構描述，在您可以參考從 Cedar 政策中的身分或存取權杖擷取的屬性之前，您必須更新結構描述，讓 Cedar 了解您的身分來源建立的主體類型。除了結構描述之外，還必須包含您想要在 Cedar 政策中參考的屬性。如需將 Amazon Cognito 權杖屬性對應至 Cedar 主體屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

當您建立 API 連結政策存放區，或在建立政策存放區時使用設定 API Gateway 和身分提供者時，驗證許可會查詢使用者集區中的使用者屬性，並建立結構描述，其中您的主體類型會填入使用者集區屬性。

AWS CLI

建立 Amazon Cognito 使用者集區身分來源

您可以使用 CreateIdentitySource 操作來建立身分來源。下列範例會建立身分來源，從 Amazon Cognito 使用者集區存取已驗證的身分。

下列`config.txt`檔案包含 Amazon Cognito 使用者集區的詳細資訊，以供 `create-identity-source`命令中的 --configuration 參數使用。

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

```
        "groupConfiguration": {
              "groupEntityType": "MyCorp::UserGroup"
        }
    }
}
```

命令：

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

如果您的政策存放區有結構描述，在您可以參考從 Cedar 政策中的身分或存取權杖擷取的屬性之前，您必須更新結構描述，讓 Cedar 了解您的身分來源建立的主體類型。除了結構描述之外，還必須包含您想要在 Cedar 政策中參考的屬性。如需將 Amazon Cognito 權杖屬性對應至 Cedar 主體屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

當您建立 API 連結政策存放區，或在建立政策存放區時使用設定 API Gateway 和身分提供者時，驗證許可會查詢使用者集區中的使用者屬性，並建立結構描述，其中您的主體類型會填入使用者集區屬性。

如需在 Verified Permissions 中使用 Amazon Cognito 存取和身分字符給已驗證使用者的詳細資訊，請參閱《Amazon Amazon Cognito Permissions 授權。

OIDC 身分來源

AWS Management Console

建立 OpenID Connect (OIDC) 身分來源

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇身分來源。
3. 選擇建立身分來源。

4. 選擇外部 OIDC 供應商。
5. 在發行者 URL 中，輸入 OIDC 發行者的 URL。這是提供服務端點，可提供授權伺服器、簽署金鑰，以及有關提供者的其他資訊，例如 `https://auth.example.com`。您的發行者 URL 必須在 託管 OIDC 探索文件`/.well-known/openid-configuration`。
6. 在權杖類型中，選擇您希望應用程式提交以進行授權的 OIDC JWT 類型。如需詳細資訊，請參閱將身分提供者字符映射至結構描述。
7. 在將字符宣告對應至結構描述實體中，選擇身分來源的使用者實體和使用者宣告。使用者實體是政策存放區中的實體，您想要從 OIDC 提供者參考使用者。使用者宣告是來自您的 ID 或存取權杖`sub`的宣告，其保留要評估之實體的唯一識別符。來自已連線 OIDC IdP 的身分將對應至選取的委託人類型。
8. （選用）在將字符宣告映射至結構描述實體中，選擇身分來源的群組實體和群組宣告。群組實體是使用者實體的父系。群組宣告會映射到此實體。群組宣告通常是來自您的 ID 或存取權杖`groups`的宣告，其中包含要評估之實體的字串、JSON 或以空格分隔的使用者群組名稱字串。來自已連線 OIDC IdP 的身分將對應至選取的委託人類型。
9. 在驗證 - 選用中，輸入您希望政策存放區在授權請求中接受的用戶端 IDs 或對象 URLs，如果有的話。
10. 選擇建立身分來源。
11. 更新您的結構描述，讓 Cedar 了解您的身分來源建立的委託人類型。除了結構描述之外，還必須包含您想要在 Cedar 政策中參考的屬性。如需將 Amazon Cognito 權杖屬性對應至 Cedar 主體屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

    當您建立 API 連結政策存放區時，驗證許可會查詢使用者集區中的使用者屬性，並建立結構描述，其中您的主體類型會填入使用者集區屬性。

AWS CLI

建立 OIDC 身分來源

您可以使用 CreateIdentitySource 操作來建立身分來源。下列範例會建立身分來源，從 Amazon Cognito 使用者集區存取已驗證的身分。

下列`config.txt`檔案包含 OIDC IdP 的詳細資訊，以供 `create-identity-source`命令的 `--configuration` 參數使用。此範例會建立 ID 字符的 OIDC 身分來源。

```
{
    "openIdConnectConfiguration": {
        "issuer": "https://auth.example.com",
```

```
        "tokenSelection": {
                "identityTokenOnly": {
                        "clientIds":["1example23456789"],
                        "principalIdClaim": "sub"
                },
        },
        "entityIdPrefix": "MyOIDCProvider",
        "groupConfiguration": {
              "groupClaim": "groups",
              "groupEntityType": "MyCorp::UserGroup"
        }
    }
}
```

下列config.txt檔案包含 OIDC IdP 的詳細資訊，以供 create-identity-source命令的 --configuration 參數使用。此範例會建立存取權杖的 OIDC 身分來源。

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

命令：

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

```
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

您必須先更新結構描述，讓 Cedar 了解您的身分來源建立的主體類型，才能參考從 Cedar 政策中的身分或存取權杖擷取的屬性。除了結構描述之外，還必須包含您想要在 Cedar 政策中參考的屬性。如需將 Amazon Cognito 權杖屬性對應至 Cedar 主體屬性的詳細資訊，請參閱 將身分提供者字符映射至結構描述。

當您建立 API 連結政策存放區時，驗證許可會查詢使用者集區中的使用者屬性，並建立結構描述，其中您的主體類型會填入使用者集區屬性。

編輯 Amazon Verified Permissions 身分來源

您可以在建立身分來源之後編輯其某些參數。您無法變更身分來源的類型，您必須刪除身分來源，並建立新的來源，才能從 Amazon Cognito 切換到 OIDC 或 OIDC 切換到 Amazon Cognito。如果您的政策存放區結構描述符合身分來源屬性，請注意，您必須分別更新結構描述，以反映您對身分來源所做的變更。

主題

- Amazon Cognito 使用者集區身分來源
- OpenID Connect (OIDC) 身分來源

Amazon Cognito 使用者集區身分來源

AWS Management Console

更新 Amazon Cognito 使用者集區身分來源

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇身分來源。
3. 選擇要編輯的身分來源 ID。
4. 選擇編輯。
5. 在 Cognito 使用者集區詳細資訊中，選取 AWS 區域 並輸入身分來源的使用者集區 ID。
6. 在主體詳細資訊中，您可以更新身分來源的主體類型。來自已連線 Amazon Cognito 使用者集區的身分將對應至選取的主體類型。

7. 在群組組態中，如果您想要對應使用者集區`cognito:groups`宣告，請選取使用 Cognito 群組。選擇主體類型的父系實體類型。
8. 在用戶端應用程式驗證中，選擇是否要驗證用戶端應用程式 IDs。
   - 若要驗證用戶端應用程式 IDs，請選擇僅接受具有相符用戶端應用程式 IDs字符。選擇為每個要驗證的用戶端應用程式 ID 新增用戶端應用程式 ID。若要移除已新增的用戶端應用程式 ID，請選擇用戶端應用程式 ID 旁的移除。
   - 如果您不想驗證用戶端應用程式 IDs請選擇不要驗證用戶端應用程式 IDs。
9. 選擇 Save changes (儲存變更)。
10. 如果您變更了身分來源的主體類型，則必須更新您的結構描述，以正確反映更新的主體類型。

您可以選擇身分來源旁的選項按鈕，然後選擇刪除身分來源，以刪除身分來源。在文字方塊`delete`中輸入 ，然後選擇刪除身分來源以確認刪除身分來源。

AWS CLI

更新 Amazon Cognito 使用者集區身分來源

您可以使用 UpdateIdentitySource 操作來更新身分來源。下列範例會將指定的身分來源更新為使用不同的 Amazon Cognito 使用者集區。

下列`config.txt`檔案包含 Amazon Cognito 使用者集區的詳細資訊，以供 `create-identity-source`命令中的 --configuration 參數使用。

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

命令：

```
$ aws verifiedpermissions update-identity-source \
    --update-configuration file://config.txt \
    --policy-store-id 123456789012
```

```
{
    "createdDate": "2023-05-19T20:30:28.214829+00:00",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

如果您變更身分來源的主體類型，則必須更新您的結構描述，以正確反映更新的主體類型。

OpenID Connect (OIDC) 身分來源

AWS Management Console

更新 OIDC 身分來源

1. 開啟 Verified Permissions 主控台。選擇您的政策存放區。
2. 在左側導覽窗格中，選擇身分來源。
3. 選擇要編輯的身分來源 ID。
4. 選擇編輯。
5. 在 OIDC 提供者詳細資訊中，視需要變更發行者 URL。
6. 在將字符宣告對應至結構描述屬性中，視需要變更使用者和群組宣告與政策存放區實體類型的關聯。變更實體類型之後，您必須更新您的政策和結構描述屬性，才能套用至新的實體類型。
7. 在對象驗證中，新增或移除您要強制執行的對象值。
8. 選擇 Save changes (儲存變更)。

您可以選擇身分來源旁的選項按鈕，然後選擇刪除身分來源，以刪除身分來源。在文字方塊delete中輸入 ，然後選擇刪除身分來源以確認刪除身分來源。

AWS CLI

更新 OIDC 身分來源

您可以使用 UpdateIdentitySource 操作來更新身分來源。下列範例會將指定的身分來源更新為使用不同的 OIDC 供應商。

下列config.txt檔案包含 Amazon Cognito 使用者集區的詳細資訊，以供 create-identity-source命令中的 --configuration 參數使用。

```
{
```

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

命令：

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

如果您變更身分來源的主體類型，則必須更新您的結構描述，以正確反映更新的主體類型。

將身分提供者字符映射至結構描述

您可能會發現想要將身分來源新增至政策存放區，並將提供者宣告或權杖映射到您的政策存放區結構描述。您可以使用引導式設定來使用身分來源建立政策存放區，或在建立政策存放區後手動更新結構描述，藉此自動化此程序。將字符映射到結構描述後，您可以建立參考它們的政策。

使用者指南的本節包含下列資訊：

- 何時可以自動將屬性填入政策存放區結構描述
- 如何在 Verified Permissions 政策中使用 Amazon Cognito 和 OIDC 權杖宣告
- 如何手動建置身分來源的結構描述

API 連結政策存放區和具有透過引導式設定建立之身分來源的政策存放區，不需要手動映射身分 (ID) 字符屬性至結構描述。您可以為 Verified Permissions 提供使用者集區中的屬性，並建立填入使用者屬性的結構描述。在 ID 字符授權中，已驗證許可會將宣告對應至主體實體的屬性。在下列情況下，您可能需要手動將 Amazon Cognito 字符映射到您的結構描述：

- 您已從範例建立空的政策存放區或政策存放區。
- 您想要將存取權杖的使用擴展到角色型存取控制 (RBAC) 之外。
- 您可以使用 Verified Permissions REST API、 AWS SDK 或 建立政策存放區 AWS CDK。

若要在 Verified Permissions 政策存放區中使用 Amazon Cognito 或 OIDC 身分提供者 (IdP) 做為身分來源，您必須在結構描述中具有提供者屬性。結構描述是固定的，且必須對應至提供者權杖在 IsAuthorizedWithToken 或 BatchIsAuthorizedWithToken API 請求中建立的實體。如果您建立政策存放區的方式會自動從 ID 權杖中的提供者資訊填入結構描述，您就可以撰寫政策。如果您建立的政策存放區沒有身分來源的結構描述，則必須將提供者屬性新增至結構描述，以符合使用 API 請求建立的實體。然後，您可以使用提供者字符中的屬性來撰寫政策。

如需在 Verified Permissions 中使用 Amazon Cognito ID 和存取權杖給已驗證使用者的詳細資訊，請參閱《Amazon Amazon Cognito Permissions 授權。

主題

- 將 ID 字符映射至結構描述
- 映射存取權杖
- Amazon Cognito 冒號分隔宣告的替代表示法
- 結構描述映射須知

將 ID 字符映射至結構描述

Verified Permissions 會將 ID 字符宣告視為使用者的屬性：其名稱和標題、群組成員資格、其聯絡資訊。ID 字符在屬性型存取控制 (ABAC) 授權模型中最有用。當您希望 Verified Permissions 根據提出請求的人員來分析對資源的存取時，請選擇身分來源的 ID 字符。

Amazon Cognito ID 字符

Amazon Cognito ID 字符適用於大多數依賴 OIDC 的方程式庫。它們擴展了 OIDC 的功能，並具有額外的宣告。您的應用程式可以使用 Amazon Cognito 使用者集區身分驗證 API 操作，或使用使用者集區託管 UI 來驗證使用者。如需詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的使用 API 和端點。

Amazon Cognito ID 權杖中的實用宣告

cognito:username 和 *preferred_username*

使用者使用者名稱的變體。

sub

使用者的唯一使用者識別符 (UUID)

字*custom:*首為 的宣告

自訂使用者集區屬性的字首，例如 `custom:employmentStoreCode`。

標準宣告

標準 OIDC 宣告，例如 `email`和 `phone_number`。如需詳細資訊，請參閱 OpenID Connect Core 1.0 中包含錯誤集 2 的標準宣告。

cognito:groups

使用者的群組成員資格。在以角色為基礎的存取控制 (RBAC) 為基礎的授權模型中，此宣告會顯示您可以在政策中評估的角色。

暫時性宣告

不是使用者屬性的宣告，但是由使用者集區預先產生字符的 Lambda 觸發程序在執行時間新增。暫時性宣告類似於標準宣告，但超出標準範圍，例如 `tenant`或 `department`。

在參考具有:分隔符號的 Amazon Cognito 屬性的政策中，參考格式為 的屬性`principal["cognito:username"]`。角色宣告`cognito:groups`是此規則的例外狀況。已驗證的許可會將此宣告的內容映射至使用者實體的父實體。

如需 Amazon Cognito 使用者集區中 ID 字符結構的詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的使用 ID 字符。

下列範例 ID 字符具有四種類型的屬性。它包含 Amazon Cognito 特定的宣告 `cognito:username`、自訂宣告 `custom:employmentStoreCode`、標準宣告 `email`和暫時性宣告 `tenant`。

```
{
    "sub": "91eb4550-XXX",
    "cognito:groups": [
        "Store-Owner-Role",
```

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

當您使用 Amazon Cognito 使用者集區建立身分來源時，您可以指定 Verified Permissions 在 授權請求中產生的主體實體類型`IsAuthorizedWithToken`。您的政策接著可以測試該委託人的屬性，做為評估該請求的一部分。您的結構描述會定義身分來源的主體類型和屬性，然後您可以在 Cedar 政策中參考它們。

您也可以指定要從 ID 字符群組宣告衍生的群組實體類型。在授權請求中，已驗證的許可會將宣告的群組的每個成員映射到該群組實體類型。在政策中，您可以參考該群組實體做為委託人。

下列範例示範如何反映 Verified Permissions 結構描述中 範例身分字符的屬性。如需編輯結構描述的詳細資訊，請參閱 編輯政策存放區結構描述。如果您的身分來源組態指定委託人類型 `User`，則可以包含類似下列範例的內容，讓 Cedar 使用這些屬性。

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

```
            "required": false
         },
         "email": {
            "type": "String"
         },
         "tenant": {
            "type": "String",
            "required": true
         }
      }
   }
}
```

如需將對此結構描述進行驗證的範例政策，請參閱 反映 Amazon Cognito ID 字符屬性。

OIDC ID 字符

使用來自 OIDC 提供者的 ID 字符與使用 Amazon Cognito ID 字符大同小異。差異在於宣告。您的 IdP 可能會顯示標準 OIDC 屬性，或具有自訂結構描述。當您在 Verified Permissions 主控台中建立新的政策存放區時，您可以使用範例 ID 字符新增 OIDC 身分來源，或手動將字符宣告映射至使用者屬性。由於 Verified Permissions 不知道 IdP 的屬性結構描述，因此您必須提供此資訊。

如需詳細資訊，請參閱建立已驗證許可政策存放區。

以下是具有 OIDC 身分來源之政策存放區的範例結構描述。

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

```
         },
         "phone_number_verified": {
            "type": "Boolean"
         }
      }
   }
}
```

如需將對此結構描述進行驗證的範例政策，請參閱 反映 OIDC ID 字符屬性。

映射存取權杖

Verified Permissions 會處理存取金鑰宣告，而非群組宣告為 動作的屬性或內容屬性。除了群組成員資格之外，IdP 的存取權杖可能包含 API 存取的相關資訊。存取權杖在使用角色型存取控制 (RBAC) 的授權模型中很有用。依賴群組成員資格以外的存取金鑰宣告的授權模型需要額外的結構描述組態工作。

映射 Amazon Cognito 存取權杖

Amazon Cognito 存取權杖具有可用於授權的宣告：

Amazon Cognito 存取權杖中的實用宣告

client_id

OIDC 依賴方的用戶端應用程式 ID。使用用戶端 ID，已驗證許可可以驗證授權請求來自政策存放區的許可用戶端。在machine-to-machine(M2M) 授權中，請求系統會使用用戶端秘密授權請求，並提供用戶端 ID 和範圍做為授權證據。

scope

OAuth 2.0 範圍代表權杖持有人的存取許可。

cognito:groups

使用者的群組成員資格。在以角色為基礎的存取控制 (RBAC) 為基礎的授權模型中，此宣告會顯示您可以在政策中評估的角色。

暫時性宣告

不是存取許可，但由使用者集區預先產生字符 Lambda 觸發程序在執行時間新增的宣告。暫時性宣告類似於標準宣告，但超出標準範圍，例如 `tenant`或 `department`。自訂存取權杖會為您的 AWS 帳單增加成本。

如需 Amazon Cognito 使用者集區的存取字符結構的詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的使用存取字符。

Amazon Cognito 存取權杖在傳遞至 Verified Permissions 時，會對應至內容物件。您可以使用 來參考存取權杖的屬性`context.token.`*`attribute_name`*。下列範例存取字符包含 `client_id`和 `scope`宣告。

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

下列範例顯示如何反映 Verified Permissions 結構描述中存取字符範例的屬性。如需編輯結構描述的詳細資訊，請參閱 編輯政策存放區結構描述。

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

```
                ]
            }
         }
      },
      ...
      ...
      "commonTypes": {
         "ReusedContext": {
            "attributes": {
               "token": {
                  "type": "Record",
                  "attributes": {
                     "scope": {
                        "type": "Set",
                        "element": {
                           "type": "String"
                        }
                     },
                     "client_id": {
                        "type": "String"
                     }
                  }
               }
            },
            "type": "Record"
         }
      }
   }
}
```

如需將對此結構描述進行驗證的範例政策，請參閱 反映 Amazon Cognito 存取字符屬性。

映射 OIDC 存取權杖

來自外部 OIDC 提供者的大多數存取權杖都與 Amazon Cognito 存取權杖緊密一致。OIDC 存取權杖在傳遞至 Verified Permissions 時，會對應至內容物件。您可以使用 來參考存取權杖的屬性`context.token.`*`attribute_name`*。下列範例 OIDC 存取權杖包含基本宣告範例。

```
{
    "sub": "91eb4550-9091-708c-a7a6-9758ef8b6b1e",
    "groups": [
        "Store-Owner-Role",
        "Customer"
```

```
    ],
    "iss": "https://auth.example.com",
    "client_id": "1example23456789",
    "aud": "https://myapplication.example.com"
    "scope": "MyAPI-Read",
    "exp": 1688096566,
    "iat": 1688092966,
    "jti": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN2222222",
    "username": "alice"
}
```

下列範例示範如何反映 Verified Permissions 結構描述中存取字符範例的屬性。如需編輯結構描述的詳細資訊，請參閱 編輯政策存放區結構描述。

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

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

如需將對此結構描述進行驗證的範例政策，請參閱 反映 OIDC 存取權杖屬性。

Amazon Cognito 冒號分隔宣告的替代表示法

在 Verified Permissions 啟動時，Amazon Cognito 字符宣告的建議結構描述，例如 `cognito:groups`和 `custom:store`轉換這些冒號分隔字串，以使用 `.` 字元做為階層分隔符號。此格式稱為點表示法。例如， 的參考會在您的政策`principal.cognito.groups`中`cognito:groups`變成 。雖然您可以繼續使用此格式，但我們建議您建置具有括號表示法的結構描述和政策。在此格式中， 的參考會在您的政策`principal["cognito:groups"]`中`cognito:groups`變成 。從 Verified Permissions 主控台自動為使用者集區 ID 字符產生的結構描述會使用括號表示法。

您可以在手動建置的 Amazon Cognito 身分來源結構描述和政策中繼續使用點符號。您無法在結構描述`:`或政策中使用點符號搭配 或任何其他非英數字元，用於任何其他類型的 OIDC IdP。

點符號的結構描述會將`:`角色的每個執行個體巢狀化為 `cognito`或`custom`初始片語的子項，如下列範例所示：

```
"CognitoUser": {
   "shape": {
      "type": "Record",
      "attributes": {
         "cognito": {
            "type": "Record",
            "required": true,
            "attributes": {
               "username": {
                  "type": "String",
```

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

如需將對此結構描述進行驗證並使用點表示法的範例政策，請參閱 使用點表示法來參考屬性。

結構描述映射須知

屬性映射在字符類型之間有所不同

在存取權杖授權中，已驗證許可會將宣告對應至內容。在 ID 字符授權中，已驗證許可會將宣告對應至主體屬性。對於您在 Verified Permissions 主控台中建立的政策存放區，只有空白和範例政策存放區會讓您沒有身分來源，並要求您將 ID 字符授權的使用者集區屬性填入您的結構描述。存取權杖授權是以角色型存取控制 (RBAC) 搭配群組成員身分宣告為基礎，不會自動將其他宣告映射至政策存放區結構描述。

不需要身分來源屬性

當您在 Verified Permissions 主控台中建立身分來源時，不會將屬性標記為必要。這可防止遺失的宣告導致授權請求中的驗證錯誤。您可以視需要將屬性設定為必要，但這些屬性必須存在於所有授權請求中。

RBAC 不需要結構描述中的屬性

身分來源的結構描述取決於您在新增身分來源時建立的實體關聯。身分來源會將一個宣告對應至使用者實體類型，並將一個宣告對應至群組實體類型。這些實體映射是身分來源組態的核心。透過此最低資訊，您可以在角色型存取控制 (RBAC) 模型中撰寫政策，為使用者可能所屬的特定使用者和特定群組執行授權動作。在結構描述中新增權杖宣告可延長政策存放區的授權範圍。來自 ID 字符的使用者屬性具有有關使用者的資訊，這些使用者可以提供屬性型存取控制 (ABAC) 授權。存取字符的內容屬性具有 OAuth 2.0 範圍等資訊，可以提供來自提供者的其他存取控制資訊，但需要額外的結構描述修改。

Verified Permissions 主控台中的使用 API Gateway 和身分提供者設定和引導式設定選項會將 ID 字符宣告指派給結構描述。存取字符宣告的情況並非如此。若要將非群組存取金鑰宣告新增至您的結構描述，您必須在 JSON 模式中編輯結構描述，並新增 commonTypes 屬性。如需詳細資訊，請參閱映射存取權杖。

OIDC 群組宣告支援多種格式

當您新增 OIDC 提供者時，您可以選擇 ID 中的群組宣告名稱，或您要映射到政策存放區中使用者群組成員資格的存取權杖。已驗證的許可會以下列格式辨識群組宣告：

1. 不含空格的字串：`"groups": "MyGroup"`
2. 空格分隔清單：`"groups": "MyGroup1 MyGroup2 MyGroup3"`。每個字串都是一個群組。
3. JSON（逗號分隔）清單：`"groups": ["MyGroup1", "MyGroup2", "MyGroup3"]`

> **Note**
>
> Verified Permissions 會將空格分隔群組中的每個字串宣告解譯為個別群組。若要將具有空格字元的群組名稱解譯為單一群組，請取代或移除宣告中的空格。例如，`My Group`將名為 的群組格式化`MyGroup`。

選擇字符類型

您的政策存放區與身分來源搭配使用的方式取決於身分來源組態中的金鑰決策：您是否將處理 ID 或存取權杖。使用 Amazon Cognito 身分提供者，您可以在建立 API 連結政策存放區時選擇權杖類型。建立 API 連結政策存放區時，您必須選擇是否要設定 ID 或存取權杖的授權。此資訊會影響驗證許可套用至您政策存放區的結構描述屬性，以及 API Gateway API 的 Lambda 授權方語法。使用 OIDC 供應商時，您必須在新增身分來源時選擇字符類型。您可以選擇 ID 或存取權杖，而且您選擇的權杖類型不會在政策存放區中處理。特別是如果您想要從 ID 字符宣告自動映射到 Verified Permissions 主控台中的

屬性中受益，請在建立身分來源之前，儘早決定您要處理的字符類型。變更字符類型需要大量努力來重構您的政策和結構描述。下列主題說明搭配政策存放區使用 ID 和存取權杖。

Cedar 剖析器需要某些字元的括號

政策通常會參考 等格式的結構描述屬性principal.username。如果大多數非英數字元，例如 :、 .或 / 可能出現在字符宣告名稱中，驗證許可無法剖析 principal.cognito:username或等條件值context.ip-address。您必須改為將這些條件格式化為括號表示法context["ip-address"]，分別採用 格式principal["cognito:username"]或 。底線字元_是宣告名稱中的有效字元，也是此要求的唯一非英數字元例外狀況。

此類型主體屬性的部分範例結構描述如下所示：

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

此類型內容屬性的部分範例結構描述如下所示：

```
"GetOrder": {
   "memberOf": [],
   "appliesTo": {
      "resourceTypes": [
         "Order"
      ],
      "context": {
```

```
         "type": "Record",
         "attributes": {
            "ip-address": {
               "required": false,
               "type": "String"
            }
   }
   },
      "principalTypes": [
         "User"
      ]
   }
}
```

如需將對此結構描述進行驗證的範例政策，請參閱 使用括號表示法來參考字符屬性。

Amazon Verified Permissions 的整合

Amazon Verified Permissions 整合可協助您在應用程式中實作精細的授權，同時將程式碼降至最低並遵循架構特定的最佳實務。這些整合提供中介軟體元件和公用程式，可將您的應用程式與 Verified Permissions 無縫連線。

透過整合，您可以：

- 在幾分鐘內實作授權
- 遵循架構特定的模式和慣例
- 減少維護開銷
- 將潛在的安全實作錯誤降至最低
- 專注於商業邏輯而非授權碼

新增到應用程式時，整合會執行下列動作：

1. 透過架構特定的中介軟體攔截傳入請求
2. 從請求中擷取相關的授權內容
3. 使用 Verified Permissions 判斷授權決策
4. 根據授權結果強制執行存取控制

Verified Permissions 目前支援下列架構：

- 適用於 Node.js 應用程式的 Express.js

將 Express 與 Amazon Verified Permissions 整合

Verified Permissions Express 整合提供在 Express.js 應用程式中實作授權的中介軟體型方法。透過此整合，您可以使用精細的授權政策來保護 API 端點，而無需修改現有的路由處理常式。整合會透過攔截請求、針對您定義的政策進行評估，以及確保只有授權使用者可以存取受保護的資源，來自動處理授權檢查。

本主題會逐步引導您設定 Express 整合，從建立政策存放區到實作和測試授權中介軟體。遵循這些步驟，您可以將強大的授權控制新增至您的 Express 應用程式，並將程式碼變更降至最低。

本主題會參考下列GitHub儲存庫：

- cedar-policy/authorization-for-expressjs - Express.js 的 Cedar 授權中介軟體
- verifiedpermissions/authorization-clients-js - JavaScript 的 Verified Permissions 授權用戶端
- verifiedpermissions/examples/express-petstore - 使用 Express.js 中介軟體的範例實作

先決條件

實作 Express 整合之前，請確定您有：

- 可存取 Verified Permissions AWS 的帳戶
- 已安裝 Node.js 和 npm
- Express.js 應用程式
- OpenID Connect (OIDC) 身分提供者 （ 例如 Amazon Cognito)
- AWS CLI 已設定適當的許可

設定整合

步驟 1：建立政策存放區

使用 建立政策存放區 AWS CLI：

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

> **Note**
>
> 儲存回應中傳回的政策存放區 ID，以用於後續步驟。

步驟 2：安裝相依性

在 Express 應用程式中安裝必要的套件：

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

設定授權

步驟 1：產生和上傳 Cedar 結構描述

結構描述會定義應用程式的授權模型，包括應用程式中的實體類型，以及允許使用者採取的動作。建議您為結構描述定義命名空間。在此範例中，我們使用 YourNamespace. 您可以將結構描述連接至 Verified Permissions 政策存放區，並在新增或修改政策時，服務會自動針對結構描述驗證政策。

`@cedar-policy/authorization-for-expressjs` 套件可以分析應用程式的 OpenAPI 規格，並產生 Cedar 結構描述。具體而言，您的規格中需要路徑物件。

如果您沒有 OpenAPI 規格，您可以遵循 express-openapi-generator 套件的快速指示來產生 OpenAPI 規格。

從 OpenAPI 規格產生結構描述：

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

接著，格式化要與 搭配使用的 Cedar 結構描述 AWS CLI。如需所需特定格式的詳細資訊，請參閱 政策存放區結構描述。如果您需要格式化結構描述的說明，在已驗證許可/範例GitHub儲存庫`prepare-cedar-schema.sh`中會有一個名為 的指令碼。 https://github.com/verifiedpermissions/examples/tree/main/express-petstore/start/scripts以下是對該指令碼的呼叫範例，該指令碼會在 `v2.cedarschema.forAVP.json` 檔案中輸出 Verified Permissions 格式的結構描述。

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

將格式化的結構描述上傳至您的政策存放區，`policy-store-id`以您的政策存放區 ID 取代 ：

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

步驟 2：建立授權政策

如果未設定任何政策，Cedar 會拒絕所有授權請求。Express 架構整合可根據先前產生的結構描述產生範例政策，協助引導此程序。

在生產應用程式中使用此整合時，我們建議您使用基礎設施做為程式碼 (IaaC) 工具建立新的政策。如需詳細資訊，請參閱使用 AWS CloudFormation。

產生範例 Cedar 政策：

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

這將在 /policies目錄中產生範例政策。然後，您可以根據您的使用案例自訂這些政策。例如：

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

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

格式化要與 搭配使用的政策 AWS CLI。如需所需格式的詳細資訊，請參閱 AWS CLI 參考中的 create-policy。如果您需要格式化政策的說明，在已驗證許可/範例GitHub儲存庫convert_cedar_policies.sh中會有一個名為 的指令碼。 https://github.com/verifiedpermissions/examples/tree/main/express-petstore/start/scripts以下是對該指令碼的呼叫：

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

將格式化政策上傳至 Verified Permissions，將 取代policy_1.json為政策檔案的路徑和名稱，並將policy-store-id取代為政策存放區 ID：

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

步驟 3：連接身分提供者

根據預設，Verified Permissions 授權方中介軟體會讀取 API 請求的授權標頭中提供的 JSON Web Token (JWT)，以取得使用者資訊。除了執行授權政策評估之外，已驗證許可還可以驗證權杖。

使用 `userPoolArn`和 建立名為 的身分來源組態檔案`identity-source-configuration.txt`，如下所示`clientId`：

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

執行下列 AWS CLI 命令來建立身分來源，將 取代`policy-store-id`為您的政策存放區 ID：

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

實作授權中介軟體

更新您的 Express 應用程式以包含授權中介軟體。在此範例中，我們使用身分字符，但您也可以使用存取字符。如需詳細資訊，請參閱 上的 authorization-for-expressjsGitHub。

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

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

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

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

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

測試整合

您可以使用不同的使用者字符向 API 端點提出請求，以測試授權實作。授權中介軟體會根據您定義的政策自動評估每個請求。

例如，如果您已設定具有不同許可的不同使用者群組：

- 管理員：完整存取所有資源和管理函數
- 員工：可以檢視、建立和更新資源
- 客戶：只能檢視資源

您可以向不同的使用者登入並嘗試各種操作，以驗證許可政策是否如預期般運作。在 Express 應用程式的終端機中，您可以看到提供授權決策其他詳細資訊的日誌輸出。

故障診斷

如果您有授權失敗，請嘗試下列動作：

- 驗證您的政策存放區 ID 是否正確
- 確保您的身分來源已正確設定
- 檢查您的政策格式是否正確
- 驗證您的 JWT 權杖是否有效

後續步驟

實作基本整合之後，請考慮：

- 針對特定授權案例實作自訂映射器
- 設定授權決策的監控和記錄
- 為不同的使用者角色建立其他政策

在 Amazon Verified Permissions 中實作授權

在您建置政策存放區、政策、範本、結構描述和授權模型之後，您就可以開始使用 Amazon Verified Permissions 來授權請求。若要實作 Verified Permissions 授權，您必須將 中的授權政策組態 AWS 與應用程式的整合結合。若要將 Verified Permissions 與您的應用程式整合，請新增 AWS SDK，並實作叫用 Verified Permissions API 的方法，並針對您的政策存放區產生授權決策。

Verified Permissions 的授權對於應用程式中的 UX 許可和 API 許可很有用。

UX 許可

控制使用者存取您的應用程式 UX。您可以允許使用者只檢視他們需要存取的確切表單、按鈕、圖形和其他資源。例如，當使用者登入時，您可能想要判斷他們的帳戶中是否顯示「轉移資金」按鈕。您也可以控制使用者可採取的動作。例如，在相同的銀行應用程式中，您可能想要判斷您的使用者是否被允許變更交易的類別。

API 許可

控制使用者對資料的存取。應用程式通常是分散式系統的一部分，並從外部 APIs引入資訊。在 Verified Permissions 允許顯示「轉移資金」按鈕的銀行應用程式中，當您的使用者啟動轉移時，必須做出更複雜的授權決策。已驗證許可可以授權 API 請求，該請求會列出符合資格的轉移目標目的地帳戶，然後請求將轉移推送到另一個帳戶。

說明此內容的範例來自範例政策存放區。若要遵循，請在測試環境中建立 DigitalPetStore 範例政策存放區。

如需使用批次授權實作 UX 許可的端對端範例應用程式，請參閱AWS 安全部落格上的使用 Amazon Verified Permissions 進行大規模精細授權。

主題

- 授權可用的 API 操作
- 測試您的授權模型
- 整合您的授權模型與應用程式

授權可用的 API 操作

Verified Permissions API 具有下列授權操作。

IsAuthorized

IsAuthorized API 操作是使用 Verified Permissions 授權請求的進入點。您必須提交主體、動作、資源、內容和實體元素。Verified Permissions 會根據您的政策存放區結構描述，驗證請求中的實體。然後，驗證許可會根據請求的政策存放區中套用至請求中實體的所有政策來評估您的請求。

IsAuthorizedWithToken

IsAuthorizedWithToken 操作會從 JSON Web 字符 (JWTs中的使用者資料產生授權請求。Verified Permissions 可直接與 OIDC 提供者搭配使用，例如 Amazon Cognito，作為政策存放區中的身分來源。Verified Permissions 會將請求中的所有屬性填入使用者 ID 或存取權杖中宣告的主體。您可以從身分來源中的使用者屬性或群組成員資格授權動作和資源。

您無法在IsAuthorizedWithToken請求中包含群組或使用者主體類型的相關資訊。您必須將所有主體資料填入您提供的 JWT。

BatchIsAuthorized

BatchIsAuthorized 操作會針對單一 API 請求中的單一主體或資源處理多個授權決策。此操作會將請求分組為單一批次操作，可將配額用量降至最低，並傳回最多 30 個複雜巢狀動作的授權決策。透過單一資源的批次授權，您可以篩選使用者可以對資源採取的動作。透過單一主體的批次授權，您可以篩選使用者可以採取動作的資源。

BatchIsAuthorizedWithToken

BatchIsAuthorizedWithToken 操作會在單一 API 請求中處理單一主體的多個授權決策。委託人是由您的政策存放區身分來源在 ID 或存取權杖中提供。此操作會將請求分組為單一批次操作，可將配額用量降至最低，並針對動作和資源的每一個請求傳回授權決策。在您的政策中，您可以從其屬性或使用者目錄中的群組成員資格授權其存取權。

如同 IsAuthorizedWithToken，您無法在BatchIsAuthorizedWithToken請求中包含群組或使用者主體類型的相關資訊。您必須將所有主體資料填入您提供的 JWT。

測試您的授權模型

若要了解部署應用程式時 Amazon Verified Permissions 授權決策的影響，您可以在使用 使用 Amazon Verified Permissions 測試工作台和 HTTPS REST API 請求對 Verified Permissions 開發政策時評估政策。測試工作台是 中的工具 AWS Management Console ，用於評估政策存放區中的授權請求和回應。

Verified Permissions REST API 是您從概念理解到應用程式設計的下一個開發步驟。Verified Permissions API 接受具有 IsAuthorized、IsAuthorizedWithToken 和 BatchIsAuthorized 的授權請求，

做為區域服務端點的簽章 AWS API 請求。若要測試您的授權模型，您可以使用任何 API 用戶端產生請求，並驗證您的政策是否如預期傳回授權決策。

例如，您可以使用下列程序在範例政策存放`IsAuthorized`區中測試 。

Test bench

1. 在 Verified Permissions 主控台開啟 Verified Permissions 主控台。從名為 DigitalPetStore 的範本政策存放區建立政策存放區。
2. 選取新政策存放區中的測試工作台。
3. 在 Verified Permissions API 參考中，從 IsAuthorized 填入您的測試台請求。下列詳細資訊會複寫範例 4 中參考 DigitalPetStore 範例的條件。

   a. 將 Alice 設定為主體。針對主體採取動作，選擇 `DigitalPetStore::User`並輸入 `Alice`。

   b. 將 Alice 的角色設定為客戶。選擇新增父系，選擇 `DigitalPetStore::Role`，然後輸入客戶。

   c. 將資源設定為順序 "1234"。對於委託人正在執行的資源，選擇 `DigitalPetStore::Order`並輸入 `1234`。

   d. `DigitalPetStore::Order` 資源需要 `owner` 屬性。將 Alice 設定為訂單的擁有者。選擇`DigitalPetStore::User`並輸入 `Alice`

   e. Alice 請求檢視訂單。針對委託人正在採取的動作，選擇 `DigitalPetStore::Action::"GetOrder"`。
4. 選擇執行授權請求。在未修改的政策存放區中，此請求會導致 `ALLOW`決策。請注意傳回決策的滿意政策。
5. 從左側導覽列中選擇政策。檢閱靜態政策並說明客戶角色 - 取得訂單。
6. 觀察 Verified Permissions 允許請求，因為主體是客戶角色，並且是資源的擁有者。

REST API

1. 在 Verified Permissions 主控台開啟 Verified Permissions 主控台。從名為 DigitalPetStore 的範本政策存放區建立政策存放區。
2. 請注意新政策存放區的政策存放區 ID。
3. 在 Verified Permissions API 參考中的 IsAuthorized 中，複製參考 DigitalPetStore 範例的範例 4 請求內文。

4. 開啟您的 API 用戶端，並為政策存放區建立區域服務端點的請求。填入標頭，如範例所示。
5. 在範例請求內文中貼上 ，並將 的值變更為您先前記下policyStoreId的政策存放區 ID。
6. 提交請求並檢閱結果。在預設 DigitalPetStore 政策存放區中，此請求會傳回 ALLOW 決策。

您可以在測試環境中變更政策、結構描述和請求，以變更結果並產生更複雜的決策。

1. 變更請求的方式會變更驗證許可的決策。例如，將 Alice 的角色變更為 Employee，或將順序 owner 1234 的屬性變更為 Bob。
2. 以影響授權決策的方式變更政策。例如，使用描述客戶角色 - 取得訂單來修改政策，以移除 User 必須是 擁有者的條件，Resource並修改請求，以便 Bob想要檢視訂單。
3. 變更結構描述，以允許政策做出更複雜的決策。更新請求實體，讓 Alice 可以滿足新的要求。例如，編輯結構描述User以允許 成為 ActiveUsers或 的成員InactiveUsers。更新政策，以便只有作用中的使用者才能檢視自己的訂單。更新請求實體，讓 Alice 成為作用中或非作用中的使用者。

整合您的授權模型與應用程式

若要在應用程式中實作 Amazon Verified Permissions，您必須定義您希望應用程式強制執行的政策和結構描述。在您的授權模型就地並經過測試後，下一步是從強制執行點開始產生 API 請求。若要這樣做，您必須設定應用程式邏輯來收集使用者資料，並將其填入授權請求。

應用程式如何使用 Verified Permissions 授權請求

1. 收集目前使用者的相關資訊。一般而言，使用者的詳細資訊會在已驗證工作階段的詳細資訊中提供，例如 JWT 或 Web 工作階段 Cookie。此使用者資料可能來自連結至政策存放區的 Amazon Cognito 身分來源，或來自其他 OpenID Connect (OIDC) 供應商。
2. 收集使用者想要存取的資源相關資訊。一般而言，當使用者選擇需要您的應用程式載入新資產時，您的應用程式會收到資源的相關資訊。
3. 判斷您的使用者想要採取的動作。
4. 使用使用者嘗試操作的委託人、動作、資源和實體來產生驗證許可的授權請求。驗證許可會根據政策存放區中的政策評估請求，並傳回授權決策。
5. 您的應用程式會從 Verified Permissions 讀取允許或拒絕回應，並強制執行使用者請求的決定。

Verified Permissions API 操作已內建於 AWS SDKs中。若要在應用程式中包含 Verified Permissions，請將所選語言的 AWS SDK 整合到應用程式套件中。

若要進一步了解 和 AWS SDKs，請參閱適用於 的工具 Amazon Web Services。

以下是各種 AWS SDKs 中已驗證許可資源的文件連結。

- 適用於 .NET 的 AWS SDK
- 適用於 C++ 的 AWS SDK
- 適用於 Go 的 AWS SDK
- 適用於 Java 的 AWS SDK
- 適用於 JavaScript 的 AWS SDK
- 適用於 PHP 的 AWS SDK
- AWS SDK for Python (Boto)
- 適用於 Ruby 的 AWS SDK
- 適用於 Rust 的 AWS SDK

的下列 適用於 JavaScript 的 AWS SDK 範例`IsAuthorized`源自使用 Amazon Verified Permissions 和 Amazon Cognito 簡化精細授權。

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

```
});
```

更多開發人員資源

- Amazon Verified Permissions 研討會
- Amazon Verified Permissions - 資源
- 使用 Amazon Verified Permissions 實作 ASP.NET Core 應用程式的自訂授權政策提供者
- 使用 Amazon Verified Permissions 為商業應用程式建置權利服務
- 使用 Amazon Verified Permissions 和 Amazon Cognito 簡化精細授權

Amazon Verified Permissions 中的安全性

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構是為了滿足最安全敏感組織的需求而建置。

安全性是 AWS 和 之間的共同責任。共同責任模型將其描述為雲端的安全性和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 中執行 AWS 服務的基礎設施 AWS 雲端。 AWS 也提供您可以安全使用的服務。第三方稽核人員會定期測試和驗證我們安全的有效性，做為AWS 合規計畫的一部分。若要了解適用於 Amazon Verified Permissions 的合規計劃，請參閱AWS 合規計劃範圍內的服務。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您了解如何在使用 Verified Permissions 時套用共同責任模型。下列主題說明如何設定 Verified Permissions 以符合您的安全和合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 Verified Permissions 資源。

主題

- Amazon Verified Permissions 中的資料保護
- Amazon Verified Permissions 的身分和存取管理
- Amazon Verified Permissions 的合規驗證
- Amazon Verified 許可中的彈性

Amazon Verified Permissions 中的資料保護

AWS 共同責任模型適用於 Amazon Verified Permissions 中的資料保護。如此模型所述， AWS 負責保護執行所有 AWS 雲端的全球基礎設施。您負責維護在此基礎設施上託管內容的控制權。此內容包含 AWS 服務 您使用之 的安全組態和管理任務。如需有關資料隱私權的更多相關資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

- 基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management () 設定個別使用者IAM。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。

- 建議您以下列方式保護您的資料：
  - 每個帳戶均要使用多重要素驗證 (MFA)。
  - 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2。
  - 使用 設定 API 和使用者活動記錄 AWS CloudTrail。
  - 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
  - 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
  - 如果您在透過命令列介面或 API 存取 AWS 時，需要 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。
- 我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 Verified Permissions 或其他 AWS 服務 使用主控台、API AWS CLI或 AWS SDKs。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。
- 您的動作名稱不應包含任何敏感資訊。
- 我們也強烈建議您一律為實體（資源和主體）使用唯一、不可變和不可重複使用的識別符。在測試環境中，您可以選擇使用簡單的實體識別符，例如 `jane`或 bob 做為類型 之實體的名稱`User`。不過，在生產系統中，基於安全考量，您使用無法重複使用的唯一值至關重要。我們建議您使用通用唯一識別碼 (UUIDs等值。例如，請考慮離開公司的`jane`使用者。稍後，您會讓其他人使用名稱 `jane`。該新使用者會自動存取仍參考 的政策授予的所有內容`User::"jane"`。已驗證的許可和 Cedar 無法區分新使用者和先前的使用者。

  本指南同時適用於主體和資源識別符。一律使用保證唯一且永遠不會重複使用的識別符，以確保您不會因為政策中存在舊識別符而意外授予存取權。

- 請確定您提供用來定義 `Long`和 `Decimal`值的字串，在每種類型的有效範圍內。此外，請確保您使用任何算術運算子不會導致值超出有效範圍。如果超過範圍，操作會導致溢位例外狀況。導致錯誤的政策會被忽略，這表示許可政策可能會意外地無法允許存取，或禁止政策可能會意外地無法封鎖存取。

資料加密

Amazon Verified Permissions 會使用 自動加密所有客戶資料，例如政策 AWS 受管金鑰，因此不需要也不支援使用客戶受管金鑰。

Amazon Verified Permissions 的身分和存取管理

AWS Identity and Access Management (IAM) 是 AWS 服務 ，可協助管理員安全地控制對 AWS resources 的存取。 IAM 管理員可控制誰可以經過身分驗證 （登入） 和授權 （具有許可） 來使用 Verified Permissions 資源。 IAM 是 AWS 服務 您可以免費使用的 。

主題

- 目標對象
- 使用身分驗證
- 使用政策管理存取權
- Amazon Verified Permissions 如何使用 IAM
- IAM Verified Permissions 的 政策
- Amazon Verified Permissions 的身分型政策範例
- AWS Amazon Verified Permissions 的 受管政策
- 對 Amazon Verified Permissions 身分和存取進行故障診斷

目標對象

使用方式 AWS Identity and Access Management (IAM) 會有所不同，取決於您在 Verified Permissions 中執行的工作。

服務使用者 – 如果您使用 Verified Permissions 服務來執行您的任務，則您的管理員會為您提供所需的登入資料和許可。當您使用更多 Verified Permissions 功能來執行工作時，您可能需要額外的許可。了解存取許可的管理方式可協助您向管理員請求正確的許可。如果您無法存取 Verified Permissions 中的功能，請參閱 對 Amazon Verified Permissions 身分和存取進行故障診斷。

服務管理員 – 如果您在公司負責 Verified Permissions 資源，您可能擁有 Verified Permissions 的完整存取權。您的任務是判斷服務使用者應存取哪些 Verified Permissions 功能和資源。然後，您必須向 IAM 管理員提交請求，以變更服務使用者的許可。檢閱此頁面上的資訊，以了解 的基本概念 IAM。若要進一步了解貴公司如何 IAM 搭配 Verified Permissions 使用 ，請參閱 Amazon Verified Permissions 如何使用 IAM。

IAM 管理員 – 如果您是 IAM 管理員，建議您了解如何撰寫政策以管理 Verified Permissions 存取的詳細資訊。若要檢視您可以在 中使用的 Verified Permissions 身分型政策範例 IAM，請參閱 Amazon Verified Permissions 的身分型政策範例。

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以身分驗證 （ 登入 AWS) 做為 AWS 帳戶根使用者、以 IAM 使用者身分驗證，或擔任 IAM 角色。

您可以使用透過身分來源提供的憑證，以聯合身分 AWS 身分身分身分登入 。 AWS IAM Identity Center (IAM Identity Center) 使用者、您公司的單一登入身分驗證，以及您的 Google 或 Facebook 登入資料，都是聯合身分的範例。當您以聯合身分身分登入時，您的管理員先前會使用 IAM 角色設定聯合身分。當您使用聯合 AWS 身分存取 時，您會間接擔任角色。

根據您身分的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需登入的詳細資訊 AWS，請參閱AWS 登入 《 使用者指南》中的如何登入您的 AWS 帳戶 。

如果您以 AWS 程式設計方式存取 ， AWS 會提供軟體開發套件 (SDK) 和命令列界面 (CLI)，以使用您的 憑證以密碼編譯方式簽署您的請求。如果您不使用 AWS 工具，則必須自行簽署請求。如需使用建議方法自行簽署請求的詳細資訊，請參閱IAM 《 使用者指南》中的 AWS API 請求的簽章版本 4。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如， AWS 建議您使用多重要素驗證 (MFA) 來提高帳戶的安全性。若要進一步了解，請參閱AWS IAM Identity Center 《 使用者指南》中的多重要素驗證和》 IAM 使用者指南》AWS 中的多重要素驗證 IAM。

AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個登入身分開始，該身分可完整存取 帳戶中的所有 AWS 服務 和資源。此身分稱為 AWS 帳戶 Theroot 使用者，可透過使用您用來建立帳戶的電子郵件地址和密碼登入來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需需要您以根使用者身分登入的任務完整清單，請參閱IAM 《 使用者指南》中的需要根使用者憑證的任務。

聯合身分

最佳實務是， 要求人類使用者，包括需要管理員存取權的使用者，使用臨時登入資料 AWS 服務 來使用與身分提供者的聯合來存取 。

聯合身分是來自您企業使用者目錄、Web 身分提供者、 AWS Directory Service、身分中心目錄，或 AWS 服務 是透過身分來源提供的登入資料存取的任何使用者。當聯合身分存取時 AWS 帳戶，它們會擔任 角色，而角色會提供臨時登入資料。

對於集中式存取權管理，我們建議您使用 AWS IAM Identity Center。您可以在 IAM Identity Center 中建立使用者和群組，或者您可以連接並同步到您自己的身分來源中的一組使用者 AWS 帳戶 和群

組，以便在所有 和應用程式中使用。如需 IAM Identity Center 的詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

IAM 使用者和群組

IAM 使用者是 中具有單一個人或應用程式特定許可 AWS 帳戶 的身分。建議您盡可能依賴臨時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需詳細資訊，請參閱IAM 《 使用者指南》中的針對需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是指定 IAM 使用者集合的身分。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為 IAM Admin 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供臨時憑證。若要進一步了解，請參閱IAM 《 使用者指南》中的IAM 使用者使用案例。

IAM 角色

IAM 角色是 中具有特定許可 AWS 帳戶 的身分。它類似 IAM 使用者，但不與特定的人員相關聯。您可以 AWS Management Console 切換 IAM 角色，暫時在 中擔任 角色。您可以透過呼叫 AWS CLI 或 AWS API 操作或使用自訂 URL 來擔任角色。如需使用角色方法的詳細資訊，請參閱IAM 《 使用者指南》中的使用 IAM 角色。

IAM 具有臨時登入資料的 角色在下列情況中很有用：

- 聯合身分使用者存取 — 如需向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需聯合角色的相關資訊，請參閱IAM 《 使用者指南》中的為第三方身分提供者（聯合）建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與IAM中的角色相關聯。如需有關許可集的資訊，請參閱 AWS IAM Identity Center 使用者指南中的許可集。
- 臨時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色，暫時接受特定任務的不同許可。
- 跨帳戶存取權 – 您可以使用 IAM 角色，允許不同帳戶中的某人 (受信任的主體) 存取您帳戶的資源。角色是授予跨帳戶存取權的主要方式。不過，對於某些 AWS 服務，您可以直接將政策連接到資源（而不是使用角色做為代理）。若要了解跨帳戶存取的角色和資源型政策之間的差異，請參閱IAM 《 使用者指南》中的IAM 角色與資源型政策的差異。

- 執行於 的應用程式 Amazon EC2 – 您可以使用 IAM 角色來管理在 EC2 執行個體上執行之應用程式的臨時登入資料，以及提出 AWS CLI 或 AWS API 請求。這是在 EC2 執行個體內儲存存取金鑰的較好方式。若要將 AWS 角色指派給 EC2 執行個體並將其提供給其所有應用程式，您可以建立連接至執行個體的執行個體描述檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得臨時憑證。如需詳細資訊，請參閱IAM 《 使用者指南》中的使用 IAM 角色將許可授予在 Amazon EC2 執行個體上執行的應用程式。

若要了解如何使用 IAM 角色或 IAM 使用者，請參閱IAM 《 使用者指南》中的何時建立 IAM 角色 （ 而非使用者 ）。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策是 中的物件， AWS 當與身分或資源建立關聯時， 會定義其許可。當委託人 （ 使用者、根使用者或角色工作階段 ） 發出請求時， 會 AWS 評估這些政策。政策中的許可決定是否允許或拒絕請求。大多數政策會以 JSON 文件 AWS 的形式存放在 中。如需 JSON 政策文件結構和內容的詳細資訊，請參閱IAM 《 使用者指南》中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授予使用者對所需資源執行動作的許可， IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者可以擔任角色。

IAM 無論您用來執行操作的方法為何， 政策都會定義動作的許可。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該政策的使用者可以從 AWS Management Console AWS CLI、 或 API AWS 取得角色資訊。

身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分型政策，請參閱IAM 《 使用者指南》中的使用客戶受管政策定義自訂 IAM 許可。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管政策是獨立的政策，您可以連接到 中的多個使用者、群組和角色 AWS 帳戶。受管政策包括 AWS 受管政策和客戶受管政策。若要了解如何在受管政策或內嵌政策之間進行選擇，請參閱IAM 《 使用者指南》中的在受管政策和內嵌政策之間進行選擇。

資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的範例包括 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主體可以包含帳戶、使用者、角色、聯合身分使用者，或 AWS 服務。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策 IAM 中使用來自 的 AWS 受管政策。

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon S3 AWS WAF和 Amazon VPC 是支援 ACLs的服務範例。如需進一步了解 ACL，請參閱 Amazon Simple Storage Service 開發人員指南中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較不常見的政策類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 許可界限 – 許可界限是一種進階功能，您可以在其中設定身分型政策可授予 IAM 實體 (IAM 使用者或角色） 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可界限的詳細資訊，請參閱IAM 《 使用者指南》中的IAM 實體的許可界限。
- 服務控制政策 SCPs) – SCPs是 JSON 政策，可指定 中組織或組織單位 (OU) 的最大許可 AWS Organizations。 AWS Organizations 是用於分組和集中管理您企業擁有 AWS 帳戶 之多個的服務。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 會限制成員帳戶中實體的許可，包括每個實體 AWS 帳戶根使用者。如需 Organizations 和 SCP 的詳細資訊，請參閱《AWS Organizations 使用者指南》中的服務控制政策。
- 資源控制政策 RCPs) – RCPs是 JSON 政策，您可以使用它來設定帳戶中資源的可用許可上限，而無需更新連接到您擁有的每個資源 IAM 的政策。RCP 會限制成員帳戶中資源的許可，並可能影響身分的有效許可，包括 AWS 帳戶根使用者，無論它們是否屬於您的組織。如需 Organizations 和 RCPs的詳細資訊，包括 AWS 服務 支援 RCPs的 清單，請參閱AWS Organizations 《 使用者指南》中的資源控制政策 RCPs)。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過撰寫程式的方式建立角色或聯合使用者的暫時工作階段時，做為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作

階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需詳細資訊，請參閱 IAM 使用者指南中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。若要了解如何 AWS 在涉及多種政策類型時決定是否允許請求，請參閱 IAM 使用者指南中的政策評估邏輯。

Amazon Verified Permissions 如何使用 IAM

在您使用 IAM 管理 Verified Permissions 的存取權之前，請先了解哪些 IAM 功能可與 Verified Permissions 搭配使用。

IAM 您可以搭配 Amazon Verified Permissions 使用的功能

IAM 功能	已驗證許可支援
身分型政策	是
資源型政策	否
政策動作	是
政策資源	是
政策條件索引鍵	否
ACL	否
ABAC (政策中的標籤)	是
臨時憑證	是
主體許可	是
服務角色	否
服務連結角色	否

若要全面了解 Verified Permissions 和其他 AWS 服務如何與大多數 IAM 功能搭配使用，請參閱IAM 《 使用者指南》中的 AWS 服務 IAM。

已驗證許可的身分型政策

支援身分型政策	是

身分型政策是可以連接到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分型政策，請參閱IAM 《 使用者指南》中的使用客戶受管政策定義自訂 IAM 許可。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及允許或拒絕動作的條件。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。若要了解您可以在 JSON 政策中使用的所有元素，請參閱IAM 《 使用者指南》中的 IAM JSON 政策元素參考。

Verified Permissions 的身分型政策範例

若要檢視 Verified Permissions 身分型政策的範例，請參閱 Amazon Verified Permissions 的身分型政策範例。

Verified Permissions 中的資源型政策

支援以資源基礎的政策	否

資源型政策是附加到資源的 JSON 政策文件。資源型政策的範例包括 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

若要啟用跨帳戶存取，您可以將另一個帳戶中的整個帳戶或 IAM 實體指定為資源型政策中的委託人。新增跨帳戶主體至資源型政策，只是建立信任關係的一半。當委託人和資源位於不同的 時 AWS 帳戶，信任帳戶中的 IAM 管理員也必須授予委託人實體 （使用者或角色） 存取資源的許可。其透過將身分型政策連接到實體來授與許可。不過，如果資源型政策會為相同帳戶中的主體授予存取，這時就不需要額外的身分型政策。如需詳細資訊，請參閱IAM 《 使用者指南》中的 中的跨帳戶資源存取 IAM。

Verified Permissions 的政策動作

支援政策動作	是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼条件下可以對什麼資源執行哪些動作。

JSON 政策的 `Action` 元素描述您可以用來允許或拒絕政策中存取的動作。政策動作通常具有與相關聯 AWS API 操作相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

若要查看 Verified Permissions 動作的清單，請參閱《服務授權參考》中的 Amazon Verified Permissions 定義的動作。

Verified Permissions 中的政策動作在動作之前使用以下字首：

```
verifiedpermissions
```

如需在單一陳述式中指定多個動作，請用逗號分隔。

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

您也可以使用萬用字元 (*) 來指定多個動作。例如，若要指定開頭是 `Get` 文字的所有動作，請包含以下動作：

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

若要檢視 Verified Permissions 身分型政策的範例，請參閱 Amazon Verified Permissions 的身分型政策範例。

Verified Permissions 的政策資源

支援政策資源	是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Resource JSON 政策元素可指定要套用動作的物件。陳述式必須包含 Resource 或 NotResource 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

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

若要查看 Verified Permissions 資源類型及其 ARNs的清單，請參閱《服務授權參考》中的 Amazon Verified Permissions 定義的資源類型。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 Amazon Verified Permissions 定義的動作。

Verified Permissions 的政策條件索引鍵

支援服務特定政策條件金鑰	否

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Condition 元素 (或 Condition 區塊) 可讓您指定使陳述式生效的條件。Condition 元素是選用項目。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。

若您在陳述式中指定多個 Condition 元素，或是在單一 Condition 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值， 會使用邏輯OR操作 AWS 評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需詳細資訊，請參閱IAM 《 使用者指南》中的IAM 政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定條件金鑰。若要查看所有 AWS 全域條件索引鍵，請參閱IAM 《 使用者指南》中的AWS 全域條件內容索引鍵。

已驗證許可中的 ACLs

支援 ACL	否

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

具有已驗證許可的 ABAC

支援 ABAC (政策中的標籤)	是

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在 中 AWS，這些屬性稱為標籤。您可以將標籤連接到 IAM 實體 （使用者或角色） 和許多 AWS 資源。為實體和資源加上標籤是 ABAC 的第一步。您接著要設計 ABAC 政策，允許在主體的標籤與其嘗試存取的資源標籤相符時操作。

ABAC 在成長快速的環境中相當有幫助，並能在政策管理變得繁瑣時提供協助。

如需根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的條件元素中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱IAM 《 使用者指南》中的使用 ABAC 授權定義許可。若要檢視包含設定 ABAC 步驟的教學課程，請參閱IAM 《 使用者指南》中的使用屬性型存取控制 (ABAC)。

搭配 Verified Permissions 使用臨時憑證

支援臨時憑證	是

當您使用臨時登入資料登入時，有些 AWS 服務 無法運作。如需其他資訊，包括哪些 AWS 服務 使用臨時登入資料，請參閱IAM 《 使用者指南》中的AWS 服務 使用 IAM 。

如果您 AWS Management Console 使用使用者名稱和密碼以外的任何方法登入 ，則會使用臨時登入資料。例如，當您 AWS 使用公司的單一登入 (SSO) 連結存取 時，該程序會自動建立臨時登入資料。當您以使用者身分登入主控台，然後切換角色時，也會自動建立臨時憑證。如需切換角色的詳細資訊，請參閱IAM 《 使用者指南》中的從使用者切換到 IAM 角色 （主控台）。

您可以使用 AWS CLI 或 AWS API 手動建立臨時登入資料。然後，您可以使用這些臨時登入資料來存取 AWS。 AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時性安全憑證。

已驗證許可的跨服務主體許可

支援主體許可	是

當您使用 IAM 使用者或角色在 中執行動作時 AWS，您會被視為委託人。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 使用呼叫 的委託人許可 AWS 服務，結合 AWS 服務 請求向下游服務提出請求。只有當服務收到需要與其他 AWS 服務 或 資源互動才能完成的請求時，才會提出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱轉發存取工作階段。

Verified Permissions 的服務角色

支援服務角色	否

服務角色是服務擔任以代表您執行動作IAM 的角色。 IAM 管理員可以從內部建立、修改和刪除服務角色 IAM。如需詳細資訊，請參閱IAM 《 使用者指南》中的建立角色以將許可委派給 AWS 服務。

Verified Permissions 的服務連結角色

支援服務連結角色。	否

服務連結角色是連結至 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。 IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需建立或管理服務連結角色的詳細資訊，請參閱 AWS 使用的服務 IAM。在表格中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

IAM Verified Permissions 的 政策

Verified Permissions 會管理應用程式中使用者的許可。為了讓您的應用程式呼叫 Verified Permissions APIs 或 AWS Management Console 允許使用者在 Verified Permissions 政策存放區中管理 Cedar 政策，您必須新增必要的 IAM 許可。

身分型政策是您可以連接到身分的 JSON 許可政策文件，例如 IAM 使用者、使用者群組或角色。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立以身分為基礎的政策，請參閱 IAM 《使用者指南》中的建立 IAM 政策。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及允許或拒絕動作的條件（如下所示）。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。若要了解您可以在 JSON 政策中使用的所有元素，請參閱 IAM 《使用者指南》中的 IAM JSON 政策元素參考。

Action	Description
CreateIdentitySource	建立新身分來源的動作。
CreatePolicy	在政策存放區中建立 Cedar 政策的動作。您可以建立靜態政策或連結至政策範本的政策。
CreatePolicyStore	建立新政策存放區的動作。
CreatePolicyTemplate	建立新政策範本的動作。
DeleteIdentitySource	刪除身分來源的動作。
DeletePolicy	從政策存放區刪除政策的動作。
DeletePolicyStore	刪除政策存放區的動作。
DeletePolicyTemplate	刪除政策範本的動作。
GetIdentitySource	取得身分來源的動作。
GetPolicy	擷取指定政策相關資訊的動作。
GetPolicyStore	擷取指定政策存放區相關資訊的動作。
GetPolicyTemplate	取得政策範本的動作。
GetSchema	取得結構描述的動作。
IsAuthorized	根據授權請求中所述的參數取得授權回應的動作。 ???

Action	Description
IsAuthorizedWithToken	根據委託人來自身分字符的授權請求中所述的參數，取得授權回應的動作。
ListIdentitySources	在 中列出所有身分來源的動作 AWS 帳戶。
ListPolicies	列出政策存放區中所有政策的動作。
ListPolicyStores	列出 中所有政策存放區的動作 AWS 帳戶。
ListPolicyTemplates	在 中列出所有政策範本的動作 AWS 帳戶。
ListTagsForResource	列出資源所有標籤的動作。
PutSchema	將結構描述新增至政策存放區的動作。
TagResource	將標籤新增至資源的動作。
UpdateIdentitySource	更新身分來源的動作。
UpdatePolicy	在政策存放區中更新政策的動作。
UpdatePolicyStore	更新政策存放區的動作。
UpdatePolicyTemplate	更新政策範本的動作。
UntagResource	從資源移除標籤的動作。

CreatePolicy 動作的許可範例 IAM 政策：

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

```
    ]
}
```

Amazon Verified Permissions 的身分型政策範例

根據預設，使用者和角色沒有建立或修改已驗證許可資源的許可。他們也無法使用 AWS Management Console、 AWS Command Line Interface (AWS CLI) 或 AWS API 來執行任務。 IAM 管理員必須建立 IAM 政策，授予使用者和角色對所需資源執行動作的許可。管理員接著必須將這些政策連接至需要這些許可的使用者。

若要了解如何使用這些範例 JSON 政策文件來建立以 IAM 身分為基礎的政策，請參閱IAM 《 使用者指南》中的建立 IAM 政策。

如需 Verified Permissions 定義的動作和資源類型的詳細資訊，包括每種資源類型的 ARNs 格式，請參閱服務授權參考中的 Amazon Verified Permissions 的動作、資源和條件索引鍵。

主題

- 政策最佳實務
- 使用 Verified Permissions 主控台
- 允許使用者檢視他們自己的許可

政策最佳實務

身分型政策會判斷是否有人可以在您的帳戶中建立、存取或刪除 Verified Permissions 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用 AWS 受管政策來授予許多常見使用案例的許可。它們可在您的 中使用 AWS 帳戶。我們建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。如需詳細資訊，請參閱IAM 《 使用者指南》中的 AWS 受管政策或 AWS 任務函數的受管政策。
- 套用最低權限許可 – 當您使用 IAM 政策設定許可時， 只會授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的詳細資訊，請參閱IAM 《 使用者指南》中的政策和許可 IAM。
- 使用 IAM 政策中的條件來進一步限制存取：您可以將條件新增至政策，以限制對動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。如果透過特定 使用服務動作，您也可以使用條件來授予存取服務動作的權限 AWS 服務，例如 AWS CloudFormation。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素：條件。

- 使用 IAM Access Analyzer 驗證您的 IAM 政策以確保安全且功能正常的許可 – IAM Access Analyzer 會驗證新的和現有的政策，讓政策遵守 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱IAM 《 使用者指南》中的使用 IAM Access Analyzer 驗證政策。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以增加安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱IAM 《 使用者指南》中的使用 MFA 保護 API 存取。

如需 中最佳實務的詳細資訊 IAM，請參閱IAM 《 使用者指南》中的安全最佳實務 IAM。

使用 Verified Permissions 主控台

若要存取 Amazon Verified Permissions 主控台，您必須擁有一組最低許可。這些許可必須允許您列出和檢視 中已驗證許可資源的詳細資訊 AWS 帳戶。如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (使用者或角色) 而言，主控台就無法如預期運作。

對於僅對 AWS CLI 或 AWS API 進行呼叫的使用者，您不需要允許最低主控台許可。反之，只需允許存取符合他們嘗試執行之 API 操作的動作就可以了。

為了確保使用者和角色仍然可以使用 Verified Permissions 主控台，也請將 Verified Permissions *ConsoleAccess*或*ReadOnly* AWS 受管政策連接到實體。如需詳細資訊，請參閱IAM 《 使用者指南》中的新增許可給使用者。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在主控台上完成此動作的許可，或使用 AWS CLI 或 AWS API 以程式設計方式完成此動作的許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
```

```
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS Amazon Verified Permissions 的 受管政策

若要將許可新增至使用者、群組和角色，使用 AWS 受管政策比自行撰寫政策更容易。建立 IAM 客戶受管政策需要時間和專業知識，為您的團隊提供他們所需的許可。若要快速開始使用，您可以使用我們的 AWS 受管政策。這些政策涵蓋常見的使用案例，並可在您的 AWS 帳戶中使用。如需受 AWS 管政策的詳細資訊，請參閱IAM 《 使用者指南》中的受AWS 管政策。

AWS 服務會維護和更新 AWS 受管政策。您無法變更 AWS 受管政策中的許可。服務偶爾會在 AWS 受管政策中新增其他許可以支援新功能。此類型的更新會影響已連接政策的所有身分識別 (使用者、群組和角色)。當新功能啟動或新操作可用時，服務很可能會更新 AWS 受管政策。服務不會從 AWS 受管政策中移除許可，因此政策更新不會破壞現有的許可。

此外， AWS 支援跨多個 服務之任務函數的受管政策。例如，ReadOnlyAccess AWS 受管政策提供所有 AWS 服務和資源的唯讀存取權。當服務啟動新功能時， 會為新操作和資源 AWS 新增唯讀許可。如需任務函數政策的清單和說明，請參閱IAM 《 使用者指南》中的AWS 任務函數的受管政策。

AWS 受管政策：AmazonVerifiedPermissionsFullAccess

`AmazonVerifiedPermissionsFullAccess` 受管政策會授予驗證許可的完整存取權。若要使用以 Amazon Cognito 為基礎的身分來源，您需要連接單獨的政策，例如 AmazonCognitoReadOnly 政策。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:CreatePolicyStore",
        "verifiedpermissions:ListPolicyStores"
      ],
      "Resource": "*"
    },
    {
      "Sid": "PolicyStoreLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:*"
      ],
      "Resource": [
        "arn:aws:verifiedpermissions::*:policy-store/*"
      ]
    }
  ]
}
```

AWS 受管政策：AmazonVerifiedPermissionsReadOnlyAccess

`AmazonVerifiedPermissionsReadOnlyAccess` 受管政策會授予 Verified Permissions 的唯讀存取權。

此政策授予 Amazon Verified Permissions 所有讀取操作的存取權，包括授權查詢 APIs`IsAuthorized`和 `IsAuthorizedWithToken`。

> **Note**
>
> 分別將存取權BatchIsAuthorizedWithToken授予 BatchIsAuthorized和時IsAuthorizedWithToken，會自動授予對 IsAuthorized和 的存取權。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:ListPolicyStores"
      ],
      "Resource": "*"
    },
    {
      "Sid": "PolicyStoreLevelPermissions",
      "Effect": "Allow",
      "Action": [
        "verifiedpermissions:GetIdentitySource",
        "verifiedpermissions:GetPolicy",
        "verifiedpermissions:GetPolicyStore",
        "verifiedpermissions:GetPolicyTemplate",
        "verifiedpermissions:GetSchema",
        "verifiedpermissions:IsAuthorized",
        "verifiedpermissions:IsAuthorizedWithToken",
        "verifiedpermissions:ListIdentitySources",
        "verifiedpermissions:ListPolicies",
        "verifiedpermissions:ListPolicyTemplates"
      ],
      "Resource": [
        "arn:aws:verifiedpermissions::*:policy-store/*"
      ]
    }
  ]
}
```

AWS 受管政策的已驗證許可更新

檢視自此服務開始追蹤這些變更以來，已驗證許可的 AWS 受管政策更新詳細資訊。如需此頁面變更的自動提醒，請訂閱 Verified Permissions 文件歷史記錄頁面上的 RSS 摘要。

變更	描述	日期
AmazonVerifiedPermissionsFullAccess – 新政策	Verified Permissions 新增了新的政策，以允許完整存取 Verified Permissions。	2024 年 10 月 11 日
AmazonVerifiedPermissionsReadOnlyAccess – 新政策	Verified Permissions 新增了新的政策，以允許存取 Amazon Verified Permissions 的所有讀取操作，包括授權查詢 APIsIsAuthorized 和 IsAuthorizedWithToken 。	2024 年 10 月 11 日
已驗證的許可已開始追蹤變更	Verified Permissions 已開始追蹤其 AWS 受管政策的變更。	2024 年 10 月 11 日

對 Amazon Verified Permissions 身分和存取進行故障診斷

使用下列資訊來協助您診斷和修正使用 Verified Permissions 和 時可能遇到的常見問題 IAM。

主題

- 我無權在已驗證的許可中執行 動作
- 我未獲得執行 iam:PassRole 的授權
- 我想要允許 以外的人員 AWS 帳戶 存取我的 Verified Permissions 資源

我無權在已驗證的許可中執行 動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 verifiedpermissions:*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 verifiedpermissions:GetWidget on resource: my-example-widget
```

在此情況下，必須更新 mateojackson 使用者的政策，允許使用 verifiedpermissions:*GetWidget* 動作存取 *my-example-widget* 資源。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我未獲得執行 iam:PassRole 的授權

如果您收到錯誤，表示您無權執行iam:PassRole動作，則必須更新您的政策，以允許您將角色傳遞給已驗證的許可。

有些 AWS 服務 可讓您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

當名為 的 IAM marymajor 使用者嘗試使用主控台在 Verified Permissions 中執行動作時，會發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 iam:PassRole 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想要允許 以外的人員 AWS 帳戶 存取我的 Verified Permissions 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要了解 Verified Permissions 是否支援這些功能，請參閱 Amazon Verified Permissions 如何使用 IAM。
- 若要了解如何 AWS 帳戶 在您擁有的 資源間提供存取權，請參閱《 使用者指南》中的在您擁有 AWS 帳戶 的另一個 IAM 使用者中提供存取權。 IAM

- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱IAM 《 使用者指南》中的提供存取權給第三方 AWS 帳戶 擁有。
- 若要了解如何透過聯合身分提供存取權，請參閱IAM 《 使用者指南》中的提供存取權給外部驗證的使用者 （ 聯合身分 ）。
- 若要了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱IAM 《 使用者指南》中的跨帳戶資源存取 IAM。

Amazon Verified Permissions 的合規驗證

若要了解 是否 AWS 服務 在特定合規計劃的範圍內，請參閱AWS 服務 合規計劃範圍內然後選擇您感興趣的合規計劃。如需一般資訊，請參閱 AWS Compliance Programs。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱在 中下載報告 AWS Artifact。

使用 時的合規責任 AWS 服務 取決於資料的敏感度、您公司的合規目標，以及適用的法律和法規。AWS 提供下列資源以協助合規：

- 安全合規與治理 - 這些解決方案實作指南內容討論了架構考量，並提供部署安全與合規功能的步驟。
- HIPAA 合格服務參考 – 列出 HIPAA 合格服務。並非所有 AWS 服務 都符合 HIPAA 資格。
- AWS 合規資源 – 此工作手冊和指南的集合可能適用於您的產業和位置。
- AWS 客戶合規指南 – 透過合規的角度了解共同責任模型。本指南摘要說明保護 的最佳實務， AWS 服務 並將指南映射到跨多個架構的安全控制 （ 包括國家標準和技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準化組織 (ISO))。
- AWS Config 開發人員指南中的使用規則評估資源 – AWS Config 服務會評估資源組態符合內部實務、產業準則和法規的程度。
- AWS Security Hub – 這 AWS 服務 可讓您全面檢視其中的安全狀態 AWS。Security Hub 使用安全控制，可評估您的 AWS 資源並檢查您的法規遵循是否符合安全業界標準和最佳實務。如需支援的服務和控制清單，請參閱「Security Hub 控制參考」。
- Amazon GuardDuty – 這可透過監控您的環境是否有可疑和惡意活動，來 AWS 服務 偵測對您 AWS 帳戶、工作負載、容器和資料的潛在威脅。GuardDuty 可滿足特定合規架構所規定的入侵偵測需求，以協助您因應 PCI DSS 等各種不同的合規需求。
- AWS Audit Manager – 這 AWS 服務 可協助您持續稽核 AWS 用量，以簡化您管理風險的方式，以及符合法規和產業標準的方式。

Amazon Verified 許可中的彈性

AWS 全域基礎設施是以 AWS 區域 和 可用區域為基礎建置。 AWS 區域 提供多個實體隔離和隔離的可用區域，這些區域與低延遲、高輸送量和高度備援聯網連接。透過可用區域，您可以設計與操作的應用程式和資料庫，在可用區域之間自動容錯移轉而不會發生中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

當您建立 Verified Permissions 政策存放區 時，它會在個別 內建立 AWS 區域，並自動跨組成該區域的可用區域的資料中心進行複寫。目前，已驗證的許可不支援任何跨區域複寫。

如需 AWS 區域 和 可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

監控 Amazon Verified Permissions API 呼叫

監控是維護 Amazon Verified Permissions 和其他 AWS 解決方案可靠性、可用性和效能的重要部分。AWS 提供下列工具來監控 Verified Permissions、報告錯誤，並在適當時採取自動動作：

- AWS CloudTrail 會擷取由您的帳戶或代表 AWS 您的帳戶發出的 API 呼叫和相關事件，並將日誌檔案交付至您指定的 Amazon S3 儲存貯體。您可以識別呼叫的使用者和帳戶 AWS、進行呼叫的來源 IP 地址，以及呼叫的時間。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》https://docs.aws.amazon.com/awscloudtrail/latest/userguide/。

如需使用 CloudTrail 監控已驗證許可的詳細資訊，請參閱 使用 記錄 Amazon Verified Permissions API 呼叫 AWS CloudTrail。

使用 記錄 Amazon Verified Permissions API 呼叫 AWS CloudTrail

Amazon Verified Permissions 已與 整合 AWS CloudTrail，此服務提供使用者、角色或 Verified Permissions 中 AWS 服務所採取動作的記錄。CloudTrail 會將已驗證許可的所有 API 呼叫擷取為事件。擷取的呼叫包括從 Verified Permissions 主控台呼叫，以及對 Verified Permissions API 操作的程式碼呼叫。如果您建立線索，您可以啟用 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括已驗證許可的事件。如果您未設定線索，仍然可以在 CloudTrail 主控台的事件歷史記錄中檢視最新的管理動作事件，但無法檢視 API 呼叫的事件，例如 `isAuthorized`。使用 CloudTrail 收集的資訊，您可以判斷對 Verified Permissions 提出的請求、提出請求的 IP 地址、提出請求的人員、提出請求的時間，以及其他詳細資訊。

若要進一步了解 CloudTrail，請參閱「AWS CloudTrail 使用者指南」。

CloudTrail 中的已驗證許可資訊

建立帳戶 AWS 帳戶 時，您的 上會啟用 CloudTrail。當活動在 Verified Permissions 中發生時，該活動會與事件歷史記錄中的其他服務 AWS 事件一起記錄在 CloudTrail 事件中。您可以檢視、搜尋和下載 AWS 帳戶的最新事件。如需詳細資訊，請參閱「使用 CloudTrail 事件歷史記錄檢視事件」。

若要持續記錄 中的事件 AWS 帳戶，包括已驗證許可的事件，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。依預設，當您在主控台中建立追蹤時，該追蹤會套用至所有的 AWS 區域。追蹤會記錄 AWS 分割區中所有 區域的事件，並將日誌檔案交付至您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析 CloudTrail 日誌中收集的事件資料並對其採取行動。如需詳細資訊，請參閱下列內容：

- 建立追蹤的概觀
- CloudTrail 支援的服務和整合
- 設定 CloudTrail 的 Amazon SNS 通知
- 接收多個區域的 CloudTrail 日誌檔案和接收多個帳戶的 CloudTrail 日誌檔案

CloudTrail 會記錄所有驗證許可動作，並記載於 Amazon Verified Permissions API 參考指南。例如，對 `CreateIdentitySource`、`DeletePolicy` 以及 `ListPolicyStores` 動作發出的呼叫會在 CloudTrail 日誌檔案中產生項目。

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 請求是使用根還是 AWS Identity and Access Management (IAM) 使用者登入資料提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 請求是否由其他 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

建立線索或事件資料存放區時，預設不會記錄 IsAuthorized 和 IsAuthorizedWithToken 等資料事件。若要記錄 CloudTrail 資料事件，您必須明確地新增欲收集之活動的受支援資源或資源類型。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的資料事件。

了解 Verified Permissions 日誌檔案項目

追蹤是一種組態，能讓事件以日誌檔案的形式交付到您指定的 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含一或多個日誌專案。一個事件為任何來源提出的單一請求，並包含請求動作、請求的日期和時間、請求參數等資訊。CloudTrail 日誌檔並非依公有 API 呼叫的堆疊追蹤排序，因此不會以任何特定順序出現。

對於授權 API 呼叫，回應元素，例如決策，包含在 下，`additionalEventData`而不是 `responseElements`。

主題

- IsAuthorized
- BatchIsAuthorized
- CreatePolicyStore
- ListPolicyStores

- DeletePolicyStore
- PutSchema
- GetSchema
- CreatePolicyTemplate
- DeletePolicyTemplate
- CreatePolicy
- GetPolicy
- CreateIdentitySource
- GetIdentitySource
- ListIdentitySources
- DeleteIdentitySource

> **Note**
>
> 已修改資料隱私權範例的某些欄位。

IsAuthorized


```
{
    "eventVersion": "1.08",
    "userIdentity": {
  "type": "AssumedRole",
  "principalId": "EXAMPLE_PRINCIPAL_ID",
  "arn": "arn:aws:iam::123456789012:role/ExampleRole",
  "accountId": "123456789012",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
    },
    "eventTime": "2023-11-20T22:55:03Z",
    "eventSource": "verifiedpermissions.amazonaws.com",
    "eventName": "IsAuthorized",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "aws-cli/2.11.18 Python/3.11.3 Linux/5.4.241-160.348.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/verifiedpermissions.is-authorized",
    "requestParameters": {
        "principal": {
```

```
            "entityType": "PhotoFlash::User",
            "entityId": "alice"
        },
        "action": {
            "actionType": "PhotoFlash::Action",
            "actionId": "ViewPhoto"
        },
        "resource": {
            "entityType": "PhotoFlash::Photo",
            "entityId": "VacationPhoto94.jpg"
        },
        "policyStoreId": "PSEXAMPLEabcdefg111111"
    },
    "responseElements": null,
    "additionalEventData": {
        "decision": "ALLOW"
    },
    "requestID": "346c4b6a-d12f-46b6-bc06-6c857bd3b28e",
    "eventID": "8a4fed32-9605-45dd-a09a-5ebbf0715bbc",
    "readOnly": true,
    "resources": [
        {
    "accountId": "123456789012",
    "type": "AWS::VerifiedPermissions::PolicyStore",
    "ARN": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data"
}
```

BatchIsAuthorized

```
{
    "eventVersion": "1.08",
    "userIdentity": {
  "type": "AssumedRole",
  "principalId": "EXAMPLE_PRINCIPAL_ID",
  "arn": "arn:aws:iam::123456789012:role/ExampleRole",
  "accountId": "123456789012",
```

```
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
    },
    "eventTime": "2023-11-20T23:02:33Z",
    "eventSource": "verifiedpermissions.amazonaws.com",
    "eventName": "BatchIsAuthorized",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "aws-cli/2.11.18 Python/3.11.3 Linux/5.4.241-160.348.amzn2int.x86_64
exe/x86_64.amzn.2 prompt/off command/verifiedpermissions.is-authorized",
    "requestParameters": {
        "requests": [
            {
                "principal": {
                    "entityType": "PhotoFlash::User",
                    "entityId": "alice"
                },
                "action": {
                    "actionType": "PhotoFlash::Action",
                    "actionId": "ViewPhoto"
                },
                "resource": {
                    "entityType": "PhotoFlash::Photo",
                    "entityId": "VacationPhoto94.jpg"
                }
            },
            {
                "principal": {
                    "entityType": "PhotoFlash::User",
                    "entityId": "annalisa"
                },
                "action": {
                    "actionType": "PhotoFlash::Action",
                    "actionId": "DeletePhoto"
                },
                "resource": {
                    "entityType": "PhotoFlash::Photo",
                    "entityId": "VacationPhoto94.jpg"
                }
            }
        ],
        "policyStoreId": "PSEXAMPLEabcdefg111111"
    },
    "responseElements": null,
    "additionalEventData": {
```

```
        "results": [
            {
                "request": {
                    "principal": {
                        "entityType": "PhotoFlash::User",
                        "entityId": "alice"
                    },
                    "action": {
                        "actionType": "PhotoFlash::Action",
                        "actionId": "ViewPhoto"
                    },
                    "resource": {
                        "entityType": "PhotoFlash::Photo",
                        "entityId": "VacationPhoto94.jpg"
                    }
                },
                "decision": "ALLOW"
            },
            {
                "request": {
                    "principal": {
                        "entityType": "PhotoFlash::User",
                        "entityId": "annalisa"
                    },
                    "action": {
                        "actionType": "PhotoFlash::Action",
                        "actionId": "DeletePhoto"
                    },
                    "resource": {
                        "entityType": "PhotoFlash::Photo",
                        "entityId": "VacationPhoto94.jpg"
                    }
                },
                "decision": "DENY"
            }
        ]
    },
    "requestID": "a8a5caf3-78bd-4139-924c-7101a8339c3b",
    "eventID": "7d81232f-f3d1-4102-b9c9-15157c70487b",
    "readOnly": true,
    "resources": [
        {
    "accountId": "123456789012",
    "type": "AWS::VerifiedPermissions::PolicyStore",
```

```
    "ARN": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data"
}
```

CreatePolicyStore

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:43:33Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreatePolicyStore",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "clientToken": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "validationSettings": {
      "mode": "OFF"
    }
  },
  "responseElements": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
    "createdDate": "2023-05-22T07:43:33.962794Z",
    "lastUpdatedDate": "2023-05-22T07:43:33.962794Z"
  },
  "requestID": "1dd9360e-e2dc-4554-ab65-b46d2cf45c29",
  "eventID": "b6edaeee-3584-4b4e-a48e-311de46d7532",
  "readOnly": false,
```

```
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

ListPolicyStores

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:43:33Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "ListPolicyStores",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "maxResults": 10
  },
  "responseElements": null,
  "requestID": "5ef238db-9f87-4f37-ab7b-6cf0ba5df891",
  "eventID": "b0430fb0-12c3-4cca-8d05-84c37f99c51f",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

DeletePolicyStore

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
```

```
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:43:32Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "DeletePolicyStore",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "1368e8f9-130d-45a5-b96d-99097ca3077f",
  "eventID": "ac482022-b2f6-4069-879a-dd509123d8d7",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

PutSchema

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
```

```
  "eventTime": "2023-05-16T12:58:57Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "PutSchema",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": {
    "lastUpdatedDate": "2023-05-16T12:58:57.513442Z",
    "namespaces": "[some_namespace]",
    "createdDate": "2023-05-16T12:58:57.513442Z",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
  },
  "requestID": "631fbfa1-a959-4988-b9f8-f1a43ff5df0d",
  "eventID": "7cd0c677-733f-4602-bc03-248bae581fe5",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "ARN": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```


GetSchema


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::222222222222:role/ExampleRole",
    "accountId": "222222222222",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
```

```
  "eventTime": "2023-05-25T01:12:07Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "GetSchema",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "a1f4d4cd-6156-480a-a9b8-e85a71dcc7c2",
  "eventID": "0b3b8e3d-155c-46f3-a303-7e9e8b5f606b",
  "readOnly": true,
  "resources": [
    {
      "accountId": "222222222222",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "ARN": "arn:aws:verifiedpermissions::222222222222:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "222222222222",
  "eventCategory": "Management"
}
```


CreatePolicyTemplate


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-16T13:00:24Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreatePolicyTemplate",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
```

```
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": {
    "lastUpdatedDate": "2023-05-16T13:00:23.444404Z",
    "createdDate": "2023-05-16T13:00:23.444404Z",
    "policyTemplateId": "PTEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
  },
  "requestID": "73953bda-af5e-4854-afe2-7660b492a6d0",
  "eventID": "7425de77-ed84-4f91-a4b9-b669181cc57b",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```


DeletePolicyTemplate


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::222222222222:role/ExampleRole",
    "accountId": "222222222222",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-25T01:11:48Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "DeletePolicyTemplate",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
```

```
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyTemplateId": "PTEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "5ff0f22e-6bbd-4b85-a400-4fb74aa05dc6",
  "eventID": "c0e0c689-369e-4e95-a9cd-8de113d47ffa",
  "readOnly": false,
  "resources": [
    {
      "accountId": "222222222222",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "ARN": "arn:aws:verifiedpermissions::222222222222:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "222222222222",
  "eventCategory": "Management"
}
```

CreatePolicy

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-22T07:42:30Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreatePolicy",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "clientToken": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
```

```
  },
  "responseElements": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyType": "STATIC",
    "principal": {
        "entityType": "PhotoApp::Role",
        "entityId": "PhotoJudge"
    },
    "resource": {
        "entityType": "PhotoApp::Application",
        "entityId": "PhotoApp"
    },
    "lastUpdatedDate": "2023-05-22T07:42:30.70852Z",
    "createdDate": "2023-05-22T07:42:30.70852Z"
  },
  "requestID": "93ffa151-3841-4960-9af6-30a7f817ef93",
  "eventID": "30ab405f-3dff-43ff-8af9-f513829e8bde",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

GetPolicy

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:role/ExampleRole",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
```

```
  },
  "eventTime": "2023-05-22T07:43:29Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "GetPolicy",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyId": "SPEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "23022a9e-2f5c-4dac-b653-59e6987f2fac",
  "eventID": "9b4d5037-bafa-4d57-b197-f46af83fc684",
  "readOnly": true,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```


CreateIdentitySource


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-19T01:27:44Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "CreateIdentitySource",
```

```
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "clientToken": "a1b2c3d4-e5f6-a1b2-c3d4-TOKEN1111111",
    "configuration": {
      "cognitoUserPoolConfiguration": {
        "userPoolArn": "arn:aws:cognito-idp:000011112222:us-east-1:userpool/us-
east-1_aaaaaaaaaa"
      }
    },
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "principalEntityType": "User"
  },
  "responseElements": {
    "createdDate": "2023-07-14T15:05:01.599534Z",
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "lastUpdatedDate": "2023-07-14T15:05:01.599534Z",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
    },
  "requestID": "afcc1e67-d5a4-4a9b-a74c-cdc2f719391c",
  "eventID": "f13a41dc-4496-4517-aeb8-a389eb379860",
  "readOnly": false,
  "resources": [
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```


GetIdentitySource


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
```

```
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-24T19:55:31Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "GetIdentitySource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "7a6ecf79-c489-4516-bb57-9ded970279c9",
  "eventID": "fa158e6c-f705-4a15-a731-2cdb4bd9a427",
  "readOnly": true,
  "resources": [
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```


ListIdentitySources


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
```

```
  },
  "eventTime": "2023-05-24T20:05:32Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "ListIdentitySources",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "95d2a7bc-7e9a-4efe-918e-97e558aacaf7",
  "eventID": "d3dc53f6-1432-40c8-9d1d-b9eeb75c6193",
  "readOnly": true,
  "resources": [
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```


DeleteIdentitySource


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:iam::333333333333:role/ExampleRole",
    "accountId": "333333333333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
  },
  "eventTime": "2023-05-24T19:55:32Z",
  "eventSource": "verifiedpermissions.amazonaws.com",
  "eventName": "DeleteIdentitySource",
  "awsRegion": "us-west-2",
```

```
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-rust/0.55.2 os/linux lang/rust/1.69.0",
  "requestParameters": {
    "identitySourceId": "ISEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111"
  },
  "responseElements": null,
  "requestID": "d554d964-0957-4834-a421-c417bd293086",
  "eventID": "fe4d867c-88ee-4e5d-8d30-2fbc208c9260",
  "readOnly": false,
  "resources": [
    {
      "accountId": "333333333333",
      "type": "AWS::VerifiedPermissions::PolicyStore",
      "arn": "arn:aws:verifiedpermissions::333333333333:policy-store/
PSEXAMPLEabcdefg111111"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "333333333333",
  "eventCategory": "Management"
}
```

使用 建立 Amazon Verified Permissions 資源 AWS CloudFormation

Amazon Verified Permissions 已與 整合 AWS CloudFormation，這項服務可協助您建立和設定 AWS 資源的模型，讓您可減少建立和管理資源和基礎設施的時間。您可以建立範本，描述您想要的所有 AWS 資源 （例如政策存放區），以及為您 AWS CloudFormation 佈建和設定這些資源。

使用 時 AWS CloudFormation，您可以重複使用範本，以一致且重複地設定您的 Verified Permissions 資源。描述您的資源一次，然後在多個 AWS 帳戶 和 區域中逐一佈建相同的資源。

> **Important**
>
> Amazon Cognito Identity 無法在與 Amazon Verified Permissions AWS 區域 相同的所有 中使用。如果您從 收到 AWS CloudFormation 有關 Amazon Cognito Identity 的錯誤，例如 `Unrecognized resource types: AWS::Cognito::UserPool, AWS::Cognito::UserPoolClient`，我們建議您在最接近 Amazon Cognito Identity AWS 區域 的地理位置中建立 Amazon Cognito 使用者集區和用戶端。建立 Verified Permissions 身分來源時，請使用此新建立的使用者集區。

已驗證的許可和 AWS CloudFormation 範本

若要佈建和設定已驗證許可和相關服務的資源，您必須了解 AWS CloudFormation 範本。範本是以 JSON 或 YAML 格式化的文本檔案。這些範本說明您想要在 AWS CloudFormation 堆疊中佈建的資源。如果您不熟悉 JSON 或 YAML，您可以使用 AWS CloudFormation 設計工具來協助您開始使用 AWS CloudFormation 範本。如需詳細資訊，請參閱AWS CloudFormation 《 使用者指南》中的什麼是 AWS CloudFormation 設計工具？。

Verified Permissions 支援在其中建立身分來源、政策、政策存放區和政策範本 AWS CloudFormation。如需詳細資訊，包括驗證許可資源的 JSON 和 YAML 範本範例，請參閱AWS CloudFormation 《 使用者指南》中的 Amazon 驗證許可資源類型參考。

AWS CDK 建構

AWS Cloud Development Kit (AWS CDK) 是開放原始碼軟體開發架構，可用於在程式碼中定義雲端基礎設施，並透過其佈建 AWS CloudFormation。建構或可重複使用的雲端元件可用於建立 AWS CloudFormation 範本。然後，您可以使用這些範本來部署雲端基礎設施。

若要進一步了解並下載 AWS CDK，請參閱AWS 雲端開發套件。

以下是 Verified Permissions AWS CDK 資源的文件連結，例如 constructs。

- Amazon Verified Permissions L2 CDK Construct

進一步了解 AWS CloudFormation

若要進一步了解 AWS CloudFormation，請參閱下列資源：

- AWS CloudFormation
- AWS CloudFormation 使用者指南
- AWS CloudFormation API 參考
- AWS CloudFormation 命令列界面使用者指南

使用 存取 Amazon Verified Permissions AWS PrivateLink

您可以使用 在 VPC 和 Amazon Verified Permissions 之間 AWS PrivateLink 建立私有連線。您可以像在 VPC 中一樣存取 Verified Permissions，無需使用網際網路閘道、NAT 裝置、VPN 連接或 AWS Direct Connect 連線。VPC 中的執行個體不需要公有 IP 地址即可存取 Verified Permissions。

您可以建立由 AWS PrivateLink提供支援的介面端點來建立此私有連線。我們會在您為介面端點啟用的每個子網中建立端點網路介面。這些是請求者管理的網路介面，可做為目的地為 Verified Permissions 之流量的進入點。

如需詳細資訊，請參閱「AWS PrivateLink 指南」中的透過 AWS PrivateLink存取 AWS 服務。

Verified Permissions 的考量

在您設定 Verified Permissions 的介面端點之前，請檢閱《 AWS PrivateLink 指南》中的考量事項。

Verified Permissions 支援透過界面端點呼叫其所有 API 動作。

Verified Permissions 不支援 VPC 端點政策。根據預設，允許透過介面端點完整存取 Verified Permissions。或者，您可以將安全群組與端點網路介面建立關聯，以控制透過介面端點傳送至 Verified Permissions 的流量。

建立已驗證許可的介面端點

您可以使用 Amazon VPC 主控台或 AWS Command Line Interface () 為已驗證許可建立介面端點AWS CLI。如需詳細資訊，請參閱《AWS PrivateLink 指南》中的建立介面端點。

使用下列服務名稱建立 Verified Permissions 的介面端點：

```
com.amazonaws.region.verifiedpermissions
```

如果您為介面端點啟用私有 DNS，您可以使用其預設的區域 DNS 名稱向已驗證許可提出 API 請求。例如 `verifiedpermissions.us-east-1.amazonaws.com`。

為您的介面端點建立端點政策

端點政策為 IAM 資源，您可將其連接至介面端點。預設端點政策允許透過介面端點完整存取 Verified Permissions。若要控制允許從您的 VPC 存取已驗證許可，請將自訂端點政策連接至介面端點。

端點政策會指定以下資訊：

- 可執行動作 (AWS 帳戶、IAM 使用者和 IAM 角色) 的主體。
- 可執行的動作。
- 可供執行動作的資源。

如需詳細資訊，請參閱「AWS PrivateLink 指南」中的使用端點政策控制對服務的存取。

範例：VPC 端點政策，適用於 Verified Permissions 動作

以下是自訂端點政策的範例。當您將此政策連接到介面端點時，它會授予所有資源上所有主體所列出的 Verified Permissions 動作的存取權。

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "verifiedpermissions:IsAuthorized",
            "verifiedpermissions:IsAuthorizedWithToken",
            "verifiedpermissions:GetPolicy"
         ],
         "Resource":"*"
      }
   ]
}
```

Amazon Verified Permissions 的配額

您的 AWS 帳戶 具有每個 AWS 服務的預設配額，先前稱為限制。除非另有說明，否則每個配額都是區域特定規定。您可以請求提高某些配額，而其他配額無法提高。

若要檢視已驗證許可的配額，請開啟 Service Quotas 主控台。在導覽窗格中，選擇 AWS 服務，然後選取已驗證的許可。

若要請求增加配額，請參閱 Service Quotas 使用者指南中的請求提高配額。如果 Service Quotas 中尚未提供配額，請使用增加服務配額表單。

您的 AWS 帳戶 具有下列與已驗證許可相關的配額。

主題

- 資源的配額
- 階層的配額
- 每秒操作的配額

資源的配額

名稱	預設	可調整	描述
每個帳戶每個區域的政策存放區	每個支援的區域：30，000	是	政策存放區的數目上限。
每個政策存放區的政策範本	每個受支援的區域：40	是	政策存放區中的政策範本數目上限。
每個政策存放區的身分來源	1	否	您可以為政策存放區定義的身分來源數目上限。
授權請求大小1	1 MB	否	授權請求的大小上限。
政策大小	10,000 位元組	否	個別政策的大小上限。

名稱	預設	可調整	描述
結構描述大小	200，000 位元組	否	政策存放區結構描述的大小上限。
每個資源的政策大小	200，000 位元組2	是	參考特定資源的所有政策的大小上限。

1 IsAuthorized 和 IsAuthorizedWithToken 的授權請求配額相同。

2 單一資源範圍的所有政策總大小的預設限制為 200，000 個位元組。同樣地，範圍讓資源保持未定義的所有政策總大小，因此套用至所有資源，預設限制為 200，000 個位元組。請注意，對於範本連結政策，政策範本的大小只會計算一次，加上用於執行個體化每個範本連結政策之每組參數的大小。如果您的政策設計符合特定限制，則可以提高此限制。如果您需要探索此選項，請聯絡 支援。

範本連結政策大小範例

您可以透過取得委託人和資源的長度總和，來判斷範本連結政策對每個資源配額的政策大小有何貢獻。如果未指定委託人或資源，則該片段的長度為 0。如果未指定資源，其大小會計入"unspecified"資源配額。範本內文的大小不會影響政策大小。

讓我們來看看下列範本：

```
@id("template1")
permit (
  principal in ?principal,
  action in [Action::"view", Action::"comment"],
  resource in ?resource
)
unless {
  resource.tag =="private"
};
```

讓我們從該範本建立下列政策：

```
TemplateLinkedPolicy {
```

```
  policyId: "policy1",
  templateId: "template1",
  principal: User::"alice",
  resource: Photo::"car.jpg"
}

TemplateLinkedPolicy {
  policyId: "policy2",
  templateId: "template1",
  principal: User::"bob",
  resource: Photo::"boat.jpg"
}

TemplateLinkedPolicy {
  policyId: "policy3",
  templateId: "template1",
  principal: User::"jane",
  resource: Photo::"car.jpg"

TemplateLinkedPolicy {
  policyId: "policy4",
  templateId: "template1",
  principal: User::"jane",
  resource
}
```

現在，讓我們計算這些政策的大小，方法是計算 和每個政策principalresource的字元。每個字元計為 1 個位元組。

的大小policy1為委託人 User::"alice"(13) 的長度加上資源 Photo::"car.jpg"(16) 的長度。將它們加起來，我們有 13 + 16 = 29 個位元組。

的大小policy2為委託人 User::"bob"(11) 的長度加上資源 Photo::"boat.jpg"(17) 的長度。將它們加起來，我們有 11 + 17 = 28 個位元組。

的大小policy3為委託人 User::"jane"(12) 的長度加上資源 Photo::"car.jpg"(16) 的長度。將它們加起來，我們有 12 + 16 = 28 個位元組。

的大小policy4會是委託人 User::"jane"(12) 的長度加上資源 (0) 的長度。將它們加起來，我們有 12 + 0 = 12 個位元組。

由於 policy2是唯一參考資源 的政策Photo::"boat.jpg"，因此總資源大小為 28 個位元組。

由於 policy1和 policy3都參考資源 Photo::"car.jpg"，因此總資源大小為 29 + 28 = 57 位元組。

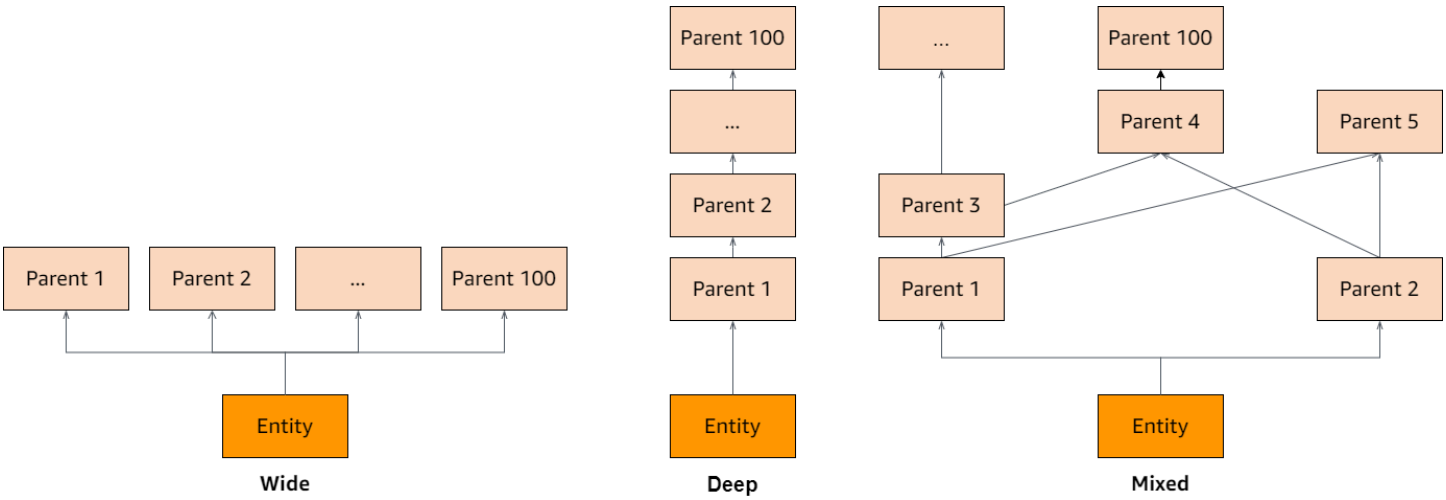
由於 policy4是唯一參考"unspecified"資源的政策，因此總資源大小為 12 個位元組。

階層的配額

> **Note**
>
> 下列配額會彙總，這表示它們會一起新增。群組的可轉移父項數量上限是列出的。例如，如果每個主體的可轉移父系限制為 100，表示在動作和資源中，可能會有 100 名主體的父系和 0 名父系，或任何總計最多 100 名父系的父系組合。

名稱	預設	可調整	描述
每個主體的暫時性父系	100	否	每個主體的可轉移父項數量上限。
每個動作的暫時性父項	100	否	每個動作的可轉移父項數量上限。
每個資源的暫時性父項	100	否	每個資源的可轉移父項數量上限。

下圖說明如何為實體（委託人、動作或資源）定義可轉移的父系。



每秒操作的配額

當應用程式請求超過 API 操作的配額 AWS 區域 時，Verified Permissions 會調節對 中服務端點的請求。當您超過每秒請求的配額，或嘗試同時寫入操作時，Verified Permissions 可能會傳回例外狀況。您可以在 Service Quotas 中檢視目前的 RPS 配額。為了防止應用程式超過 操作的配額，您必須針對重試和指數退避進行最佳化。如需詳細資訊，請參閱使用退避模式重試，以及管理和監控工作負載中的 API 限流。

名稱	預設	可調整	描述
每個帳戶每個區域每秒 BatchGetPolicy 請求數	每個受支援的區域：10	是	每秒 BatchGetPolicy 請求的數量上限。
每個帳戶每個區域每秒 BatchIsAuthorized 請求數	每個受支援的區域：30	是	每秒 BatchIsAuthorized 請求的數量上限。
每個帳戶每個區域每秒 BatchIsAuthorizedWithToken 請求數	每個受支援的區域：30	是	每秒 BatchIsAuthorizedWithToken 請求的數量上限。
每個帳戶每個區域的每秒 CreateIdentitySource 請求數	每個受支援的區域：1	是	每秒 CreateIdentitySource 請求的數量上限。

名稱	預設	可調整	描述
每個帳戶每個區域的每秒 CreatePolicy 請求數	每個受支援的區域：10	是	每秒 CreatePolicy 請求的數量上限。
每個帳戶每個區域的 CreatePolicyStore 每秒請求數	每個受支援的區域：1	否	每秒 CreatePolicyStore 請求的數量上限。
每個帳戶每個區域的每秒 CreatePolicyTemplate 請求數	每個受支援的區域：10	是	每秒 CreatePolicyTemplate 請求的數量上限。
每個帳戶每個區域的每秒 DeleteIdentitySource 請求數	每個受支援的區域：1	是	每秒 DeleteIdentitySource 請求的數量上限。
每個帳戶每個區域的每秒 DeletePolicy 請求數	每個受支援的區域：10	是	每秒 DeletePolicy 請求的數量上限。
每個帳戶每個區域的 DeletePolicyStore 每秒請求數	每個受支援的區域：1	否	每秒 DeletePolicyStore 請求的數量上限。
每個帳戶每個區域的每秒 DeletePolicyTemplate 請求數	每個受支援的區域：10	是	每秒 DeletePolicyTemplate 請求的數量上限。
每個帳戶每個區域的每秒 GetIdentitySource 請求數	每個受支援的區域：10	是	每秒 GetIdentitySource 請求的數量上限。
每個帳戶每個區域的每秒 GetPolicy 請求數	每個受支援的區域：10	是	每秒的 GetPolicy 請求數目上限。
每個帳戶每個區域的每秒 GetPolicyStore 請求數	每個受支援的區域：10	是	每秒 GetPolicyStore 請求的數量上限。
每個帳戶每個區域的每秒 GetPolicyTemplate 請求數	每個受支援的區域：10	是	每秒 GetPolicyTemplate 請求的數量上限。
每個帳戶每個區域的每秒 GetSchema 請求數	每個受支援的區域：10	是	每秒 GetSchema 請求的數量上限。

名稱	預設	可調整	描述
每個帳戶每個區域每秒的 IsAuthorized 請求數	每個受支援的區域：200	是	每秒的 IsAuthorized 請求數目上限。
每個帳戶每個區域的 IsAuthorizedWithToken 每秒請求數	每個受支援的區域：200	是	每秒的 IsAuthorizedWithToken 請求數目上限。
每個帳戶每個區域的 ListIdentitySources 每秒請求數	每個受支援的區域：10	是	每秒 ListIdentitySources 請求的數量上限。
每個帳戶每個區域每秒的 ListPolicies 請求數	每個受支援的區域：10	是	每秒 ListPolicies 請求的數量上限。
每個帳戶每個區域的 ListPolicyStores 每秒請求數	每個受支援的區域：10	是	每秒 ListPolicyStores 請求的數量上限。
每個帳戶每個區域的 ListPolicyTemplates 每秒請求數	每個受支援的區域：10	是	ListPolicyTemplates 每秒請求數上限。
每個帳戶每個區域的每秒 PutSchema 請求數	每個受支援的區域：10	是	每秒 PutSchema 請求的數量上限。
每個帳戶每個區域的每秒 UpdateIdentitySource 請求數	每個受支援的區域：1	是	每秒 UpdateIdentitySource 請求的數量上限。
每個帳戶每個區域的每秒 UpdatePolicy 請求數	每個受支援的區域：10	是	每秒 UpdatePolicy 請求的數量上限。
每個帳戶每個區域的每秒 UpdatePolicyStore 請求數	每個受支援的區域：10	否	每秒 UpdatePolicyStore 請求的數量上限。
每個帳戶每個區域的每秒 UpdatePolicyTemplate 請求數	每個受支援的區域：10	是	每秒 UpdatePolicyTemplate 請求的數量上限。

Amazon Verified Permissions 和 Cedar 政策語言術語和概念

您應該了解下列概念，才能使用 Amazon Verified Permissions。

已驗證的許可概念

- 授權模型
- 授權請求
- 授權回應
- 已考量的政策
- 內容資料
- 決定政策
- 實體資料
- 許可、授權和主體
- 政策強制執行
- 政策存放區
- 滿意的政策
- Amazon Verified Permissions 與 Cedar 政策語言之間的差異

Cedar 政策語言概念

- 授權
- 實體
- 群組和階層
- 命名空間
- 政策
- 政策範本
- 結構描述

授權模型

授權模型說明應用程式提出的授權請求範圍，以及評估這些請求的基礎。它根據不同類型的資源、對這些資源採取的動作，以及採取這些動作的委託人類型來定義。它也會考慮採取這些動作的背景。

角色型存取控制 (RBAC) 是評估基礎，其中定義角色並與一組許可相關聯。然後，可以將這些角色指派給一或多個身分。指派的身分會取得與角色相關聯的許可。如果修改與角色相關聯的許可，則修改會自動影響角色指派的任何身分。Cedar 可以透過使用主體群組來支援 RBAC 決策。

屬性型存取控制 (ABAC) 是評估基礎，其中與身分相關聯的許可是由該身分的屬性決定。Cedar 可以透過使用參考委託人屬性的政策條件來支援 ABAC 決策。

Cedar 政策語言允許為具有屬性型條件的使用者群組定義許可，以啟用單一政策中 RBAC 和 ABAC 的組合。

授權請求

授權請求是由應用程式提出的驗證許可請求，用於評估一組政策，以判斷委託人是否可以對特定內容的資源執行動作。

授權回應

授權回應是對授權請求的回應。它包含允許或拒絕的決定，以及其他資訊，例如決定政策IDs。

已考量的政策

視為政策是在評估授權請求時，由 Verified Permissions 所選取的整組政策。

內容資料

內容資料是屬性值，可提供要評估的其他資訊。

決定政策

決定政策是決定授權回應的政策。例如，如果有兩個滿意的政策，其中一個是拒絕，另一個是允許，則拒絕政策將是決定政策。如果有多個滿意的許可政策，但沒有滿意的禁止政策，則有多個決定政策。如果沒有政策相符且拒絕回應，則沒有決定性政策。

實體資料

實體資料是有關委託人、動作和資源的資料。與政策評估相關的實體資料是群組成員資格，一直增加主體和資源的實體階層和屬性值。

許可、授權和主體

Verified Permissions 會在您建置的自訂應用程式中管理精細的許可和授權。

委託人是人或機器應用程式的使用者，其身分繫結至識別符，例如使用者名稱或機器 ID。身分驗證程序會判斷委託人是否為其聲稱的真實身分。

與該身分相關聯的是一組應用程式許可，用於決定該主體在該應用程式中被允許執行的動作。授權是評估這些許可的程序，以判斷委託人是否允許在應用程式中執行特定動作。這些許可可以表示為政策。

政策強制執行

政策強制執行是在 Verified Permissions 外部的應用程式內強制執行評估決策的程序。如果 Verified Permissions 評估傳回拒絕，則強制執行會確保委託人無法存取資源。

政策存放區

政策存放區是政策和範本的容器。每個存放區都包含一個結構描述，用於驗證新增至存放區的政策。根據預設，每個應用程式都有自己的政策存放區，但多個應用程式可以共用單一政策存放區。當應用程式發出授權請求時，它會識別用來評估該請求的政策存放區。政策存放區提供隔離一組政策的方法，因此可用於多租戶應用程式中，以包含每個租戶的結構描述和政策。每個租用戶都可以有個別的政策存放區。

評估授權請求時，驗證許可只會考慮與請求相關的政策存放區中的政策子集。關聯性是根據政策的範圍來決定。範圍會識別政策適用的特定委託人和資源，以及委託人可在資源上執行的動作。定義範圍有助於透過縮小所考慮政策集來改善效能。

滿意的政策

滿意的政策是符合授權請求參數的政策。

Amazon Verified Permissions 與 Cedar 政策語言之間的差異

Amazon Verified Permissions 使用 Cedar 政策語言引擎來執行其授權任務。不過，原生 Cedar 實作與驗證許可中 Cedar 實作之間有一些差異。本主題識別這些差異。

命名空間定義

Cedar 的已驗證許可實作與原生 Cedar 實作有下列差異：

- 已驗證的許可在政策存放區中定義的結構描述中僅支援一個命名空間。
- 已驗證的許可不允許您建立空白字串或包含下列值的命名空間：aws、 amazon或 cedar。

政策範本支援

Verified Permissions 和 Cedar 都只允許 principal和 範圍內的預留位置resource。不過，已驗證的許可也要求 principal和 resource 都不會受到限制。

下列政策在 Cedar 中有效，但由於 principal不受限制，因此被驗證許可拒絕。

```
permit(principal, action == Action::"view", resource == ?resource);
```

下列兩個範例在 Cedar 和 Verified Permissions 中都有效，因為 principal和 resource 都有限制條件。

```
permit(principal == User::"alice", action == Action::"view", resource == ?resource);
```

```
permit(principal == ?principal, action == Action::"a", resource in ?resource);
```

結構描述支援

已驗證的許可要求所有結構描述 JSON 金鑰名稱都是非空白字串。Cedar 允許在少數情況下使用空字串，例如屬性或命名空間。

動作群組定義

Cedar 授權方法要求根據政策評估授權請求時，要考慮實體的清單。

您可以在結構描述中定義應用程式使用的動作和動作群組。不過，Cedar 不會將結構描述包含在評估請求中。相反地，Cedar 只會使用結構描述來驗證您提交的政策和政策範本。由於 Cedar 在評估請求期間不會參考結構描述，即使您已在結構描述中定義動作群組，您也必須將任何動作群組的清單納入實體清單中，才能傳遞給授權 API 操作。

Verified Permissions 會為您執行此操作。您在結構描述中定義的任何動作群組都會自動附加到您傳遞給 的實體清單，做為 IsAuthorized或 IsAuthorizedWithToken操作的參數。

實體格式

使用 entityList 參數的已驗證許可中實體的 JSON 格式與 Cedar 不同，方式如下：

- 在已驗證的許可中，JSON 物件必須將其所有鍵值對包裝在名為 的 JSON 物件中`Record`。
- Verified Permissions 中的 JSON 清單必須包裝在金鑰名稱為 `Set`且值為 Cedar 原始 JSON 清單的 JSON 鍵值對中。
- 對於 `String`、 `Long`和 `Boolean`類型名稱，來自 Cedar 的每個鍵/值對都會被驗證許可中的 JSON 物件取代。物件的名稱是原始金鑰名稱。在 JSON 物件中有一個索引鍵值對，其中索引鍵名稱是純量值的類型名稱 (`String`、 或 `Boolean`)`Long`，而值是來自 Cedar 實體的值。
- Cedar 實體和 Verified Permissions 實體的語法格式在下列方面有所不同：

Cedar 格式	已驗證的許可格式
`uid`	`Identifier`
`type`	`EntityType`
`id`	`EntityId`
`attrs`	`Attributes`
`parents`	`Parents`

Example - 清單

下列範例顯示實體清單如何分別以 Cedar 和 Verified Permissions 表示。

Cedar

```
[
  {
    "number": 1
  },
  {
    "sentence": "Here is an example sentence"
  },
  {
    "Question": false
  }
]
```

Verified Permissions

```
{
  "Set": [
    {
      "Record": {
        "number": {
          "Long": 1
        }
      }
    },
    {
      "Record": {
        "sentence": {
          "String": "Here is an example sentence"
        }
      }
    },
    {
      "Record": {
        "question": {
          "Boolean": false
        }
      }
    }
  ]
}
```

Example - 政策評估

下列範例顯示實體的格式如何分別評估 Cedar 和 Verified Permissions 中授權請求中的政策。

Cedar

```
[
    {
        "uid": {
            "type": "PhotoApp::User",
            "id": "alice"
        },
        "attrs": {
            "age": 25,
```

```
            "name": "alice",
            "userId": "123456789012"
        },
        "parents": [
            {
                "type": "PhotoApp::UserGroup",
                "id": "alice_friends"
            },
            {
                "type": "PhotoApp::UserGroup",
                "id": "AVTeam"
            }
        ]
    },
    {
        "uid": {
            "type": "PhotoApp::Photo",
            "id": "vacationPhoto.jpg"
        },
        "attrs": {
            "private": false,
            "account": {
                "__entity": {
                    "type": "PhotoApp::Account",
                    "id": "ahmad"
                }
            }
        },
        "parents": []
    },
    {
        "uid": {
            "type": "PhotoApp::UserGroup",
            "id": "alice_friends"
        },
        "attrs": {},
        "parents": []
    },
    {
        "uid": {
            "type": "PhotoApp::UserGroup",
            "id": "AVTeam"
        },
        "attrs": {},
```

```
        "parents": []
    }
]
```

Verified Permissions

```
[
    {
        "Identifier": {
            "EntityType": "PhotoApp::User",
            "EntityId": "alice"
        },
        "Attributes": {
            "age": {
                "Long": 25
            },
            "name": {
                "String": "alice"
            },
            "userId": {
                "String": "123456789012"
            }
        },
        "Parents": [
            {
                "EntityType": "PhotoApp::UserGroup",
                "EntityId": "alice_friends"
            },
            {
                "EntityType": "PhotoApp::UserGroup",
                "EntityId": "AVTeam"
            }
        ]
    },
    {
        "Identifier": {
            "EntityType": "PhotoApp::Photo",
            "EntityId": "vacationPhoto.jpg"
        },
        "Attributes": {
            "private": {
                "Boolean": false
            },
```

```
            "account": {
                "EntityIdentifier": {
                    "EntityType": "PhotoApp::Account",
                    "EntityId": "ahmad"
                }
            }
        },
        "Parents": []
    },
    {
        "Identifier": {
            "EntityType": "PhotoApp::UserGroup",
            "EntityId": "alice_friends"
        },
        "Parents": []
    },
    {
        "Identifier": {
            "EntityType": "PhotoApp::UserGroup",
            "EntityId": "AVTeam"
        },
        "Parents": []
    }
]
```

長度和大小限制

Verified Permissions 支援以政策存放區的形式儲存，以保留您的結構描述、政策和政策範本。該儲存體會導致 Verified Permissions 施加與 Cedar 無關的長度和大小限制。

物件	驗證許可限制（以位元組為單位）	Cedar 限制
政策大小1	10,000	無
內嵌政策描述	150	不適用於 Cedar
政策範本大小	10,000	無
結構描述大小	100,000	無

物件	驗證許可限制（以位元組為單位）	Cedar 限制
實體類型	200	無
政策 ID	64	無
政策範本 ID	64	無
實體 ID	200	無
政策存放區 ID	64	不適用於 Cedar

1 根據政策存放區中建立之政策的主體、動作和資源的合併大小，驗證許可中每個政策存放區的政策有限制。與單一資源相關的所有政策總大小不得超過 200，000 個位元組。對於範本連結政策，政策範本的大小只會計算一次，加上用於執行個體化每個範本連結政策的每組參數大小。

Amazon Verified Permissions 升級到 Cedar v4 常見問答集

Amazon Verified Permissions 正在更新至 Cedar v4。我們正在努力讓您盡可能無縫。下列FAQs應解答您的問題並協助您做好準備。

主題

- 升級的目前狀態為何？
- 我需要現在執行任何動作嗎？
- 升級主控台是否會影響授權服務？
- Cedar v3 和 Cedar v4 中的重大變更是什麼？
- 何時完成 Cedar v4 的升級？

升級的目前狀態為何？

首先，我們已升級主控台以使用 Cedar v4.3，但後端仍在 Cedar v2.5.0 上執行。這表示雖然您現在可以使用 主控台來編寫使用 `is`運算子等新功能的政策，但是當您嘗試儲存政策時，在我們完成升級之前，您仍然會收到錯誤。

我需要現在執行任何動作嗎？

否。 您可以視需要使用主控台開始探索 Cedar v4，但不需要執行任何動作。

升級主控台是否會影響授權服務？

否。 升級之前，我們會執行測試，以檢查您的政策存放區是否與 Cedar v4 正常運作。2.5.0 版和 4.3 版之間有一些輕微的重大變更，但您的政策存放區不太可能受到影響。如果是這樣，您的政策存放區將不會升級，並且會繼續使用 Cedar v2.5.0 進行授權。如果發生這種情況，我們將與您聯絡，說明您在升級之前需要進行的任何變更。

Cedar v3 和 Cedar v4 中的重大變更是什麼？

重大變更會在 Cedar 變更日誌中識別，並以 標記(*)。

> **Note**
>
> 如果您的政策存放區受到中斷變更的影響，則不會升級，我們會與您一起更新政策存放區，以便升級。

何時完成 Cedar v4 的升級？

我們的目標是在 2025 年 12 月 31 日之前升級所有帳戶。

Amazon Verified Permissions 使用者指南的文件歷史記錄

下表說明 Verified Permissions 的文件版本。

變更	描述	日期
新的 AWS 受管政策	您現在可以搭配 Verified Permissions 使用 `AmazonVerifiedPermissionsFullAccess` 和 `AmazonVerifiedPermissionsReadOnlyAccess` IAM 受管政策。	2024 年 10 月 11 日
OIDC 身分來源	您現在可以從 OpenID Connect (OIDC) 身分提供者授權使用者。	2024 年 6 月 8 日
使用身分來源字符的批次授權	您現在可以在單一 `BatchIsAuthorizedWithToken` API 請求中授權來自 Amazon Cognito 使用者集區的使用者。	2024 年 4 月 5 日
使用 API Gateway 建立政策存放區	您現在可以從現有的 API 和 Amazon Cognito 使用者集區建立政策存放區。	2024 年 4 月 1 日
內容概念和範例	新增了有關使用 Verified Permissions 授權請求中內容的資訊。	2024 年 2 月 1 日
授權概念和範例	新增了使用 Verified Permissions 授權請求的相關資訊。	2024 年 2 月 1 日
AWS CloudFormation 整合	Verified Permissions 支援在其中建立身分來源、政策、政策存放區和政策範本 AWS CloudFormation。	2023 年 6 月 30 日

初始版本	Amazon Verified Permissions 使用者指南的初始版本	2023 年 6 月 13 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。