開發人員指南

AWS Partner Central

AWS Partner Central: 開發人員指南

Table of Contents

..... vi
歡迎 1
使用AWS SDKs 1
設定和身分驗證 2
將您的AWS 帳戶連結至 Partner Central 2
設定 IAM 2
驗證 API 呼叫 3
使用自訂使用者代理程式簽署您的呼叫 4
Partner Central 代理程式 MCP 伺服器 6
概觀 6
代理程式功能 6
主要優點 7
使用範例 7
機會建立 7
機會複製 7
管道洞察 8
機會摘要 8
產生銷售遊戲 8
建立客戶設定檔 8
解決方案建議 9
資金建議 9
下一步建議 9
機會進展 9
合作夥伴加入的客服人員經驗 10
合作夥伴設定檔自動化 10
賣方設定指引 10
DHCP 合規指引 11
開始使用 11
開始使用 11
先決條件 11
步驟 1：設定 IAM 許可 12
步驟 2：連接您的 MCP 用戶端 18
使用 MCP 標頭簽署您的呼叫 20
步驟 3：驗證您的設定 21

步驟 4：執行您的第一個任務 22
安全考量 24
後續步驟 24
組態參考 25
Endpoint 25
IAM 許可 25
目錄環境 30
工作階段管理 30
檔案上傳 30
錯誤代碼 31
速率限制 32
串流 (SSE) 事件類型 32
後續步驟 33
工具參考 33
工具概觀 33
sendMessage 34
getSession 42
錯誤處理 43
關於 APIs 45
使用 API 45
使用帳戶 API 45
使用銷售 API 56
使用 優勢 API 105
使用頻道 API 117
使用營收測量 API 118
支援的區域 128
帳戶 API 的區域 129
銷售 API 的區域 129
優勢 API 的區域 129
頻道 API 的區域 129
營收測量 API 的區域 130
許可 130
帳戶 API 的存取權 131
銷售 API 的存取權 133
利益 API 的存取權 139
通道 API 的存取 141

營收測量 API 的存取權 149
配額 157
帳戶 API 的配額 158
銷售 API 的配額 160
優勢 API 的配額 162
頻道 API 的配額 164
營收測量 API 的配額 167
記錄 169
記錄帳戶 API 170
記錄銷售 API 173
記錄利益 API 174
記錄頻道 API 176
記錄營收測量 API 179
通知 181
AWS合作夥伴中央帳戶 API 的通知 181
AWS Partner Central Selling API 的通知 192
AWS Partner Central Benefits API 的通知 206
AWS Partner Central Channel API 的通知 218
在沙盒中測試 228
存取沙盒環境 228
沙盒環境的重要詳細資訊 228
在沙盒中測試帳戶 API 229
在沙盒中測試銷售 API 232
在沙盒中測試 優勢 API 236
在沙盒中測試頻道 API 240
在沙盒中測試營收測量 API 244
程式碼範例 249
基本概念 250
動作 250
案例 316
更新機會的關聯實體 316
版本備註 322
文件歷史紀錄 346

AWS Partner Central API 參考已重組。如需支援的 API 操作的詳細資訊，請參閱 AWS Partner Central API 參考。

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。

歡迎使用AWS Partner Central 開發人員指南

本開發人員指南說明如何AWS使用 服務 APIs 來整合和管理建置、行銷、銷售和成長活動AWS。

主題

- 搭配AWS SDK 使用AWS Partner Central API
- 設定和身分驗證

搭配AWS SDK 使用AWS Partner Central API

AWS軟體開發套件 SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK程式碼範例
AWS CLI	AWS CLI程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK程式碼範例

SDK 文件	代碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK程式碼範例

> **可用性範例**
>
> 找不到所需的內容嗎？ 請使用本頁面底部的提供意見回饋連結申請程式碼範例。

設定和身分驗證

使用AWS Partner Central API 設定和驗證需要三個步驟。以下是程序的概觀：

1. 將您的AWS Marketplace賣方帳戶連結至 Partner Central。
2. 使用 IAM 設定許可。
3. 使用 Signature 第 4 版 (SigV4) 驗證您的 API 呼叫。

將您的AWS 帳戶連結至 Partner Central

將您的 連結至AWS 帳戶 Partner Central 是使用 API 的先決條件。如需詳細資訊，請參閱將AWS Partner Central 帳戶與AWS Marketplace賣方帳戶連結。您必須使用具有聯盟領導或雲端管理員許可的帳戶登入 Partner Central，導覽至 **`Account Linking`**區段，然後依照提示操作。

設定 IAM

若要使用AWS Partner Central API，您需要AWS Identity and Access Management(IAM) 角色或 IAM 使用者才能開始撥打電話。如需詳細資訊，請參閱何時使用 IAM？。請遵循AWS 帳戶指南中建立 IAM 角色和建立 IAM 使用者的步驟。您必須在 Partner Central-linked AWS Marketplace Seller 帳戶中建立此 IAM 角色/使用者。建立 IAM 角色/使用者不會產生任何成本。

1. 建立 IAM 角色/使用者

   登入AWS 管理主控台，導覽至 IAM 服務，然後依照步驟建立 IAM 角色或 IAM 使用者。
2. 指派政策：

視需要連接受管政策或建立自訂政策。若要修改或展開許可，請將其他政策套用至 IAM 角色，而不是從 複製內容並將其AWSPartnerCentralOpportunityManagement與其他許可結合。避免複製 受管政策，因為這樣做會阻止您在新功能發行時自動存取這些功能，而且您未來必須手動更新您的政策。如需存取政策的詳細資訊，請參閱存取控制文件。

管理AWS Marketplace優惠

為了管理AWS Marketplace優惠並將其連結至機會，合作夥伴必須授予 IAM 角色存取目錄 APIs許可。確保角色/使用者具有許可，例如 `aws-marketplace:ListEntities`和 `aws-marketplace:SearchAgreements`。

驗證 API 呼叫

AWS Partner Central API 使用 Signature 第 4 版 (SigV4) 進行身分驗證。以下是實作方法：

使用AWS SDK

AWS SDKs會自動處理請求簽署。提供您的AWS登入資料，而 SDK 會執行其餘動作。

1. 對於 Java，請參閱提供暫時登入資料給適用於 Java 的AWS開發套件。
2. 對於 Python (Boto3)，請參閱登入資料。
3. 如需 JavaScript (Node.js)，請參閱在 Node.js 中設定登入資料。
4. 對於 .NET，請參閱登入資料和設定檔解析。
5. 如需其他程式設計語言和更多範例，請參閱在 上建置的工具AWS。

不使用AWS SDK 進行身分驗證

如果AWS SDK 不適用於您選擇的程式設計語言，身分驗證會涉及手動建立正式請求、簽署請求，以及處理工作階段字符。AWS會提供使用 SigV4 簽署的完整指引。不過，請注意，建議使用AWS SDK，因為手動請求簽署會增加複雜性，並且需要仔細管理安全字符。

awsJson1_0 通訊協定的每個請求都必須使用以下標頭，使用 HTTP "POST" 方法傳送至根 URL (/)。

必要標頭

awsJson1_0 通訊協定的每個 API 請求都必須使用以下標頭，使用 HTTP "POST" 方法傳送至根 URL (/)。

標頭	必要	說明
內容類型	true	此標頭的靜態值為 `application/x-amz-json-1.0` 。
內容長度	true	RFC 9110#section-8.6 定義的標準 Content-Length 標頭。
X-Amz-Target	請求為 true	例如， 服務`CreatePartner` 操作的值`PartnerCentralAccount` 為 `PartnerCentralAccount.CreatePartner` 。

使用自訂使用者代理程式簽署您的呼叫

向AWS Partner Central 提出 API 請求時，我們建議包含 `X-Amzn-User-Agent`標頭，以協助AWS識別用戶端應用程式的來源、監控用量和稽核效能。AWS使用此標頭來區分進行呼叫的用戶端應用程式的類型，並收集有關不同用戶端實作成功率的洞見。

自訂使用者代理程式標頭

標頭名稱： `X-Amzn-User-Agent`

目的：區分提出 API 請求的用戶端類型，並分類互動的來源。

格式：`{Integrator}|{Product}`

範例值： `AWS|AWS Partner CRM Connector`

在每個請求中包含此標頭AWS，可讓 分析請求模式、追蹤整合，並改善不同用戶端系統的 API 體驗。

在 SDKs 中使用自訂標頭

若要在 SDK 呼叫中包含 `X-Amzn-User-Agent`標頭，您可以在進行 API 呼叫之前修改用戶端請求行為。以下是使用AWS SDK for Python (Boto3) 的銷售 API 範例：

```
import boto3

# Define service and endpoint details
service_name = "partnercentral-selling"
```

```
endpoint_url = "https://partnercentral-selling.us-east-1.api.aws"

# Create a boto3 client for Partner Central
partner_central_client = boto3.client(
service_name=service_name,
='us-east-1',
endpoint_url=endpoint_url
)

# Function to add the custom User-Agent header

def add_version_header(params, **kwargs):
params["headers"]['X-Amzn-User-Agent'] = 'AWS|AWS Partner CRM Connector'

# Register the event to modify the request before the call is made
partner_central_client.meta.events.register(
f'before-call.{service_name}.*', add_version_header
)

# Now, whenever an API call is made using this client, the custom User-Agent header
 will be included
```

此範例示範如何在 Boto3 開發套件中註冊事件，以自動將 `X-Amzn-User-Agent`標頭附加到每個 API 請求。透過修改其個別的請求攔截機制，可以將相同的方法套用至其他AWS SDKs。

> **Note**
>
> `X-Amzn-User-Agent` 標頭格式已簡化。我們建議您使用新的格式。先前的格式 (`CompanyName|ProductName|CRMName|ProductVersion`) 仍支援回溯相容性。現有的整合將繼續運作，無需修改。

Partner Central 代理程式 MCP 伺服器

Partner Central 代理程式 MCP Server 透過模型內容協定 (MCP) 提供 Partner Central 工具，讓您的 AI 代理程式和工具透過自然語言探索機會管理、客戶洞察和資金計劃並與之互動。

伺服器會針對寫入操作處理身分驗證、工作階段管理和human-in-the-loop，以維持控制，同時簡化工作流程。

概觀

Partner Central 代理程式 MCP Server 是一項全受管託管服務，可將 MCP 相容 AI AWS用戶端連線至 Partner Central 代理程式。它透過 HTTPS 搭配 SigV4 身分驗證使用 JSON-RPC 2.0，並支援伺服器傳送事件 (SSE) 進行即時串流回應。

伺服器支援多迴轉對話、文件分析的檔案附件，以及在任何寫入操作執行之前需要您明確同意的內建核准工作流程。

代理程式功能

- 管道洞察 — 取得有關銷售管道的對話智慧，包括風險機會、階段進展和封閉損失分析
- 機會建立 — 透過自然語言對話、上傳會議備註、提案或通話文字記錄，或複製現有的機會，來建立新的機會
- 機會摘要 — 產生涵蓋階段、支出、結束日期等任何交易的簡潔、at-a-glance摘要
- 銷售遊戲產生 — 建立結合機會詳細資訊、產業內容和AWS解決方案建議的自訂銷售策略
- 建立客戶設定檔 — 使用涵蓋產業、商業模型、地理位置和最新發展的公開可用資訊產生公司設定檔
- 解決方案建議 — 根據機會要求交叉參考您的已註冊解決方案，以尋找最佳相符項目
- 資金建議：根據可用的AWS資金計劃評估機會、預估金額，以及建立資金請求
- 後續步驟建議 — 根據AWS共同銷售標準和階段進展指引，取得優先行動計劃
- 機會進展 — 透過管道階段上傳支援文件、擷取相關資料和進展機會
- AI 輔助產品清單：從您現有的數位資產產生高品質的AWS Marketplace產品清單、根據AWS Marketplace標準對清單強度進行評分，以及接收欄位層級建議以改善可探索性。如需詳細資訊，請參閱《 AWS Marketplace賣方指南》中的 AI 輔助產品清單。

主要優點

- 對 Partner Central 的對話存取 — 以自然語言提出問題，而不是導覽複雜的主控台工作流程
- 更多時間銷售 — 透過簡短的對話而非完成多步驟表單來建立機會，從而減少資料輸入，並讓合作夥伴銷售團隊花更多時間銷售，客服人員建議改進，以便合作夥伴提交更高品質的機會並改善管道衛生
- Human-in-the-loop安全 — 所有寫入操作都需要您的明確核准才能執行
- 多轉對話 — 精簡工作階段中的查詢，無需重複內容
- 檔案分析 — 連接代理程式的文件 (PDF、DOCX、XLSX、CSV、映像），以與您的問題一起分析
- 集中式存取管理 — 透過具有精細許可的 IAM 政策控制存取
- 沙盒測試 — 在隔離的沙盒環境中測試工作流程，然後再接觸生產資料
- 串流回應 — 在代理程式處理您的請求時，透過 SSE 取得意見回饋

使用範例

所有 AI 產生的洞見都包含工作階段 ID 以進行可追蹤性。資料會與登入合作夥伴自己的機會隔離。所有內容都具有明確的公開標籤，並受AWS負責任的 AI 政策管理。

機會建立

代理程式會從自然語言描述或上傳的文件 (PDF、DOCX、XLSX、TXT) 建立新的機會。它會擷取客戶名稱、專案範圍、預期結束日期和其他必要欄位、從公開可用的 Web 資料中豐富客戶詳細資訊、根據 AWS 整備要求驗證草稿，並在核准後透過 Partner Central Selling API 建立機會。

- 「為 Acme Corp 創造機會 — 他們想要將其資料倉儲遷移到 Redshift，目標結束日期為 Q3，預計每月 AWS 支出 40，000 美元」
- 「這是我昨天與 GlobalTech 會議的通話備註 — 從此文字記錄建立機會」
- 「使用此提案 PDF 來為客戶的 SAP 遷移建立機會」

機會複製

客服人員從現有客戶或專案詳細資訊中建立新機會、合併您提供的新客戶或專案詳細資訊，並在提交之前驗證至少一個差異化欄位已變更，因此 AWS 檢閱不會拒絕重複項目。

- 「為新客戶複製機會 O1234567890 — Globex、相同工作負載、目標結束日期結束於Q4」

- 「建立 O9876543210 等機會，但為客戶的生產環境而非沙盒建立機會」

管道洞察

取得有關銷售管道的對話式智慧。代理程式會分析階段進展、截止日期、停滯交易和封閉損失模式，以呈現最重要的內容。

- 「本週哪些機會需要我注意？」
- 「下個月有多少機會結束？」
- 「在過去 6 個月內，我們失去機會的主要原因是什麼？」

機會摘要

客服人員會將公司名稱、產業、階段、預期每月AWS支出、目標關閉日期和其他金鑰詳細資訊合成為簡潔摘要，無需掃描個別表單欄位。

- 「給我機會 O1234567890 的摘要」
- 「Acme Corp 交易的目前狀態和金鑰詳細資訊是什麼？」

產生銷售遊戲

客服人員透過將機會的詳細資訊、客戶產業內容和相關AWS解決方案建議結合到ready-to-use銷售中，來建置自訂的銷售策略。

- 「為機會 O1234567890 產生銷售活動」
- 「銷售雲端遷移給此金融服務客戶的最佳方法是什麼？」
- 「為我建立 GlobalTech 資料分析交易的銷售策略」

建立客戶設定檔

代理程式會使用公開的資訊產生公司設定檔，涵蓋產業分類、商業模型 (B2B/B2C/混合）、地理位置、公司規模、市場焦點和最近的業務發展。設定檔會標記為「使用公開可用的資料和AWS AI 洞見產生」。

- 「為 Acme Corp 建立客戶設定檔」
- 「我們對此客戶的產業和商業模式有何了解？」

- 「為我即將與 GlobalTech 進行的會議整理公司概觀」

解決方案建議

客服人員會根據機會需求交叉參考您註冊的解決方案，顯示解決方案名稱、描述，以及是否已連接到機會。

- 「哪個解決方案最符合機會 O1234567890？」
- 「我們的資料分析解決方案是否已連接到此交易？」
- 「建議客戶遷移其 SAP 工作負載的解決方案」

資金建議

客服人員會根據機會詳細資訊和計劃資格條件，針對可用的AWS資金計劃評估每個機會。找到相符項目時，會顯示程式名稱、描述和詳細推理。然後，您可以估計資金金額、建立自動填入的資金請求草稿，或以對話方式進一步了解計劃。SCA（策略協作協議）預算可用性會在相關時顯示，且所有動作都遵守 IAM 許可。

- 「我是否符合機會 O6789012345 的任何資金計劃的資格？」
- 「與此客戶一起估算 POC 的資金金額」
- 「為此機會建立 MAP 利益應用程式」

下一步建議

代理程式會根據AWS階段進展指引評估機會、將目前資料與合格機會的條件進行比較，並準確識別您仍然需要收集的資訊。結果是以AWS共同銷售標準為基礎的優先行動計劃。

- 「接下來我需要做什麼才能獲得機會 O1234567890？」
- 「這個機會準備好提交了嗎？ 缺少哪些欄位？」
- 「將此交易從 Prospect 移至 Qualified 的要求是什麼？」

機會進展

客服人員接受支援文件（會議記錄、通話備註、電子郵件摘要）、擷取相關資訊、映射至機會欄位、評估階段需求，以及更新機會。如果差距仍然存在，則會傳回滿足需求與不滿意需求的明細。

- 「這是我的通話備註 — 使用相關詳細資訊更新機會 O1234567890」
- 「我正在連接會議文字記錄。根據我們討論的內容發展這個機會。」
- 「檢閱此電子郵件摘要並告知我其滿足的機會欄位」

合作夥伴加入的客服人員經驗

客服人員可自動化 Partner Central 的合作夥伴加入，並透過自然語言提供 Marketplace 賣方設定和遵循 GDPR 的引導式協助。

代理程式功能：

- 合作夥伴設定檔自動化：掃描您的網站、自動填入您的合作夥伴設定檔，以及管理可見性、聯盟潛在客戶聯絡人、訓練網域和帳戶連線
- 賣方設定指引 — Marketplace 賣方註冊的Step-by-step指引，包括稅務表單、銀行、合規 (KYC/BAV/SU)、ESC 目錄和服務連結角色
- // 合規指引 — 評估 // 準備程度、擷取用於營收標記的產品代碼，以及驗證附屬公司帳戶連線的合併營收歸因

合作夥伴設定檔自動化

代理程式會掃描您的公司網站以擷取並自動填入您的合作夥伴設定檔，然後引導您完成任何剩餘的差距。它可以更新設定檔欄位、變更可見性、管理您的聯盟潛在客戶聯絡人、連結訓練認證網域，以及處理帳戶連線邀請。

- 「您可以查看我的網站並填寫我的設定檔嗎？」
- 「公開我們的設定檔，讓 AWS 客戶可以找到我們」
- 「在 【email】 將我們的聯盟潛在客戶聯絡人更新至 【name】」
- 「連接我們的 EMEA 附屬公司帳戶」

賣方設定指引

客服人員會逐步解說 Marketplace 賣方end-to-end、根據您的國家/地區和實體類型判斷正確的稅務表單、依區域說明銀行和支出設定，以及識別適用於您的合規要求 (KYC、銀行帳戶驗證、次要使用者驗證）。

- 「我想要開始在 Marketplace 上銷售 - 我應該從哪裡開始？」

- 「我需要什麼稅務表單？ 我是德國的公司
- 「我需要客戶開發中心嗎？ 我正在向法國的客戶銷售 」
- 「如果我在美國境外，如何設定付款？」

DHCP 合規指引

代理程式會檢查您的 帳戶是否已就緒 – 驗證您的帳戶是否已連線、清單是否存在，以及您的產品代碼是否可供標記。對於與附屬公司帳戶的合作夥伴，它會驗證所有賣方帳戶是否已正確連結合併收入歸因。

- 「我是否相容於 MCU？」
- 「在哪裡可以找到要標記的產品碼？」
- 「告知我我需要執行的所有步驟，才能符合 PMP 規範」

開始使用

準備好設定 Partner Central 代理程式 MCP Server？ 如需step-by-step設定說明，請前往 入門指南。

Partner Central Agent MCP Server 入門

本指南將逐步引導您使用自訂 MCP 用戶端設定 Partner Central Agent MCP 伺服器的程式設計存取權。伺服器使用直接 HTTPS 搭配 SigV4 身分驗證 - 不需要代理或 IDE 外掛程式。

先決條件

開始之前，請確定您已：

- 作用中的 Partner Central 帳戶 （ 遷移至AWS主控台 ）
- 具有 Partner Central IAM 許可AWS的帳戶
- AWS使用登入資料安裝和設定的 CLI
- 存取 us-east-1 （ 維吉尼亞北部 ） 區域
- HTTPS 連線至 `partnercentral-agents-mcp.us-east-1.api.aws`
- HTTP 用戶端中的 TLS 1.2+ 支援
- 支援 JSON-RPC 2.0 和 SigV4 請求簽署的 MCP 相容用戶端

步驟 1：設定 IAM 許可

Partner Central Agent MCP Server 需要兩個層級的 IAM 許可：通訊協定存取（與 MCP 端點通訊）和資料存取（執行 Partner Central 操作）。

連接 IAM 政策

若要使用AWS管理主控台將政策連接至您的 IAM 身分：

1. 開啟 IAM 主控台。
2. 在左側導覽窗格中，根據您要連接政策的身分選擇使用者、使用者群組或角色，然後選擇特定使用者、群組或角色的名稱。
3. 選擇許可索引標籤。
4. 選擇連接政策（如果是第一次新增許可)。
5. 在政策清單中，搜尋並選取您要連接的受管政策（例如，您從下列 JSON 區塊建立的自訂政策）。
6. 選擇連接政策（或下一步，然後選擇新增許可) 以確認。

許可會立即生效。您可以將多個政策連接到相同的身分。

建議：使用 受管政策

授予 MCP 通訊協定存取權的最簡單方法是將 `AWSMcpServiceActionsFullAccess`受管政策連接至您的 IAM 身分。此政策包含與 MCP 伺服器互動所需的所有許可。

對於精細控制，您可以使用 IAM 政策中的 `aws:IsMcpServiceAction`條件索引鍵來限定 MCP 服務動作的許可範圍。

MCP 通訊協定存取的最低許可

您的 IAM 身分至少需要此動作才能與 MCP 伺服器互動：

Action	說明
`partnercentral:UseSession`	建立、更新和擷取對話工作階段時需要

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "partnercentral:UseSession"
            ],
            "Resource": "*",
   "Condition": {
                "Bool": {
                    "aws:IsMcpServiceAction": "true"
                }
            }
        }
    ]


}
```

資料存取許可

若要透過 代理程式實際執行 Partner Central 操作，您需要根據使用案例的其他許可。

機會管理：

```
{
    "Effect": "Allow",
    "Action": [
        "partnercentral:List*",
        "partnercentral:Get*",
        "partnercentral:CreateOpportunity",
        "partnercentral:UpdateOpportunity",
        "partnercentral:SubmitOpportunity",
        "partnercentral:AssignOpportunity",
        "partnercentral:AssociateOpportunity",
        "partnercentral:DisassociateOpportunity"
    ],
    "Resource": "*"
}
```

資金計劃：

```
{
    "Effect": "Allow",
    "Action": [
```

```
        "partnercentral:ListBenefitAllocations",
        "partnercentral:ListBenefitApplications",
        "partnercentral:CreateBenefitApplication",
        "partnercentral:GetBenefitApplication",
        "partnercentral:UpdateBenefitApplication",
        "partnercentral:SubmitBenefitApplication",
        "partnercentral:AmendBenefitApplication",
        "partnercentral:CancelBenefitApplication",
        "partnercentral:RecallBenefitApplication",
        "partnercentral:AssociateBenefitApplicationResource",
        "partnercentral:DisassociateBenefitApplicationResource"
    ],
    "Resource": "*"
}
```

Marketplace 存取：

```
{
    "Effect": "Allow",
    "Action": [
        "aws-marketplace:DescribeEntity",
        "aws-marketplace:DescribeAgreement",
        "aws-marketplace:SearchAgreements",
        "aws-marketplace:ListEntities"
    ],
    "Resource": "*"
}
```

合作夥伴中央上線：

```
{
    "Effect": "Allow",
    "Action": [
        "partnercentral:ListPartners",
        "partnercentral:GetPartner",
        "partnercentral:GetProfileVisibility",
        "partnercentral:GetAllianceLeadContact",
        "partnercentral:GetProfileUpdateTask",
        "partnercentral:StartProfileUpdateTask",
        "partnercentral:CancelProfileUpdateTask",
        "partnercentral:PutProfileVisibility",
        "partnercentral:PutAllianceLeadContact",
        "partnercentral:SendEmailVerificationCode",
```

```
        "partnercentral:AssociateAwsTrainingCertificationEmailDomain",
        "partnercentral:DisassociateAwsTrainingCertificationEmailDomain",
        "partnercentral:GetAccountConnections",
        "partnercentral:GetConnectionInvitations",
        "partnercentral:CreateConnectionInvitation",
        "partnercentral:CancelConnectionInvitation",
        "partnercentral:RespondConnectionInvitation",
        "partnercentral:CancelConnection",
        "partnercentral:ManageConnectionPreferences"
    ],
    "Resource": "*"
}
```

對於 Marketplace 賣方設定，也新增：

```
{
    "Effect": "Allow",
    "Action": [
        "aws-marketplace:ListEntities",
        "aws-marketplace:DescribeEntity",
        "aws-marketplace:DescribeChangeSet",
        "aws-marketplace:StartChangeSet"
    ],
    "Resource": "*"
}
```

對於服務連結角色管理 (Resale Authorizations/CPPO)：

```
{
    "Effect": "Allow",
    "Action": [
        "iam:GetRole",
        "iam:CreateServiceLinkedRole"
    ],
    "Resource": "*"
}
```

完整存取政策

對於開發和測試，您可以將所有許可合併為單一政策：

```
aws iam create-policy \
```

```
    --policy-name PartnerCentralAgentsFullAccess \
    --policy-document '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "partnercentral:UseSession",
                    "partnercentral:List*",
                    "partnercentral:Get*",
                    "partnercentral:CreateOpportunity",
                    "partnercentral:UpdateOpportunity",
                    "partnercentral:SubmitOpportunity",
                    "partnercentral:AssignOpportunity",
                    "partnercentral:AssociateOpportunity",
                    "partnercentral:DisassociateOpportunity",
                    "partnercentral:CreateResourceSnapshot",
                    "partnercentral:CreateResourceSnapshotJob",
                    "partnercentral:StartResourceSnapshotJob",
                    "partnercentral:CreateEngagement",
                    "partnercentral:CreateEngagementInvitation",
                    "partnercentral:RejectEngagementInvitation",
                    "partnercentral:StartEngagementByAcceptingInvitationTask",
                    "partnercentral:StartEngagementFromOpportunityTask",
                    "partnercentral:CreateBenefitApplication",
                    "partnercentral:UpdateBenefitApplication",
                    "partnercentral:SubmitBenefitApplication",
                    "partnercentral:AmendBenefitApplication",
                    "partnercentral:CancelBenefitApplication",
                    "partnercentral:RecallBenefitApplication",
                    "partnercentral:AssociateBenefitApplicationResource",
                    "partnercentral:DisassociateBenefitApplicationResource",
                    "partnercentral:ListPartners",
                    "partnercentral:StartProfileUpdateTask",
                    "partnercentral:CancelProfileUpdateTask",
                    "partnercentral:PutProfileVisibility",
                    "partnercentral:PutAllianceLeadContact",
                    "partnercentral:SendEmailVerificationCode",
                    "partnercentral:AssociateAwsTrainingCertificationEmailDomain",
                    "partnercentral:DisassociateAwsTrainingCertificationEmailDomain",
                    "partnercentral:CreateConnectionInvitation",
                    "partnercentral:CancelConnectionInvitation",
                    "partnercentral:RespondConnectionInvitation",
                    "partnercentral:CancelConnection",
```

```
                    "partnercentral:ManageConnectionPreferences"
                ],
                "Resource": "*"
            },
            {
                "Effect": "Allow",
                "Action": [
                    "aws-marketplace:DescribeEntity",
                    "aws-marketplace:DescribeAgreement",
                    "aws-marketplace:SearchAgreements",
                    "aws-marketplace:ListEntities",
                    "aws-marketplace:DescribeChangeSet",
                    "aws-marketplace:StartChangeSet"
                ],
                "Resource": "*"
            },
            {
                "Effect": "Allow",
                "Action": [
                    "iam:GetRole",
                    "iam:CreateServiceLinkedRole"
                ],
                "Resource": "*"
            }
        ]
    }'
```

唯讀政策

對於生產環境或唯讀使用案例，限制讀取操作的許可：

```
aws iam create-policy \
    --policy-name PartnerCentralAgentReadOnly \
    --policy-document '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "partnercentral:UseSession",
                    "partnercentral:List*",
                    "partnercentral:Get*",
                    "partnercentral:ListPartners",
```

```
                "partnercentral:GetPartner",
                "partnercentral:GetProfileVisibility",
                "partnercentral:GetAllianceLeadContact",
                "partnercentral:GetProfileUpdateTask",
                "partnercentral:GetAccountConnections",
                "partnercentral:GetConnectionInvitations"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:DescribeEntity",
                "aws-marketplace:DescribeAgreement",
                "aws-marketplace:SearchAgreements",
                "aws-marketplace:ListEntities",
                "aws-marketplace:DescribeChangeSet"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole"
            ],
            "Resource": "*"
        }
    ]
}'
```

步驟 2：連接您的 MCP 用戶端

Partner Central Agent MCP Server 使用直接 HTTPS 搭配 SigV4 請求簽署。沒有代理層 — MCP 用戶端會將 JSON-RPC 2.0 請求直接傳送至端點。

Endpoint

```
https://partnercentral-agents-mcp.us-east-1.api.aws/mcp
```

身分驗證

所有請求都必須使用 AWS Signature 第 4 版簽署：

- 服務名稱： partnercentral-agents-mcp
- 區域: us-east-1

初始化 MCP 連線

傳送建立通訊協定的initialize請求：

```
{
    "jsonrpc": "2.0",
    "id": 1,
    "method": "initialize",
    "params": {
        "protocolVersion": "2025-03-26",
        "capabilities": {},
        "clientInfo": {
            "name": "my-partner-client",
            "version": "1.0.0"
        }
    }
}
```

預期回應：

```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "protocolVersion": "2025-03-26",
        "capabilities": {
            "tools": {
                "listChanged": false
            }
        },
        "serverInfo": {
            "name": "PartnerCentralAgentMCPServer",
            "version": "1.0.0"
        }
    }
}
```

列出可用的工具

```
{
    "jsonrpc": "2.0",
    "id": 2,
    "method": "tools/list",
    "params": {}
}
```

使用 MCP 標頭簽署您的呼叫

向 Partner Central 代理程式 MCP 提出請求時，我們建議您使用下列方法來包含自訂 MCP 標頭，以協助AWS識別用戶端應用程式的來源、監控用量和稽核效能。AWS使用此標頭來區分進行呼叫的用戶端應用程式的類型，並收集不同用戶端實作成功率的洞見。

方法 1：_meta 欄位 （程式設計/無狀態 MCP)

對於直接建構 MCP `tools/call`請求的程式碼，請在請求上提供 `_meta` 欄位。

```
{
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "content": [
                {
                    "type": "text",
                    "text": "List my open opportunities with expected close date in Q1
 2026"
                }
            ],
            "catalog": "AWS"
        },
        "_meta": {
            "integrator": "<Integrator's Company Name / Direct>",
            "sourceProduct": "<Integrator's Application Name>"
        }
    }
}
```

方法 2： clientInfo （工作階段型自訂代理程式）

對於建立工作階段的自訂 MCP 用戶端，請在 clientInfo 欄位內提供 MCP 標頭資訊：

```
{
    "method": "initialize",
    "params": {
        "protocolVersion": "2024-11-05",
        "clientInfo": {
            "integrator": "<Integrator's Company Name / Direct>",
            "sourceProduct": "<Integrator's Application Name>"
        }
    }
}
```

中的欄位clientInfo：

- integrator — 合作夥伴的公司名稱或「直接」

  範例：AWS
- sourceProduct — 產品/代理程式名稱

  範例：AWS CRM Connector

方法 3：URL 參數 （僅限託管 MCP)

僅適用於整合器無法修改通訊協定欄位的託管 MCP 用戶端。使用 URL 參數：

伺服器 URL： https://mcp.partnercentral.aws?appId=<Integrator's Company Name / Direct>

步驟 3：驗證您的設定

傳送簡單的訊息以確認一切正常運作。使用 Sandbox目錄進行測試：

```
{
    "jsonrpc": "2.0",
    "id": 3,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
```

```
        "arguments": {
            "content": [
                {
                    "type": "text",
                    "text": "Hello, what can you help me with?"
                }
            ],
            "catalog": "Sandbox"
        }
    }
}
```

如果您收到 的回應"status": "complete"和客服人員的文字回覆，表示您的設定正常運作。回應也會包含sessionId可用於後續訊息的 。

步驟 4：執行您的第一個任務

查詢您的機會

```
{
    "jsonrpc": "2.0",
    "id": 4,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
            "content": [
                {
                    "type": "text",
                    "text": "List my open opportunities with expected revenue over
 $50K"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

檢查資金資格

```
{
```

```
    "jsonrpc": "2.0",
    "id": 5,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
            "content": [
                {
                    "type": "text",
                    "text": "Am I eligible for MAP funding for opportunity
 O1234567890?"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

擷取工作階段歷史記錄

```
{
    "jsonrpc": "2.0",
    "id": 6,
    "method": "tools/call",
    "params": {
        "name": "getSession",
        "arguments": {
            "sessionId": "session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
            "catalog": "AWS"
        }
    }
}
```

合作夥伴加入

```
{
    "jsonrpc": "2.0",
    "id": 7,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
```

```
        "arguments": {
            "sessionId": "session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
            "content": [
                {
                    "type": "text",
                    "text": "Help me onboard to Partner Central"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

要嘗試的其他加入任務：

- 「引導我完成稅務面試程序」
- 「您可以查看我的網站並填寫我的合作夥伴設定檔嗎？」
- 「我還需要做什麼才能準備好在 Marketplace 上銷售？」

安全考量

- 請勿透過 MCP 工具參數傳遞AWS登入資料。身分驗證由傳輸層的 SigV4 請求簽署處理。
- 使用沙盒目錄進行測試和開發。"Sandbox" 目錄提供不會影響生產合作夥伴資料的隔離環境。
- 在生產環境中套用最低權限的 IAM 政策。使用唯讀政策來監控和報告使用案例。只有在使用者需要更新機會或提交資金應用程式時，才授予寫入許可。
- 仔細檢閱寫入操作。伺服器對所有寫入操作使用human-in-the-loop核准。建議寫入動作時，請在核准之前檢閱參數。
- 工作階段資料是暫時性的。工作階段會在建立後 48 小時過期。請勿依賴工作階段進行長期資料儲存。
- 檔案上傳會移至暫時性 S3 儲存貯體。上傳的檔案會暫時儲存，不會永久保留。請勿上傳包含登入資料、秘密或其他敏感資訊的檔案。

後續步驟

- 組態參考 — 端點、IAM 動作、工作階段管理和錯誤碼的完整參考
- 工具參考 — sendMessage和 getSession工具的詳細文件

組態參考

此頁面提供連線至 和設定 Partner Central Agent MCP Server 的完整參考。

Endpoint

屬性	Value
URL	`https://partnercentral-agents-mcp.us-east-1.api.aws/mcp`
區域	`us-east-1` 僅限（維吉尼亞北部）
通訊協定	透過 HTTPS 的 JSON-RPC 2.0
串流	伺服器傳送事件 (SSE)
身分驗證	AWS Signature 第 4 版
SigV4 服務名稱	`partnercentral-agents-mcp`
TLS 要求	TLS 1.2 或更新版本

IAM 許可

除非另有說明，否則所有 IAM 動作都會使用 `partnercentral:` 字首。

MCP 通訊協定存取

授予 MCP 通訊協定存取權的最簡單方法是將 `AWSMcpServiceActionsFullAccess`受管政策連接至您的 IAM 身分。此政策包含與 MCP 伺服器互動所需的所有許可。對於精細控制，您可以使用 IAM 政策中的 `aws:IsMcpServiceAction`條件索引鍵來限定 MCP 服務動作的許可範圍。

Action	說明
`partnercentral:UseSession`	建立、更新和擷取對話工作階段

機會管理

Action	說明
partnercentral:List*	列出機會、解決方案和其他 Partner Central 資源
partnercentral:Get*	擷取個別機會和其他資源的詳細資訊
partnercentral:CreateOpportunity	建立機會
partnercentral:UpdateOpportunity	修改機會欄位（舞台、結束日期、收入等）
partnercentral:SubmitOpportunity	提交AWS檢閱的機會
partnercentral:AssignOpportunity	指派機會給合作夥伴代表
partnercentral:AssociateOpportunity	將機會連結到另一個資源（解決方案等）
partnercentral:DisassociateOpportunity	移除機會與其他資源之間的連結
partnercentral:StartEngagementFromOpportunityTask	提交從機會開始參與的機會

資金計劃

Action	說明
partnercentral:ListBenefitAllocations	列出您帳戶的可用利益分配
partnercentral:ListBenefitApplications	列出提交的 利益應用程式
partnercentral:CreateBenefitApplication	建立新的利益應用程式 (MAP、POC、WMP)

Action	說明
partnercentral:GetBenefitApplication	擷取特定利益應用程式的詳細資訊
partnercentral:UpdateBenefitApplication	更新待定利益應用程式
partnercentral:AssociateBenefitApplicationResource	將資源連結至利益應用程式
partnercentral:DisassociateBenefitApplicationResource	從利益應用程式移除資源連結

Marketplace 存取

Action	說明
aws-marketplace:DescribeEntity	擷取市場實體的詳細資訊
aws-marketplace:SearchAgreements	搜尋市場協議
aws-marketplace:ListEntities	列出市集實體

加入代理程式

合作夥伴帳戶管理 (**partnercentral**)：

Action	說明
partnercentral:ListPartners	列出帳戶的合作夥伴註冊
partnercentral:GetPartner	擷取合作夥伴註冊和設定檔詳細資訊
partnercentral:GetProfileVisibility	擷取目前的設定檔可見性 (PUBLIC/PRIVATE)

Action	說明
partnercentral:GetAllianceLeadContact	擷取聯盟潛在客戶聯絡資訊
partnercentral:GetProfileUpdateTask	檢查待定非同步設定檔更新的狀態
partnercentral:StartProfileUpdateTask	提交合作夥伴設定檔更新（非同步）
partnercentral:CancelProfileUpdateTask	取消待定的設定檔更新任務
partnercentral:PutProfileVisibility	將設定檔可見性變更為 PUBLIC 或 PRIVATE
partnercentral:PutAllianceLeadContact	更新聯盟主要聯絡人（名稱、標題、電子郵件）
partnercentral:SendEmailVerificationCode	為以電子郵件為基礎的網域/聯絡人變更傳送驗證碼
partnercentral:AssociateAwsTrainingCertificationEmailDomain	連結用於訓練認證追蹤的電子郵件網域
partnercentral:DisassociateAwsTrainingCertificationEmailDomain	移除連結的訓練認證電子郵件網域
partnercentral:GetAccountConnections	列出作用中的帳戶連線（例如，附屬公司連結）
partnercentral:GetConnectionInvitations	列出待傳送和收到的連線邀請
partnercentral:CreateConnectionInvitation	將連線邀請傳送至另一個AWS帳戶

Action	說明
partnercentral:CancelConnectionInvitation	取消待定的傳出連線邀請
partnercentral:RespondConnectionInvitation	接受或拒絕傳入連線邀請
partnercentral:CancelConnection	取消作用中的帳戶連線
partnercentral:ManageConnectionPreferences	取得或更新連線帳戶之間的共用偏好設定

Marketplace 賣方設定 (**aws-marketplace**)：

Action	說明
aws-marketplace:ListEntities	列出 Marketplace 實體（例如賣方實體）
aws-marketplace:DescribeEntity	擷取 Marketplace 實體的完整詳細資訊
aws-marketplace:DescribeChangeSet	檢查已提交變更集的狀態
aws-marketplace:StartChangeSet	提交 Marketplace 變更集（例如，賣方設定檔更新）

IAM — 服務連結角色 (**iam**)：

Action	說明
iam:GetRole	檢查 Marketplace 轉售授權服務連結角色是否存在
iam:CreateServiceLinkedRole	建立AWSServiceRoleForMarketplaceResaleAuthorization 角色

目錄環境

目錄	說明
"AWS"	生產環境。所有操作都會影響即時合作夥伴資料。
"Sandbox"	隔離的測試環境。不會影響生產資料。使用 進行開發和驗證。

工作階段管理

屬性	Value
工作階段 ID 格式	UUID v4 加上session-字首 （例如 session-550e8400-e29b-41d4-a716-446655440000 ）
工作階段建立	在不使用 的第一次sendMessage 呼叫時自動執行 sessionId
工作階段到期	工作階段建立後 48 小時 （絕對過期，非閒置型）

檔案上傳

屬性	Value
每則訊息的最大檔案數	3
影像大小限制	3.75 MB
文件大小限制	4.5 MB
允許擴充功能	doc, docx, pdf, png, jpeg, xlsx, csv, txt

屬性	Value
S3 儲存貯體 （生產）	aws-partner-central-marketplace-ephemeral-writeonly-files
上傳路徑	s3：//{bucket}/{aws-account-id}/
S3 URI 需求	必須包含`versionId` 查詢參數

上傳工作流程

1. 將檔案上傳至AWS帳戶 ID 字首下適當的 S3 儲存貯體。
2. 請注意 S3 URI，包括上傳`versionId`傳回的 。
3. 在`sendMessage`呼叫中包含 檔案做為`document`內容區塊。

S3 URI 格式範例：

```
s3://aws-partner-central-marketplace-ephemeral-writeonly-files/123456789012/my-document.pdf?versionId=abc123
```

錯誤代碼

Code	名稱	描述
`-32001`	`AUTHENTICATION_FAILURE`	SigV4 簽章無效或登入資料已過期
`-31004`	`TOOL_PERMISSION_DENIED`	IAM 身分缺少此操作所需的`partnercentral:` 動作
`-32002`	`ACCESS_DENIED`	一般存取遭拒（帳戶未註冊、區域不相符等）
`-32004`	`LIMIT_EXCEEDED`	超出速率限制或配額。以指數退避重試。

Code	名稱	描述
-30001	RESOURCE_NOT_FOUND	請求的資源（工作階段、機會等）不存在
-32600	INVALID_REQUEST	格式錯誤的 JSON-RPC 請求或無效的參數
-32603	INTERNAL_ERROR	伺服器端錯誤。重試 請求。

速率限制

伺服器會強制執行每個帳戶速率限制：

作業	持續費率	爆量
sendMessage	每分鐘 2 個請求	10
所有其他操作	每分鐘 10 個請求	20

如果您超過這些限制，伺服器會傳回錯誤碼 -32004(LIMIT_EXCEEDED)。在用戶端中實作具有抖動的指數退避，以正常處理速率限制。

串流 (SSE) 事件類型

在 sendMessage 呼叫true中將 stream 設定為 時，伺服器會傳回具有下列事件類型的伺服器傳送事件：

事件類型	說明
stream_start	已建立串流連線
assistant-response.start	客服人員已開始產生回應
assistant-response.delta	代理程式回應的增量文字區塊
assistant-response.completed	客服人員已完成產生回應

事件類型	說明
server-tool-use	客服人員正在叫用內部工具 （讀取操作）
server-tool-response	內部工具調用的結果
tool_approval_request	客服人員正在請求寫入操作的人工核准
stream_end	串流連線關閉
done	指出 SSE 串流完成的最終事件

後續步驟

- 工具參考 — sendMessage和 getSession工具的詳細文件

工具參考

Partner Central Agent MCP Server 會公開兩種 MCP 工具：sendMessage用於所有客服人員互動，以及getSession用於擷取工作階段狀態。所有 Partner Central 操作 - 機會查詢、資金應用程式、文件分析 - 透過 以自然語言處理sendMessage。

工具概觀

工具	說明	Category
sendMessage	傳送訊息給 Partner Central AI 代理器。支援文字、檔案附件和human-in-the-loop核准回應。	讀取/寫入
getSession	擷取工作階段狀態，包括對話歷史記錄、事件和中繼資料。	唯讀

sendMessage

所有 Partner Central AI 代理器互動的主要工具。使用此工具來提出問題、請求動作、附加文件進行分析，以及回應寫入操作的核准請求。

客服人員會在工作階段中維護對話內容，因此您可以提出後續問題，而無需重複先前的內容。

Parameters

- content（必要）— 內容區塊陣列。每個區塊必須包含決定區塊結構type的欄位。您可以在單一訊息中包含多個區塊（例如文字 + 文件附件）。

  內容區塊類型：

Type	欄位	說明
text	type（必要）、 text（必要）	傳送給客服人員的使用者訊息文字
document	type（必要）、 filename（必要）、 s3Uri（必要）	代理程式要分析的檔案附件。s3Uri 必須包含 versionId 參數。
tool_approval_response	type（必要）、 toolUseId（必要）、 decision（必要）、 message（選用）	回應human-in-the-loop核准請求

- catalog（必要）— 操作的目標環境。

  有效值： "AWS"（生產）、 "Sandbox"（測試）

- sessionId（選用）— UUID v4 識別現有工作階段以繼續。省略 以建立新的工作階段。格式：session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

  預設：系統會自動建立新的工作階段。

- stream（選用）— 啟用伺服器傳送事件 (SSE) 串流以進行即時回應交付。

  有效值：true、false

  預設：false

回應

回應包括：

欄位	說明
sessionId	後續訊息的工作階段識別符
status	回應狀態："requires_approval" 、"complete" 或 "error"
content	來自代理程式的回應內容區塊陣列

範例

基本文字訊息（新工作階段）

要求:

```
{
    "jsonrpc": "2.0",
    "id": 1,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "content": [
                {
                    "type": "text",
                    "text": "List my open opportunities with expected close date in Q1
 2026"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

回應：

```
{
```

```
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "content": [
            {
                "type": "text",
                "text": "I found 12 open opportunities with expected close dates in Q1
 2026. Here's a summary:\n\n1. **O1234567890** - Acme Corp Cloud Migration - $250,000 -
 Qualified stage\n2. **O1234567891** - GlobalTech Data Analytics - $180,000 - Prospect
 stage\n..."
            }
        ],
        "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
        "status": "complete"
    }
}
```

後續訊息（現有工作階段）

要求:

```
{
    "jsonrpc": "2.0",
    "id": 2,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "text",
                    "text": "Tell me more about O1234567890. Is it ready for
 submission?"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

檔案連接

先將文件上傳至 S3，然後在訊息中參考文件：

```
{
    "jsonrpc": "2.0",
    "id": 3,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "text",
                    "text": "Review this customer proposal and suggest which
 opportunity it aligns with"
                },
                {
                    "type": "document",
                    "filename": "acme-proposal.pdf",
                    "s3Uri": "s3://aws-partner-central-marketplace-ephemeral-writeonly-
files/123456789012/acme-proposal.pdf?versionId=abc123def456"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

檔案上傳限制：

- 每則訊息最多 3 個檔案
- 影像大小限制：3.75 MB
- 文件大小限制：4.5 MB
- 允許擴充功能：doc、docx、pdf、png、jpeg、xlsx、csv、 txt
- 檔案必須上傳至 s3://{bucket}/{your-aws-account-id}/
- S3 URI 必須包含versionId查詢參數

Human-in-the-loop工作流程

當客服人員需要執行寫入操作（例如，更新機會、提交資金應用程式）時，它會傳回包含建議動作詳細資訊"requires_approval"的狀態。您必須使用tool_approval_response內容區塊來回應。

步驟 1 — 客服人員請求核准：

```
{
    "jsonrpc": "2.0",
    "id": 4,
    "result": {
        "content": [
            {
                "type": "text",
                "text": "I'd like to update opportunity O1234567890 with the following
 changes:\n- Target close date: 2026-03-31\n- Expected revenue: $300,000\n- Stage:
 Qualified\n\nPlease approve, reject, or override this action."
            },
            {
                "type": "tool_approval_request",
                "toolUseId": "tool-use-98765",
                "toolName": "update_opportunity_enhanced",
                "parameters": {
                    "opportunityId": "O1234567890",
                    "targetCloseDate": "2026-03-31",
                    "expectedRevenue": 300000,
                    "stage": "Qualified"
                }
            }
        ],
        "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
        "status": "requires_approval"
    }
}
```

步驟 2 — 核准動作：

```
{
    "jsonrpc": "2.0",
    "id": 5,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
```

```
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "tool_approval_response",
                    "toolUseId": "tool-use-98765",
                    "decision": "approve"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

步驟 2（替代）— 拒絕動作：

```
{
    "jsonrpc": "2.0",
    "id": 5,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "tool_approval_response",
                    "toolUseId": "tool-use-98765",
                    "decision": "reject",
                    "message": "The expected revenue should be $250,000, not $300,000"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

步驟 2（替代）— 以自訂回應覆寫：

```
{
    "jsonrpc": "2.0",
    "id": 5,
    "method": "tools/call",
```

```
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "tool_approval_response",
                    "toolUseId": "tool-use-98765",
                    "decision": "override",
                    "message": "Use expected revenue of $250,000 and keep the stage as
 Prospect instead"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

核准決策值：

決策	Behavior (行為)
"approve"	使用建議的參數執行工具
"reject"	請勿執行工具。選用message說明原因。
"override"	透過 提供自訂回應或修改的指示 message

使用 SSE 串流

在代理程式處理您的請求時，啟用串流以接收增量回應區塊：

要求:

```
{
    "jsonrpc": "2.0",
    "id": 6,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
```

```
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "text",
                    "text": "Analyze my pipeline and identify opportunities at risk"
                }
            ],
            "catalog": "AWS",
            "stream": true
        }
    }
}
```

伺服器會以 SSE 事件串流回應：

```
event: stream_start
data: {"sessionId": "session-550e8400-e29b-41d4-a716-446655440000"}

event: assistant-response.start
data: {}

event: server-tool-use
data: {"toolName": "analyze_pipeline", "parameters": {}}

event: server-tool-response
data: {"toolName": "analyze_pipeline", "result": {"opportunitiesAnalyzed": 47,
 "atRisk": 5}}

event: assistant-response.delta
data: {"text": "I analyzed your pipeline of 47 opportunities and identified "}

event: assistant-response.delta
data: {"text": "5 that are at risk of slipping:\n\n"}

event: assistant-response.delta
data: {"text": "1. **O2345678901** - Close date is past due by 15 days\n"}

event: assistant-response.completed
data: {"status": "complete"}

event: stream_end
data: {}
```

getSession

擷取對話工作階段的目前狀態，包括完整的對話歷史記錄、事件和中繼資料。使用此項目可檢查工作階段狀態、檢閱過去的互動，或繼續對話。

Parameters

- `sessionId`（必要）— 要擷取之工作階段的 UUID。格式：`session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`。
- `catalog`（必要）— 工作階段所屬的環境。

  有效值："AWS"、"Sandbox"

回應

欄位	Type	Description
`sessionId`	string	工作階段識別符
`createdAt`	string	工作階段建立的 ISO 8601 時間戳記
`lastActivity`	string	上次活動的 ISO 8601 時間戳記
`sequenceNumber`	integer	目前事件序號
`stateType`	string	目前工作階段狀態
`events`	陣列	完整的對話歷史記錄（使用者訊息、客服人員回應、工具使用）
`variables`	object	工作階段變數和中繼資料
`eventCount`	integer	工作階段中的事件總數

範例

要求:

```
{
    "jsonrpc": "2.0",
    "id": 7,
    "method": "tools/call",
    "params": {
        "name": "getSession",
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "catalog": "AWS"
        }
    }
}
```

回應：

```
{
    "jsonrpc": "2.0",
    "id": 7,
    "result": {
        "content": [
            {
                "type": "text",
                "text": "{\"sessionId\":\"session-550e8400-e29b-41d4-
a716-446655440000\",\"createdAt\":\"2026-01-15T10:30:00Z\",\"lastActivity\":
\"2026-01-15T11:45:00Z\",\"sequenceNumber\":8,\"stateType\":\"END_TURN\",\"eventCount
\":8,\"events\":[...],\"variables\":{}}"
            }
        ]
    }
}
```

錯誤處理

所有錯誤都遵循 JSON-RPC 2.0 錯誤格式：

```
{
    "jsonrpc": "2.0",
    "id": 1,
```

```
    "error": {
        "code": -32001,
        "message": "Authentication failed. Verify your SigV4 credentials and ensure
 they have not expired."
    }
}
```

the section called “錯誤代碼” 如需錯誤代碼的完整清單及其意義，請參閱 。

建議的重試策略

- 對於 -32004(LIMIT_EXCEEDED)：從 1 秒開始以指數退避重試
- 對於 -32603(INTERNAL_ERROR)：以指數退避重試最多 3 次
- 對於 -32001(AUTHENTICATION_FAILURE)：重新整理憑證並重試
- 對於所有其他錯誤：請勿自動重試 — 檢查錯誤訊息並更正請求

關於AWS Partner Central APIs

下列各節提供 APIs的一般資訊。

主題

- 使用AWS Partner Central API
- 支援 AWS 的區域
- 許可
- AWS Partner Central API 的配額
- 登入AWS Partner Central API
- AWS Partner Central API 的通知
- 在沙盒中測試

使用AWS Partner Central API

下列各節提供使用AWS Partner Central APIs的相關資訊。

主題

- 使用AWS Partner Central Account API
- 使用AWS Partner Central Selling API
- 使用AWS Partner Central Benefits API
- 使用AWS Partner Central Channel API
- 使用營收測量 API

使用AWS Partner Central Account API

管理合作夥伴帳戶

合作夥伴帳戶已註冊並與現有AWS帳戶連結，以提供完整的 IAM 型體驗。這種方法消除了使用者層級登入資料的需求，確保所有註冊和操作都透過AWS帳戶中的 IAM 實體進行管理。AWS Partner Central可實現與AWS服務的無縫整合，此模型提供合作夥伴實體管理、支援多個目錄，並建立AWS帳戶層級權利系統。

合作夥伴可以使用可用的合作夥伴帳戶 APIs 來註冊、管理和更新其帳戶：

合作夥伴註冊 API

合作夥伴可以使用其帳戶註冊其AWS帳戶，以進行合作夥伴參與。使用建立 API，合作夥伴將提供必要的聯盟潛在客戶資訊、接受 APN 條款，並提供唯一的法定名稱。

合作夥伴設定檔 API

合作夥伴會管理包含公司詳細資訊（名稱、描述、網站、標誌）的設定檔，其中包含公有/私有可見性控制、非同步驗證，以及允許一次更新一次的狀態追蹤。

網域管理 API

合作夥伴可以透過電子郵件驗證來註冊和驗證業務網域，以建立組織身分並關聯員工的培訓和認證。

合作夥伴連線 API

合作夥伴會將其AWS帳戶與其他AWS帳戶連線，以用於各種目的，例如與其他合作夥伴共同銷售、共用AWS Marketplace 在帳戶之間提供資料、授權經銷商/通路合作夥伴分發/重新銷售其產品等。

合作夥伴驗證 API

合作夥伴驗證是合作夥伴帳戶註冊的必要先決條件。合作夥伴必須先完成業務驗證和身分驗證程序，才能建立合作夥伴帳戶。這些驗證會驗證企業是否已合法註冊，並確認註冊AWS帳戶的人員身分。

主題

- 使用合作夥伴註冊
- 使用合作夥伴設定檔
- 使用網域管理
- 使用合作夥伴帳戶連線
- 使用合作夥伴驗證

使用合作夥伴註冊

合作夥伴註冊

合作夥伴可以使用其帳戶註冊其AWS帳戶，以進行合作夥伴參與。合作夥伴將使用建立 API 提供必要的聯盟潛在客戶資訊、接受 APN 條款，並提供唯一的法定名稱。

註冊期間

1. 同步合作夥伴實體建立 – 客戶透過呼叫建立 API 來啟動合作夥伴註冊程序，該 API 會執行輸入驗證，並在相同的請求中建立合作夥伴實體。
2. 聯盟潛在客戶資訊要求 – 在合作夥伴註冊期間，客戶必須提供AWS合作夥伴網路 (APN) 相關通訊的聯盟潛在客戶聯絡資訊。
3. 強制執行條款與條件 – 客戶必須在註冊期間接受 APN 條款與條件。需要接受才能完成註冊並存取任何功能。這些條款適用於所有 APN 計畫優點和功能。
4. Tag-On-Create支援 – 作為AWS一致性授權體驗 (CAE) 和標籤型授權的一部分，客戶可以在建立合作夥伴實體時新增標籤。
5. 以法定名稱為基礎的驗證 – 註冊將根據合作夥伴法定名稱強制執行唯一性，確保相同實體下沒有重複的註冊。

註冊後

1. 標籤管理支援 – 註冊後，合作夥伴可以標記、取消標記和列出現有合作夥伴實體的標籤。
2. 擷取合作夥伴實體 - 建立合作夥伴實體之後，客戶可以使用清單 API 來擷取合作夥伴 ID，然後使用取得 API 來擷取特定合作夥伴實體的詳細資訊。

API 摘要

1. CreatePartner API：客戶透過呼叫 CreatePartner API 來啟動合作夥伴註冊程序，該 API 會執行輸入驗證，並在相同的請求中建立合作夥伴實體。
2. ListPartners API：註冊後，客戶可以使用清單 API 來擷取合作夥伴實體的清單。
3. GetPartner API：註冊後，客戶可以使用 Get API 擷取特定合作夥伴實體的詳細資訊。
4. PutAllianceLeadContact API：更新主要聯盟潛在客戶聯絡資訊。此操作允許合作夥伴轉移主要聯盟潛在客戶指定或修改聯絡詳細資訊，同時保持組織連續性。
5. GetAllianceLeadContact API：更新主要聯盟潛在客戶聯絡資訊。此操作允許合作夥伴轉移主要聯盟潛在客戶指定或修改聯絡詳細資訊，同時保持組織連續性。

使用合作夥伴設定檔

設定檔管理

合作夥伴透過公有/私有可見性控制、非同步驗證和狀態追蹤，管理包含公司詳細資訊（名稱、描述、網站、標誌）的設定檔，一次允許更新一次。

1. 設定檔資訊 – 合作夥伴設定檔必須包含基本的公司詳細資訊，例如顯示名稱、描述、網站 URL、標誌、IndustrySegments和 PrimarySolutionType。
2. 多語言支援 – 合作夥伴設定檔支援多種語言。設定檔的每個當地語系化版本都包含地區設定的屬性。
3. 設定檔可見性控制 – 合作夥伴可以設定其設定檔可見性（公有或私有）。
4. 設定檔儲存 – 設定檔資訊將存放在合作夥伴帳戶物件中。
5. 設定檔狀態管理 – 發佈之前，設定檔更新會以非同步方式驗證。合作夥伴可以檢查最新的設定檔更新狀態 (IN_PROGRESS、SUCCEED、FAILED 和 CANCELLED)。只有核准的設定檔才會公開顯示。
6. 請求並行控制 – 一次僅允許一個設定檔更新請求。如果設定檔更新已在進行中，客戶必須先透過 CancelPartnerProfileUpdateTask API 明確取消持續更新，才能啟動新的更新。
7. 取消更新任務 – 合作夥伴可以透過呼叫 CancelPartnerProfileUpdateTask API 來取消進行中的設定檔更新。只有在更新處於 IN_PROGRESS 狀態時，才允許取消，並且必須在開始新的更新之前完成。

API 摘要

1. StartPutProfileTask API：此 API 接受合作夥伴設定檔更新，並執行基本同步輸入驗證。
2. GetProfileUpdateTask API：此 API 允許合作夥伴擷取目前的設定檔更新狀態。它適用於客戶透過 StartPutPartnerProfile API 開始設定檔更新後，讓合作夥伴檢查其請求是 IN_PROGRESS、FAILED、SUCCEEDED 還是 CANCELLED。
3. CancelProfileTask API：允許合作夥伴明確取消目前處於 IN_PROGRESS 狀態的設定檔更新任務。這包括自動審核和手動審核階段。取消請求可讓合作夥伴啟動新的設定檔更新，而無需等待目前的審核程序完成。
4. PutProfileVisibility API：透過呼叫此 API，無論地區設定為何，合作夥伴都可以控制設定檔的可見性。這可讓合作夥伴將設定檔設為公有或私有，而無需修改內容。
5. GetProfileVisibility API：透過呼叫 UpdatePartnerProfileVisibility API，無論地區設定為何，合作夥伴都可以控制設定檔的可見性。這可讓合作夥伴將設定檔設為公有或私有，而無需修改內容。

使用網域管理

網域管理

合作夥伴可以透過電子郵件驗證來註冊和驗證業務網域，以建立組織身分並關聯員工的培訓和認證。

API 摘要

1. SendEmailVerificationCode API：將驗證碼傳送至指定的電子郵件地址，以啟動電子郵件驗證程序。用於新聯絡人建立和電子郵件地址更新。
2. AssociateAwsTrainingCertificationEmailDomain API：將電子郵件網域與合作夥伴帳戶的 AWS 訓練和認證建立關聯，以便自動驗證員工認證。
3. DisassociateAwsTrainingCertificationEmailDomain API：移除電子郵件網域與 AWS 訓練之間的關聯，以及合作夥伴帳戶的認證。

使用合作夥伴帳戶連線

合作夥伴可以使用 Partner Connection API AWS管理他們自己的帳戶之間的連線，或與其他合作夥伴AWS帳戶的連線。此程序包含兩個階段：

1. 連線邀請階段：啟動和回應連線請求
2. 作用中連線階段：管理已建立的連線及其相關聯的業務活動

建立和傳送連線邀請

步驟 1：建立連線邀請

建立合作夥伴連線的第一步是使用 `CreateConnectionInvitation`動作建立和傳送連線邀請。建立邀請時，您必須指定：

- 目錄：將管理連線的目錄AWS(AWS或沙盒）
- ConnectionType：您要建立的關係類型。選擇下列其中一項：
  - **`OPPORTUNITY_COLLABORATION`**：當您想要將連線請求傳送至其他合作夥伴的帳戶時，請使用此類型，以便您可以向他們傳送機會參與邀請以進行機會協作
  - **`SUBSIDIARY`**：當您想要將您擁有的多個AWS Marketplace 賣方帳戶連接到您已與 APN 連結或您註冊為合作夥伴的「主要」帳戶時，請使用此類型
- ReceiverIdentifier：接收者的公有合作夥伴設定檔 ID （適用於機會協作連線類型） 或AWS帳戶 ID （適用於次要連線類型）

- 電子郵件：您的通訊聯絡人電子郵件地址
- 名稱：您做為寄件者的名稱
- 訊息：說明連線請求用途的說明

建立時，邀請會進入待定狀態，等待來自接收者的動作。

> **Important**
>
> 透過傳送連線邀請，即表示您同意在接收者接受邀請時向他們公開AWS您的帳戶 ID。此揭露對於建立帳戶層級連線是必要的。

步驟 2：監控您傳送的邀請

您可以使用 ParticipantType 參數設為 ListConnectionInvitations的動作，追蹤您已傳送的邀請狀態SENDER。這可讓您：

- 檢視所有傳出邀請
- 依狀態 (PENDING、ACCEPTED、REJECTED、EXPIRED) 篩選
- 依連線類型篩選
- 檢查對其他合作夥伴的特定邀請

步驟 3：取消邀請 （選用）

如果您需要在邀請被接受之前撤銷待定的邀請，您可以使用 CancelConnectionInvitation動作。請注意：

- 您只能取消處於待定狀態的邀請
- 接受邀請並建立連線後，就無法取消該邀請
- 若要結束已建立的連線，您必須改用 CancelConnection動作

接收和回應連線邀請

步驟 1：列出傳入邀請

若要檢視您收到的連線邀請，請使用 ListConnectionInvitations動作，並將 ParticipantType 參數設為 RECEIVER。您可以依下列條件篩選：

- 連線類型
- 狀態 (PENDING、ACCEPTED、REJECTED)
- 寄件者的識別符

步驟 2：檢閱邀請詳細資訊

使用 GetConnectionInvitation動作來檢視特定邀請的完整詳細資訊，包括：

- 寄件者名稱和聯絡人電子郵件
- 解釋目的的邀請訊息
- 正在請求的連線類型
- 過期日期
- 寄件者的公有設定檔資訊

步驟 3：接受或拒絕邀請

檢閱邀請詳細資訊後，您有兩個選項：

選項 A：接受邀請

使用 AcceptConnectionInvitation動作來建立連線。當您接受時：

- 在作用中狀態下建立新的連線
- 邀請的狀態變更為已接受
- AWS您的帳戶 ID 會向寄件者公開
- 您可以開始執行連線類型啟用的業務活動

> **Important**
>
> 重要同意：接受連線邀請，即表示您同意向寄件者公開AWS您的帳戶 ID。此揭露對於建立帳戶層級連線是必要的。

選項 B：拒絕邀請

如果您不想建立連線，請使用 RejectConnectionInvitation動作。您可以選擇性地提供拒絕的原因。拒絕後：

- 邀請的狀態變更為已拒絕
- 未建立連線
- 邀請將不再出現在您的待定邀請清單中

自動過期

如果在過期期間內未採取任何動作，系統會自動將邀請轉換為過期狀態。無法接受或拒絕過期的邀請。

管理作用中連線

檢視您的連線

建立連線後，雙方可以使用下列動作來檢視和管理連線：

`ListConnections`：使用篩選選項擷取連線清單：

- 依連線類型和狀態篩選 （例如 `OPPORTUNITY_COLLABORATION:ACTIVE`)
- 依其他方的識別符篩選
- 檢視每個連線的摘要資訊

`GetConnection`：擷取特定連線的完整詳細資訊，包括：

- 與此合作夥伴關係相關聯的所有連線類型
- 每個連線類型的狀態 （作用中或已終止）
- 來自原始邀請的聯絡資訊
- AWS雙方的帳戶 IDs 和公開設定檔 IDs
- 每個連線類型的建立和終止時間戳記

將連線類型新增至現有連線

如果您已與合作夥伴建立作用中的連線，並且想要建立新的關係類型，您可以傳送具有不同連線類型的新連線邀請。接受時：

- 新的連線類型會新增至現有的連線
- 這兩種連線類型都可以獨立管理
- 關係會維持相同的連線 ID

> **Note**
>
> 在任何時間，指定目錄中的兩個帳戶之間只能有一個指定類型的作用中連線。

終止連線類型

任一方都可以使用 CancelConnection動作終止特定連線類型。終止時：

- 指定要終止的連線 ID 和連線類型
- 提供終止的原因
- 連線類型的狀態變更為 CANCELED
- 相同連線中的其他連線類型不會受到影響

完成連線終止

當連線中的所有連線類型終止時，連線本身會轉換為已終止狀態。完全終止的連線：

- 無法重新啟用，但您可以傳送新的連線請求，以繼續與該帳戶進行業務
- 保留在系統中以保留歷史記錄
- 可以使用 GetConnection API 檢視
- 需要新的邀請才能重新建立關係

監控連線事件

合作夥伴應使用 Amazon EventBridge 監控連線相關事件，以隨時掌握下列事項：

- 新的傳入連線邀請
- 已傳送邀請的狀態變更 (ACCEPTED、REJECTED、EXPIRED)
- 由另一方啟動的連線終止

收到事件時，請使用適當的取得 API 動作來擷取最新詳細資訊：

- GetConnectionInvitation 邀請更新
- GetConnection 用於連線更新

可能的事件

- 已建立連線邀請：通知收件人帳戶傳入連線邀請。
- 已取消連線邀請：當寄件者取消傳入連線邀請時，通知收件人帳戶。
- 連線邀請已過期：當連線邀請過期而不採取動作時，通知寄件者和收件人帳戶。
- 連線邀請遭拒：當收件人拒絕其連線邀請時，通知寄件者帳戶。
- 已接受連線邀請：當寄件者帳戶接受連線邀請並建立連線時， 會通知寄件者帳戶。
- 連線已取消：當所有連線完全終止時，通知兩個連線方。

了解連線狀態

連線邀請狀態

State	說明	可以轉換為
PENDING	建立時的初始狀態；等待接收者動作	ACCEPTED, REJECTED, EXPIRED, CANCELED
ACCEPTED	接受收件人；已建立連線	無 （終端狀態）
REJECTED	收件人拒絕；包含選用拒絕原因	無 （終端狀態）
EXPIRED	由於過期期間內沒有動作而導致系統過期	無 （終端狀態）
CANCELED	寄件者在接受之前撤回邀請	無 （終端狀態）

連線類型狀態

State	說明
ACCEPTED	連線類型可運作；相關的商業活動已啟用
CANCELED	任一方結束的連線類型

最佳實務

1. 明確溝通：在連線邀請中提供詳細且明確的訊息，以說明目的和預期的協同合作。
2. 及時回應：立即監控和回應連線邀請，以避免自動過期。
3. 定期審查：定期審查您的作用中連線，以確保其與您目前的業務需求保持相關。
4. 適當終止：結束商業關係時，請使用 CancelConnection 動作，並清楚說明維持專業溝通的原因。
5. 事件監控：設定 EventBridge 規則以自動接收連線狀態變更的通知，以協助隨時掌握重要的更新。
6. 連線類型管理：在建立連線之前，請考慮需要哪些連線類型，因為每種類型都啟用不同的業務功能。
7. 安全意識：請記住，建立連線會在各方之間顯示AWS帳戶 IDs。僅與信任的合作夥伴連線。

連線類型及其目的

連線類型	用途	使用案例
OPPORTUNITY_COLLABORATION	啟用與其他合作夥伴共同銷售機會的協作	如果您希望其他合作夥伴能夠存取您的機會，反之亦然，請使用此連線類型。此連線可讓您傳送機會參與邀請給連線的合作夥伴。
SUBSIDIARY	在您的多個AWS Marketplace賣方帳戶與主要合作夥伴帳戶之間建立連線	當您有多個AWS Marketplace Seller 帳戶想要連線到您註冊為合作夥伴和賣方的主要帳戶時，請使用 。

使用合作夥伴驗證

管理合作夥伴驗證

合作夥伴驗證是合作夥伴帳戶註冊的必要先決條件。合作夥伴必須先完成業務驗證和身分驗證程序，才能建立合作夥伴帳戶。這些驗證會驗證企業是否已合法註冊，並確認註冊AWS帳戶的人員身分。

合作夥伴可以使用可用的驗證 APIs來啟動和監控驗證程序：

驗證期間

1. 商業驗證啟動 – 合作夥伴透過呼叫具有商業註冊詳細資訊的 `StartVerification` API 來啟動商業驗證，包括法定名稱、註冊 ID、國家/地區代碼和註冊的司法管轄區。
2. 身分驗證啟動 – 合作夥伴透過呼叫具有註冊者資訊的 `StartVerification` API 來啟動身分驗證。API 會傳回安全完成 URL，其中註冊者會使用政府核發的身分驗證來完成身分驗證工作流程。
3. 驗證狀態監控 – 合作夥伴使用 `GetVerification` API 擷取驗證程序的目前狀態和詳細資訊。API 會傳回驗證類型、狀態、時間戳記和驗證特定的詳細資訊。
4. 驗證完成 – 業務驗證和身分驗證都必須達到 SUCCEEDED 狀態，合作夥伴才能繼續建立合作夥伴帳戶。失敗或過期的驗證必須在帳戶註冊之前解決。

後置驗證

1. 合作夥伴帳戶建立 – 成功驗證後，合作夥伴會繼續使用合作夥伴帳戶文件中所述的 `CreatePartner` API 建立其合作夥伴帳戶。
2. 驗證記錄擷取 – 合作夥伴可以隨時使用 `GetVerification` API 和啟動期間傳回的驗證 ID 擷取驗證詳細資訊。

API 摘要

1. **`StartVerification`** API：合作夥伴透過呼叫 `StartVerification` API 來啟動驗證程序。對於商業驗證，API 會驗證商業註冊詳細資訊。針對身分驗證，API 會產生限時完成 URL，讓註冊者完成身分驗證。
2. **`GetVerification`** API：啟動驗證後，合作夥伴會使用 `GetVerification` API 來擷取驗證狀態和詳細資訊。API 會傳回目前狀態、時間戳記和驗證特定資訊。

使用AWS Partner Central Selling API

此AWS Partner Central API 參考旨在協助AWS合作夥伴將客戶關係管理 (CRM) 系統與 Partner Central 整合。API 會自動與 Partner Central 互動，這有助於確保在聯合業務活動中進行有效的互動。

API 提供標準AWS API 功能。使用 API 動作或使用專為您的程式設計語言或平台量身打造的AWS SDK 來存取它。如需詳細資訊，請參閱 入門AWS和要建置的工具AWS。

AWS Partner Central API 提供的功能

1. 機會管理：AWS使用 `CreateOpportunity`、`UpdateOpportunity`、、`ListOpportunitiesGetOpportunity`和 等 API 動作，協助管理銷售機會`AssignOpportunity`。
2. AWS推薦管理：AWS使用 `ListEngagementInvitations`、`StartEngagementByAcceptingInvitation`、`GetEngagementInvitation`和 等動作，協助接收共用的推薦`RejectEngagementInvitation`。
3. 實體關聯：使用 動作和 將AWS產品、合作夥伴解決方案、AWS Marketplace解決方案、AWS Marketplace產品和 AWS Marketplace Private Offer `AssociateOpportunity` 等相關實體與機會建立關聯`DisassociateOpportunity`。
4. 檢視AWS機會詳細資訊：使用 `GetAWSOpportunitySummary`動作來擷取連結至您機會的AWS即時機會摘要。
5. 清單解決方案：提供清單 APIs以列出合作夥伴使用 提供的解決方案`ListSolutions`。
6. 事件訂閱：合作夥伴可以透過聆聽機會建立、機會更新、參與邀請接受、參與邀請遭拒和使用 Amazon EventBridge 建立的參與邀請等事件，訂閱機會的即時更新。

支援的區域和端點

AWS Partner Central API 在美國東部（維吉尼亞北部）區域提供。

區域	Endpoint
us-east-1	partnercentral-selling.us-east-1.api.aws

合作夥伴可以在安全的沙盒環境中測試和驗證 API 整合。這可讓您測試 API 動作，而不會影響即時資料。如需詳細資訊，請參閱在沙盒中測試AWS Partner Central Selling API。

銷售 API 實體

AWS Partner Central 實體代表在合作夥伴與 之間的銷售互動中使用的關鍵業務元件AWS。這些實體會封裝與機會、解決方案、產品和優惠相關的資訊，以便順利協同合作和管理聯合銷售活動。下列各節提供 Partner Central Selling API 中核心實體的說明。

機會

機會是企業識別並積極追求的潛在銷售或交易。這是合格的潛在客戶，或具有公司產品或服務可以解決的特定需求。機會通常會在銷售管道或 CRM 系統中追蹤，並構成未來收入的基礎。有效的機會管理涉及透過銷售程序培養潛在客戶，從初始資格到結束。如需詳細資訊，請參閱使用機會和資料類型

合作夥伴解決方案

代表合作夥伴解決方案（在AWS Partner Central 上稱為產品），這是由AWS Partner 建立和交付的軟體產品或諮詢實務。合作夥伴解決方案可協助客戶解決特定業務挑戰，或使用AWS服務實現特定目標。如需詳細資訊，請參閱什麼是解決方案？

AWS產品

代表特定AWS服務或產品。AWS提供各種產品和服務，旨在提供可擴展、可靠且符合成本效益的基礎設施解決方案。合作夥伴可以從 Partner Central AWS（開始匯入 >產品和解決方案）上的大量匯入頁面取得最新的AWS產品清單。 https://partnercentral.awspartner.com/OpportunityBulkImportPage如需詳細資訊，您可以了解AWS產品或檢視所有AWS產品的清單。

AWS Marketplace私有優惠

AWS Marketplace私有優惠是一項功能，可讓AWS Marketplace賣方向個別AWS客戶提供特定定價和條款。透過私有優惠，賣方可以交涉自訂價格、付款排程和最終使用者授權條款。AWS客戶可以取得符合其特定需求的軟體解決方案，同時還可能受益於比標準優惠更有利的條款或定價。私有優惠程序涉及賣方建立具有量身訂做條款的獨特優惠，然後私下與指定的AWS客戶共用，以供其檢閱和接受。如需詳細資訊，請參閱 中的私有優惠AWS Marketplace。

參與邀請

參與邀請是指來自 的正式請求，AWS供合作夥伴在特定推薦時協作。這可讓AWS和 合作夥伴共同推動機會向前。合作夥伴可以接受或拒絕邀請。

主題

- 處理您的機會
- 使用來自 的機會AWS
- 使用機會更新
- 使用多合作夥伴機會
- 關聯、取消關聯和指派機會

- 使用潛在客戶
- 使用交易規模洞察
- 最佳實務

處理您的機會

什麼是機會？

在銷售程序的初始階段，銷售代表會評估感興趣的個人（稱為潛在客戶) 是否可能成為客戶。此評估和驗證階段稱為資格。一旦潛在客戶被視為符合資格，且被視為轉換為客戶的機率較高，就會將其分類為機會。

處理您的機會

合作夥伴可以管理在其 CRM 系統中建立的機會，並使用AWS Partner Central Selling API 將其與AWS Partner Central 同步。這可讓合作夥伴追蹤和管理從啟動到關閉的機會。

建立機會

管理機會的第一步是使用 `CreateOpportunity`動作建立機會。這會建立將 `Lifecycle.ReviewStatus`設為 的機會`Pending Submission`。不過，機會尚未提交至AWS進行驗證。

建立機會之後，您必須使用 `AssociateOpportunity`動作，將至少一個合作夥伴解決方案、AWS Marketplace解決方案或AWS Marketplace產品與機會建立關聯。此動作明確定義了嘗試銷售的機會。您可以使用 `ListSolutions` API 檢視帳戶中可用解決方案的完整清單，並且可以在 1 到 10 個解決方案之間建立關聯。

或者，您也可以使用 `AssociateOpportunity`動作，將相關AWS產品與機會建立關聯。此步驟可協助AWS銷售團隊了解預期將與機會一起銷售AWS的產品。

在所需的解決方案或產品建立關聯，並選擇性地AWS連結產品後，您可以使用 `StartEngagementFromOpportunityTask`動作開始參與機會。這是當您提交開始參與的機會時。

檢閱程序

使用 `StartEngagementFromOpportunityTask`動作開始機會的參與後，機會會進入AWS驗證階段，並將其`Lifecycle.ReviewStatus`設定為 `Submitted`。在檢閱程序完成之前，無法對機會進行任何變更。

在此驗證階段，AWS會確保機會詳細資訊準確且完整。驗證進行中時，Lifecycle.ReviewStatus會設定為 In-Review。

如果合作夥伴需要變更或其他詳細資訊，AWS請將 設定為 Lifecycle.ReviewStatus Action Required，並透過 Lifecycle.ReviewComments 欄位傳達任何必要的更新。

一旦機會通過驗證， 會Lifecycle.ReviewStatus變更 Approved，讓機會準備好共同銷售活動。

合作夥伴應使用 Amazon EventBridge 監控Opportunity Updated事件。這將通知他們任何狀態變更或意見回饋AWS。收到事件時，合作夥伴可以使用 GetOpportunity API 動作來擷取最新的機會詳細資訊並驗證 Lifecycle.ReviewStatus 欄位。

解決驗證問題

如果 Lifecycle.ReviewStatus 設定為 Action Required，合作夥伴需要解決 反白顯示的問題 AWS。若要解決這些問題，合作夥伴可以使用 UpdateOpportunity API 動作來更新機會。

在 Action Required 狀態期間，只能編輯特定欄位。這些欄位包括：

1. Customer.Account.Address.City
2. Customer.Account.Address.Country
3. Customer.Account.Address.PostalCode
4. Customer.Account.Address.StateOrRegion
5. Customer.Account.Address.StreetAddress
6. Customer.Account.WebsiteUrl
7. LifeCycle.TargetCloseDate
8. Project.ExpectedCustomerSpend.Amount
9. Project.ExpectedCustomerSpend.Currency
10. Project.CustomerBusinessProblem
11. PartnerOpportunityIdentifier

進行必要的變更後，機會會重新進入驗證階段，而程序會重複，直到機會的Lifecycle.ReviewStatus 設定為 Approved或 為止Disqualified。

核准後更新

一旦機會成為 Approved，合作夥伴就可以視需要使用 UpdateOpportunity動作繼續更新機會，促進無縫的共同銷售活動。

合作夥伴應繼續透過 Amazon EventBridge 監控Opportunity Updated事件，以隨時更新任何變更。如需追蹤AWS更新的詳細資訊，請參閱「使用機會更新」一節。合作夥伴也可以根據業務驗證規則更新選取欄位。

使用來自 的機會AWS

1. 接收AWS機會

當AWS銷售主管將合作夥伴連接到 CRM AWS系統中的機會時，會與合作夥伴共用機會。這些稱為AWS機會，與合作夥伴自己的帳戶中建立的機會不同。

當AWS機會已連接合作夥伴時， 會AWS建立包含AWS機會資料子集的參與邀請。合作夥伴將收到建立的參與邀請事件。

2. 檢閱業務開發邀請

業務開發邀請包含重要資訊，例如 Project.Title、Project.CustomerBusinessProblem、、Project.CustomerUseCase Lifecycle.Stage和一些額外的欄位。合作夥伴可以使用此資料來決定是否追求機會。

不過，來自AWS機會的下列欄位不包含在業務開發邀請中：

```
Customer.Account.Address.StreetAddress
Customer.Account.Address.City
Customer.Account.Address.PostalCode
Customer.Contact
Customer.Account.AWSAccountId
Project.OtherSolutionDescription
RelatedEntityIdentifiers
```

3. 處理業務開發邀請

收到建立的參與邀請事件時，合作夥伴可以使用 GetEngagementInvitation動作來擷取機會的詳細資訊AWS。

如果合作夥伴決定不追求機會，他們可以使用 RejectEngagementInvitation動作以及必要的 來拒絕邀請RejectionReason。一旦拒絕，就會失去對機會的存取。

若要接受邀請並繼續，合作夥伴應使用 StartEngagementByAcceptingInvitationTask動作。這是循序執行下列任務的非同步動作：

1. 接受業務開發邀請。
2. 使用機會中的AWS資料，在合作夥伴的帳戶中建立新的機會。
3. 包括識別合作夥伴帳戶中客戶所需的其他詳細資訊。

完成後，會使用對應的機會 ID 觸發機會建立事件。然後，合作夥伴可以在 GetOpportunity動作中使用此 ID 來擷取機會的完整詳細資訊。

如果合作夥伴未使用事件，他們可以使用相同的承載StartEngagementByAcceptingInvitationTask再次呼叫 ，以檢查最新狀態。

4. 管理AWS機會接受後

一旦機會進入合作夥伴的帳戶中，就可以像系統中的任何其他機會一樣進行管理和更新。合作夥伴可以使用 等動作UpdateOpportunity進行任何必要的變更。

如需如何追蹤AWS更新和管理機會更新的詳細資訊，請參閱 使用機會更新一節。

使用機會更新

更新機會

合作夥伴可以使用 UpdateOpportunity動作來更新機會，其中包含管理哪些欄位已更新以及更新時間的特定規則：

1. 如果 Lifecycle.ReviewStatus是 Submitted或 ，則無法進行更新In-Review。
2. 在提交之前，合作夥伴可以進行更新，但除非叫用 StartEngagementFromOpportunityTask動作，否則AWS不會處理它們。
3. 當機會處於 Submitted或 In-review 狀態時，會封鎖所有更新。
4. 如果機會處於 Action Required 狀態，請AWS開啟選取欄位以進行更新。這些欄位包括：
   - Customer.Account.Address.City
   - Customer.Account.Address.Country
   - Customer.Account.Address.PostalCode
   - Customer.Account.Address.StateOrRegion
   - Customer.Account.Address.StreetAddress
   - Customer.Account.WebsiteUrl
   - LifeCycle.TargetCloseDate

- Project.ExpectedCustomerSpend.Amount
- Project.ExpectedCustomerSpend.CurrencyCode
- Project.ExpectedCustomerSpend.EstimationURL
- Project.ExpectedCustomerSpend.Frequency
- Project.ExpectedCustomerSpend.TargetCompany
- Project.CustomerBusinessProblem
- PartnerOpportunityIdentifier

5. 在檢閱程序後 (即當 Lifecycle.ReviewStatus 設定為 時Approved)，無法更新下列欄位：
   - Customer.Account.Address.Country
   - Customer.Account.Address.PostalCode
   - Customer.Account.Industry
   - Customer.Account.WebsiteUrl
   - Project.CustomerBusinessProblem
   - PartnerOpportunityIdentifier
   - Project.Title
6. 對於所有其他欄位，可以使用 UpdateOpportunity動作進行更新。不過，根據特定計劃或機會類型的業務規則，可能會有其他限制。如需詳細資訊，請參閱欄位層級驗證。
7. 對於透過 UI 和 API 進行的所有更新，會產生機會更新事件。

從 接收有關機會AWS的更新

AWS通常會更新AWS機會，而且每次進行更新時都會產生機會更新事件。

若要從 擷取最新更新AWS，合作夥伴需要叫用兩個不同的動作

1. GetOpportunity 擷取合作夥伴機會的詳細資訊。
2. GetAWSOpportunitySummary 擷取AWS機會資料的即時摘要。

的大多數定期更新AWS將透過 GetAWSOpportunitySummary回應提供。不過，AWS有時可能會直接更新合作夥伴機會中的屬性。

使用來自 的更新AWS

1. 叫用 GetAWSOpportunitySummary動作以擷取AWS機會的最新詳細資訊。

2. 如果需要在合作夥伴的機會中反映變更，請使用 UpdateOpportunity動作將相關資料複製到合作夥伴的機會。

合作夥伴可以選擇將此程序自動化為直接更新機制，或實作手動檢閱程序來驗證和更新資料。

使用多合作夥伴機會

AWS Partner可以與AWS作用中參與者一起處理機會。

主題

- 業務開發和快照
- 建立 的自訂政策 ResourceSnapshotJobRole
- 邀請合作夥伴參與機會
- 擷取參與邀請詳細資訊
- 回應參與邀請
- 檢閱和更新多合作夥伴機會
- 使用快照來接收合作夥伴更新
- 檢視參與成員
- 監控資源快照任務狀態

業務開發和快照

參與是 擁有的資源，AWS用於多個AWS Partner帳戶與客戶AWS機會之間的協同合作。業務開發有助於合作夥伴和協同合作之間共用安全資訊，同時維持個別的機會擁有權和控制權。業務開發包含來自AWS和 合作夥伴的機會，並以作用中參與者AWS的身分加入 。合作夥伴可以使用 邀請AWS或其他合作夥伴加入參與EngagementInvitations。當受邀的合作夥伴接受時，他們會在相同的互動中收到自己的機會。

參與成員透過快照共用進度 - 時間點、來自 機會等基礎資源的特定欄位的不可變副本。快照是在參與中建立並與成員共用。當基礎資源變更時，擁有者可以建立新的快照修訂以反映更新。AWS提供快照任務 - 客戶擁有的任務，可在資源變更時自動建立新的修訂。共用後，參與成員仍可永久存取快照。

建立 的自訂政策 ResourceSnapshotJobRole

在多合作夥伴機會上進行協作需要與參與中的其他合作夥伴共用您機會的快照。若要維持最新機會詳細資訊的存取權，您需要建立名為 的自訂角色ResourceSnapshotJobRole。此角色可讓系統代表您

建立機會的快照，並從相同互動中的其他合作夥伴擷取快照。如果沒有此角色，您對機會所做的任何更新都不會讓參與中的其他合作夥伴看到，而且他們將繼續看到機會資料的過時快照。

> **Note**
>
> 您可以使用 以外的名稱`ResourceSnapshotJobRole`。如果您使用不同的名稱，請將下列政策`ResourceSnapshotJobRole`中的所有 執行個體取代為您的政策名稱。

若要建立此角色，請前往您的 IAM 主控台，並使用下列自訂信任政策建立`ResourceSnapshotJobRole`角色：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "resource-snapshot-job.partnercentral-selling.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

建立此角色之後，您需要連接 AWSPartnerCentralSellingResourceSnapshotJobExecutionRolePolicyAWS受管政策，或連接下列許可政策：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "partnercentral:CreateResourceSnapshot"
            ],
            "Resource": [
                "arn:aws:partnercentral:*::catalog/AWS/engagement/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetOpportunity"
            ],
            "Resource": [
                "arn:aws:partnercentral:*:111122223333:catalog/AWS/opportunity/*"
            ]
        }
    ]
}
```

使用您的 AWS 帳戶 ID 取代 {account}。

建立ResourceSnapshotJobRole角色並連接許可後，您需要ResourceSnapshotJobRole為組織設定 。使用 PutSellingSystemSettings動作來設定您ResourceSnapshotJobRole為公司建立的 角色 （ 由AWS您的帳戶識別 ）：

```
aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
Host: partner-central.aws.amazon.com
Content-Type: application/json

{
  "ResourceSnapshotJobRoleIdentifier": "arn:aws:iam::{account}:role/
ResourceSnapshotJobRole"
}
```

{account} 將 取代為AWS您的帳戶 ID，並將 ResourceSnapshotJobRole取代為您建立的角色名稱。資源快照任務會隱含擔任此角色，以存取您的機會並建立快照，以便與參與中的其他合作夥伴共用。

邀請合作夥伴參與機會

合作夥伴可以合作處理來自合作夥伴和AWS發起參與邀請的機會。您最多可以邀請 9 個合作夥伴加入機會。

邀請合作夥伴加入機會

1. 遵循 AWS合作夥伴中央銷售指南中的尋找和與合作夥伴連線中的步驟。您必須先與合作夥伴連線，才能邀請他們參與機會。此連線授予彼此帳戶 IDs的交互存取權，以確保邀請到達預期的合作夥伴帳戶。
2. 使用 StartEngagementFromOpportunityTask動作來建立參與，並將機會與其建立關聯。此非同步動作會依序執行下列任務：
    1. 建立參與。
    2. 將機會與參與建立關聯。
    3. 叫用 CreateEngagementInvitation動作至AWS。
    4. 提交機會給AWS。
    5. 啟動 ResourceSnapshotJob以建立快照。
3. 邀請其他合作夥伴加入。
    a. 若要取得與您的機會相關聯的參與 ID，請使用 ListEngagementFromOpportunityTasks動作，由上一個步驟中TaskIdentifier傳回的 篩選。
    b. 使用接收合作夥伴的參與 ID 和帳戶 ID 向 傳送邀請CreateEngagementInvitation。

成功的邀請會將建立參與邀請的事件傳送給接收合作夥伴。

擷取參與邀請詳細資訊

當您收到建立的參與邀請事件時，請使用 GetEngagementInvitation動作來擷取邀請詳細資訊。回應包含與業務開發相關聯之客戶機會的基本資訊。

檢閱邀請詳細資訊，特別是 InvitationMessage、Project.CustomerUseCase、Project.Title和 Project.CustomerBusinessProblem 欄位。此資訊提供有關客戶機會的內容，以及邀請合作夥伴對協作的期望。

使用這些詳細資訊來評估您是否想要透過接受或拒絕參與邀請來追求機會。

回應參與邀請

當合作夥伴傳送參與邀請給您時，它會建立參與邀請建立的事件，以通知您該邀請。您可以選擇接受或拒絕邀請。此決策決定您對多合作夥伴機會的參與。

若要拒絕參與邀請：

使用 RejectEngagementInvitation動作來拒絕邀請。您必須提供RejectionReason參數來解釋您的決策。一旦拒絕，您就無法存取邀請詳細資訊，而參與邀請拒絕事件會通知傳送合作夥伴您已拒絕他們的邀請。

若要接受參與邀請：

若要接受邀請並繼續多合作夥伴機會，請使用 StartEngagementByAcceptingInvitationTask動作。此非同步動作會依序執行下列任務：

1. 接受參與邀請。
2. 使用傳送合作夥伴的機會中的資料，在您的合作夥伴帳戶中建立新的機會。您將會收到機會建立的事件。
3. 包含識別您帳戶中客戶所需的其他詳細資訊。
4. 發佈參與邀請接受的事件，通知合作夥伴您已接受邀請並已新增至參與。

已過期的參與邀請

如果您未在十五天內回應參與邀請，則會過期，而且您無法參與多合作夥伴機會。參與邀請過期事件會通知傳送合作夥伴邀請已過期。

> **Note**
>
> 如果您仍然想要協作，傳送合作夥伴必須啟動新的參與邀請。

檢閱和更新多合作夥伴機會

當合作夥伴對機會啟動互動時，接收合作夥伴帳戶中新建立機會的檢閱狀態取決於傳送合作夥伴的機會狀態。

已提交檢閱狀態的機會：

如果傳送合作夥伴的機會Lifecycle.ReviewStatus已設為 Submitted，則會發生下列程序：

1. 在接收合作夥伴的帳戶中建立的新機會將Lifecycle.ReviewStatus設定為 Submitted。
2. Submitted 狀態會持續到傳送合作夥伴的機會為 為止Approved。
3. 驗證傳送合作夥伴的機會且 Lifecycle.ReviewStatus 設定為 後Approved，其他合作夥伴的機會會自動繼承Approved狀態。

此程序可確保跨多合作夥伴機會的一致機會詳細資訊，無需進行多次檢閱。

完成後，會使用對應的機會 ID 觸發機會建立的事件。合作夥伴可以將此 ID 與 GetOpportunity動作搭配使用，以擷取完整的機會詳細資訊。

具有核准檢閱狀態的機會：

如果傳送合作夥伴在將 Lifecycle.ReviewStatus設定為 的機會上啟動參與Approved，則會發生下列程序：

1. 在接收合作夥伴的帳戶中建立的新機會將Lifecycle.ReviewStatus設定為 Approved。
2. 使用對應的機會 ID 觸發機會建立的事件。
3. 合作夥伴可以使用 GetOpportunity動作搭配機會 ID 來擷取完整的機會詳細資訊。

不過，已核准的機會可能沒有從傳送合作夥伴的機會複製的一些合作夥伴特定詳細資訊。當機會階段更新為 Qualified或更新版本階段時，接收合作夥伴應使用 UpdateOpportunity動作更新這些詳細資訊：

- Project.CustomerUseCase
- Project.DeliveryModels
- Solution

- PrimaryNeedsFromAws
- LifeCycle.TargetCloseDate
- Project.ExpectedCustomerSpend.Amount
- Project.ExpectedCustomerSpend.Currency
- Marketing.source
- Marketing.AwsFundingUsed

在接收合作夥伴的帳戶中建立機會後，即可像合作夥伴系統中的任何其他機會一樣進行管理和更新。合作夥伴可以使用 UpdateOpportunity動作進行變更，或提供有關他們參與機會的詳細資訊。

使用快照來接收合作夥伴更新

在互動中，合作夥伴會獨立維護和更新其機會。當有人修改業務開發的資源時，例如新增機會，則會發佈業務開發資源快照建立的事件。

若要隨時掌握變更，並從其他合作夥伴的機會存取最新資訊，您可以使用下列動作：

- ListEngagementResourceAssociations – 使用此動作來擷取與機會相關聯的參與 ID。
- ListResourceSnapshots – 使用此動作可擷取與參與相關聯的所有機會快照的完整清單，提供所有合作夥伴可用快照的概觀。
- GetResourceSnapshot – 使用此動作取得特定機會快照的即時摘要，可讓您檢視up-to-date，而無需直接存取其他合作夥伴的機會。

如果您從其他合作夥伴的機會中識別應該反映在您自己的相關變更或更新，請使用UpdateOpportunity動作。此動作有助於選擇性地將相關資料納入您的機會中，確保整個參與的一致性。

檢視參與成員

多合作夥伴機會的參與最多可以有十個合作夥伴。每當新的合作夥伴接受參與邀請時，就會發佈參與成員新增的事件，通知所有目前成員有關變更的資訊。

若要檢視參與內協作的所有成員，請遵循下列步驟：

1. 使用 ListEngagementResourceAssociations動作來擷取與機會相關聯的參與 ID。
2. 將步驟 1 的 ID 提供給 ListEngagementMembers動作，以擷取參與成員的合作夥伴詳細資訊。

> **Note**
>
> 只有參與的成員可以叫用 `ListEngagementMembers`動作。

監控資源快照任務狀態

使用多合作夥伴機會時，您必須建立一個角色，允許您為其他合作夥伴建立機會的快照，並從參與中的其他合作夥伴擷取機會快照。如果沒有此角色，您對機會所做的任何更新都不會提供給參與中的其他合作夥伴，他們將繼續看到您機會的過時快照。

若要追蹤與參與中多合作夥伴機會相關聯的資源快照任務狀態，請遵循下列步驟：

1. 使用 ListEngagementResourceAssociations動作來擷取與機會相關聯的參與 ID。
2. 將步驟 1 的 ID 提供給 ListResourceSnapshotJobs動作，以產生參與中發起人擁有的所有快照任務清單。從回應擷取任務 ID。

3. 將任務 ID 提供給 GetResourceSnapshotJob動作，以追蹤任務狀態，並查看是否正在執行。

ResourceSnapshotJob 操作會將指標發佈至 Amazon CloudWatch 以進行非同步操作。CloudWatch 會將資料處理為可讀且幾近即時的指標，協助您監控服務的效能和運作狀態。

下列指標可供使用：

- 故障 – 計算任務執行期間的內部問題。使用此指標來監控服務穩定性，並設定故障閾值的警示。
- 錯誤 – 在任務處理期間追蹤客戶相關問題。此指標有助於識別客戶輸入或組態的問題。
- 調用 – 代表任務調用的總數，包括成功和失敗的嘗試。使用此項目追蹤一段時間內的服務用量趨勢。

> **Note**
>
> 只有在ResourceSnapshotJob服務中有活動時，才會向 CloudWatch 報告指標。如果沒有任務或特定指標的資料，則不會報告該指標。

若要確保 操作的順暢ResourceSnapshotJob操作：

- 使用 指標來驗證服務是否如預期般執行。
- 建立 CloudWatch 警示，以在指標超過可接受範圍時監控指標和觸發動作 （例如傳送通知）。
- 設定主動監控以識別和解決問題。

如需使用 CloudWatch 指標的詳細資訊，請參閱 Amazon CloudWatch 使用者指南。

關聯、取消關聯和指派機會

機會可以在機會生命週期的任何階段與合作夥伴解決方案、AWS產品、AWS Marketplace解決方案、AWS Marketplace產品、AWS Marketplace優惠和AWS Marketplace優惠集相關聯或取消關聯。

將機會與其他實體建立關聯

關聯的實體會從 RelatedEntityIdentifiers 物件內的 GetOpportunity方法擷取。您可以使用 RelatedEntityIdentifiers更新 AssociateOpportunity。請注意，此欄位無法使用UpdateOpportunity或 CreateOpportunity方法更新。

解決方案

在開始使用 StartEngagementFromOpportunityTask動作與AWS的參與之前，至少必須建立 1 個和最多 10 個 Partner Solutions 的關聯。AWS推薦不一定包含合作夥伴解決方案。

若要檢視現有的解決方案，請使用 ListSolutions動作。

合作夥伴可以在 AWS Partner Central 的 Build區段中建立、更新和管理其解決方案。

> **Important**
>
> 若要將機會階段進展到已遞交或已啟動，您必須將有效的解決方案或AWS Marketplace產品清單與機會建立關聯。

> **Note**
>
> 仍然支援使用 Solutions實體類型的舊版合作夥伴解決方案 (S-XXXX 識別符）。不過，我們建議您將AwsMarketplaceSolutions實體類型與機會建立關聯，因為Solution實體類型將在未來被取代。

AWS產品

最多可以建立 20 個AWS產品與機會的關聯。若要檢視可用的AWS產品清單，請使用 AWS GitHub 上託管的產品清單。與AWS產品的關聯只使用 AssociateOpportunity動作來完成。若要取代或移除產品，請使用 DisassociateOpportunity動作。

解決方案

機會可以與您的賣方帳戶的解決方案相關聯。使用 AssociateOpportunity動作，並將 RelatedEntityType設定為 AwsMarketplaceSolutions。

實體必須處於公有或有限狀態。服務會拒絕草稿或非作用中的實體。

若要關聯解決方案，需要 ARN。下列範例顯示解決方案的 ARN 格式：

```
arn:aws:aws-marketplace:us-east-1:123456789012:AWSMarketplace/Solution/soln-
ab1c2de3fgh4i
```

您也可以透過合作夥伴AWS Marketplace帳戶連線，將連接至主要帳戶之分公司帳戶的解決方案建立關聯。輸入完整的 ARN，包括附屬公司帳戶 ID。

ListSolutions 動作AwsMarketplaceSolutionArn會傳回每個解決方案摘要，並支援依 ARN 篩選。

AWS Marketplace產品

機會可以與您的賣方帳戶中AWS Marketplace的產品相關聯。使用 AssociateOpportunity動作，並將 RelatedEntityType設定為 AwsMarketplaceProducts。

實體必須處於公有或有限狀態。服務會拒絕草稿或非作用中的實體。

若要關聯產品，需要 ARN。ARN 包含產品類型（例如，、AmiProductSaaSProduct、ContainerProduct)。下列範例顯示 AMI 產品的 ARN 格式：

```
arn:aws:aws-marketplace:us-east-1:123456789012:AWSMarketplace/AmiProduct/prod-
ab1c2de3fgh4i
```

您也可以透過合作夥伴AWS Marketplace帳戶連線，將連接至主要帳戶之分公司帳戶的產品建立關聯。輸入完整的 ARN，包括附屬公司帳戶 ID。

AWS Marketplace優惠和優惠集

AWS Marketplace優惠

機會可以與AWS Marketplace私有優惠相關聯。若要檢視可用的優惠，請使用AWS Marketplace目錄 API 中的 ListEntities。目前，您只能從連結至AWS Partner Central 的AWS Marketplace賣方帳戶關聯優惠。

若要關聯私有優惠 ARN，則需要 。範例：

```
arn:aws:aws-marketplace:us-east-1:123123123123:AWSMarketplace/Offer/offer-
dtn3example1tg
```

AWS Marketplace優惠集

機會也可以與AWS Marketplace優惠集相關聯。優惠集可讓您將多個相關市集優惠分組在一起，以獲得全方位的解決方案封裝。若要檢視可用的優惠集，請使用AWS Marketplace目錄 API 中的 ListEntities。

若要關聯優惠集，需要 ARN。範例：

```
arn:aws:aws-marketplace:us-east-1:123123123123:AWSMarketplace/OfferSet/offerset-
dtn3example1tg
```

重要：一個機會可以與一個優惠或一個優惠集相關聯，但不能同時與兩者相關聯。一次只能將其中一個實體類型與機會相關聯。

取消機會與其他實體的關聯

使用 DisassociateOpportunity動作從下列實體類型取消連結機會：

- 解決方案
- AWS產品
- AWS Marketplace解決方案
- AWS Marketplace產品
- AWS Marketplace優惠
- AWS Marketplace優惠集

根據機會的狀態，取消連結這些相關物件時，會套用不同的驗證規則。

指派機會

使用 AssignOpportunity 變更機會的擁有者。您可以將任何 Partner Central 使用者設定為機會擁有者。

使用潛在客戶

什麼是潛在客戶？

在business-to-business(B2B) 銷售中，潛在客戶代表對公司的產品或服務表達興趣，但尚未完全符合銷售機會資格的潛在客戶。潛在客戶是銷售管道的初始階段，需要培養和資格，才能轉換為主動共同銷售的機會AWS。

AWS與合作夥伴共用的潛在客戶是以客戶參與訊號為基礎，例如網路研討會出席、行銷活動參與或合作夥伴解決方案查詢。這些潛在客戶包括寶貴的內容，例如客戶公司資訊、業務問題和參與詳細資訊，以協助合作夥伴排定優先順序並確定潛在機會的資格。

使用潛在客戶邀請

當潛在客戶對合作夥伴解決方案或服務表達興趣AWS時，合作夥伴會收到來自 的潛在客戶邀請。潛在客戶邀請程序可讓合作夥伴在承諾參與之前評估潛在客戶，確保有效率的資源分配和更高的轉換率。

接收潛在客戶邀請

AWS會建立潛在客戶邀請，並透過銷售 API 與合作夥伴共用。當有新的潛在客戶邀請可用時，AWS會將含潛在客戶內容的參與邀請傳送給符合資格的合作夥伴。

合作夥伴應使用 Amazon EventBridge 監控Engagement Invitation Created事件。此事件通知包含設為 payloadType的欄位LeadInvitation，可讓合作夥伴區分潛在客戶邀請與機會邀請。收到事件時，合作夥伴可以擷取邀請詳細資訊來評估潛在客戶。

列出潛在客戶邀請

合作夥伴可以使用 ListEngagementInvitations API 動作檢視其所有主要邀請。此動作會擷取呼叫委託人為寄件者或接收者的邀請。

若要特別針對潛在客戶邀請進行篩選，合作夥伴應使用值為 的payloadType篩選條件LeadInvitation。此篩選條件可確保只傳回潛在客戶邀請，從結果中排除機會邀請。

潛在客戶邀請可以處於下列狀態：

- 待定：等待合作夥伴接受或拒絕
- 已接受：合作夥伴已接受邀請並取得完整潛在客戶詳細資訊的存取權
- 已拒絕：合作夥伴已拒絕邀請
- 已過期：邀請已超過其過期日期，無需採取動作

評估潛在客戶邀請

接受潛在客戶邀請之前，合作夥伴應該使用 GetEngagementInvitation API 動作擷取詳細的邀請資訊。這為做出明智的接受或拒絕決策提供了基本內容。

領導邀請承載包括：

客戶資訊（可用的預先接受）：

- 公司名稱、產業和國家/地區

- 網站 URL
- 市場客群（企業、大型、中型、小型、微型）
- AWS成熟度層級（評估、單一帳戶、多帳戶）

客戶互動：

- 來源類型和行銷活動資訊
- 客戶採取的動作（表單完成、參加網路研討會等）
- 客戶提供的業務問題描述
- 使用案例類別
- 聯絡人標題（提供 BANT 資格的授權內容）

> **Important**
>
> 邀請階段看不到客戶聯絡資訊（名稱、電子郵件、電話號碼）。只有在接受邀請、確保客戶隱私權並防止潛在客戶耕作行為之後，聯絡人詳細資訊才會變成可用。

接受潛在客戶邀請

當合作夥伴決定尋求潛在客戶時，他們必須使用 `AcceptEngagementInvitation` API 動作接受邀請。此動作會將合作夥伴新增至業務開發，並授予完整潛在客戶詳細資訊的存取權，包括客戶聯絡資訊。

接受後：

- 合作夥伴會新增為參與的成員
- 客戶聯絡資訊會透過業務開發主管內容顯示
- 潛在客戶在AWS系統中被歸類為合作夥伴合格潛在客戶 (PQL)
- 潛在客戶會出現在合作夥伴的作用中潛在客戶清單中
- 合作夥伴可以開始培養潛在客戶並建立客戶聯絡

合作夥伴可以透過`AcceptEngagementInvitation`為每個邀請呼叫，以程式設計方式接受多個潛在客戶邀請，從而有效地大量處理高品質潛在客戶。

拒絕潛在客戶邀請

如果潛在客戶不符合合作夥伴的功能、容量或業務重點，合作夥伴可以使用 RejectEngagementInvitation API 動作拒絕邀請。拒絕潛在客戶邀請時，合作夥伴應提供拒絕原因，以協助AWS改善潛在客戶路由和比對。常見的拒絕原因包括：

- 缺乏地理覆蓋範圍
- 使用案例的技術專業知識不足
- 目前的容量限制條件
- 客戶產業不相符
- 預算不一致

拒絕之後：

- 潛在客戶已從合作夥伴的待定邀請佇列中移除
- 絕不會與合作夥伴共用客戶聯絡資訊
- 潛在客戶在AWS系統中被歸類為合作夥伴拒絕潛在客戶 (PRL)
- 邀請仍會顯示在合作夥伴拒絕的邀請清單中以供參考

如果符合客戶同意要求，拒絕的潛在客戶可能有資格重新指派給其他合作夥伴。

管理已接受的潛在客戶

接受潛在客戶邀請後，合作夥伴可以存取完整的潛在客戶資訊、培養客戶關係，以及追蹤潛在客戶的資格階段。

檢視潛在客戶詳細資訊

合作夥伴會使用 GetEngagement API 動作存取完整的潛在客戶詳細資訊。業務開發包含潛在客戶內容，其中包含有關客戶及其互動的完整資訊AWS。

潛在客戶內容包括：

完成客戶設定檔：

- 來自原始邀請的所有公司資訊
- 所有客戶互動的完整聯絡詳細資訊（姓名、電子郵件、電話、公司職稱）
- AWS成熟度層級和市場客群

互動歷史記錄：

- 將多個客戶接觸點合併為單一互動
- 每個互動都包含來源資訊、客戶動作、業務問題和聯絡詳細資訊
- 互動會依時間順序列出，以提供完整的客戶旅程檢視

資格狀態：

- 潛在客戶資格的目前狀態（不合格、合格、取消資格或自訂狀態）
- 合作夥伴可以在資格程序進行時更新此狀態

此全方位檢視可讓合作夥伴了解客戶跨多個AWS行銷活動和接觸點的完整互動歷程記錄，而不是將每次互動視為單獨的潛在客戶。

列出作用中潛在客戶

合作夥伴可以使用 `ListEngagements` API 動作將`contextType`篩選條件設定為 來擷取其所有作用中的潛在客戶`Lead`。這會傳回合作夥伴是成員且參與包含潛在客戶內容的參與。

清單回應包含重要資訊，例如：

- 業務開發 ID 和標題
- 客戶公司名稱
- 目前的參與度分數
- 資格狀態
- 互動次數
- 建立和修改時間戳記

合作夥伴可以使用此清單來建置儀表板、追蹤潛在客戶管道運作狀態，以及識別需要注意的潛在客戶。

使用 Prospecting Insights 豐富潛在客戶

合作夥伴可以利用AWS洞察來豐富接受的潛在客戶，在投資推廣之前更好地排定優先順序並讓他們符合資格。透過探勘任務以非同步方式執行富集，該任務可增強與AWS衍生訊號的領導參與，以支援資格決策。

啟動探勘任務

合作夥伴使用 StartProspectingFromEngagementTask API 動作啟動擴充。此動作在單一請求中接受最多 100 個參與識別符，允許合作夥伴批次富集潛在客戶。任務會以非同步方式執行，並立即傳回 TaskId和 合作夥伴用來追蹤進度TaskArn的 。初始TaskStatus是 PENDING、IN_PROGRESS、COMPLETED或 之一FAILED。

合作夥伴可以選擇性地提供 TaskName來標記任務，也可以提供 ClientToken來確保重試之間的冪等性。

輪詢結果

由於探勘是非同步的，合作夥伴會使用 GetProspectingFromEngagementTask API 動作與開始時TaskId傳回的 輪詢完成。回應會針對每個提交的識別符，報告整體任務狀態和每個參與結果。

每個業務開發都是獨立處理的，因此個別業務開發可以成功或失敗，而不會影響相同任務中的其他業務開發。對於每個互動，結果包括：

- 業務開發識別符：已處理的業務開發。
- 狀態：PENDING、COMPLETED、 IN_PROGRESS或 FAILED。
- 勘探內容識別符：成功處理參與時填入。此識別符會參考新增至參與的豐富探勘內容，並可用於後續操作。
- 原因代碼和訊息：僅在參與失敗時填入，提供列舉的失敗代碼和具有建議復原步驟的人類可讀取描述。

監控多個任務

為了追蹤許多潛在客戶的擴充，合作夥伴會使用 ListProspectingFromEngagementTasks API 動作。此動作會列出發起人帳戶啟動的所有探勘任務，並支援依任務識別符、任務名稱或啟動時間範圍的選用篩選條件，以及可設定的排序。回應會分頁；使用每個回應中的 NextToken 值來擷取後續頁面。

當擴充成功完成時，AWS洞見會新增至業務開發中，做為潛在客戶開發內容。合作夥伴可以使用 擷取富集的詳細資訊GetEngagement，並使用它們來設定潛在客戶的資格狀態，並決定哪些會導致轉換進展。

更新潛在客戶資訊

當合作夥伴培養潛在客戶並收集其他資訊時，他們可以使用 UpdateEngagementContext API 動作更新潛在客戶詳細資訊。此動作可讓合作夥伴修改業務開發中的潛在客戶內容。

合作夥伴可以更新：

資格狀態：合作夥伴應在進行內部資格程序時更新資格狀態：

- 不合格：接受潛在客戶後的預設狀態；表示潛在客戶需要評估
- 合格：潛在客戶符合資格條件，並顯示轉換為機會的高潛力
- 不符合資格：潛在客戶不符合條件或不適合合作夥伴的解決方案
- 自訂狀態：合作夥伴可以定義自己的資格狀態，以符合其內部程序

潛在客戶中繼資料：合作夥伴可以新增或更新與其資格和培養程序相關的其他資訊。

更新資格狀態有助於合作夥伴追蹤潛在客戶進度、產生準確的管道報告，並識別準備好轉換為機會的潛在客戶。

監控AWS更新

AWS可能會根據新的客戶互動訊號或精簡互動評分來更新潛在客戶資訊。AWS更新互動時，合作夥伴會透過 Amazon EventBridge 接收Engagement Updated事件。

這些更新可能包括：

- 根據新客戶活動來精簡參與度分數
- 來自新行銷活動或接觸點的其他客戶互動
- 更新客戶資訊

合作夥伴應監控這些事件並呼叫 GetEngagement以擷取最新的潛在客戶詳細資訊。這可確保合作夥伴擁有排定優先順序和培養潛在客戶的最新資訊。

Engagement Updated 事件包含 contextTypes 欄位，允許合作夥伴專門針對潛在客戶進行篩選（使用潛在客戶內容進行互動）。

將潛在客戶轉換為機會

當潛在客戶符合資格並展現嚴重的購買意圖時，合作夥伴可以將其轉換為正式與 共同銷售合作的機會 AWS。此轉換會標記從領導培養 （主要由合作夥伴驅動） 到機會管理 （與 合作） 的轉換AWS。

建議方法：便利 API

合作夥伴可以使用StartOpportunityFromEngagementTask方便的 API 動作，簡化機會建立和連結程序。此任務 API 會自動協調多個動作：

1. 使用潛在客戶內容的初步資訊建立草案機會
2. 透過資源快照將機會連結至來源參與
3. 將 CustomerProject 內容新增至業務開發

此便利方法可減少所需的 API 呼叫數量，並確保適當的歸因追蹤。使用此方法的合作夥伴仍需要：

- 使用 完成機會詳細資訊 `UpdateOpportunity`
- 使用 關聯合作夥伴解決方案 `AssociateOpportunity`
- 準備就緒`StartEngagementFromOpportunityTask`時使用 提交機會

對於希望將整合複雜性降至最低lead-to-opportunity工作流程合作夥伴來說，便利 API 特別有用。

替代方法：Step-by-step API

建立草稿機會

轉換潛在客戶的第一步是使用 `CreateOpportunity` API 動作建立草案機會。這會建立將 `Lifecycle.ReviewStatus`設為 的機會`Pending Submission`。在此階段，機會尚未提交至AWS進行驗證。

合作夥伴應將潛在客戶培養期間收集的資訊填入機會，包括：

- 客戶帳戶詳細資訊
- 專案資訊和業務問題
- 預期的客戶支出
- 目標結束日期
- 資格期間收集的任何其他相關詳細資訊

草案機會可讓合作夥伴在提交 之前準備完整的資訊AWS，確保更高的核准率和更快的驗證。

將機會連結至領導參與

建立草案機會後，合作夥伴必須使用 `CreateResourceSnapshot` API 動作將其連結至來源潛在客戶參與。此步驟對於下列項目至關重要：

- 屬性追蹤：從潛在客戶邀請到機會關閉建立來源鏈，為行銷活動和潛在客戶來源提供準確的投資報酬率計算。

- 轉換指標：允許AWS和 合作夥伴測量lead-to-opportunity轉換率，並識別成功的潛在客戶來源。
- 內容保留：維護完整的客戶旅程歷史記錄，包括所有原始互動和互動訊號。

建立資源快照時，合作夥伴會指定：

- 業務開發 ID （來源領導業務開發）
- 機會識別符
- 資源類型 （機會）

此動作會自動將 CustomerProject 內容新增至參與，表示潛在客戶已轉換為作用中的機會。參與現在同時包含原始潛在客戶內容 （保留歷史記錄） 和新的 CustomerProject 內容 （指出作用中的機會）。

完成機會詳細資訊

建立並連結機會後，合作夥伴必須先完成其他機會要求，才能提交。如需詳細資訊，請參閱 `AssociateOpportunity` API 動作。

更新專案詳細資訊：合作夥伴可以使用 `UpdateOpportunity` API 動作來精簡專案資訊、客戶業務問題、預期支出，以及在潛在客戶資格期間收集的其他相關詳細資訊。

提交機會

所有必要資訊完成後，合作夥伴會使用`StartEngagementFromOpportunityTask`方便的 API 提交AWS驗證機會。這會將機會從草稿狀態轉換為AWS檢閱程序。

驗證要求：參與在提交之前必須具有 CustomerProject 內容。使用 `CreateResourceSnapshot`將機會連結至參與時，會自動建立此內容。如果缺少此內容，提交將會失敗。

提交後：

- 機會`Lifecycle.ReviewStatus`的變更 `Submitted`
- 潛在客戶在AWS系統中被歸類為合作夥伴銷售合格潛在客戶 (PSQL)
- AWS開始驗證，以確保機會詳細資訊準確且完整
- 在檢閱程序完成之前，無法對機會進行任何變更

機會接著會遵循「使用機會」文件中描述的標準驗證工作流程，逐步完成審核中、必要動作、已核准或取消資格等狀態。

使用交易規模洞察

什麼是 Deal Sizing？

Deal Sizing 是AWS Partner Central 中 APN Customer Engagements (ACE) 機會的功能，使用 AI 在合作夥伴建立或更新機會時提供自動交易大小預估AWS和服務建議。

AI 預測 MRR 估計值

交易規模調整洞見會根據合作夥伴的歷史AWS機會、使用案例和業務描述，提供預測每月經常性收入 (MRR) 範圍的中位數。當合作夥伴收集有關交易和銷售週期進度的詳細資訊時，合作夥伴應該檢閱和更新總 MRR。

AI 建議AWS產品

AI 建議產品是根據客戶業務問題描述和機會詳細資訊來識別。AI 建議符合技術需求和使用案例的相關AWS產品。合作夥伴應檢閱這些建議，並自訂產品選擇以符合客戶的特定需求。

具有定價計算器 URLs增強型洞見

合作夥伴也可以匯入AWS定價計算器 URLs，以自動填入服務選擇，並接收更深入的洞見，包括遷移加速計劃 (MAP) 資格指標、最佳化建議和潛在的成本節省分析。

了解來源分隔資料

Deal Sizing 提供兩個不同來源的洞見，讓您全面了解機會價值：

- AWS來源：根據類似歷史機會的模式的 AI 建議產品
- 合作夥伴來源：從AWS定價計算器匯入您自己的預估

處理來源之間的差異

當合作夥伴提供的預估值與AWS建議大不相同時，合作夥伴應該：

1. 檢閱機會詳細資訊，以確保擷取所有相關資訊
2. 確認定價計算器 URL 組態符合實際需求
3. 將AWS建議視為類似歷史機會告知的資料點
4. 根據特定客戶內容和需求做出明智的決策

存取 Deal Sizing Insights

合作夥伴透過 GetAwsOpportunitySummary API 動作存取 Deal Sizing 資料，該動作會傳回包含交易大小預測、產品建議和最佳化洞見的延伸AwsOpportunitySummary物件。

當交易規模調整資料變成可用時

在建立具有足夠客戶和專案詳細資訊的機會之後，Deal Sizing 洞見就會變成可用。系統會分析機會屬性並產生：

1. AI 預測 MRR：根據機會特性自動計算
2. AWS產品建議：與客戶業務問題和技術需求相關的服務
3. 增強的洞見 （提供定價計算器 URL 時）：MAP 資格、最佳化建議和成本節省分析

> **Note**
>
> Deal Sizing 洞見不適用於所有機會。資格取決於合作夥伴歷史記錄和所提供機會詳細資訊的完整性等因素。

提供交易規模調整輸入

合作夥伴透過 CreateOpportunity和 UpdateOpportunity API 動作提供 Deal Sizing 的輸入。系統會使用機會屬性來產生建議。

提供合約總值 (TCV)

預估合約總價值而非每月經常性收入的合作夥伴可以直接提供其交易價值。AWS 會使用採用 AI 技術的轉換模型，自動將 TCV 轉換為 AWS MRR。

對於 TCV 中的交易大小，請提供 ExpectedContractDuration為 Months（有效值範圍為 1 到 144 個月），以及 TargetCompany 為 Self和 Frequency None。

```
{
  "Project": {
    "ExpectedContractDuration": { "Term": "Months", "Value": "36" },
    "ExpectedCustomerSpend": [
      {
        "Amount": "1200000.00",
        "CurrencyCode": "USD",
        "Frequency": "None",
```

```
        "TargetCompany": "Self"
      }
    ]
  }
}
```

AWS 衍生 AWS MRR 預估，並在 上傳回兩個項目GetOpportunity：

```
{
  "Project": {
    "ExpectedContractDuration": { "Term": "Months", "Value": "36" },
    "ExpectedCustomerSpend": [
      {
        "Amount": "5600.00",
        "CurrencyCode": "USD",
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      },
      {
        "Amount": "1200000.00",
        "CurrencyCode": "USD",
        "Frequency": "None",
        "TargetCompany": "Self"
      }
    ]
  }
}
```

在上述範例中，ExpectedCustomerSpend第一個陣列包含 AWS MRR 估計值 (TargetCompany: "AWS"、Frequency: "Monthly")，而第二個陣列是合作夥伴的總合約值 (TargetCompany: "Self"、Frequency: "None")。

TCV 驗證規則

所有 TCV 驗證錯誤都會傳回INVALID_VALUE具有 的錯誤代碼Reason: "BUSINESS_VALIDATION_FAILED"。區分因素是 FieldName和 Message：

條件	欄位	訊息
沒有 的自我輸入 ExpectedContractDuration	project.expectedContractDuration	存在自我項目時，需要 ExpectedContractDuration

條件	欄位	訊息
ExpectedContractDuration 沒有自我輸入（僅建立）	project.expectedCustomerSpend	存在 ExpectedContractDuration 時，需要自我項目
多個自我項目	project.expectedCustomerSpend	僅允許一個自我項目
當持續時間存在時 TargetCompany 無效	project.expectedCustomerSpend.targetCompany	TargetCompany 必須是 'AWS' 或 'Self'
AWS 與 Self 之間的貨幣不相符	project.expectedCustomerSpend.currencyCode	項目之間的 CurrencyCode 必須相符
自我輸入 + EstimationUrl	project.expectedCustomerSpend.estimationUrl	自我輸入和 EstimationUrl 無法共存

> **Note**
>
> 在 上UpdateOpportunity，如果 ExpectedContractDuration 在沒有自我項目的情況下存在，但合作夥伴交易值存在於儲存體中，則系統會自動重建自我項目，不會擲出錯誤。

在 TCV 和手動 MRR 之間切換

若要從 TCV 模式切換回手動 MRR，請明確ExpectedContractDuration設為 null並僅提供AWS項目：

```
{
  "Project": {
    "ExpectedContractDuration": null,
    "ExpectedCustomerSpend": [
      {
        "Amount": "8000.00",
        "CurrencyCode": "USD",
```

```
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      }
    ]
  }
}
```

> **Note**
>
> ExpectedContractDuration 從承載省略（未傳送欄位）表示「無變更」— 保留現有的 TCV 資料。明確傳送null表示「清除 TCV 模式」。

TCV 和定價計算器 URLs

如果合作夥伴同時提供Self項目 (TCV 模式）和 EstimationUrl，API 會傳回驗證錯誤。自我輸入和 EstimationUrl 不能在相同的請求中共存。

提供手動 MRR 估算

合作夥伴可以使用 Project.ExpectedCustomerSpend[].Amount 欄位手動輸入 MRR 預估值：

> **Note**
>
> CurrencyCode 欄位接受 USD和 EUR。如果 AWS 分割區是 aws-eusc(AWS 歐洲主權雲端），則貨幣代碼必須為 EUR。

```
{
  "Project": {
    "ExpectedCustomerSpend": [
      {
        "Amount": "25000.00",
        "CurrencyCode": "USD",
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      }
    ]
  }
}
```

此欄位適用於所有 合作夥伴，並可透過設定相同的值或以合作夥伴提供的值覆寫來接受 AI 預測。

提供定價計算器 URLs

合作夥伴可以選擇性地透過 `Project.ExpectedCustomerSpend[].EstimationUrl` 欄位提供AWS定價計算器 URLs，以自動匯入服務組態，並接收增強的洞見，包括 MAP 資格指標、最佳化建議和成本節省分析。

```
{
  "Project": {
    "ExpectedCustomerSpend": [
      {
        "Amount": null,
        "CurrencyCode": "USD",
        "EstimationUrl": "https://calculator.aws/#/estimate?id=abc123",
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      }
    ]
  }
}
```

提供有效的定價計算器 URL 時，系統會自動從計算器組態計算 MRR 數量，並在 的合作夥伴來源中填入金額欄位和詳細的產品層級支出資料`AwsProductsSpendInsightsBySource`。合作夥伴在提供 EstimationUrl 時，應將 Amount 設定為 null - 系統會自動計算它。

無法解析格式無效和 URLs的 URLs 將遭 拒絕`ValidationException`。

金額和 EstimationUrl 如何一起運作

您可以靈活地提供機會的每月經常性收入 (MRR) 預估。API 可智慧地處理各種手動金額和AWS定價計算器 URLs的組合，以確保您的機會始終可以成功建立。

僅使用手動量

當您僅提供金額欄位時，API 會儲存您的手動預估值，並將 EstimationUrl 設定為 null。當您從客戶對話或自己的計算中擁有特定的 MRR 預估時，此方法很有用。

範例請求：

```
{
    "Project": {
```

```
        "ExpectedCustomerSpend": [{
            "Amount": "25000.00",
            "CurrencyCode": "USD",
            "Frequency": "Monthly",
            "TargetCompany": "AWS"
        }]
    }
}
```

僅使用AWS定價計算器 URL

當您僅提供 EstimationUrl 欄位時：

- 有效 URL – API 會從計算器組態計算 MRR 數量，並同時儲存 URL 和計算數量。
- 無效的 URL – API 會傳回 ValidationException，其中包含 URL 無效原因的詳細資訊。

範例請求：

```
{
    "Project": {
        "ExpectedCustomerSpend": [{
            "EstimationUrl": "https://calculator.aws/#/estimate?id=abc123",
            "CurrencyCode": "USD",
            "Frequency": "Monthly",
            "TargetCompany": "AWS"
        }]
    }
}
```

錯誤回應範例（無效的 URL)：

```
{
    "ErrorList": [{
        "Code": "INVALID_VALUE",
        "FieldName": "project.expectedCustomerSpend.estimationUrl",
        "Message": "Invalid Estimation URL: https://calculator.aws/#/estimate?
id=invalid"
    }],
    "Message": "The provided Estimation URL is invalid",
    "Reason": "INVALID_VALUE"
}
```

同時提供 Amount 和 EstimationUrl

當您提供這兩個欄位時，API 會優先確保成功建立您的機會：

1. 如果兩個值都保持不變 – 不會對這些欄位執行更新
2. URL 無效 – API 會儲存您提供的金額，並將 EstimationUrl 設定為 null，讓您可繼續建立機會。
3. URL 有效，且計算金額與您提供的金額相符 – 這兩個值都會儲存。
4. URL 有效，但計算金額與您提供的金額不符 – API 會儲存您提供的金額，並將 EstimationUrl 設定為 null，而不會傳回錯誤。

> **Note**
>
> 當這兩個值都不符合現有的值時，API 會先嘗試從 URL 計算數量。當發生不相符或驗證問題時，您手動提供的金額一律優先。

主要優點：無效的 URLs絕不會封鎖建立機會

此方法可確保無效或無法存取的AWS定價計算器 URLs 永遠不會阻止您建立或更新機會。發生衝突或驗證問題時，您手動提供的金額一律優先，讓您完全掌控機會資料。

了解擴充 AwsOpportunitySummary Data Model

本節說明 `GetAwsOpportunitySummary` API 動作傳回的回應結構。

> **Note**
>
> AI 預測的 MRR 傳回於 `Project.ExpectedCustomerSpend[].Amount`

`GetAwsOpportunitySummary` API 動作會透過提供具有來源歸因之完整支出分析的新`Insights.AwsProductsSpendInsightsBySource`結構，傳回交易規模分析洞見。

來源分隔：AWS vs 合作夥伴資料

Deal Sizing 會依來源區隔洞見，以提供透明度並防止重複計數：

- AWS來源：來自 的 AI 產品建議AWS

- 合作夥伴來源：從合作夥伴提供的定價計算器 URLs 匯入的支出資料和產品詳細資訊

此區隔可讓合作夥伴：

1. 比較其預估值與AWS建議
2. 驗證其大小假設
3. 不小心合併這兩個來源，以避免膨脹總計
4. 保持對資料來源的可見性

結構概觀

```
{
  "Insights": {
    "AwsProductsSpendInsightsBySource": {
      "<AWS | Partner>": {
        "CurrencyCode": "string",
        "Frequency": "string",
        "TotalAmount": "string",
        "TotalOptimizedAmount": "string",
        "TotalPotentialSavingsAmount": "string",
        "TotalAmountByCategory": { },
        "AwsProducts": [ ]
      }
    }
  },
  "Project": {
    "ExpectedCustomerSpend": [ ]
  },
  "RelatedEntityIds": {
    "AwsProducts": [ ]
  }
}
```

欄位參考

AwsProductsSpendInsightsBySource 地圖

以來源分隔的支出洞察，提供AWS建議和合作夥伴預估的獨立分析。

`Insights.AwsProductsSpendInsightsBySource` 欄位是最多包含兩個索引鍵的映射：

金鑰	Type	說明
AWS	AwsProductsSpendInsights	AI 產生的洞見，包括來自 的建議產品AWS
合作夥伴	AwsProductsSpendInsights	定價計算器 URLs包括詳細的服務成本和最佳化

> **Note**
>
> 只有具有可用資料的來源才會包含在回應中。新機會通常從僅填入AWS來源開始。

AwsProductInsights 物件

單一來源 (AWS或合作夥伴） 的全面支出分析，包括總金額、最佳化節省、計劃類別明細，以及詳細的產品層級洞察。

每個來源物件都包含下列欄位：

欄位	Type	說明
CurrencyCode	string	ISO 4217 貨幣代碼。支援的值為 "USD" 和 "EUR"。
Frequency (頻率)	string	指出預期客戶花費預計金額的頻率。頻率欄位只允許每月值，代表經常性每月支出。
TotalAmount	string	最佳化之前此來源的預估總支出
TotalOptimizedAmount	string	套用建議的最佳化後的總預估花費
TotalPotentialSavingsAmount	string	透過實作最佳化來實現量化的節省

欄位	Type	說明
TotalAmountByCategory	object	映射至AWS計劃和現代化途徑的支出金額
AwsProducts	陣列	產品層級詳細資訊，包括成本和最佳化建議

目前狀態限制：對於AWS來源，當每個產品支出建議不可用時，可能會省略 TotalAmount、TotalOptimizedAmount 和 TotalPotentialSavingsAmount。在這些情況下`Project.ExpectedCustomerSpend[].Amount`，合作夥伴應該參考總AWS MRR 估計。

TotalAmountByCategory 物件

將支出金額映射至AWS計劃和策略計劃：

類別索引鍵	說明
符合 MAP 資格	符合 Migration Acceleration Program 資金資格的服務支出
CAT 合格	支援成本分配和追蹤的服務
路徑 - 移至雲端原生	與雲端原生現代化一致的服務
路徑 - 移至 受管服務	支援受管服務採用的服務
路徑 - 移至開放原始碼	開放原始碼服務替代方案
路徑 - 移至容器	容器型服務選項
路徑 - 移至無伺服器	無伺服器運算服務
路徑 - 移至資料庫	資料庫現代化服務
路徑 - 移至分析	分析和資料處理服務

> **Note**
>
> 類別總量可能會超過 TotalAmount，因為個別產品可以屬於多個類別。

AwsProducts 陣列

具有計劃資格指標 (MAP、現代化途徑）、成本估算和最佳化建議的AWS服務清單。

每個產品物件都包含：

欄位	類型	必要	說明
ProductCode	string	是	AWS用於機會關聯的 Partner Central 產品識別符
ServiceCode	string	否	定價計算器服務代碼（原始計算器 URL 的連結）
類別	陣列	是	此產品符合的計劃和路徑類別清單
Amount	string	否	最佳化前的基準服務成本AWS（來源建議可能為 null)
OptimizedAmount	string	否	套用最佳化後的服務成本 (AWS來源建議可能為 null)
PotentialSavingsAmount	string	否	透過最佳化降低服務特定的成本 (AWS對於來源建議可能為 null)
最佳化	陣列	否	此產品的特定最佳化建議清單

Null 處理：當每個產品支出建議不可用時，AWS來源產品、金額、OptimizedAmount 和 PotentialSavingsAmount 欄位可能為 Null。系統會`Project.ExpectedCustomerSpend[0].Amount`改為透過 提供總 MRR。

最佳化物件

每個最佳化建議都包含：

欄位	Type	說明
Description	string	最佳化策略的人類可讀說明
SavingsAmount	string	實作此最佳化可實現量化的成本節省

範例

範例 1：僅限AWS建議（目前狀態）

此範例顯示只有 AI 建議可用且尚未建議每個產品支出金額時的典型回應。

```
{
  "Catalog": "AWS",
  "Insights": {
    "AwsProductsSpendInsightsBySource": {
      "AWS": {
        "CurrencyCode": "USD",
        "Frequency": "Monthly",
        "AwsProducts": [
          {
            "ProductCode": "AmazonEC2",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging"],
            "Amount": null,
            "OptimizedAmount": null,
            "PotentialSavingsAmount": null,
            "Optimizations": []
          },
          {
            "ProductCode": "AWSLambda",
            "Categories": ["Cost Allocation Tagging", "Pathway - Move to Cloud
 Native"],
            "Amount": null,
```

```
            "OptimizedAmount": null,
            "PotentialSavingsAmount": null,
            "Optimizations": []
          },
          {
            "ProductCode": "AmazonRDS",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging", "Pathway - Move
 to Managed Services"],
            "Amount": null,
            "OptimizedAmount": null,
            "PotentialSavingsAmount": null,
            "Optimizations": []
          }
        ]
      }
    }
  },
  "Project": {
    "ExpectedCustomerSpend": [
      {
        "Amount": "28000.00",
        "CurrencyCode": "USD",
        "EstimationUrl": null,
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      }
    ]
  },
  "RelatedEntityIds": {
    "AwsProducts": [
      "AmazonEC2",
      "AWSLambda",
      "AmazonRDS"
    ]
  }
}
```

Key Points：AWS source 會顯示具有類別但沒有每個產品數量的建議產品。可透過 取得的總 MRR 預估`Project.ExpectedCustomerSpend[].Amount`。

範例 2：合作夥伴定價計算器匯入

此範例顯示合作夥伴提供有效定價計算器 URL 時的增強洞察。

```
{
  "Catalog": "AWS",
  "Insights": {
    "AwsProductsSpendInsightsBySource": {
      "Partner": {
        "CurrencyCode": "USD",
        "Frequency": "Monthly",
        "TotalAmount": "32500.00",
        "TotalOptimizedAmount": "30000.00",
        "TotalPotentialSavingsAmount": "2500.00",
        "TotalAmountByCategory": {
          "MAP Eligible": "22500.00",
          "Cost Allocation Tagging": "32500.00",
          "Pathway - Move to Cloud Native": "5000.00",
          "Pathway - Move to Managed Services": "7500.00"
        },
        "AwsProducts": [
          {
            "ServiceCode": "ec2",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging"],
            "Amount": "15000.00",
            "OptimizedAmount": "13500.00",
            "PotentialSavingsAmount": "1500.00",
            "Optimizations": [
              {
                "Description": "Convert to 3-year reserved instances",
                "SavingsAmount": "1000.00"
              },
              {
                "Description": "Migrate to Graviton-based instances",
                "SavingsAmount": "500.00"
              }
            ]
          },
          {
            "ServiceCode": "lambda",
            "Categories": ["Cost Allocation Tagging", "Pathway - Move to Cloud
 Native"],
            "Amount": "5000.00",
            "OptimizedAmount": "5000.00",
            "PotentialSavingsAmount": "0.00",
            "Optimizations": []
          },
```

```
          {
            "ServiceCode": "AmazonRDS",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging", "Pathway - Move
 to Managed Services"],
            "Amount": "7500.00",
            "OptimizedAmount": "6500.00",
            "PotentialSavingsAmount": "1000.00",
            "Optimizations": [
              {
                "Description": "Optimize Multi-AZ for dev/test environments",
                "SavingsAmount": "1000.00"
              }
            ]
          },
          {
            "ServiceCode": "AmazonS3",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging"],
            "Amount": "5000.00",
            "OptimizedAmount": "5000.00",
            "PotentialSavingsAmount": "0.00",
            "Optimizations": []
          }
        ]
      }
    }
  },
  "Project": {
    "ExpectedCustomerSpend": [
      {
        "Amount": "32500.00",
        "CurrencyCode": "USD",
        "EstimationUrl": "https://calculator.aws/#/estimate?id=abc123",
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      }
    ]
  },
  "RelatedEntityIds": {
    "AwsProducts": []
  }
}
```

關鍵點：合作夥伴來源包含每個產品金額的完整支出詳細資訊。最佳化建議提供可行的成本降低策略。類別明細顯示 MAP 資格 （總計 32，500 USD)。產品包含連結至定價計算器組態的 ServiceCode。

範例 3：兩個來源都存在 （比較案例）

此範例示範AWS建議和合作夥伴預估如何同時顯示，以便進行比較。

```
{
  "Catalog": "AWS",
  "Insights": {
    "AwsProductsSpendInsightsBySource": {
      "AWS": {
        "CurrencyCode": "USD",
        "Frequency": "Monthly",
        "AwsProducts": [
          {
            "ProductCode": "AmazonEC2",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging"],
            "Amount": null,
            "OptimizedAmount": null,
            "PotentialSavingsAmount": null,
            "Optimizations": []
          },
          {
            "ProductCode": "AWSLambda",
            "Categories": ["Cost Allocation Tagging", "Pathway - Move to Cloud
 Native"],
            "Amount": null,
            "OptimizedAmount": null,
            "PotentialSavingsAmount": null,
            "Optimizations": []
          },
          {
            "ProductCode": "EBS",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging"],
            "Amount": null,
            "OptimizedAmount": null,
            "PotentialSavingsAmount": null,
            "Optimizations": []
          },
          {
            "ProductCode": "RDS",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging", "Pathway - Move
 to Managed Services"],
```

```
            "Amount": null,
            "OptimizedAmount": null,
            "PotentialSavingsAmount": null,
            "Optimizations": []
          }
        ]
      },
      "Partner": {
        "CurrencyCode": "USD",
        "Frequency": "Monthly",
        "TotalAmount": "32500.00",
        "TotalOptimizedAmount": "30000.00",
        "TotalPotentialSavingsAmount": "2500.00",
        "TotalAmountByCategory": {
          "MAP Eligible": "22500.00",
          "Cost Allocation Tagging": "32500.00",
          "Pathway - Move to Cloud Native": "5000.00",
          "Pathway - Move to Managed Services": "7500.00"
        },
        "AwsProducts": [
          {
            "ServiceCode": "ec2",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging"],
            "Amount": "15000.00",
            "OptimizedAmount": "13500.00",
            "PotentialSavingsAmount": "1500.00",
            "Optimizations": [
              {
                "Description": "Convert to 3-year reserved instances",
                "SavingsAmount": "1000.00"
              }
            ]
          },
          {
            "ServiceCode": "lambda",
            "Categories": ["Cost Allocation Tagging", "Pathway - Move to Cloud
Native"],
            "Amount": "5000.00",
            "OptimizedAmount": "5000.00",
            "PotentialSavingsAmount": "0.00",
            "Optimizations": []
          },
          {
            "ServiceCode": "amazonRDS",
```

```
            "Categories": ["MAP Eligible", "Cost Allocation Tagging", "Pathway - Move
 to Managed Services"],
            "Amount": "7500.00",
            "OptimizedAmount": "6500.00",
            "PotentialSavingsAmount": "1000.00",
            "Optimizations": [
              {
                "Description": "Optimize Multi-AZ for dev/test environments",
                "SavingsAmount": "1000.00"
              }
            ]
          },
          {
            "ServiceCode": "amazonS3",
            "Categories": ["MAP Eligible", "Cost Allocation Tagging"],
            "Amount": "5000.00",
            "OptimizedAmount": "5000.00",
            "PotentialSavingsAmount": "0.00",
            "Optimizations": []
          }
        ]
      }
    }
  },
  "Project": {
    "ExpectedCustomerSpend": [
      {
        "Amount": "28000.00",
        "CurrencyCode": "USD",
        "EstimationUrl": "https://calculator.aws/#/estimate?id=abc123",
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      }
    ]
  },
  "RelatedEntityIds": {
    "AwsProducts": [
      "AmazonEC2",
      "AWSLambda",
      "EBS",
      "RDS"
    ]
  }
```

```
}
```

重點：這兩個來源都存在：AWS建議 （總計 28，000 美元） 和合作夥伴計算器 （總計 32.5，000 美元）。AWS建議顯示 4 個產品；合作夥伴計算器顯示 4 個產品 (3 個重疊）。合作夥伴可以將AWS建議與其詳細的預估值進行比較。可透過 取得AWS總計`Project.ExpectedCustomerSpend[].Amount`。

最佳實務

處理 Null 值

1. EstimationUrl 欄位：預期大多數合作夥伴的 EstimationUrl 為 null。這是標準行為，不應視為錯誤。將手動 MRR 項目選項顯示為主要路徑。
2. 缺少類別：如果AWS來源不存在 TotalAmountByCategory，這表示每個產品建議不可用。提供定價計算器 URLs 時，類別明細適用於合作夥伴來源的資料。

最佳化建議

1. 顯示可行的洞見：顯著顯示最佳化描述和節省金額，以協助合作夥伴了解降低成本的機會。
2. 總節省：跨產品的 PotentialSavingsAmount 總和，以顯示總最佳化潛力。
3. 依影響排定優先順序：依 SavingsAmount 排序最佳化，以協助合作夥伴專注於影響程度最高的建議。

監控更新

1. 定期輪詢：在機會建立工作流程期間`GetAwsOpportunitySummary`定期輪詢 （建議：每 5 分鐘）。
2. 處理非同步產生：在建立機會之後，可能不會立即提供 Deal Sizing 洞見。實作適當的載入狀態並重試邏輯。

API 整合模式

1. 利用現有的整合：已呼叫 的合作夥伴`GetAwsOpportunitySummary`只需要使用回應中的其他欄位，不需要新的 API 呼叫。
2. 緩慢降級：設計 UIs 來處理 Deal Sizing 資料尚未可用或不完整的情況。

資料品質和準確性

1. 驗證定價計算器 URLs：確認 URLs提交之前有效，以避免 null EstimationUrl 回應。
2. 提供豐富的機會詳細資訊：更完整的機會資訊（客戶詳細資訊、專案描述、技術需求）可產生更準確的 AI 建議。
3. 檢閱差異：當合作夥伴預估與AWS建議有顯著差異時，請檢閱機會詳細資訊，以確保擷取所有相關內容。

最佳實務

對事件做出反應

從AWS Partner Central API 處理事件時，請確保您的處理邏輯具有處理重複事件的等冪性。請考慮根據您的應用程式需求批次處理或選擇性擷取詳細資訊，而不是針對每個事件立即進行 GetOpportunity 呼叫。對於不中斷的操作，請注意配額。

實作樂觀鎖定

樂觀鎖定可防止在並行更新期間意外的資料覆寫。以下是典型的案例：

1. 合作夥伴從其 CRM 系統擷取機會。
2. 使用者A更新AWS Partner Central 上的機會。
3. 使用者透過 CRM 整合同時B更新相同的機會。
4. 如果資料變更，CRM 系統會嘗試上傳資料，但會傳回 `ConflictException`。
5. 使用者檢閱錯誤並手動解決衝突的資料。

若要避免這種情況，所有 UpdateOpportunity 請求都必須包含 參數，您可以從先前的 CreateOpportunity、UpdateOpportunity 和 GetOpportunity 動作取得該`LastModifiedDate`參數。只有在 `LastModifiedDate`符合我們的系統時，更新才會成功。如果沒有，您必須`LastModifiedDate`使用 GetOpportunity 擷取最新的 ，並重新嘗試更新。

在 CRM 和AWS Partner Central 之間同步資料

請務必讓您的系統與 Partner Central 的最新資料保持同步。以下是確保系統反映最新資料的兩個策略：

使用事件（建議）

1. 使用 ListOpportunities 載入資料。

2. 訂閱機會事件。
3. 回應新的機會或變更。
   - 當您收到 Opportunity Created、 Opportunity Updated或 Engagement Invitation Accepted事件時，請使用 GetOpportunity來擷取最新的資料。
   - 當您收到Engagement Invitation Rejected事件時，請移除對應的機會。

使用 ListOpportunities 輪詢

1. 使用 ListOpportunities 載入資料。
2. 選擇輪詢頻率，確保不會太頻繁，以免耗盡您的每日讀取配額。
3. LastModifiedDate 從儲存的資料中識別最新的 ，確保其源自於AWS。
4. 呼叫 ListOpportunities 時，請使用AfterLastModifiedDate篩選條件中的時間戳記。

```
{
   "FilterList": [
       {
           "Name": "AfterLastModifiedDate",
           "ValueList": [ "2023-05-01T20:37:46Z" ] // Replace with actual timestamp
 of your last synced data
        }
   ]
}
```

5. AWS將傳回在時間戳記上指定的值之後建立或更新的機會。
6. 使用 逐一查看所有傳回的頁面NextToken，並使用 GetOpportunity 更新系統的資料。

```
{
   "NextToken": "AAMA-EFRSN...PZa942D",
   "FilterList": [
       {
           "Name": "AfterLastModifiedDate",
           "ValueList": [ "2023-05-01T20:37:46Z" ] // Replace with actual timestamp
 of your last synced data
        }
   ]
}
```

使用AWS Partner Central Benefits API

AWS Partner Central Benefits API 簡化了合作夥伴在所有AWS合作夥伴計劃中探索、申請和管理利益的方式。透過將利益管理集中到單一直覺式界面，該服務會建立透明的功績制，獎勵合作夥伴，同時簡化操作。

API 提供標準AWS API 功能。使用 API 動作或使用專為您的程式設計語言或平台量身打造的AWS SDK 來存取它。如需詳細資訊，請參閱 入門AWS和要建置的工具AWS。

什麼是利益？

在AWS合作夥伴網路 (APN) 中，利益代表合作夥伴可以從中取得的優勢或功能，AWS以支援其業務成長和客戶成功。優勢旨在根據合作夥伴的資格和貢獻獎勵合作夥伴，同時提供實際價值，協助合作夥伴擴展其AWS實務。

優點可能有多種形式，包括：

- 財務獎勵 - 點數、現金支出和資金計劃，例如遷移加速計劃 (MAP) 或行銷開發資金 (MDF)
- 存取優點 - 預覽對新服務、工具或專業資源的存取
- 辨識 - 數位徽章、認證、能力指定和合作夥伴層狀態
- 技術資源 - 解決方案架構協助、技術支援和 ProServe 參與
- 行銷支援 - 聯合行銷機會、精選合作夥伴狀態和成功案例出版物

探索可用的優勢

合作夥伴透過探索他們可獲得哪些好處並了解資格要求，開始他們的利益之旅。Benefits API 透過 Benefit Discovery APIs提供全面的探索功能。

列出可用的優勢

合作夥伴可以使用 `ListBenefits` API 動作檢視所有可用的利益。此動作會擷取具有選用篩選功能的優點目錄，以協助合作夥伴快速尋找相關機會。

為了有效地探索好處，合作夥伴可以依下列條件進行篩選：

- 計劃 - 依特定AWS合作夥伴計劃篩選，例如 MAP （遷移加速計劃）、MDF （行銷開發資金）、ISV 工作負載遷移、創新沙盒、概念驗證或合作夥伴計畫資金
- 履行類型 - 依優點的交付方式篩選：`CREDITS`、`CASH`、`DISCOUNTACCESS`、、 `RECOGNITION`或 `RESOURCE`

- 狀態 - 依利益狀態篩選： ACTIVE 或 DEPRECATED

ListBenefits API 會傳回利益摘要，其中包含基本資訊，例如利益 ID、名稱、描述、相關聯的計劃和履行類型。此摘要檢視可讓合作夥伴快速掃描可用的利益，並識別與其業務目標最相關的利益。

篩選方法範例：

專注於財務獎勵的合作夥伴可能會依履行類型篩選CASH，CREDITS並查看資金機會。尋求技術啟用的合作夥伴可能會依履行類型篩選ACCESS，RESOURCE或尋找技術支援優勢和工具存取。

檢視詳細利益資訊

一旦合作夥伴識別感興趣的利益，他們就可以使用 GetBenefit API 動作擷取完整詳細資訊。這可提供完整的資訊，包括：

- 資格條件 - 合作夥伴必須符合的詳細要求，才能獲得利益
- 利益請求結構描述 - 申請時合作夥伴需要提供的特定資訊和文件
- 授予詳細資訊 - 核准時會收到哪些合作夥伴
- 關聯的計劃 - 利益支援的AWS合作夥伴計劃

利益請求結構描述特別重要，因為它定義利益應用程式的結構和必要欄位。合作夥伴應在建立利益應用程式之前仔細檢閱此結構描述，以確保他們預先收集所有必要資訊。

> **Important**
>
> 每個利益的資格判斷由用戶端或 UI layer 處理。對於與資金相關的利益，資格通常是以合作夥伴的計分卡效能和計劃參與為基礎。

主題

- 使用利益應用程式
- 使用利益分配

使用利益應用程式

利益應用程式會為合作夥伴對特定利益的請求建立模型。它會擷取根據利益特定條件評估和處理請求所需的所有資訊。利益應用程式生命週期會經歷多種狀態，從建立草稿到最終核准或拒絕。

建立利益應用程式

合作夥伴使用 CreateBenefitApplication API 動作建立利益應用程式，以啟動利益請求程序。建立時，應用程式會進入 PENDING_SUBMISSION 狀態，讓合作夥伴在提交審核之前準備完整的資訊。

建立利益應用程式時，合作夥伴必須提供：

- 利益識別符 - 所請求利益的 ID 或 ARN
- 履行類型 - 合作夥伴預期如何獲得利益 DISCOUNT(CREDITS、CASH、RECOGNITION、、ACCESS或 RESOURCE)
- 利益應用程式詳細資訊 - 包含利益請求結構描述所定義之利益特定資訊的 JSON 文件
- 用戶端字符 - 防止重複提交的唯一冪等字符

合作夥伴可以選擇性地提供：

- 名稱和描述 - 應用程式的人類可讀取識別符
- 合作夥伴聯絡人 - 管理此利益請求的人員的聯絡資訊（最多 1 位聯絡人）
- 檔案附件 - 支援文件，例如專案計劃、SOWs、成本證明或客戶滿意度調查（最多 10 個檔案）
- 相關資源 - 相關機會或現有利益分配的連結（最多 10 個資源）
- 標籤 - 資源組織和追蹤的鍵值對（最多 200 個標籤）

CreateBenefitApplication API 會執行軟驗證，只檢查基本欄位類型和模式。完整商業邏輯驗證會在提交期間進行，允許合作夥伴儲存不完整的應用程式，並在稍後傳回以完成它們。

最佳實務：合作夥伴應使用來自 的利益請求結構描述GetBenefit，確切了解在建立應用程式之前需要哪些資訊。這可降低因資料遺失或無效而提交失敗的可能性。

更新利益應用程式

合作夥伴可以使用 UpdateBenefitApplication API 動作修改草稿利益應用程式。這可讓合作夥伴在提交之前精簡資訊、新增文件或更正錯誤。

更新利益應用程式時，合作夥伴必須提供：

- 識別符 - 要更新之利益應用程式的 ID 或 ARN
- 修訂 - 樂觀鎖定的目前修訂編號
- 利益應用程式詳細資訊 - 完整、更新的 JSON 文件

合作夥伴必須提交完整的利益應用程式物件，即使只有特定欄位正在變更。最佳實務是先使用 擷取最新的應用程式詳細資訊GetBenefitApplication、修改必要欄位，然後將完整更新的承載提交至UpdateBenefitApplication。

樂觀鎖定：修訂欄位可確保只有在應用程式自上次擷取以來未變更時，才會套用更新。如果修訂版不符合目前的資料庫值，則更新會因衝突錯誤而遭到拒絕。這可防止合作夥伴意外覆寫其他程序或系統所做的變更。

更新只能在應用程式處於 PENDING_SUBMISSION 狀態時執行。提交後，應用程式會進入檢閱程序，且無法再透過此 API 更新。

將資源與利益應用程式建立關聯

合作夥伴可以使用 AssociateBenefitApplicationResource API 動作將受益應用程式連結至相關資源。這會在利益與使用利益的業務內容之間建立寶貴的關聯。

支援的資源類型包含：

- 機會 - 將利益應用程式連結至 中的特定客戶機會。這對於特定機會的好處特別有用，例如 MAP 資金或與客戶互動相關的 POC 點數。
- BENEFIT_ALLOCATION - 連結至現有的利益分配，讓合作夥伴能夠鏈結利益或示範如何利用先前的利益

資源關聯只能在提交之前發生。提交利益應用程式後 （狀態變更為 IN_REVIEW)，即不允許進一步關聯。合作夥伴最多可將 10 個資源與每個利益應用程式建立關聯。

若要取消資源的關聯，合作夥伴會使用 DisassociateBenefitApplicationResource API 動作。與關聯類似，取消關聯只能以 PENDING_SUBMISSION 狀態發生。

> **Important**
>
> 合作夥伴必須擁有適當的許可，才能存取利益應用程式並讀取相關聯的特定資源。API 會在關聯操作期間驗證這些許可。

提交利益應用程式

當利益應用程式完成並準備好AWS檢閱時，合作夥伴會使用 SubmitBenefitApplication API 動作提交它。提交會觸發從 PENDING_SUBMISSION 到 的狀態轉換，IN_REVIEW並啟動利益特定的核准工作流程。

提交時，API 會執行全面的驗證，包括：

- 必要欄位驗證 - 確保利益應用程式詳細資訊中的所有必要欄位都存在
- 欄位格式驗證 - 驗證欄位值是否符合預期的模式和資料類型
- 業務規則驗證 - 套用利益特定的業務邏輯和限制條件
- 資源驗證 - 確認所有相關資源都有效且可存取
- 檔案驗證 - 驗證所有檔案附件是否已成功完成處理

如果驗證失敗，API 會傳回 ValidationException，其中包含詳細的錯誤代碼和訊息，指出哪些欄位需要更正。合作夥伴必須使用 解決這些問題，UpdateBenefitApplication然後再嘗試再次提交。

檔案處理需求：如果任何連接的檔案仍處於 PENDING 狀態，則無法提交利益應用程式。合作夥伴必須等待檔案處理完成（狀態變更為 SUCCEEDED)，才能提交。如果檔案處理失敗（狀態 FAILED)，合作夥伴應該上傳更正的版本。

成功提交後：

- 應用程式狀態會變更為 IN_REVIEW
- 利益擁有者的團隊會收到新提交的通知
- 合作夥伴無法再透過標準更新 APIs修改應用程式詳細資訊
- 應用程式會進入定義的核准工作流程，其中可能包括業務核准、技術核准和財務核准階段

合作夥伴可以透過擷取應用程式並監控階段欄位來追蹤提交進度，該欄位指出目前的核准階段（例如「商業核准」、「技術核准」、「財務核准」)。

管理提交的應用程式

提交後，合作夥伴在管理利益應用程式方面的選項有限，但 API 為常見案例提供特定動作。

回收利益應用程式

如果合作夥伴在提交後發現錯誤或需要進行變更，他們可以使用 RecallBenefitApplication API 動作叫用應用程式。Recall 會將應用程式轉換回 PENDING_SUBMISSION 狀態，允許更新和重新提交。

召回應用程式時，合作夥伴應該提供：

- 識別符 - 要回收之利益應用程式的 ID 或 ARN

- 原因 - 回收的選用說明 （最多 1000 個字元）

原因欄位可實現更好的可追蹤性，並有助於AWS了解導致召回的常見問題，以通知未來改善利益應用程式程序。

取回後，合作夥伴可以使用 UpdateBenefitApplication進行必要的變更，然後在準備就緒SubmitBenefitApplication時使用 重新提交。

> **Important**
>
> 召回通常僅在早期審核階段可用。已進入稍後核准階段或已核准的應用程式可能不符合召回資格。

修改利益應用程式

對於提交後的次要更正，合作夥伴可以使用 AmendBenefitApplication API 動作來更新特定欄位，而不會叫用整個應用程式。當AWS檢閱者在檢閱過程中請求釐清或更正時，此功能特別有用。

修訂使用合作夥伴指定的 JSON 修補程式樣式方法：

- 路徑 - 識別要更新之欄位的 JSONPath 表達式 （例如 $.CreditDisbursementDetails.AwsAccountIdForCredits)
- 值 - 欄位的新值
- 操作 - 要執行的操作 （目前僅REPLACE支援）

合作夥伴可以在單一 API 呼叫中提交最多 10 個修訂。每個修訂都必須包含更新的修訂編號，才能進行樂觀鎖定。

修訂應用於次要更正。對於重大變更，合作夥伴應該使用 RecallBenefitApplication 將應用程式傳回至草稿狀態，以進行全面更新。

取消利益應用程式

如果合作夥伴不再需要利益或希望撤銷其應用程式，他們可以使用 CancelBenefitApplication API 動作取消它。取消會將應用程式轉換為CANCELED狀態，永久結束應用程式程序。

取消應用程式時，合作夥伴應提供：

- 識別符 - 要取消之利益應用程式的 ID 或 ARN

- 原因 - 取消的選用說明 （最多 1000 個字元）

一旦取消，應用程式就無法重新啟用。如果合作夥伴稍後決定他們需要利益，他們必須建立新的利益應用程式。

取消的常見原因包括：

- 客戶機會遺失或延遲
- 合作夥伴容量限制已變更
- 業務優先順序轉移
- 預期用途不再需要利益

檢視利益應用程式詳細資訊

合作夥伴可以使用 GetBenefitApplication API 動作擷取有關利益應用程式的完整資訊。這可提供應用程式的完整檢視，包括：

應用程式中繼資料：

- 唯一識別碼 (ID) 和 Amazon Resource Name (ARN)
- 相關聯的利益識別符
- 應用程式名稱和描述
- 目前狀態 (PENDING_SUBMISSION、IN_REVIEW、ACTION_REQUIREDAPPROVED、REJECTED、或 CANCELED)
- 目前處理階段

狀態資訊：

- 狀態原因 - 人類可讀取的目前狀態說明
- 狀態原因代碼 - 表示特定問題或要求的結構化代碼 （例如，「檢查清單未完成」、「專案計畫遺失」、「設計成功遺失」）

應用程式內容：

- 完整利益應用程式詳細資訊 JSON 文件
- 合作夥伴聯絡資訊

- 具有處理狀態的檔案附件
- 關聯的資源 （機會或配置）
- 資源標籤

稽核資訊：

- 建立時間戳記
- 上次修改時間戳記
- 目前的修訂版編號

當應用程式處於 ACTION_REQUIRED 狀態時，狀態原因代碼特別重要。這些代碼針對合作夥伴需要處理的應用程式以繼續，提供具體、可行的指引。

列出利益應用程式

合作夥伴可以使用 ListBenefitApplications API 動作檢視其所有利益應用程式。這會傳回具有強大篩選功能的應用程式摘要分頁清單。

合作夥伴可以依下列條件篩選應用程式：

- 程式 - 檢視特定程式 (MAP、MDF、沙盒、POC 等） 的應用程式
- 履行類型 - 依交付方法篩選 (CREDITS、CASH、 ACCESS等）
- 利益識別符 - 檢視應用程式以取得特定利益
- 狀態 - 依應用程式狀態 (PENDING_SUBMISSION、IN_REVIEW、REJECTED、、CANCELED) APPROVED篩選
- 階段 - 依核准階段篩選 （業務核准、技術核准、財務核准）
- 關聯的資源 ARNs - 尋找連結至特定機會或配置的應用程式

清單回應包含基本摘要資訊，例如：

- 應用程式 ID、ARN 和名稱
- 關聯的利益 ID
- 程式和履行類型
- 目前狀態和階段
- 建立和修改時間戳記

- 相關資源
- 目前的修訂版編號
- 從利益應用程式詳細資訊中選取的欄位 （由利益擁有者定義）

合作夥伴可以使用排序參數設定結果排序，並可選擇依建立日期、狀態、程式或利益識別符以遞增或遞減順序排序。

Dashboard Building：ListBenefitApplicationsAPI 旨在支援合作夥伴儀表板建立。透過篩選和排序應用程式，合作夥伴可以建立檢視，例如：

- 需要動作的應用程式 (ACTION_REQUIRED 狀態）
- 最近提交的應用程式 （依建立日期、IN_REVIEW狀態排序）
- 核准的應用程式待分配 (APPROVED 狀態）
- 依程式的應用程式 （由特定程式篩選）

使用利益分配

利益分配代表已授予合作夥伴的利益。當利益應用程式獲得核准時， 會AWS建立一或多個利益分配，為合作夥伴提供實際的點數、資金、存取權或其他利益履行。

了解利益分配

利益分配可以代表各種類型的履行：

- 財務支出 (CASH) - 使用支出追蹤將財務付款直接傳送給合作夥伴
- AWS抵用金 (CREDITS) - 適用於使用量追蹤AWS帳戶的抵用金代碼
- 消耗分配 - 使用率追蹤的預先核准資金金額或錢包
- Access Grants (ACCESS) - 存取系統、工具或資源
- 辨識 （辨識） - 數位徽章、認證或狀態指定
- 資源 (RESOURCE) - 技術支援、諮詢服務或 ProServe 業務

每個利益分配都有透過其狀態欄位追蹤的生命週期：

- ACTIVE - 配置可供使用
- 非作用中 - 配置暫時無法使用
- FULFILLED - 配置已完全使用或交付

利益分配也包括暫時性資訊，定義它們何時生效 (StartsAt) 和何時過期 (ExpiresAt)，讓合作夥伴了解利用利益的時間範圍。

列出利益分配

合作夥伴可以使用 ListBenefitAllocations API 動作檢視其所有利益配置。這會傳回具有完整篩選功能的配置摘要分頁清單。

合作夥伴可以依下列條件篩選配置：

- 利益識別符 - 檢視特定利益的配置
- 利益應用程式識別符 - 檢視特定應用程式產生的配置
- 履行類型 - 依配置類型 (CREDITS、ACCESS、 等） CASH篩選
- 狀態 - 依配置狀態篩選 (ACTIVE、INACTIVE、FULFILLED)

清單回應包含配置摘要，其中包含：

- 配置 ID、ARN 和名稱
- 關聯的利益 ID 和利益應用程式 ID （ 如適用 ）
- 履行類型
- 目前狀態
- 建立時間戳記
- 開始日期

此清單檢視可讓合作夥伴建置配置追蹤儀表板，並識別：

- 可用於立即使用的作用中配置
- 即將過期的配置
- 歷史追蹤的完成配置
- 依程式或履行類型的分配總值

檢視詳細配置資訊

合作夥伴可以使用 GetBenefitAllocation API 動作擷取完整的配置詳細資訊。這可提供配置的完整資訊，包括根據配置類型而有所不同的履行特定詳細資訊。

核心配置資訊：

- 唯一識別碼 (ID) 和 Amazon Resource Name (ARN)
- 配置名稱和狀態
- 狀態原因 （目前狀態的說明）
- 關聯的利益 ID 和利益應用程式 ID
- 履行類型
- 建立、更新、啟動和過期時間戳記

履行詳細資訊 （類型特定）：

FulfillmentDetail 欄位根據履行類型包含不同的資訊結構：

現金支出詳細資訊 （現金類型）

對於直接現金付款，履行詳細資訊包括：

- 已支付金額 - 已支付資金的總金額，包括金額值和貨幣代碼
- 發行詳細資訊 （如適用） - 有關採購訂單或發行事件的資訊：
  - 發行 ID - 支出的唯一識別符
  - 發行金額 - 以本地貨幣為單位的金額
  - 發出時間 - 分配時間戳記

此結構支援單一支出和預先核准的資金案例，其中單一配置可能會發生多個發行。

點數詳細資訊 (CREDITS 類型）

對於AWS點數，履行詳細資訊包括：

- 已分配金額 - 已分配的點數總計
- 已發行金額 - 已發行的點數總額 （可能低於分階段發行的配置）
- 點數代碼 - 個別點數代碼詳細資訊的陣列：
  - AWS帳戶 ID - 套用點數的帳戶
  - 值 - 點數碼的貨幣值
  - AWS點數代碼 - 實際點數代碼字串
  - 狀態 - 點數碼狀態 (ACTIVE、INACTIVE、FULFILLED)

- 發出時間 - 發出點數碼時
- 過期時間 - 當點數碼過期時

此詳細的點數追蹤可讓合作夥伴監控多個帳戶的點數使用率，並了解哪些點數可用、部分使用或完全使用。

消耗性配置詳細資訊（預先核准的金額/貨盤）

對於預先核准的資金集區，履行詳細資訊包括：

- 配置金額 - 配置中的總金額
- 剩餘金額 - 未使用的金額仍然可用
- 使用量 - 已使用量
- 發行詳細資訊（如適用）- 配置的採購訂單資訊

消耗性配置可讓合作夥伴視需要為合格活動提取資金，並即時追蹤餘額。

存取詳細資訊 (ACCESS 類型)

對於存取授權，履行詳細資訊包括：

- 描述 - 有關正在授予的存取權及其使用方式的詳細說明

存取配置可能會提供預覽服務、專用工具或受限資源的存取權。描述欄位提供合作夥伴如何啟用和使用授予的存取權的指示。

將配置套用至新的利益應用程式

利益分配中的 ApplicableBenefitIds 欄位可識別可利用此配置的其他優點。這可讓 合作夥伴使用現有配置來申請其他利益，進而實現利益鏈結。

例如，合作夥伴可能會：

1. 接收預先核准的 MAP 資金分配
2. 為個別客戶遷移專案建立利益應用程式
3. 將這些應用程式與預先核准的配置建立關聯
4. 當專案獲得核准時，從配置中提取資金

此方法可簡化具有預先核准資金之合作夥伴的利益申請程序，減少個別專案請求的審查週期。

使用AWS Partner Central Channel API

AWS合作夥伴中央管道 APIs 可讓您以程式設計方式管理授權AWS的轉售業務。這些 APIs AWS解決方案供應商、AWS經銷商和AWS分發賣方：

- 報告用於向 Partner Central 轉售程式的AWS管理帳戶
- 傳送邀請至AWS帳戶以接受合作夥伴關聯
- 建立加入轉售最終客戶AWS帳戶和套用合作夥伴折扣所需的關係

使用AWS Partner Central Channel APIs，您可以建置自訂自動化以：

- 將新的合作夥伴擁有AWS的帳戶加入 APN 轉售計畫
- 加入新的最終客戶帳戶以轉售計劃
- 加速客戶加入和折扣資格

使用程式管理帳戶 (PMAs)

計劃管理帳戶 (PMA) 會將您的AWS管理帳戶與頻道計劃註冊建立關聯。程式管理帳戶是使用頻道交握來啟用自助服務。當您建立和啟用 PMA 時，管道計劃利益和折扣會套用至交付至相關聯AWS管理帳戶的所有發票。使用這些 APIs 來報告和管理處理轉售和管道折扣AWS的帳戶：

- `CreateProgramManagementAccount`：新增帳戶以進行客戶帳單管理
- `UpdateProgramManagementAccount`：修改現有帳戶
- `DeleteProgramManagementAccount`：從轉售計劃中移除帳戶
- `ListProgramManagementAccounts`：檢視現有帳戶及其狀態

使用頻道交握

頻道交握可為關鍵頻道管理操作提供安全、相互的同意。AWS Partner Central 使用頻道交握來驗證AWS管理帳戶對 的同意有三個不同的目的：啟用計劃管理帳戶 (PMAs)、建立服務期間，以及提早終止服務期間。共有三種類型：

- `PROGRAM_MANAGEMENT_ACCOUNT`：在啟用 PMAs 以套用管道計劃利益時，程式管理帳戶交握會驗證同意

- START_SERVICE_PERIOD：服務期間建立交握會建立帳單轉移承諾的相互協議，其中包含最短通知期間或固定期限
- REVOKE_SERVICE_PERIOD：服務期間終止交握可在雙方同意時提早終止作用中的服務期間。所有交握類型都需要目標AWS管理帳戶接受才能生效。

使用這些 APIs來管理 PMA 啟用和服務期間管理的頻道交握：

- CreateChannelHandshake：建立 PMA 邀請、建立服務期間或服務期間終止的交握
- AcceptChannelHandshake：接受來自目標AWS帳戶的交握 （ 可由合作夥伴或客戶根據交握類型使用 ）
- RejectChannelHandshake：拒絕來自目標AWS帳戶的交握
- CancelChannelHandshake：在接受、拒絕或過期待處理交握之前取消交握
- ListChannelHandshakes：追蹤和檢視依類型和狀態篩選的交握

使用頻道關係

關係可讓您管理最終客戶、內部組織或下游賣方。每個關係包括：

- 關聯AWS組織的AWS管理帳戶
- 頻道折扣資格所需的AWS合作夥伴網路中繼資料

當您建立關係時，關聯AWS組織的所有使用都會獲得相關的頻道計劃利益。使用這些 APIs來報告賣方和最終客戶：

- CreateRelationship：加入新的終端客戶、經銷商或內部AWS帳戶
- GetRelationship：檢視特定關係詳細資訊
- ListRelationships：檢視所有關係
- UpdateRelationship：修改現有關係
- DeleteRelationship：移除關係

使用營收測量 API

主題

- 使用營收歸因 IDs

- 使用 Marketplace 收入共享

使用營收歸因 IDs

收入歸因 ID 是交易層級識別符，可將您的AWS Marketplace 產品收入映射至特定的AWS Marketplace 優惠和/或AWS合作夥伴中心機會。它以產品層級的合作夥伴收入衡量 (PRM) 實作為基礎：PRMS 會擷取 Marketplace 產品的總歸屬收入，而收入歸屬 ID 會根據您指定的每月成本分配百分比，將收入映射到特定交易。

Revenue Attribution Service 公開 APIs 以建立、擷取、列出和更新營收屬性 IDs，並以非同步方式管理其每月成本分配項目。

什麼是營收歸因 ID？

營收歸因 ID 有兩個層級：

- 由ra-字首 ID （例如，ra-aabbccddee001) 和 ARN 識別的合作夥伴命名、合作夥伴描述資源的屬性。每個屬性的範圍都限定為 Catalog(AWS 用於生產、Sandbox測試） 和合作夥伴的 帳戶。
- 每個項目的一或多個每月成本分配項目會將單一 （優惠 ID 或機會 ID、帳單月） 組合映射至成本分配百分比。項目會透過配置任務模式以非同步方式批次管理。

營收歸因 ID 有兩種使用方式：

- 作為現有產品層級 PRM 實作上的交易層級浮水印。建立營收歸因 ID、將適用的 Marketplace 優惠和/或 ACE 機會建立關聯，並將您已衡量的產品營收AWS對應至這些特定交易，無需變更現有的資源標籤或使用者代理程式字串。
- 作為獨立識別符，直接用作資源標籤值 (aws-apn-id = <RA ID>) 或在使用者代理程式字串 (APN_1.1/pc_<RA ID>$) 中，從一開始就將AWS耗用量歸因於特定交易。

使用營收歸因 IDs

合作夥伴可以透過營收歸因服務管理營收歸因 IDs 及其每月成本分配項目。生命週期會進行兩個獨立的流程：屬性記錄流程 （建立、更新、擷取、列出） 和配置流程 （啟動批次任務、輪詢結果、擷取個別項目、列出屬性的項目）。

建立營收屬性 ID

第一步是使用 CreateRevenueAttribution API 動作建立營收歸因 ID。傳回的 Id和 Arn 可立即用作資源標籤值，或在使用者代理程式字串中使用屬性AWS。

建立營收歸因 ID 時，合作夥伴必須提供：

- **Catalog** — AWS Sandbox用於生產或測試。
- **Name** — 屬性的人類可讀取名稱。在目錄和合作夥伴的帳戶中必須是唯一的。最多 128 個字元。

合作夥伴可以選擇性地提供：

- **Description** — 屬性的任意文字描述。最多 1024 個字元。
- **MarketplaceProduct** —AWS要與此屬性建立關聯的 Marketplace 產品。提供 ProductIdentifier(25 個字元的 Marketplace 產品 ID) 和 TenancyModel(MULTI_TENANT 或 SINGLE_TENANT)。如果在建立時省略，則無論客戶購買的 Marketplace 產品為何，歸因適用於營收歸因 ID 下測量的所有消耗，當單一部署跨越多個 Marketplace 清單時很有用。
- **Tags** — 資源組織最多 200 個鍵值對。標籤索引鍵在請求中必須是唯一的。

最佳實務：每當您的交易錨定至特定 Marketplace 清單MarketplaceProduct.ProductIdentifier時提供 。這可讓AWS驗證產品是否由呼叫的合作夥伴帳戶或透過合作夥伴帳戶連線 (PAC) 連線的分公司帳戶所擁有，並在回應中顯示已解決的 SaaSProductCode和 ProductType（例如，AMI、ML)。如果產品不是授權帳戶所擁有，API 會傳回ValidationException原因為 PRODUCT_NOT_FOUND_OR_NOT_OWNED。

回應會傳回新的 Id（格式 ra-[a-z0-9]{13})、Arn、已解析的MarketplaceProduct屬性和初始Version數字 （從 1 開始，每次後續更新時遞增）。

新增每月成本分配項目

建立營收歸因 ID 後，合作夥伴會透過 StartRevenueAttributionAllocationsTask API 動作提交一批每月成本分配項目，將AWS Marketplace 優惠和/或 ACE 機會與其建立關聯。這是一個非同步操作，每個任務最多可接受 250 個配置變更 (CREATE 和/或 UPDATE)。

每個配置項目指定：

- 優惠 ID 或機會 ID — 與交易相關聯的唯一識別符。如果 Marketplace 優惠已連結至 ACE 機會，合作夥伴只需要提供優惠 ID；AWS自動將收入歸納至連結的機會。
- 帳單月份 — 套用成本分配的行事曆月份 （例如 2026-04)。
- 成本分配 % — 此計費月份中可歸因於此交易的產品總AWS耗用量部分。多租戶 SaaS 產品的必要項目，包括客戶共用基礎設施的混合部署合作夥伴託管元件。
- 客戶AWS帳戶 ID — 取用產品的客戶AWS帳戶。多租用戶 SaaS 產品需要。

指定營收歸因 ID 和相同帳單月份之所有配置項目的總成本分配 % 不得超過 100%。如果總計超過 100%，違規項目會在非同步業務驗證期間，以每筆記錄錯誤碼拒絕。

同步驗證： 會同步StartRevenueAttributionAllocationsTask執行基本形狀驗證 （欄位類型、模式、批次大小）。如果基本驗證通過，API 會傳回狀態TaskId為 的 IN_PROGRESS。業務驗證（上限檢查、不可變性規則、針對 Marketplace 和 ACE 的相依性查詢） 會以非同步方式執行，並透過探索每個記錄的結果GetRevenueAttributionAllocationsTask。

若要監控提交的任務，合作夥伴GetRevenueAttributionAllocationsTask會使用營收歸因資源 ID 或 ARN 輪詢。回應會從 進展IN_PROGRESS至 COMPLETED（無論個別記錄成功或失敗） 並傳回：

- **TaskStatus** — IN_PROGRESS、 COMPLETED或 FAILED（最上層任務失敗）。
- **RecordResults** — 針對每個輸入記錄：指派的 AllocationId（如果成功）、每個記錄的狀態 (SUCCEEDED 或 FAILED)，以及記錄失敗時的結構化錯誤代碼和訊息。

合作夥伴應該立即擷取任務結果。任務完成後，每個記錄的結果可以進行對帳，並且可以在新任務中更正和重新提交任何失敗的項目。

編輯每月成本分配項目

成本分配項目會依計費月份管理，規則如下：

- 合作夥伴可以隨時為目前或任何未來計費月份新增每月項目。
- 合作夥伴可以隨時更新目前或未來任何計費月份的每月項目。
- 合作夥伴可以更新上個月的項目，直到當月 7 日為止。此時段允許合作夥伴在AWS Cost Explorer 中檢閱實際用量，然後再完成上個月的配置。

在當月 7 日之後，無法再修改上一個計費月份的屬性。更新目前或未來月份的項目會從下一個每月帳單週期開始套用。前幾個月的歷史屬性不會追溯重新計算。

配置項目的更新會透過相同的 StartRevenueAttributionAllocationsTask API 動作提交，UPDATE並針對每個受影響的項目將 Operation 設定為 。非同步任務模式會統一處理更新和建立。

更新營收歸因 ID

合作夥伴可以使用 UpdateRevenueAttribution API 動作更新Description現有營收歸因 ID 的 。屬性記錄本身不可變 — Name、 MarketplaceProduct和tag-on-create值無法在建立後變更。若要重新標記資源，請在屬性的 ARN 上使用標準AWS標記操作。

更新營收歸因 ID 時，合作夥伴必須提供：

- **Catalog** — AWS或 Sandbox。
- **Identifier** — 要更新的屬性 ra- ID。
- **Version** — 目前版本的屬性，用於樂觀鎖定。

合作夥伴可以選擇性地提供：

- **Description** — 更新的任意文字描述。最多 1024 個字元。
- **ClientToken** — 用於更新的等冪符記。

樂觀鎖定： Version 欄位可確保只有在屬性自上次擷取以來未變更時，才會套用更新。如果提交的 Version 不符合目前的資源版本，則會使用 拒絕更新，ConflictException並顯示包含提交和目前版本編號的訊息。最佳實務是在每次更新GetRevenueAttribution之前，使用 擷取最新版本。

檢視營收歸因 ID 詳細資訊

合作夥伴可以使用 GetRevenueAttribution API 動作擷取單一營收歸因 ID 的完整資訊。將傳回 。

資源中繼資料：

- 唯一識別碼 (Id) 和 Amazon Resource Name (Arn)。
- 屬性Catalog所在的 。
- Name 與 Description。
- 擷取的 Version和 LatestVersion (使用最新的 進行後續更新)。

Marketplace 產品屬性 （ 如果MarketplaceProduct在建立時設定 ）：

- ProductId — 合作夥伴提供的 Marketplace 產品 ID。
- ProductCode — 從 解析的AWS Marketplace 產品程式碼ProductId。
- ProductType — SaaS、AMI、Container、Data、 ML或 Professional Services。
- TenancyModel — MULTI_TENANT或 SINGLE_TENANT。

稽核資訊：

- CreatedDate 與 LastModifiedDate。

合作夥伴可以選擇在請求Version中提供特定 ，以擷取屬性的歷史版本。省略 Version 以傳回最新的 。

若要擷取特定的每月成本分配項目，合作夥伴會使用 GetRevenueAttributionAllocation API 動作搭配項目的 AllocationId。這會傳回優惠 ID 或機會 ID、帳單月份、成本分配 %、客戶AWS帳戶 ID，以及項目的狀態和稽核資訊。

列出營收歸因 IDs

合作夥伴可以使用 ListRevenueAttributions API 動作檢視其帳戶中的所有營收歸因 IDs。這會傳回具有篩選和排序功能的屬性摘要分頁清單。

合作夥伴可以依下列條件篩選結果：

- **Catalog** — AWS或 Sandbox（必要）。
- **Identifiers** — 最多可擷取 100 個特定 ra- ID 的清單。 IDs
- **CreatedAfter** / **CreatedBefore** — 依建立時間戳記範圍 （包含） 篩選。

合作夥伴可以使用 Sort 參數設定結果排序：

- SortBy — CreatedDate或 LastModifiedDate。
- SortOrder — ASCENDING或 DESCENDING。

回應包含每個屬性的摘要：Arn、Id、Catalog、Name、已解析MarketplaceProduct屬性、CreatedDate、LastModifiedDate、最新 Version和 TotalRevenueAttributionAssociationCount（連結至屬性的作用中每月成本分配項目數量）。

使用 MaxResults（預設值 25，上限為 100) 和 NextToken來分頁大型結果集。NextToken 如果有其他頁面可用，回應會包含 。

列出每月成本分配項目

合作夥伴可以使用 ListRevenueAttributionAllocations API 動作列出特定營收歸因 ID 的每月成本分配項目。這會傳回具有篩選功能的配置摘要分頁清單。

合作夥伴可以依下列條件篩選結果：

- **MarketplaceOfferId** — 僅列出與特定 Marketplace 優惠相關聯的項目。

- **AwsPartnerCentralOpportunityId** — 僅列出與特定 Partner Central Opportunity 相關聯的項目。
- **BillingMonth** — 僅列出特定日曆月的項目。

回應包含每個項目的摘要資訊：AllocationId、相關聯的優惠 ID 或機會 ID、客戶AWS帳戶 ID、帳單月份、成本分配 %、項目的名稱和狀態，以及稽核時間戳記。

儀表板建置： ListRevenueAttributions和 的組合ListRevenueAttributionAllocations旨在支援合作夥伴儀表板的建立。合作夥伴可以建立檢視，例如：

- 過去 30 天內建立的所有營收歸因 IDs，依建立日期排序。
- 跨多個營收歸因 IDs 的特定 Marketplace 優惠的所有每月成本分配項目。
- 目前計費月份的所有成本分配項目，以驗證每個屬性總計不超過 100%。
- 至少有一個作用中配置項目 () 的所有營收歸因 IDsTotalRevenueAttributionAssociationCount > 0。

使用 Marketplace 收入共享

Marketplace 收入共享是AWS Marketplace 產品的輸入，它會宣告透過AWS Marketplace 收集的產品總收入部分。AWS使用此輸入將合作夥伴的 Marketplace 收集收入與其他收入訊號相互關聯。

Marketplace Revenue Share Service 公開 APIs 來建立、擷取和列出 Marketplace Revenue Share 資源，以及管理其配置（合作夥伴宣告的收入共享百分比和生效日期時段）。Marketplace Revenue Share 資源支援標準AWS標記操作。

什麼是 Marketplace 收入共享？

Marketplace 收入共享有兩個層級：

- 共用 — 單一AWS Marketplace 產品的輸入，由 Marketplace 識別ProductId。每個產品只有一個 Marketplace 收入共享。
- 一或多個配置 – 每個配置都會宣告 RevenueSharePercent和有效日期時段 (EffectiveDate，選擇性為 ExpirationDate)。共享可以隨著時間的推移有許多配置，但在任何時候都只能套用一個ACTIVE配置。

配置具有下列關鍵屬性：

- 狀態 — ACTIVE或 INACTIVE。建立時，狀態預設為 ACTIVE。只有ACTIVE配置參與收入計算和重疊檢查。
- 重疊不可變 - 在單一共享中，沒有兩個ACTIVE配置可能有重疊[EffectiveDate, ExpirationDate]的範圍。隨時允許每個共享最多一個開放式ACTIVE配置 （不允許 ExpirationDate)。

使用 Marketplace 收入共享

合作夥伴會透過 Marketplace Revenue Share Service 管理 Marketplace Revenue Shares 及其配置。生命週期會進行兩個流程：共享流程 （建立、擷取、列出、標籤） 和配置流程 （建立、更新、軟刪除、擷取、列出）。

建立 Marketplace 收入共享

第一步是使用 CreateMarketplaceRevenueShare API 動作為 Marketplace 產品建立AWS Marketplace 收入共享。共享由 Marketplace 識別ProductId，並做為所有後續配置的容器。

建立 Marketplace Revenue Share 時，合作夥伴必須提供：

- **Catalog** — AWS Sandbox用於生產或測試。
- **ProductId** — 25 個字元的 Marketplace 產品 ID。模式：[a-z0-9]{25}。

合作夥伴可以選擇性地提供：

- **Tags** — 建立時最多 50 個資源組織的鍵值對。使用 TagResource 在建立後新增標籤。
- **ClientToken** — 等冪字符，可防止在重試時重複建立。最多 64 個字元。SDKs 會自動產生此字符。

產品驗證：建立時，服務會向AWS Marketplace 驗證產品是否由發起人帳戶或透過合作夥伴帳戶連線 (PAC) 連接到發起人的子公司帳戶所擁有。如果產品不存在或擁有的帳戶未獲授權給發起人，API 會傳回ValidationException原因為 PRODUCT_NOT_FOUND_OR_NOT_OWNED。

每個產品一個共用：相同 ProductId中相同 的第二個CreateMarketplaceRevenueShare呼叫會Catalog傳回 ConflictException。

回應會傳回共享的 Arn （格式 arn:{partition}:partnercentral:{region}:{account-id}:catalog/{catalog}/marketplace-revenue-share/

{productId})Catalog、ProductId、、初始Version數字（從 1 開始，並在每個配置變動上遞增），以及在建立時套用的任何標籤。

新增配置

建立 Marketplace 收入共享後，合作夥伴會透過 CreateMarketplaceRevenueShareAllocation API 動作建立配置，宣告收入共享百分比和生效日期時段。

建立配置時，合作夥伴必須提供：

- **Catalog** — AWS或 Sandbox。
- **MarketplaceRevenueShareIdentifier** — 父共享的識別符。
- **RevenueSharePercent** — 透過 Marketplace 以十進位形式（例如 45%) 0.45 提供的產品收入百分比。
- **EffectiveDate** — 此配置生效的行事曆日期。格式：YYYY-MM-DD。

回應會傳回配置的 Id（格式 mrsa-[A-Za-z0-9]{13})、配置狀態 (ACTIVE 建立時） 和父共享的凹凸 Version（每個配置變動都會凹凸，用於後續配置更新的樂觀鎖定）。

更新配置

合作夥伴可以使用 UpdateMarketplaceRevenueShareAllocation API 動作更新配置。可更新的欄位取決於配置是未來日期、目前有效還是過去日期。

更新配置時，合作夥伴必須提供：

- **Catalog** — AWS或 Sandbox。
- **MarketplaceRevenueShareIdentifier** — 父共享的識別符。
- **AllocationId** — 要更新的配置 mrsa- ID。
- **MarketplaceRevenueShareVersion** — 配置上樂觀鎖定的父共用的目前版本。碰撞衝突會傳回 ConflictException。

合作夥伴可以選擇性地提供：

- **RevenueSharePercent** — 已更新的收入共享百分比。只能對未來日期的配置進行編輯。
- **EffectiveDate** — 更新的生效日期。只能對未來日期的配置進行編輯。
- **ExpirationDate** — 更新的過期日期。可在未來日期的配置（任何值 > EffectiveDate) 和目前有效或開放式配置（任何值）上編輯≥ today。

- **Status** — ACTIVE或 INACTIVE。設定為 INACTIVE 軟刪除配置。只能對未來日期的配置進行編輯。
- **ClientToken** — 等冪符記。

檢視 Marketplace 收入共享詳細資訊

合作夥伴可以使用 GetMarketplaceRevenueShare API 動作擷取單一 Marketplace Revenue Share 的完整資訊。這會傳回共用的 Arn、Catalog、ProductId、 Version (父共用的目前版本) LastModifiedDate、、 CreatedDate和套用至共用的任何標籤。

合作夥伴可以使用 GetMarketplaceRevenueShareAllocation API 動作搭配配置的 來擷取單一配置Id。將傳回 。

- 配置的 Id和父共享的 MarketplaceRevenueShareArn。
- RevenueSharePercent, EffectiveDate, ExpirationDate.
- Status (ACTIVE 或 INACTIVE)。
- 父共享在後續更新上MarketplaceRevenueShareVersion樂觀鎖定的 。
- CreatedDate 與 LastModifiedDate。

列出 Marketplace 收入共享

合作夥伴可以使用 ListMarketplaceRevenueShares API 動作檢視其帳戶擁有的所有 Marketplace 營收共享。這會傳回共享摘要的分頁清單。

合作夥伴可以依下列條件篩選結果：

- **Catalog** — AWS或 Sandbox (必要) 。
- **ProductIds** — 要擷取的特定 Marketplace IDs 清單。

回應包含每個共享的摘要：Arn、Catalog、ProductId、最新 Version、 CreatedDate和 LastModifiedDate。

使用 MaxResults和 NextToken來分頁大型結果集。

列出配置

合作夥伴可以使用 ListMarketplaceRevenueShareAllocations API 動作列出特定 Marketplace Revenue Share 的配置。這會傳回具有篩選功能的配置摘要分頁清單。

合作夥伴可以依下列條件篩選結果：

- **Status** — ACTIVE或 INACTIVE。
- 有效日期範圍 — EffectiveDateOnOrAfter / EffectiveDateOnOrBefore 可擷取有效日期落在特定時段內的配置。

回應包含每個配置的摘要資訊：Id、RevenueSharePercent、EffectiveDate、MarketplaceRevenueShareVersion、、ExpirationDate Status和稽核時間戳記。

標記 Marketplace 營收份額

Marketplace Revenue Share 資源支援標準AWS標記操作：

- **TagResource** — 在 Marketplace Revenue Share 上新增或覆寫標籤。每個共用最多 50 個標籤。
- **UntagResource** — 依金鑰移除特定標籤。
- **ListTagsForResource** — 擷取目前套用至 Marketplace 收入共享的所有標籤。

配置不支援標籤。標記父共用以組織相關配置。

每個標記操作都會接受共享的 Arn (格式 arn:{partition}:partnercentral:{region}:{account-id}:catalog/{catalog}/marketplace-revenue-share/{productId}) 作為資源識別符。也支援透過 上的 Tags 欄位建立 Standard AWS tag-on-createCreateMarketplaceRevenueShare。

支援 AWS 的區域

下列各節提供有關支援的 AWS 區域的資訊。

主題

- Partner AWS Central Account API 支援的 AWS 區域
- AWS Partner Central Selling API 支援的 AWS 區域
- Partner AWS Central Benefits API 支援的 AWS 區域
- Partner AWS Central Channel API 支援的 AWS 區域
- Partner AWS Central Revenue Measurement API 支援 AWS 的區域

Partner AWS Central Account API 支援的 AWS 區域

您可以使用下列端點，從任何 AWS 美國東部 （維吉尼亞北部） 區域存取 AWS Partner Central 帳戶服務。

```
partnercentral-account.us-east-1.api.aws
```

AWS Partner Central Selling API 支援的 AWS 區域

您可以使用下列端點，從任何 AWS 美國東部 （維吉尼亞北部） 區域存取 AWS Partner Central 銷售服務。

```
partnercentral-selling.us-east-1.api.aws
```

Partner AWS Central Benefits API 支援的 AWS 區域

您可以使用下列端點，從任何 AWS 美國東部 （維吉尼亞北部） 區域存取 AWS Partner Central 利益服務。

```
partnercentral-benefits.us-east-1.api.aws
```

Partner AWS Central Channel API 支援的 AWS 區域

您可以使用下列端點，從任何 AWS 商業區域存取 AWS Partner Central 頻道管理服務。

```
partnercentral-channel.global.api.aws
```

使用 SigV4a 簽署請求

由於 AWS Partner Central Channel API 使用全域端點，請求會使用 Signature 第 4a 版 (SigV4a) 簽署，SigV4 的延伸支援使用非對稱密碼編譯簽署，從而啟用跨多個 AWS 區域驗證的簽章。

使用 AWS SDK 或 CLI

如果您使用 AWS SDK 或 AWS CLI，當您呼叫全域端點時，會自動處理 SigV4a 簽署。不需任何其他設定。

手動簽署請求

如果您在沒有 SDK 的情況下手動簽署請求，請注意與標準 SigV4 的下列差異：

- SigV4a 會從您現有的 AWS 私密存取金鑰衍生橢圓曲線數位簽章演算法 (ECDSA) 金鑰對，而不是使用 HMAC 型金鑰衍生。
- 登入資料範圍會*針對區域元件使用 ，而非特定區域名稱，因為簽章跨區域有效。

如需 SigV4a 如何運作的詳細資訊，請參閱適用於 API 請求的 AWS Signature 第 4 版。如需 SigV4a 簽署的程式碼範例，請參閱 GitHub 上的 sigv4a-signing-examples 專案。

Partner AWS Central Revenue Measurement API 支援 AWS 的區域

您可以使用下列端點，從任何 AWS 商業區域存取 AWS Partner Central Revenue Measurement 服務。

```
partnercentral-prm.global.api.aws
```

使用 SigV4a 簽署請求

由於營收衡量服務使用全域端點，請求會使用 Signature 第 4a 版 (SigV4a) 簽署，SigV4 的延伸支援使用非對稱密碼編譯簽署，從而啟用跨多個 AWS 區域驗證的簽章。

使用 AWS SDK 或 CLI

如果您使用 AWS SDK 或 AWS CLI，當您呼叫全域端點時，會自動處理 SigV4a 簽署。不需任何其他設定。

手動簽署請求

如果您在沒有 SDK 的情況下手動簽署請求，請注意與標準 SigV4 的下列差異：

- SigV4a 會從您現有的 AWS 私密存取金鑰衍生橢圓曲線數位簽章演算法 (ECDSA) 金鑰對，而不是使用 HMAC 型金鑰衍生。
- 登入資料範圍會*針對區域元件使用 ，而非特定區域名稱，因為簽章跨區域有效。

如需 SigV4a 如何運作的詳細資訊，請參閱適用於 API 請求的 AWS Signature 第 4 版。如需 SigV4a 簽署的程式碼範例，請參閱 GitHub 上的 sigv4a-signing-examples 專案。

許可

下列各節提供許可的相關資訊。

主題

- 帳戶 API 的存取權
- 銷售 API 的存取權
- 利益 API 的存取權
- 通道 API 的存取
- 營收測量 API 的存取權

帳戶 API 的存取權

存取控制和許可由AWS Identity and Access Management(IAM) 管理。本節提供設定與帳戶 API 互動必要許可的指引。

先決條件

設定許可之前，請確定您的AWS 帳戶已連結至 ，且您已建立必要的 IAM 角色和使用者。如需詳細資訊，請參閱設定和身分驗證。

使用AWS受管政策

AWS提供受管政策，授予與帳戶 API 互動所需的許可。若要提供管理帳戶資源的必要存取權，請將`AWSPartnerCentralFullAccess`政策連接至您的 IAM 身分。如需詳細資訊，請參閱 AWS使用者的 受管政策。

將政策指派給 IAM 角色和使用者

請依照下列步驟將政策指派給 IAM 角色和使用者：

1. 登入AWS 管理主控台。
2. 導覽至 IAM 服務。
3. 選取角色或使用者，然後選擇您要連接政策的 IAM 角色或使用者。
4. 將`AWSPartnerCentralFullAccess`政策連接至選取的 IAM 角色或使用者。

如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

使用條件索引鍵管理許可

IAM 政策中的條件索引鍵可為何時強制執行陳述式政策提供資源層級許可。您可以使用條件金鑰來指定允許或拒絕特定許可的條件。

如需詳細資訊，請參閱 IAM JSON 政策元素：Condition 運算子。

條件索引鍵概觀

條件金鑰	說明	適用的動作	有效值
partnercentral：Catalog	依相關聯的目錄實體類型篩選存取權	所有動作	AWS、沙盒

必要許可的摘要

必要許可的摘要

Action	說明
partnercentral：AcceptConnectionInvitation	允許接受連線邀請
partnercentral：AssociateAwsTrainingCertificationEmailDomain	允許關聯AWS訓練認證電子郵件網域
partnercentral：CancelConnection	允許取消連線
partnercentral：CancelConnectionInvitation	允許取消連線邀請
partnercentral：CancelProfileUpdateTask	允許取消設定檔更新任務
partnercentral：CreateConnectionInvitation	允許建立連線邀請
partnercentral：CreatePartner	允許建立合作夥伴
partnercentral：DisassociateAwsTrainingCertificationEmailDomain	允許取消AWS訓練認證電子郵件網域的關聯
partnercentral：GetAllianceLeadContact	允許擷取聯盟潛在客戶聯絡詳細資訊
partnercentral：GetConnection	允許擷取連線詳細資訊
partnercentral：GetConnectionInvitation	允許擷取連線邀請詳細資訊
partnercentral：GetConnectionPreferences	允許擷取連線偏好設定

Action	說明
partnercentral：GetPartner	允許擷取合作夥伴詳細資訊
partnercentral：GetProfileUpdateTask	允許擷取設定檔更新任務詳細資訊
partnercentral：GetProfileVisibility	允許擷取設定檔可見性設定
partnercentral：GetVerification	允許擷取驗證詳細資訊
partnercentral：ListConnectionInvitations	允許列出連線邀請
partnercentral：ListConnections	允許列出連線
partnercentral：ListPartners	允許列出合作夥伴
partnercentral：PutAllianceLeadContact	允許更新聯盟潛在客戶聯絡詳細資訊
partnercentral：PutProfileVisibility	允許更新設定檔可見性設定
partnercentral：RejectConnectionInvitation	允許拒絕連線邀請
partnercentral：SendEmailVerificationCode	允許傳送電子郵件驗證碼
partnercentral：StartProfileUpdateTask	允許開始設定檔更新任務
partnercentral：StartVerification	允許開始驗證程序
partnercentral：UpdateConnectionPreferences	允許更新連線偏好設定

銷售 API 的存取權

存取控制和許可由AWS Identity and Access Management(IAM) 管理。本節提供設定與 API 互動必要許可的指引，包括列出AWS Marketplace實體所需的許可。

先決條件

在設定許可之前，請確定您的AWS 帳戶已連結至 Partner Central，而且您已建立必要的 IAM 角色和使用者。如需詳細資訊，請參閱設定和身分驗證。

使用AWS受管政策

AWS提供受管政策，授予與 API 互動所需的許可。若要提供管理機會的必要存取權，請將`AWSPartnerCentralOpportunityManagement`政策連接至您的 IAM 身分。如需詳細資訊，請參閱 AWSAWS Partner Central 使用者的 受管政策。

AWSPartnerCentralOpportunityManagement 政策

此政策授予 Partner Central 機會管理動作的完整存取權。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "OpportunityManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:AcceptEngagementInvitation",
                "partnercentral:AssignOpportunity",
                "partnercentral:AssociateOpportunity",
                "partnercentral:CreateEngagement",
                "partnercentral:CreateEngagementInvitation",
                "partnercentral:CreateOpportunity",
                "partnercentral:CreateResourceSnapshot",
                "partnercentral:CreateResourceSnapshotJob",
                "partnercentral:DeleteResourceSnapshotJob",
                "partnercentral:DisassociateOpportunity",
                "partnercentral:GetAwsOpportunitySummary",
                "partnercentral:GetEngagement",
                "partnercentral:GetEngagementInvitation",
                "partnercentral:GetOpportunity",
                "partnercentral:GetResourceSnapshot",
                "partnercentral:GetResourceSnapshotJob",
                "partnercentral:ListEngagementByAcceptingInvitationTasks",
                "partnercentral:ListEngagementFromOpportunityTasks",
                "partnercentral:ListEngagementInvitations",
                "partnercentral:ListEngagementMembers",
                "partnercentral:ListEngagementResourceAssociations",
                "partnercentral:ListEngagements",
                "partnercentral:ListOpportunities",
                "partnercentral:ListResourceSnapshotJobs",
```

```
                "partnercentral:ListResourceSnapshots",
                "partnercentral:ListSolutions",
                "partnercentral:RejectEngagementInvitation",
                "partnercentral:StartEngagementByAcceptingInvitationTask",
                "partnercentral:StartEngagementFromOpportunityTask",
                "partnercentral:StartResourceSnapshotJob",
                "partnercentral:StopResourceSnapshotJob",
                "partnercentral:SubmitOpportunity",
                "partnercentral:UpdateOpportunity"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListingAWSMarketplaceEntities",
            "Effect": "Allow",
            "Action": ["aws-marketplace:ListEntities"],
            "Resource": "*"
        },
        {
            "Sid": "AWSMarketplaceOffersAccess",
            "Effect": "Allow",
            "Action": ["aws-marketplace:DescribeEntity"],
            "Resource": [
              "arn:aws:aws-marketplace:*:*:AWSMarketplace/Offer/*",
              "arn:aws:aws-marketplace:*:*:AWSMarketplace/OfferSet/*"
            ]
        }
    ]
}
```

自訂政策

如果受管政策不符合您的需求，請建立自訂 IAM 政策，授予使用案例所需的許可。下列範例是授予列出AWS Marketplace實體許可的自訂政策：

Example自訂政策範例

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListOpportunities",
                "aws-marketplace:ListEntities"
            ],
            "Resource": "*"
        }
    ]
}
```

自訂寬鬆政策

此政策提供 Partner Central 銷售動作的廣泛存取權，包括未來可能新增的功能，而無需更新政策。透過使用萬用字元動作 `partnercentral:*`，此政策會在新的 Partner Central 銷售功能可用時自動授予存取權，以減少手動更新的需求。此政策也包含與AWS Marketplace實體互動的許可，這有助於確保維持銷售和 Marketplace 動作的存取權。

JSON

```
{
    "Version":"2012-10-17",
    "Statement":
    [
        {
            "Effect": "Allow",
            "Action":
            [
                "partnercentral:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action":
            [
                "aws-marketplace:ListEntities",
                "aws-marketplace:DescribeEntity"
            ],
            "Resource": "*"
```

```
        },
        {
            "Effect": "Allow",
            "Action":
            [
                "aws-marketplace:SearchAgreements",
                "aws-marketplace:DescribeAgreement"
            ],
            "Resource": "*",
            "Condition":
            {
                "StringEquals":
                {
                    "aws-marketplace:PartyType": "Proposer"
                }
            }
        }
    ]
}
```

將政策指派給 IAM 角色和使用者

請依照下列步驟將政策指派給 IAM 角色和使用者：

1. 登入AWS 管理主控台。
2. 導覽至 IAM 服務。
3. 選取角色或使用者，然後選擇您要連接政策的 IAM 角色或使用者。
4. 將`AWSPartnerCentralOpportunityManagement`政策或自訂政策連接至選取的 IAM 角色或使用者。

如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

使用條件索引鍵管理許可

IAM 政策中的條件索引鍵可為何時強制執行陳述式政策提供資源層級許可。您可以使用條件金鑰來指定允許或拒絕特定許可的條件。

如需詳細資訊，請參閱 IAM JSON 政策元素：Condition 運算子。

條件索引鍵概觀

條件金鑰	說明	適用的動作	有效值
partnercentral：Catalog	依相關聯的目錄實體類型篩選存取權	所有動作	AWS、沙盒
aws-marketplace：PartyType	根據方的類型（例如提案者）篩選存取權	SearchAgreements、DescribeAgreement	提案者

所需許可的摘要

所需許可的摘要

Action	說明
partnercentral：CreateOpportunity	允許建立機會
partnercentral：UpdateOpportunity	允許更新機會
partnercentral：ListOpportunities	允許列出機會
partnercentral：GetOpportunity	允許擷取機會詳細資訊
partnercentral：ListSolutions	允許列出解決方案
partnercentral：AssociateOpportunity	允許將機會與其他實體建立關聯
partnercentral：DisassociateOpportunity	允許取消機會與其他實體的關聯
partnercentral：AcceptEngagementInvitation	允許接受參與邀請
partnercentral：RejectEngagementInvitation	允許拒絕參與邀請
partnercentral：GetEngagementInvitation	允許擷取參與邀請詳細資訊
partnercentral：ListEngagementInvitations	允許列出參與邀請
partnercentral：SubmitOpportunity	允許提交機會

Action	說明
partnercentral：GetAwsOpportunitySummary	允許擷取AWS機會摘要
partnercentral：StartProspectingFromEngagementTask	從業務開發建立探勘任務
partnercentral：GetProspectingFromEngagementTask	傳回探勘任務詳細資訊
partnercentral：ListProspectingFromEngagementTasks	傳回探勘任務的清單
aws-marketplace：ListEntities	允許列出AWS Marketplace實體
aws-marketplace：DescribeEntity	允許描述AWS Marketplace實體
aws-marketplace：SearchAgreements	允許在 中搜尋協議AWS Marketplace
aws-marketplace：DescribeAgreement	允許在 中描述協議AWS Marketplace

利益 API 的存取權

存取控制和許可由AWS Identity and Access Management(IAM) 管理。本節提供設定與 利益 API 互動必要許可的指引。

先決條件

設定許可之前，請確定您的AWS 帳戶已連結至 ，且您已建立必要的 IAM 角色和使用者。如需詳細資訊，請參閱設定和身分驗證。

使用AWS受管政策

AWS提供受管政策，授予與 Benefits API 互動所需的許可。若要提供管理利益資源的必要存取權，請將`AWSPartnerCentralFullAccess`政策連接至您的 IAM 身分。如需詳細資訊，請參閱 AWS使用者的 受管政策。

將政策指派給 IAM 角色和使用者

請依照下列步驟將政策指派給 IAM 角色和使用者：

1. 登入AWS 管理主控台。
2. 導覽至 IAM 服務。
3. 選取角色或使用者，然後選擇您要連接政策的 IAM 角色或使用者。
4. 將`AWSPartnerCentralFullAccess`政策連接至選取的 IAM 角色或使用者。

如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

使用條件索引鍵管理許可

IAM 政策中的條件索引鍵可為何時強制執行陳述式政策提供資源層級許可。您可以使用條件金鑰來指定允許或拒絕特定許可的條件。

如需詳細資訊，請參閱 IAM JSON 政策元素：Condition 運算子。

條件索引鍵概觀

條件金鑰	說明	適用的動作	有效值
partnercentral：Catalog	依相關聯的目錄實體類型篩選存取權	所有動作	AWS、沙盒

所需許可的摘要

所需許可的摘要

Action	說明
partnercentral：AmendBenefitApplication	允許修改利益應用程式
partnercentral：AssociateBenefitApplicationResource	允許將資源與利益應用程式建立關聯
partnercentral：CancelBenefitApplication	允許取消利益應用程式
partnercentral：CreateBenefitApplication	允許建立利益應用程式
partnercentral：DisassociateBenefitApplicationResource	允許取消資源與利益應用程式的關聯

Action	說明
partnercentral：GetBenefit	允許擷取利益詳細資訊
partnercentral：GetBenefitAllocation	允許擷取利益分配詳細資訊
partnercentral：GetBenefitApplication	允許擷取利益應用程式詳細資訊
partnercentral：ListBenefitAllocations	允許列出利益分配
partnercentral：ListBenefitApplications	允許列出利益應用程式
partnercentral：ListBenefits	允許列出優點
partnercentral：RecallBenefitApplication	允許回收利益應用程式
partnercentral：SubmitBenefitApplication	允許提交利益應用程式
partnercentral：UpdateBenefitApplication	允許更新利益應用程式

通道 API 的存取

存取控制和許可由AWS Identity and Access Management(IAM) 管理。本節提供設定與AWS Partner Central Channel API 互動必要許可的指引。

頻道管理帳戶設定

AWS Partner Central 上的頻道管理功能需要將 IAM 角色部署在兩種類型的AWS帳戶中：

1. AWS Partner Central 連結AWS 帳戶：

   這是與您的AWS Partner Network 和 Partner Central 帳戶AWS 帳戶相關聯的 。AWS 帳戶每個AWS合作夥伴只有一個 Partner Central 連結。作為一次性設定活動，您將AWS 帳戶使用 Partner Central 頻道受管政策在此 中建立 IAM 角色。

2. 程式管理帳戶AWS 帳戶：

   這是AWS 帳戶用作 Bill-Transfer 帳戶的 ，用於管理終端客戶消費的帳單和付款。這AWS 帳戶會在 AWS Partner Central 頻道管理中報告為程式管理帳戶。您可能有多個計劃管理帳戶，而且您需要在每個AWS 帳戶報告為計劃管理帳戶的 中建立 IAM 角色。

對於每種AWS 帳戶類型，您將需要將不同的 IAM 角色與特定許可和信任政策建立關聯。

存取已連結的AWS Partner Central AWS 帳戶

在AWS Partner Central 連結中AWS 帳戶，建立 IAM 角色來管理頻道資源，這會授予 Partner Central 使用者。此 IAM 角色允許使用者在AWS Partner Central 上檢視和管理頻道管理資源。角色名稱必須以 開頭`PartnerCentralRoleFor`，建議名稱為 `PartnerCentralRoleForChannel`。若要設定角色，請完成下列步驟：

- 新增下列自訂信任政策，以允許AWS Partner Central 雲端管理員和聯盟將 IAM 角色映射至 Partner Central 使用者：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "partnercentral-account-management.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- 將 `AWSPartnerCentralChannelManagement`受管政策與`PartnerCentralRoleForChannel`角色建立關聯。

存取（程式管理帳戶）AWS帳戶

在AWS Partner Central 中AWS 帳戶每個報告為程式管理帳戶的 中，建立下列 IAM 角色，其必要名稱如下：

- `PartnerCentralChannelHandshakeApprovalManagement`：利用此角色來管理 Partner Central 與程式管理之間的交握AWS 帳戶。此 IAM 角色可用來回應，以在AWS Partner Central 中啟用您的程式管理帳戶。建立此角色時，請將`AWSPartnerCentralChannelHandshakeApprovalManagement`受管政策建立關聯。
- `PartnerCentralChannelBillingTransferReadOnly`：利用此跨帳戶角色，可從AWS Partner Central 中查看帳單轉移狀態。此角色會將帳單轉移狀態從您的 共用AWS 帳戶到AWS Partner Central，以集中檢視帳單轉移狀態。如果建立此角色，AWS Partner Central 中的使用者可

以在AWS Partner Central 中檢視帳單轉移狀態。不過，此角色不會授予AWS Partner Central 使用者在AWS Billing and Cost Management 主控台中存取帳單傳輸的許可。

- 信任政策：
  - 在下面的 JSON 中，在建立角色時將 取代`{PartnerCentralLinkedAccountID}`為您實際的AWS Partner Central linked AWS 帳戶 12 位數 ID。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::{PartnerCentralLinkedAccountId}:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- 許可政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BillingTransferReadOnly",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeResponsibilityTransfer",
        "organizations:ListHandshakesForOrganization",
        "organizations:ListInboundResponsibilityTransfers",
        "organizations:ListOutboundResponsibilityTransfers"
    ],
      "Resource": "*"
    }
  ]
}
```

- `PartnerCentralChannelBillingTransferManagement`（選用）：利用此跨帳戶角色，從AWS Partner Central 建立和管理帳單轉移。此角色將可讓AWS Partner Central 使用者存取包含`AWSPartnerCentralChannelManagement`受管政策的 IAM 角色，以管理計劃管理帳戶中的帳

單轉移AWS 帳戶。使用此跨帳戶角色，AWS Partner Central 使用者可以使用AWS Partner Central 管道管理中的「管理帳單」按鈕，在AWS Billing and Cost Management 主控台中存取和管理AWS 帳單轉移。

- 信任政策：
  - 在下面的 JSON 中，在建立角色時將 取代`{PartnerCentralLinkedAccountID}`為您實際的AWS Partner Central linked AWS 帳戶 12 位數 ID。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::{PartnerCentralLinkedAccountId}:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- 許可政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BillingTransferManagement",
      "Effect": "Allow",
      "Action": [
        "billingconductor:CreateBillingGroup",
        "billingconductor:ListPricingPlans",
        "organizations:AcceptHandshake",
        "organizations:CancelHandshake",
        "organizations:DeclineHandshake",
        "organizations:DescribeResponsibilityTransfer",
        "organizations:InviteOrganizationToTransferResponsibility",
        "organizations:ListHandshakesForAccount",
        "organizations:ListHandshakesForOrganization",
        "organizations:ListInboundResponsibilityTransfers",
        "organizations:ListOutboundResponsibilityTransfers",
        "organizations:TerminateResponsibilityTransfer",
        "organizations:UpdateResponsibilityTransfer"
```

```
    ],
      "Resource": "*"
    }
  ]
}
```

使用AWS受管政策

AWS提供受管政策，授予與頻道 API 互動所需的許可。若要提供管理所有頻道資源的必要存取權，請將AWSPartnerCentralChannelManagement政策連接至您的 IAM 身分。若要提供必要的存取權來檢視和回應頻道交握請求，請將AWSPartnerCentralChannelHandshakeApprovalManagement政策連接到您的 IAM 身分。如需詳細資訊，請參閱 AWSAWS Partner Central 使用者的 受管政策。

AWSPartnerCentralChannelManagement 政策

此政策授予 Partner Central 頻道管理動作的完整存取權。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ChannelManagement",
      "Effect": "Allow",
      "Action": [
        "partnercentral:CreateProgramManagementAccount",
        "partnercentral:UpdateProgramManagementAccount",
        "partnercentral:DeleteProgramManagementAccount",
        "partnercentral:ListProgramManagementAccounts",
        "partnercentral:GetProgramManagementAccount",
        "partnercentral:CreateRelationship",
        "partnercentral:UpdateRelationship",
        "partnercentral:DeleteRelationship",
        "partnercentral:GetRelationship",
        "partnercentral:ListRelationships",
        "partnercentral:CreateChannelHandshake",
        "partnercentral:AcceptChannelHandshake",
        "partnercentral:RejectChannelHandshake",
        "partnercentral:CancelChannelHandshake",
        "partnercentral:ListChannelHandshakes"
      ],
      "Resource": "*",
```

```
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "AWS",
            "Sandbox"
          ]
        }
      }
    },
    {
      "Sid": "ChannelBillingTransferRoleAccess",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/PartnerCentralChannelBillingTransferManagement",
        "arn:aws:iam::*:role/PartnerCentralChannelBillingTransferReadOnly"
      ]
    },
    {
      "Sid": "TaggingAccess",
      "Effect": "Allow",
      "Action": [
        "partnercentral:TagResource",
        "partnercentral:UntagResource",
        "partnercentral:ListTagsForResource"
      ],
      "Resource": [
        "arn:aws:partnercentral:*:*:catalog/*/program-management-account/*",
        "arn:aws:partnercentral:*:*:catalog/*/channel-handshake/*"
      ],
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "AWS",
            "Sandbox"
          ]
        }
      }
    }
  ]
}
```

AWSPartnerCentralChannelHandshakeApprovalManagement 政策

此政策授予檢視和回應頻道交握請求的存取權。此政策應套用至接收頻道交握請求之AWS帳戶中的角色。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ChannelHandshakeManagement",
      "Effect": "Allow",
      "Action": [
        "partnercentral:ListChannelHandshakes",
        "partnercentral:AcceptChannelHandshake",
        "partnercentral:RejectChannelHandshake"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "AWS",
            "Sandbox"
          ]
        }
      }
    }
  ]
}
```

將政策指派給 IAM 角色和使用者

請依照下列步驟將政策指派給 IAM 角色和使用者：

1. 登入AWS 管理主控台。
2. 導覽至 IAM 服務。
3. 選取角色或使用者，然後選擇您要連接政策的 IAM 角色或使用者。
4. 將AWSPartnerCentralChannelManagement政策或自訂政策連接至選取的 IAM 角色或使用者。

如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

使用條件索引鍵管理許可

IAM 政策中的條件索引鍵可為何時強制執行陳述式政策提供資源層級許可。您可以使用條件金鑰來指定允許或拒絕特定許可的條件。

如需詳細資訊，請參閱 IAM JSON 政策元素：Condition 運算子。

條件索引鍵概觀

條件金鑰	說明	適用的動作	有效值
partnercentral：Catalog	依相關聯的目錄實體類型篩選存取權	所有頻道管理動作	AWS、沙盒
partnercentral：ChannelHandshakeType	根據頻道交握類型篩選存取權	CreateChannelHandshake、AcceptChannelHandshake、RejectChannelHandshake、CancelChannelHandshake、ListChannelHandshakes	PROGRAM_MANAGEMENT_ACCOUNT、START_SERVICE_PERIOD、REVOKE_SERVICE_PERIOD

必要許可的摘要

必要許可的摘要

Action	說明
partnercentral：CreateProgramManagementAccount	允許建立程式管理帳戶
partnercentral：UpdateProgramManagementAccount	允許更新程式管理帳戶
partnercentral：DeleteProgramManagementAccount	允許刪除程式管理帳戶
partnercentral：ListProgramManagementAccounts	允許列出程式管理帳戶

Action	說明
partnercentral：GetProgramManagementAccount	允許擷取程式管理帳戶詳細資訊
partnercentral：CreateRelationship	允許建立關係
partnercentral：UpdateRelationship	允許更新關係
partnercentral：DeleteRelationship	允許刪除關係
partnercentral：GetRelationship	允許擷取關係詳細資訊
partnercentral：ListRelationships	允許列出關係
partnercentral：CreateChannelHandshake	允許建立頻道交握
partnercentral：AcceptChannelHandshake	允許接受頻道交握
partnercentral：RejectChannelHandshake	允許拒絕頻道交握
partnercentral：CancelChannelHandshake	允許取消頻道交握
partnercentral：ListChannelHandshakes	允許列出頻道交握

營收測量 API 的存取權

存取控制和許可由AWS Identity and Access Management(IAM) 管理。本節提供設定必要許可以與營收衡量 API 互動的指引，包括列出AWS Marketplace實體所需的許可。

先決條件

在設定許可之前，請確定您的AWS 帳戶已連結至 Partner Central，而且您已建立必要的 IAM 角色和使用者。如需詳細資訊，請參閱設定和身分驗證。

使用AWS受管政策

AWS提供受管政策，授予與營收測量 API 互動所需的許可。根據您的角色，有兩個 受管政策可用：

- 合作夥伴 — 將`AWSPartnerCentralRevenueAttributionManagement`政策連接到您的 IAM 身分。

- 客戶 — 將AWSRevenueAttributionManagement政策連接到您的 IAM 身分。

如需詳細資訊，請參閱 AWSAWS Partner Central 使用者的 受管政策。

AWSPartnerCentralRevenueAttributionManagement 政策

此政策為向AWS Partner Central 註冊AWS的帳戶提供收入歸因管理活動的必要存取權。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RevenueMeasurementCreation",
            "Effect": "Allow",
            "Action": [
                "partnercentral:CreateRevenueAttribution",
                "partnercentral:CreateMarketplaceRevenueShare",
                "partnercentral:CreateMarketplaceRevenueShareAllocation"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueMeasurementListing",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListRevenueAttributions",
                "partnercentral:ListRevenueAttributionAllocations",
                "partnercentral:ListMarketplaceRevenueShares",
                "partnercentral:ListMarketplaceRevenueShareAllocations"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
```

```
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueAttributionManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetRevenueAttribution",
                "partnercentral:UpdateRevenueAttribution",
                "partnercentral:GetRevenueAttributionAllocation",
                "partnercentral:StartRevenueAttributionAllocationsTask",
                "partnercentral:GetRevenueAttributionAllocationsTask"
            ],
            "Resource": "arn:aws:partnercentral:*:*:catalog/*/revenue-attribution/*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "MarketplaceRevenueShareManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetMarketplaceRevenueShare",
                "partnercentral:GetMarketplaceRevenueShareAllocation",
                "partnercentral:UpdateMarketplaceRevenueShareAllocation"
            ],
            "Resource": "arn:aws:partnercentral:*:*:catalog/*/marketplace-revenue-
share/*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
```

```
        {
            "Sid": "TaggingAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:TagResource",
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": [
                "arn:aws:partnercentral:*:*:catalog/*/revenue-attribution/*",
                "arn:aws:partnercentral:*:*:catalog/*/marketplace-revenue-share/*"
            ],
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "OpportunityAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListOpportunities",
                "partnercentral:GetOpportunity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "PartnerResourceAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListPartners",
                "partnercentral:GetPartner"
```

```
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "ListingAWSMarketplaceEntities",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:ListEntities"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSMarketplaceEntityAccess",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:DescribeEntity"
            ],
            "Resource": [
                "arn:aws:aws-marketplace:*:*:AWSMarketplace*/Offer/*"
            ]
        },
        {
            "Sid": "AmazonQPartnerAssistantAccess",
            "Effect": "Allow",
            "Action": [
                "q:StartConversation",
                "q:SendMessage",
                "q:GetConversation",
                "q:ListConversations",
                "q:PassRequest"
            ],
            "Resource": "*"
        }
    ]
}
```

AWSRevenueAttributionManagement 政策

此政策為未向AWS Partner Central 註冊AWS的帳戶提供收入歸因管理活動的必要存取權。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RevenueAttributionCreation",
            "Effect": "Allow",
            "Action": [
                "partnercentral:CreateRevenueAttribution"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueAttributionListing",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListRevenueAttributions"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueAttributionManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetRevenueAttribution",
```

```
                "partnercentral:UpdateRevenueAttribution"
            ],
            "Resource": "arn:aws:partnercentral:*:*:catalog/*/revenue-attribution/*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "TaggingAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:TagResource",
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": [
                "arn:aws:partnercentral:*:*:catalog/*/revenue-attribution/*"
            ],
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "AWS",
                        "Sandbox"
                    ]
                }
            }
        }
    ]
}
```

將政策指派給 IAM 角色和使用者

請依照下列步驟將政策指派給 IAM 角色和使用者：

1. 登入AWS 管理主控台。
2. 導覽至 IAM 服務。
3. 選取角色或使用者，然後選擇您要連接政策的 IAM 角色或使用者。

4. 將AWSPartnerCentralRevenueAttributionManagement政策（適用於合作夥伴）或AWSRevenueAttributionManagement政策（適用於客戶）連接至選取的 IAM 角色或使用者。

如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

使用條件索引鍵管理許可

IAM 政策中的條件索引鍵可為何時強制執行陳述式政策提供資源層級許可。您可以使用條件金鑰來指定允許或拒絕特定許可的條件。

如需詳細資訊，請參閱 IAM JSON 政策元素：Condition 運算子。

條件索引鍵概觀

條件金鑰	說明	適用的動作	有效值
partnercentral：Catalog	依相關聯的目錄實體類型篩選存取權	所有動作	AWS、沙盒

所需許可的摘要

所需許可的摘要

Action	說明
partnercentral：CreateRevenueAttribution	允許建立新的收入歸因記錄
partnercentral：UpdateRevenueAttribution	允許更新現有的營收歸因記錄
partnercentral：GetRevenueAttribution	允許擷取特定收入歸因的詳細資訊
partnercentral：ListRevenueAttributions	允許列出具有選用篩選條件的營收屬性
partnercentral：GetRevenueAttributionAllocation	允許依其 ID 擷取單一配置
partnercentral：ListRevenueAttributionAllocations	允許列出具有篩選支援的遞交配置

Action	說明
partnercentral：StartRevenueAttributionAllocationsTask	允許提交最多 250 個分配變更批次以進行非同步處理
partnercentral：GetRevenueAttributionAllocationsTask	允許擷取先前提交的配置任務狀態
partnercentral：CreateMarketplaceRevenueShare	允許建立新的市場收入共享資源
partnercentral：GetMarketplaceRevenueShare	允許擷取特定市場收入份額的詳細資訊
partnercentral：ListMarketplaceRevenueShares	允許列出具有選用篩選條件的市場收入共享
partnercentral：CreateMarketplaceRevenueShareAllocation	允許建立新的市場收入共享分配
partnercentral：GetMarketplaceRevenueShareAllocation	允許擷取特定市場收入分配的詳細資訊
partnercentral：UpdateMarketplaceRevenueShareAllocation	允許更新現有的市場收入分配
partnercentral：ListMarketplaceRevenueShareAllocations	允許列出市場收入份額下的配置
partnercentral：TagResource	允許新增或覆寫資源的標籤
partnercentral：UntagResource	允許從資源移除標籤
partnercentral：ListTagsForResource	允許列出與資源相關聯的標籤

AWS Partner Central API 的配額

下列各節提供有關服務配額的資訊。

主題

- AWS合作夥伴中央帳戶 API 的配額
- AWS Partner Central Selling API 的配額
- AWS Partner Central Benefits API 的配額
- AWS Partner Central Channel API 的配額
- AWS Partner Central Revenue Measurement API 的配額

AWS合作夥伴中央帳戶 API 的配額

AWS Partner Central Account API 具有下列配額。

其他配額

其他配額

顯示名稱	目錄	說明	預設值
每個帳戶的開啟連線邀請	AWS	您可以使用AWS目錄中的合作夥伴帳戶來維護的開放連線邀請數目上限	1,000
每個帳戶的作用中連線數	AWS	您可以在AWS目錄中與合作夥伴帳戶維持的作用中連線數目上限	1,000
每個合作夥伴的電子郵件網域	AWS	AWS目錄中可與合作夥伴帳戶相關聯的AWS訓練認證電子郵件網域數量上限	50
每個帳戶的連線邀請率	AWS	您每天可在AWS目錄中傳送的連線邀請數量上限	50
每個帳戶的設定檔更新任務速率	AWS	您每天可在AWS目錄中啟動的設定檔更新任務數目上限	10

顯示名稱	目錄	說明	預設值
每個帳戶的設定檔可見性更新率	AWS	您每天可以在AWS目錄中啟動的設定檔可見性更新數目上限	5
每個電子郵件地址的電子郵件驗證碼速率	AWS	您每天可在AWS目錄中請求的電子郵件驗證碼數量上限	5
每個帳戶的開啟連線邀請	沙盒	您可以在沙盒目錄中使用合作夥伴帳戶維持的開啟連線邀請數量上限	1,000
每個帳戶的作用中連線數	沙盒	您可以在沙盒目錄中與合作夥伴帳戶維持的作用中連線數目上限	1,000
每個合作夥伴的電子郵件網域	沙盒	在沙盒目錄中，可與合作夥伴帳戶相關聯的AWS訓練認證電子郵件網域數量上限	50
每個帳戶的連線邀請率	沙盒	您可以在沙盒目錄中每天傳送的連線邀請數量上限	50
每個帳戶的設定檔更新任務速率	沙盒	您可以在沙盒目錄中啟動的每天設定檔更新任務數目上限	10
每個帳戶的設定檔可見性更新率	沙盒	您可以在沙盒目錄中每天啟動的設定檔可見性更新數目上限	5

顯示名稱	目錄	說明	預設值
每個電子郵件地址的電子郵件驗證碼速率	沙盒	您可以在沙盒目錄中請求的電子郵件驗證碼數量上限。沙盒目錄不會傳送電子郵件。	N/A

了解和管理配額

速率限制

達到 API 速率限制時，服務將以 ThrottlingException 回應。為了更好地處理速率限制，AWS 建議在應用程式中實作指數退避和重試策略。

請求提高配額

如果預設配額不符合您的需求，您可以透過 Service Quotas 頁面請求增加配額。Service Quotas 主控台是瀏覽器型界面，可用來檢視和管理服務配額。您可以從任何AWS 管理主控台頁面存取 Service Quotas，方法是在頂端導覽列中選擇它，或在 中搜尋 Service Quotas AWS 管理主控台。

AWS Partner Central Selling API 的配額

AWS Partner Central 銷售 API 會強制執行配額，以確保公平使用，並防止服務遭到濫用。以下是每個機會的各種 API 操作和關聯的詳細配額。

API 操作配額

Type	API 操作	配額（每個合作夥伴帳戶）
讀取動作	GetOpportunity	每秒 10 次；每 24 小時 100，000 次
	GetAwsOpportunitySummary	
	ListOpportunities	
	ListSolutions	
	GetEngagementInvitation	

Type	API 操作	配額（每個合作夥伴帳戶）
	ListEngagementInvitations	
探勘讀取動作	GetProspectingFrom EngagementTask	每秒 100 次
探勘讀取動作	ListProspectingFro mEngagementTasks	每秒 50
寫入動作	CreateOpportunity	每秒 1 次；每 24 小時 10，000 次
	UpdateOpportunity	
	AssociateOpportunity	
	DisassociateOpportunity	
	RejectEngagementInvitation	
	AssignOpportunity	
	StartEngagementFro mOpportunityTask	
	StartEngagementByAcceptingI nvitationTask	
探勘寫入動作	StartProspectingFr omEngagementTask	每秒 2 個
參與寫入動作	CreateEngagement	每秒 15 個

每個機會的關聯配額

相關實體	配額
AWS產品	每個機會 20 個

相關實體	配額
合作夥伴解決方案	每個機會 10 個
AWS Marketplace解決方案	每個機會 10 個
AWS Marketplace產品	每個機會 10 個
AWS Marketplace私有優惠	每個機會 1 個

了解和管理配額

速率限制

達到 API 速率限制時，服務將以 ThrottlingException 回應。為了更好地處理速率限制，AWS 建議在應用程式中實作指數退避和重試策略。

配額的時間範圍

在連續 24 小時期間重設每日配額。您的請求會受到調節，例如，如果您在過去 24 小時內已執行 10，000 個寫入動作，並嘗試執行第 10，001 個請求。確保應用程式的使用模式將此納入考量，以防止意外限流。

請求提高配額

如果預設配額不符合您的需求，您可以透過 Service Quotas 頁面請求增加配額。Service Quotas 主控台是瀏覽器型界面，可用來檢視和管理服務配額。您可以從任何AWS 管理主控台頁面存取 Service Quotas，方法是在頂端導覽列中選擇它，或在 中搜尋 Service Quotas AWS 管理主控台。

AWS Partner Central Benefits API 的配額

AWS Partner Central Benefits API 具有下列配額。

請求配額

請求配額

API 操作	配額 （每個AWS帳戶）
ListBenefits	每秒 20 個

API 操作	配額（每個AWS帳戶）
GetBenefit	每秒 20 個
AmendBenefitApplication	每秒 10 個
CancelBenefitApplication	每秒 10 個
CreateBenefitApplication	每秒 10 個
GetBenefitApplication	每秒 20 個
ListBenefitApplications	每秒 30 個
RecallBenefitApplication	每秒 10 個
SubmitBenefitApplication	每秒 10 個
UpdateBenefitApplication	每秒 10 個
GetBenefitAllocation	每秒 20 個
ListBenefitAllocations	每秒 20 個
AssociateBenefitApplicationResource	每秒 10 個
DisassociateBenefitApplicationResource	每秒 10 個

其他配額

其他配額

顯示名稱	目錄	說明	預設值
效益應用程式	AWS	您可以在AWS目錄中建立的最大利益應用程式數量	10,000

顯示名稱	目錄	說明	預設值
效益應用程式	沙盒	您可以在沙盒目錄中建立的最大利益應用程式數量	10,000

了解和管理配額

速率限制

達到 API 速率限制時，服務將以 ThrottlingException 回應。為了更好地處理速率限制，AWS 建議在應用程式中實作指數退避和重試策略。

請求提高配額

如果預設配額不符合您的需求，您可以透過 Service Quotas 頁面請求增加配額。Service Quotas 主控台是瀏覽器型界面，可用來檢視和管理服務配額。您可以從任何AWS 管理主控台頁面存取 Service Quotas，方法是在頂端導覽列中選擇它，或在 中搜尋 Service Quotas AWS 管理主控台。

AWS Partner Central Channel API 的配額

AWS Partner Central Channel API 具有下列配額。

請求配額

請求配額

API 操作	配額 （每個AWS帳戶）
CreateProgramManagementAccount	每秒 3 個
UpdateProgramManagementAccount	每秒 3 個
ListProgramManagementAccounts	每秒 5
DeleteProgramManagementAccount	每秒 3 個
CreateChannelHandshake	每秒 3 個
ListChannelHandshakes	每秒 5

API 操作	配額（每個AWS帳戶）
AcceptChannelHandshake	每秒 3 個
RejectChannelHandshake	每秒 3 個
CancelChannelHandshake	每秒 3 個
CreateRelationship	每秒 3 個
GetRelationship	每秒 5
ListRelationships	每秒 5
UpdateRelationship	每秒 3 個
DeleteRelationship	每秒 3 個

其他配額

其他配額

顯示名稱	目錄	說明	預設值
程式管理帳戶	AWS	您可以在AWS目錄中建立的程式管理帳戶數目上限	50
每個程式管理帳戶的關係	AWS	您可以在AWS目錄中為每個程式管理帳戶建立的關係數目上限	1,000
每個帳戶的程式管理帳戶交握	AWS	AWS目錄中作用中程式管理帳戶交握的最大數量	100
啟動每個帳戶的服務期間交握	AWS	在AWS目錄中建立服務期間的作用中交握數量上限	200

顯示名稱	目錄	說明	預設值
撤銷每個帳戶的服務期間交握	AWS	AWS目錄中撤銷服務期間的作用中交握數量上限	200
程式管理帳戶每天交握的速率	AWS	您可以每天傳送至AWS目錄中相同AWS帳戶的程式管理帳戶交握數目上限	5
每天開始服務期間交握的速率	AWS	您可以每天傳送到AWS目錄中相同AWS帳戶的啟動服務期間交握次數上限	5
每天撤銷服務期間交握的速率	AWS	您可以每天傳送至AWS目錄中相同AWS帳戶的撤銷服務期間交握次數上限	5
程式管理帳戶	沙盒	您可以在沙盒目錄中建立的程式管理帳戶數目上限	50
每個程式管理帳戶的關係	沙盒	您可以在沙盒目錄中為每個程式管理帳戶建立的關係數目上限	1,000
每個帳戶的程式管理帳戶交握	沙盒	沙盒目錄中的作用中程式管理帳戶交握數目上限	100
啟動每個帳戶的服務期間交握	沙盒	在沙盒目錄中建立服務期間的作用中交握數量上限	200

顯示名稱	目錄	說明	預設值
撤銷每個帳戶的服務期間交握	沙盒	沙盒目錄中撤銷服務期間的作用中交握數量上限	200
程式管理帳戶每天交握的速率	沙盒	您可以在沙盒目錄中每天傳送至相同AWS帳戶的程式管理帳戶交握數目上限	5
每天開始服務期間交握的速率	沙盒	您可以在沙盒目錄中每天傳送至相同AWS帳戶的啟動服務期間交握次數上限	5
每天撤銷服務期間交握的速率	沙盒	您可以每天傳送至沙盒目錄中相同AWS帳戶的撤銷服務期間交握次數上限	5

了解和管理配額

速率限制

達到 API 速率限制時，服務將以 ThrottlingException 回應。為了更好地處理速率限制，AWS 建議在應用程式中實作指數退避和重試策略。

請求提高配額

如果預設配額不符合您的需求，您可以透過 Service Quotas 頁面請求增加配額。Service Quotas 主控台是瀏覽器型界面，可用來檢視和管理服務配額。您可以在頂端導覽列上選擇 Service Quotas，或在 中搜尋 Service Quotas，以從任何AWS 管理主控台頁面存取 Service Quotas AWS 管理主控台。Service Quotas

AWS Partner Central Revenue Measurement API 的配額

AWS Partner Central Revenue Measurement API 會強制執行配額，以確保公平使用，並防止服務遭到濫用。以下是各種 API 操作的詳細配額。

API 操作配額

Type	API 操作	配額（每個合作夥伴帳戶）
讀取動作	GetRevenueAttribution	每秒 50
	GetRevenueAttributionAllocation	
	GetRevenueAttributionAllocationsTask	
	GetMarketplaceRevenueShare	
	GetMarketplaceRevenueShareAllocation	
列出動作	ListRevenueAttributions	每秒 10 個
	ListRevenueAttributionAllocations	
	ListMarketplaceRevenueShares	
	ListMarketplaceRevenueShareAllocations	
	ListTagsForResource	
寫入動作	CreateRevenueAttribution	每秒 2 個
	UpdateRevenueAttribution	
	CreateMarketplaceRevenueShare	
	CreateMarketplaceRevenueShareAllocation	

Type	API 操作	配額（每個合作夥伴帳戶）
	UpdateMarketplaceRevenueShareAllocation	
	TagResource	
	UntagResource	
非同步寫入動作	StartRevenueAttributionAllocationsTask	每秒 1 個

了解和管理配額

速率限制

達到 API 速率限制時，服務將以 ThrottlingException 回應。為了更好地處理速率限制，AWS建議在應用程式中實作指數退避和重試策略。

請求提高配額

如果預設配額不符合您的需求，您可以透過 Service Quotas 頁面請求增加配額。Service Quotas 主控台是瀏覽器型界面，可用來檢視和管理服務配額。您可以從任何AWS 管理主控台頁面存取 Service Quotas，方法是在頂端導覽列中選擇它，或在 中搜尋 Service Quotas AWS 管理主控台。

登入AWS Partner Central API

下列各節提供記錄的相關資訊。

主題

- 記錄AWS Partner Central Account API
- 記錄AWS Partner Central Selling API
- 記錄AWS Partner Central Benefits API
- 記錄AWS Partner Central Channel API
- 記錄AWS Partner Central Revenue Attribution API

記錄AWS Partner Central Account API

AWS CloudTrail 是一項服務，可對您的 進行控管、合規、營運稽核和風險稽核AWS 帳戶。使用AWS CloudTrail，您可以記錄、持續監控和保留與AWS基礎設施中動作相關的帳戶活動。AWS合作夥伴中央帳戶 API 活動會記錄為 CloudTrail 中的事件。您可以建立線索，這是可將事件做為日誌檔案交付至 Amazon S3 儲存貯體的組態。

概觀

AWS Partner Central 帳戶 API 已與 整合AWS CloudTrail，此服務提供由使用者、角色或AWS Partner Central 中的AWS服務所採取之動作的記錄。CloudTrail 會將AWS Partner Central Account API 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自AWS Partner Central 的呼叫，以及來自AWS Partner Central Account API 操作的程式碼呼叫。

如果您建立線索，則可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括AWS Partner Central Account API 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。

您可以使用 CloudTrail 所收集的資訊，判斷對AWS Partner Central Account API 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。

了解AWS Partner Central Account API 日誌檔案項目

線索是一種組態，可將事件做為日誌檔案交付至 Amazon S3 儲存貯體。當您的線索追蹤AWS Partner Central Account API 事件時，CloudTrail 會將事件處理為所有區域的日誌檔案。每個日誌檔案可以包含一或多個事件。

下列範例顯示 CloudTrail 日誌項目，示範AWS Partner Central Account API 上的 `CreatePartner`動作：

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/CloudTrailTestUser",
        "accountId": "123456789012",
        "accessKeyId": "ABCDEFGHIJKLMNOP1234",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAEXAMPLE52AGFT725JGDZ",
```

```
                "arn": "arn:aws:iam::123456789012:role/ExampleRole",
                "accountId": "123456789012",
                "userName": "ExampleRole"
            },
            "attributes": {
                "creationDate": "2025-11-07T19:45:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-11-07T19:45:58Z",
    "eventSource": "partnercentral-account.amazonaws.com",
    "eventName": "CreatePartner",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "PostmanRuntime/7.18.0",
    "requestParameters": {
        "catalog": "AWS",
        "clientToken": "abcdef12-3456-7890-bcde-f123456789ab",
        "legalName": "ExampleLegalName",
        "primarySolutionType": "VALUE_ADDED_RESALE_AWS_SERVICES",
        "allianceLeadContact": {
            "firstName": "ExampleFirstName",
            "lastName": "ExampleLastName",
            "email": "example@domain.com",
            "businessTitle": "ExampleBusinessTitle"
        },
        "emailVerificationCode": "ExampleVerificationCode",
        "tags": [
            {
                "key": "office",
                "value": "1"
            }
        ]
    },
    "responseElements": {
        "catalog": "AWS",
        "arn": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/partner/
EXAMPLE_PARTNER_ID",
        "id": "EXAMPLE_PARTNER_ID",
        "legalName": "ExampleLegalName",
        "createdAt": "2025-11-07T19:45:57.867007329Z",
        "profile": {
            "primarySolutionType": "VALUE_ADDED_RESALE_AWS_SERVICES"
```

```
        },
        "allianceLeadContact": {
            "firstName": "ExampleFirstName",
            "lastName": "ExampleLastName",
            "email": "example@domain.com",
            "businessTitle": "ExampleBusinessTitle"
        }
    },
    "requestID": "12345678-1234-5678-9abc-def012345678",
    "eventID": "87654321-4321-8765-cba9-fed098765432",
    "readOnly": false,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::PartnerCentral::Partner",
            "ARN": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/partner/
EXAMPLE_PARTNER_ID"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "partnercentral-account.partner.us-east-1.api.aws"
    }
}
```

在此範例中，名為 CloudTrailTestUser 的 IAM 使用者呼叫了 `CreatePartner`動作。該請求於 2025 年 11 月 7 日 19：45：58 UTC 提出。請求建立了具有 AWS `EXAMPLE_PARTNER_ID` 目錄 ID 的新合作夥伴，其法定名稱為「ExampleLegalName」。

AWS Partner Central Account API 日誌檔項目中的欄位

CloudTrail 日誌檔案中的每個項目都包含提出請求的人員、請求中處理的資源，以及AWS Partner Central Account API 傳回的回應元素的相關資訊。日誌項目中的欄位清單，例如 `eventVersion`、`userIdentity`和 `eventTime`，提供動作的詳細資訊。例如， `sourceIPAddress` 欄位會顯示提出請求的 IP 地址。

記錄AWS Partner Central Selling API

AWS CloudTrail 是一項服務，可對您的 進行控管、合規、營運稽核和風險稽核AWS 帳戶。使用AWS CloudTrail，您可以記錄、持續監控和保留AWS與基礎設施中動作相關的帳戶活動。AWS Partner Central API 活動會記錄為 CloudTrail 中的事件。您可以建立線索，這是可將事件做為日誌檔案交付至 Amazon S3 儲存貯體的組態。

概觀

AWS Partner Central API 已與 整合AWS CloudTrail，此服務提供由使用者、角色或AWS Partner Central 中的AWS服務所採取之動作的記錄。CloudTrail 會將AWS Partner Central 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自AWS Partner Central 的呼叫，以及來自AWS Partner Central API 操作的程式碼呼叫。

如果您建立追蹤，則可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括AWS Partner Central 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。

使用 CloudTrail 收集的資訊，您可以判斷向AWS Partner Central 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。

了解AWS Partner Central Selling API 日誌檔案項目

線索是一種組態，可將事件做為日誌檔案交付至 Amazon S3 儲存貯體。當您的線索追蹤AWS Partner Central 事件時，CloudTrail 會將事件處理為所有區域的日誌檔案。每個日誌檔案可以包含一或多個事件。

下列範例顯示 CloudTrail 日誌項目，示範AWS Partner Central 上的 `ListOpportunities`動作：

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDEFGHIJKLMNOP12345",
        "arn": "arn:aws:iam::123456789010:user/CloudTrailTestUser",
        "accountId": "123456789010",
        "accessKeyId": "ABCDEFGHIJKLMNOP1234",
        "userName": "CloudTrailTestUser"
    },
    "eventTime": "2023-10-17T21:49:23Z",
    "eventSource": "partnercentral-selling.amazonaws.com",
    "eventName": "ListOpportunities",
```

```
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "PostmanRuntime/7.18.0",
    "requestParameters": {
        "MaxResults": 20
    },
    "responseElements": null,
    "requestID": "fEXAMPLE-cb3e-4e21-86fd-6b3EXAMPLEd1",
    "eventID": "7EXAMPLE-97d6-4139-91e3-01aEXAMPLE48",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789010"
}
```


在此範例中，名為 CloudTrailTestUser 的 IAM 使用者呼叫了 `ListOpportunities`動作。已於 us-east-1 中呼叫 動作，AWS 區域並於 2023 年 10 月 17 日 21：49：23 UTC 提出請求。

AWS Partner Central Selling API 日誌檔項目中的欄位

CloudTrail 日誌檔中的每個項目都包含提出請求的人員、請求中處理的資源，以及AWS Partner Central 傳回的回應元素的相關資訊。日誌項目中的欄位清單，例如 `eventVersion`、`userIdentity`和 `eventTime`，提供動作的詳細資訊。例如， `sourceIPAddress` 欄位會顯示提出請求的 IP 地址。

記錄AWS Partner Central Benefits API

AWS CloudTrail 是一項服務，可對您的 進行控管、合規、營運稽核和風險稽核AWS 帳戶。使用 AWS CloudTrail，您可以在整個AWS基礎設施中記錄、持續監控和保留與動作相關的帳戶活動。AWS Partner Central Benefits API 活動會記錄為 CloudTrail 中的事件。您可以建立線索，這是可將事件做為日誌檔案交付至 Amazon S3 儲存貯體的組態。

概觀

AWS Partner Central Benefits API 已與 整合AWS CloudTrail，此服務提供由使用者、角色或AWS Partner Central 中的AWS服務所採取之動作的記錄。CloudTrail 會將AWS Partner Central Benefits API 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自AWS Partner Central 的呼叫，以及來自AWS Partner Central Benefits API 操作的程式碼呼叫。

如果您建立線索，您可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括AWS Partner Central Benefits API 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。

您可以使用 CloudTrail 所收集的資訊，判斷對AWS Partner Central Benefits API 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。

了解AWS Partner Central Benefits API 日誌檔案項目

追蹤是一種組態，可讓您將事件做為日誌檔案交付至 Amazon S3 儲存貯體。當您的線索追蹤AWS Partner Central Benefits API 事件時，CloudTrail 會將事件視為所有區域的日誌檔案來處理。每個日誌檔案可以包含一或多個事件。

下列範例顯示 CloudTrail 日誌項目，示範AWS Partner Central Benefits API 上的`ListBenefitApplications`動作：

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/CloudTrailTestUser",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EX_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/TestRole",
                "accountId": "123456789012",
                "userName": "TestRole"
            },
            "attributes": {
                "creationDate": "2025-10-21T03:08:23Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-10-21T03:10:59Z",
    "eventSource": "partnercentral-benefits.amazonaws.com",
    "eventName": "ListBenefitApplications",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "python-requests/2.32.4",
    "requestParameters": {
        "catalog": "AWS"
    },
```

```
    "responseElements": null,
    "requestID": "12345678-1234-5678-9abc-def012345678",
    "eventID": "87654321-4321-8765-cba9-fed098765432",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

在此範例中，名為 Alice 的 IAM 使用者呼叫了 `ListBenefitApplications`動作。該請求於 2025 年 10 月 21 日 UTC 03：10：59 提出。列出 AWS 目錄之利益應用程式的請求。

AWS Partner Central Benefits API 日誌檔項目中的欄位

CloudTrail 日誌檔中的每個項目都包含提出請求的人員、請求中處理的資源，以及AWS Partner Central Benefits API 傳回的回應元素的相關資訊。日誌項目中的欄位清單，例如 `eventVersion`、`userIdentity`和 `eventTime`，提供動作的詳細資訊。例如， `sourceIPAddress` 欄位會顯示提出請求的 IP 地址。

記錄AWS Partner Central Channel API

AWS CloudTrail 是一項服務，可對您的 進行控管、合規、營運稽核和風險稽核AWS 帳戶。使用AWS CloudTrail，您可以在整個AWS基礎設施中記錄、持續監控和保留與動作相關的帳戶活動。AWS合作夥伴中央通道 API 活動會記錄為 CloudTrail 中的事件。您可以建立線索，此組態可讓事件做為日誌檔案交付至 Amazon S3 儲存貯體。

概觀

AWS Partner Central Channel API 已與 整合AWS CloudTrail，此服務提供由使用者、角色或AWS Partner Central 中的AWS服務所採取之動作的記錄。CloudTrail 會將AWS Partner Central Channel API 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自AWS Partner Central 的呼叫，以及來自AWS Partner Central Channel API 操作的程式碼呼叫。

如果您建立追蹤，則可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括AWS Partner Central Channel API 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。

您可以使用 CloudTrail 所收集的資訊，判斷對AWS Partner Central Channel API 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。

了解AWS Partner Central Channel API 日誌檔案項目

線索是一種組態，可將事件做為日誌檔案交付至 Amazon S3 儲存貯體。當您的線索追蹤AWS Partner Central Channel API 事件時，CloudTrail 會將事件視為所有區域的日誌檔案來處理。每個日誌檔案可以包含一或多個事件。

下列範例顯示 CloudTrail 日誌項目，示範AWS Partner Central Channel API 上的`CreateProgramManagementAccount`動作：


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAEXAMPLE52AGFT725JGDZ:example-user-Isengard",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/example-user-Isengard",
        "accountId": "123456789012",
        "accessKeyId": "ASIAEXAMPLE52AGL6IXQLZ5",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAEXAMPLE52AGFT725JGDZ",
                "arn": "arn:aws:iam::123456789012:role/ExampleRole",
                "accountId": "123456789012",
                "userName": "ExampleRole"
            },
            "attributes": {
                "creationDate": "2025-10-21T17:06:47Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-10-21T17:07:26Z",
    "eventSource": "partnercentral-channel.amazonaws.com",
    "eventName": "CreateProgramManagementAccount",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "PostmanRuntime/7.18.0",
    "requestParameters": {
        "catalog": "AWS",
        "program": "SOLUTION_PROVIDER",
        "displayName": "ExampleDisplayName",
        "accountId": "987654321098",
        "clientToken": "abcdef12-3456-7890-bcde-f123456789ab"
```

```
    },
    "responseElements": {
        "programManagementAccountDetail": {
            "id": "pma-example123456789",
            "arn": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/program-
management-account/pma-example123456789"
        }
    },
    "requestID": "12345678-1234-5678-9abc-def012345678",
    "eventID": "87654321-4321-8765-cba9-fed098765432",
    "readOnly": false,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::PartnerCentralChannel::ProgramManagementAccount",
            "ARN": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/program-
management-account/pma-example123456789"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "clientProvidedHostHeader": "partnercentral-channel.global.api.aws"
    }
}
```

在此範例中，名為 ExampleRole 的 IAM 角色透過擔任的角色工作階段呼叫 `CreateProgramManagementAccount`動作。該請求於 2025 年 10 月 21 日 UTC 17：07：26 提出。請求建立了具有解決方案供應商計劃 ID `pma-example123456789` 的新計劃管理帳戶。

AWS Partner Central Channel API 日誌檔項目中的欄位

CloudTrail 日誌檔中的每個項目都包含有關提出請求的人員、請求中處理的資源，以及AWS Partner Central Channel API 傳回的回應元素的資訊。日誌項目中的欄位清單，例如 `eventVersion`、`userIdentity`和 `eventTime`，提供動作的詳細資訊。例如， `sourceIPAddress` 欄位會顯示提出請求的 IP 地址。

記錄AWS Partner Central Revenue Attribution API

AWS CloudTrail 是一項服務，可讓您的 進行控管、合規、營運稽核和風險稽核AWS 帳戶。使用AWS CloudTrail，您可以記錄、持續監控和保留AWS與基礎設施中動作相關的帳戶活動。AWS合作夥伴中央營收測量 API 活動會記錄為 CloudTrail 中的事件。您可以建立線索，此組態可讓事件做為日誌檔案交付至 Amazon S3 儲存貯體。

概觀

AWS Partner Central Revenue Measurement API 已與 整合AWS CloudTrail，此服務提供由使用者、角色或AWS Partner Central 中的AWS服務所採取之動作的記錄。CloudTrail 會將AWS Partner Central Revenue Attribution 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自AWS Partner Central 主控台的呼叫，以及來自AWS Partner Central Revenue Measurement API 操作的程式碼呼叫。

如果您建立線索，您可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括AWS Partner Central Revenue Measurement 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。

您可以使用 CloudTrail 所收集的資訊，判斷對AWS Partner Central Revenue Measurement 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。

下列AWS Partner Central Revenue Measurement API 動作會記錄在 CloudTrail 中：

- `CreateRevenueAttribution`
- `UpdateRevenueAttribution`
- `GetRevenueAttribution`
- `ListRevenueAttributions`
- `TagResource`
- `UntagResource`
- `ListTagsForResource`
- `CreateMarketplaceRevenueShare`
- `GetMarketplaceRevenueShare`
- `ListMarketplaceRevenueShares`
- `CreateMarketplaceRevenueShareAllocation`
- `GetMarketplaceRevenueShareAllocation`
- `UpdateMarketplaceRevenueShareAllocation`

- ListMarketplaceRevenueShareAllocations
- StartRevenueAttributionAllocationsTask
- GetRevenueAttributionAllocationsTask
- ListRevenueAttributionAllocations
- GetRevenueAttributionAllocation

了解AWS Partner Central Revenue Measurement API 日誌檔案項目

線索是一種組態，可將事件做為日誌檔案交付至 Amazon S3 儲存貯體。當您的線索追蹤AWS Partner Central Revenue Measurement 事件時，CloudTrail 會將事件視為所有區域的日誌檔案來處理。每個日誌檔案可以包含一或多個事件。

下列範例顯示 CloudTrail 日誌項目，示範AWS Partner Central Revenue Attribution 上的 CreateRevenueAttribution動作：

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAEXAMPLEID",
        "arn": "arn:aws:iam::123456789012:user/CloudTrailTestUser",
        "accountId": "123456789012",
        "accessKeyId": "AKIAEXAMPLEKEY",
        "userName": "CloudTrailTestUser"
    },
    "eventTime": "2026-06-02T11:53:54Z",
    "eventSource": "partnercentral-prm.amazonaws.com",
    "eventName": "CreateRevenueAttribution",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "aws-sdk-java/2.20.0",
    "requestParameters": {
        "catalog": "AWS",
        "name": "my-revenue-attribution",
        "marketplaceProduct": {
            "tenancyModel": "MULTI_TENANT"
        }
    },
    "responseElements": {
        "id": "ra-0123456789abcdef0",
```

```
        "arn": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/revenue-
attribution/ra-0123456789abcdef0",
        "name": "my-revenue-attribution",
        "revision": "1"
    },
    "requestID": "5ce2f907-3850-412a-b1a4-r012r1234567",
    "eventID": "80b39b72-eefe-4578-8db0-01ee2e34ee56",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


在此範例中，名為 CloudTrailTestUser 的 IAM 使用者呼叫了 `CreateRevenueAttribution`動作。已於 us-east-1AWS 區域中呼叫 動作，並於 2026 年 6 月 2 日 11：53：54 UTC 提出請求。已使用 ID 建立新的收入歸因資源`ra-0123456789abcdef0`。

AWS Partner Central Revenue Measurement API 日誌檔項目中的欄位

CloudTrail 日誌檔中的每個項目都包含提出請求的人員、請求中處理的資源，以及AWS Partner Central Revenue Measurement 傳回的回應元素的相關資訊。日誌項目中的欄位清單，例如 `eventVersion`、 `userIdentity`和 `eventTime`，提供動作的詳細資訊。例如，`sourceIPAddress` 欄位會顯示提出請求的 IP 地址。

AWS Partner Central API 的通知

下列各節提供通知的相關資訊。

主題

- AWS合作夥伴中央帳戶 API 的通知
- AWS Partner Central Selling API 的通知
- AWS Partner Central Benefits API 的通知
- AWS Partner Central Channel API 的通知

AWS合作夥伴中央帳戶 API 的通知

合作夥伴帳戶連線通知可讓合作夥伴隨時掌握直接影響其商業關係和營運工作流程的連線生命週期事件。這些事件對於合作夥伴來說至關重要：

- 迅速回應新的協同合作

- 保持對其連線產品組合狀態的意識
- 建立或終止連線時採取適當的動作
- 了解關係何時變更，以確保業務持續性

主題

- 完成監控事件的先決條件
- 設定 Amazon EventBridge 以監控事件
- 進一步了解帳戶連線 API 事件

完成監控事件的先決條件

使用者需要適當的 IAM 許可，才能存取和管理帳戶連線 API 發佈的事件。如需 EventBridge 可用動作、資源和條件索引鍵的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的在 Amazon EventBridge 中使用 IAM 政策條件。 EventBridge 其中一個條件索引鍵是 `events:detail-type`，可用於將許可範圍限定為特定事件類型。

下列範例政策示範如何自訂和限制提議事件的許可。`AllowPutRuleForPartnercentralAccountEvents` 陳述式允許建立規則，但僅適用於來源的事件`aws.partnercentral-account`。

如需詳細的 IAM 政策範例，請參閱 EventBridge 許可上的 AWS 文件。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralAccountEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-account.connection"
        }
      }
    }
  ]
}
```

設定 Amazon EventBridge 以監控事件

若要監控帳戶連線 API 事件，您可以建立符合您要擷取之事件的 EventBridge 規則。您可以使用AWS管理主控台或AWS SDKs來建立和管理規則。下列各節說明如何使用兩種方法建立規則。無論您使用何種方法，都必須在美國東部（維吉尼亞北部）us-east-1區域中建立規則。

AWS 管理主控台設定

若要使用 設定 EventBridge 規則AWS 管理主控台，請遵循《Amazon EventBridge 使用者指南》中的建立對 Amazon EventBridge 中的事件做出反應的規則中的步驟。 EventBridge 建立規則時，您必須將事件匯流排設定為預設值，並在美國東部（維吉尼亞北部）us-east-1區域中建立規則。

以下是事件規則的範例：

```
{
    "source": ["aws.partnercentral-account"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS開發套件設定

您可以使用AWS SDKs以程式設計方式建立和管理 EventBridge 規則。如需詳細資訊，請參閱《Amazon EventBridge API 參考》中的 PutRule。

下列範例使用適用於 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
    Name='MyConnectionInvitationReceivedRule',
    EventPattern=
    '{
        "source": ["aws.partnercentral-account"],
        "detail-type": ["Partner Connection Invitation Received"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
```

```
print('Rule ARN:', response['RuleArn'])
```

進一步了解帳戶連線 API 事件

下列各節說明帳戶連線 API 事件類型、觸發它們的案例，以及事件範例。

Event types (事件類型)

以下是事件類型及其觸發條件。

- 收到合作夥伴連線邀請：通知接收者另一個合作夥伴想要建立連線
- 已接受合作夥伴連線邀請：通知寄件者已接受其邀請，且連線現在處於作用中狀態
- 拒絕合作夥伴連線邀請：通知寄件者其邀請遭拒
- 已取消合作夥伴連線：當作用中連線終止時，通知其他連線的參與者
- 已取消合作夥伴連線邀請：在採取任何動作之前，通知接收者寄件者已撤銷其邀請
- 合作夥伴連線邀請已過期：通知寄件者和接收者邀請已過期

範例事件

下列各節提供上一節先前列出的事件範例。

主題

- 已接收的合作夥伴連線邀請
- 已接受合作夥伴連線邀請
- 合作夥伴連線邀請遭拒
- 已取消合作夥伴連線
- 已取消合作夥伴連線邀請
- 合作夥伴連線邀請已過期

已接收的合作夥伴連線邀請

觸發內容

當透過 CreateConnectionInvitation API 成功建立連線邀請並保留在 PAC 的資料存放區時，就會產生此事件。建立邀請成功後會立即傳送事件，並保留在 PAC 的資料存放區中，而不只是在進行 API 呼叫時。

建立新的連線邀請時，系統會自動傳送事件。每個建立的邀請一個事件，沒有相同邀請的重複事件。

收件人

連線邀請中接收者AWS的帳戶。

預期的處理方式

典型的客戶動作：

- 即時通知：觸發業務團隊有關新合作夥伴關係機會的提醒
- 工作流程自動化：使用待定邀請自動更新合作夥伴管理系統
- 決策支援：根據連線類型，將邀請路由給適當的決策者
- 稽核記錄：記錄邀請接收以進行合規和合作夥伴關係追蹤
- 回應自動化：對於信任的合作夥伴，可能會自動接受特定邀請類型

常見整合模式：

- Lambda 函數 → 更新合作夥伴入口網站儀表板
- SQS 佇列 → 批次程序邀請以供檢閱
- SNS 主題 → 向業務團隊傳送電子郵件/Slack 通知
- API 目的地 → Webhook 到外部 CRM/合作夥伴管理系統

範例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type": "Partner Connection Invitation Received",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
  "account": "<corresponding partner AWS account>",
  "resources": [ "<<Connection Invitation ARN>>" ],
  "detail": {
    "catalog": "AWS",
    "connectionInvitation" :{
        "arn": "<<Connection Invitation ARN>>",
        "id": "pacinv-****",
```

```
        "connectionType": "OpportunityCollaboration",
        "inviterEmail": "abc@def.com",
        "invitationMessage": "We'd like to collaborate on a joint solution for cloud
 security. Please accept this connection invitation to proceed.",
        "senderCompanyName": "<<Sender Partner Account Name>>",
        "senderProfileId": "pprofile-****",
        "expiresAt": "<ISO 8601 date time>"
    }
  }
}
```

已接受合作夥伴連線邀請

觸發內容

當成功接受連線邀請，並透過 AcceptConnectionInvitation API 建立連線並保留在 PAC 的資料存放區時，就會產生此事件。邀請接受成功完成並建立連線後，會立即傳送事件。

接受連線邀請時，系統會自動傳送事件。每個接受的邀請一個事件，沒有相同接受的重複事件。

收件人

涉及連線邀請的兩個AWS帳戶：最初傳送邀請的寄件者帳戶，以及接受邀請的接收者帳戶。

預期的處理方式

典型的客戶動作：

- 合作夥伴關係啟用：觸發工作流程以啟用第 2 層業務活動
- 狀態更新：更新具有作用中連線狀態的合作夥伴管理系統
- 通知：提醒業務團隊成功建立合作夥伴關係
- 存取佈建：自動授予適當的協同合作許可
- 分析：追蹤合作夥伴關係轉換率和成功指標

常見整合模式：

- Lambda 函數 → 啟用資料共用許可並更新合作夥伴入口網站
- SQS 佇列 → 業務設定的批次處理連線啟用
- SNS 主題 → 向業務和技術團隊傳送電子郵件/Slack 通知

- API 目的地 → Webhook 到 CRM 系統以更新合作夥伴關係狀態

範例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type":  "Partner Connection Invitation Accepted",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
  "resources": [ "<<Connection Invitation ARN>>" , "<<Connection ARN>>" ],
  "account": "<corresponding partner AWS account>",
  "detail": {
    "catalog": "AWS",
    "connectionInvitation" :{
        "arn": "<<Connection Invitation ARN>>",
        "id": "pacinv-****",
        "connectionType": "OpportunityCollaboration"
    },
    "connection": {
        "arn": "<<Connection ARN>>",
         "id": "pac-*****",
         "Participant1AccountId": "<<Sender AWS AccountId>>",
         "Participant2AccountId": "<<Receiver AWS AccountId>>",
         "status": "ACTIVE"
     }
  }
}
```

合作夥伴連線邀請遭拒

觸發內容

當透過 RejectConnectionInvitation API 成功拒絕連線邀請時，就會產生此事件。邀請拒絕處理後立即傳送事件，並保留在 PAC 的資料存放區中。

拒絕連線邀請時，系統會自動傳送事件。每個邀請拒絕一個事件，相同拒絕沒有重複的事件。

收件人

最初傳送連線邀請AWS的帳戶 （寄件者帳戶）。

預期的處理方式

典型的客戶動作：

- 狀態更新：更新具有拒絕狀態的合作夥伴管理系統
- 後續動作：觸發替代合作夥伴關係方法的工作流程
- 通知：提醒業務團隊有關合作夥伴關係決策
- 清除：從內部系統移除擱置中的邀請參考

常見整合模式：

- Lambda 函數 → 更新合作夥伴入口網站並觸發後續工作流程
- SQS 佇列 → 批次處理拒絕以進行業務分析
- SNS 主題 → 向業務開發團隊傳送電子郵件/Slack 通知
- API 目的地 → Webhook 到 CRM 系統以更新機會狀態

範例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type": "Partner Connection Invitation Rejected",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
  "account": "<corresponding partner AWS account>",
  "resources": [ "<<Connection Invitation ARN>>" ],
  "detail": {
    "catalog": "AWS",
    "connectionInvitation" :{
        "arn": "<<Connection Invitation ARN>>",
        "id": "pacinv-****",
        "connectionType": "OpportunityCollaboration",
        "invitationMessage": "We'd like to collaborate on a joint solution for cloud
 security. Please accept this connection invitation to proceed.",
        "receiverProfileId": "pprofile-****"
    }
  }
}
```

已取消合作夥伴連線

觸發內容

當作用中連線成功取消，且作用中狀態更新為透過 CancelConnection API 在 PAC 的資料存放區中終止時，就會產生此事件。處理連線取消並更新連線狀態後，會立即傳送事件。

連線終止時，系統會自動傳送事件。每個連線取消一個事件，相同取消沒有重複的事件。

收件人

連線中其他參與者AWS的帳戶（而非啟動取消的參與者）。

預期的處理方式

典型的客戶動作：

- 存取撤銷：立即撤銷許可並停用資料共用
- 狀態更新：使用終止的連線狀態更新合作夥伴管理系統
- 通知：提醒業務和技術團隊終止合作夥伴關係
- 清除：移除連線參考並清除共用資源
- 分析：追蹤連線持續時間和終止模式

常見整合模式：

- Lambda 函數 → 撤銷許可並更新合作夥伴入口網站狀態
- SQS 佇列 → 清除工作流程的批次處理連線終止
- SNS 主題 → 向業務和技術團隊傳送電子郵件/Slack 通知
- API 目的地 → Webhook 到 CRM 系統以更新合作夥伴關係狀態

範例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type": "Partner Connection Cancelled",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
```

```
  "resources": [ "<<Connection ARN>>" ],
  "account": "<corresponding partner AWS account>",
  "detail": {
    "catalog": "AWS",
    "connection" :{
        "arn": "<<Connection Invitation ARN>>",
        "id": "pac-****",
        "connectionType": "OpportunityCollaboration",
        "canceledBy": "<<AWS account ID>>"
    }
  }
}
```

已取消合作夥伴連線邀請

觸發內容

當透過 CancelConnectionInvitation API 成功取消待定連線邀請時，就會產生此事件。邀請取消處理後立即傳送事件，並將邀請狀態更新為 CANCELED。

取消連線邀請時，系統會自動傳送事件。每個連線邀請取消一個事件，相同取消沒有重複的事件。

收件人

應該接收連線邀請AWS的帳戶（接收者帳戶）。

預期的處理方式

典型的客戶動作：

- 狀態更新：更新合作夥伴管理系統以移除擱置中的邀請參考
- 通知：提醒業務團隊潛在的合作夥伴關係機會已撤銷
- 清除：從內部追蹤系統移除邀請參考
- 分析：追蹤合作夥伴關係洞見的邀請取消模式

常見整合模式：

- Lambda 函數 → 更新合作夥伴入口網站並移除待定邀請通知
- SQS 佇列 → 取消批次程序以進行業務分析
- SNS 主題 → 向業務開發團隊傳送電子郵件/Slack 通知
- API 目的地 → Webhook 到 CRM 系統以更新機會狀態

範例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type": "Partner Connection Invitation Cancelled",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
  "account": "<corresponding partner AWS account>",
  "resources": [ "<<Connection Invitation ARN>>" ],
  "detail": {
    "catalog": "AWS",
    "connectionInvitation" :{
        "arn": "<<Connection Invitation ARN>>",
        "id": "pacinv-****",
        "connectionType": "OpportunityCollaboration",
        "invitationMessage": "We'd like to collaborate on a joint solution for cloud
 security. Please accept this connection invitation to proceed.",
        "senderProfileId": "pprofile-****"
    }
  }
}
```

合作夥伴連線邀請已過期

觸發內容

當待定連線邀請在到達其 ExpiresAt 時間戳記後自動過期，而不被接收者接受或拒絕時，就會產生此事件。到達邀請過期時間時，系統會自動觸發事件。

連線邀請過期時，系統會自動傳送事件。每個邀請一個事件已過期，但沒有相同過期的重複事件。

收件人

涉及連線邀請的兩個AWS帳戶：最初傳送邀請的寄件者帳戶，以及收到邀請的接收者帳戶。

預期的處理方式

典型的客戶動作：

- 狀態更新：更新合作夥伴管理系統，將邀請標記為過期
- 後續動作：觸發重新參與的工作流程或其他合作夥伴關係方法

- 通知：提醒業務團隊錯失合作夥伴關係機會
- 清除：從內部追蹤系統移除過期的邀請參考

常見整合模式：

- Lambda 函數 → 更新合作夥伴入口網站並觸發後續工作流程
- SQS 佇列 → 用於業務分析的批次程序過期
- SNS 主題 → 向業務開發團隊傳送電子郵件/Slack 通知
- API 目的地 → Webhook 到 CRM 系統以更新機會狀態

範例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type": "Partner Connection Invitation Expired",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
  "resources": [ "<<Connection Invitation ARN>>" ],
  "detail": {
    "catalog": "AWS",
    "connectionInvitation": {
        "arn": "<<Connection Invitation ARN>>",
        "id": "pacinv-****",
        "connectionType": "OpportunityCollaboration",
        "inviterEmail": "abc@def.com",
        "invitationMessage": "We'd like to collaborate on a joint solution for cloud
 security. Please accept this connection invitation to proceed.",
        "senderProfileId": "pprofile-****",
        "receiverProfileId": "pprofile-****"
    }
  }
}
```

AWS Partner Central Selling API 的通知

銷售 API 的事件提供有關狀態變更或機會詳細資訊的即時通知。這些事件有助於讓您的系統與AWS Partner Central 保持同步，並有助於確保及時回應和更新。

主題

- 完成監控事件的先決條件
- 設定 Amazon EventBridge 以監控事件
- 進一步了解銷售 API 事件

完成監控事件的先決條件

使用者需要適當的 IAM 許可，才能存取和管理 Partner Central 銷售 API 發佈的事件。如需 EventBridge 可用動作、資源和條件索引鍵的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的在 Amazon EventBridge 中使用 IAM 政策條件。 EventBridge 其中一個條件索引鍵是 `events:detail-type`，可用於將許可範圍限定為特定事件類型。

下列範例政策示範如何自訂和限制提議事件的許可。`AllowPutRuleForPartnercentralSellingEvents` 陳述式允許建立規則，但僅適用於來自`aws.partnercentral-selling`來源的事件。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AllowPutRuleForPartnercentralSellingEvents",
    "Effect": "Allow",
    "Action": "events:PutRule",
    "Resource": "*",
    "Condition": {
      "ForAllValues:StringEquals": {
        "events:source": "aws.partnercentral-selling"
      }
    }
  }
}
```

設定 Amazon EventBridge 以監控事件

若要監控銷售 API 事件，您可以建立符合您要擷取之事件的 EventBridge 規則。您可以使用AWS 管理主控台或AWS SDKs來建立和管理規則。下列各節說明如何使用兩種方法建立規則。無論您使用何種方法，都必須在美國東部（維吉尼亞北部）`us-east-1`區域中建立規則。

AWS 管理主控台設定

若要使用 設定 EventBridge 規則AWS 管理主控台，請遵循《Amazon EventBridge 使用者指南》中的建立對 Amazon EventBridge 中的事件做出反應的規則中的步驟。 EventBridge 建立規則時，您必須將事件匯流排設定為預設值，並在美國東部（維吉尼亞北部）us-east-1區域中建立規則。

以下是事件規則的範例：

```
{
    "source": ["aws.partnercentral-selling"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS開發套件設定

您可以使用AWS SDKs以程式設計方式建立和管理 EventBridge 規則。如需詳細資訊，請參閱《Amazon EventBridge API 參考》中的 PutRule。

下列範例使用適用於 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
    Name='MyOpportunityCreatedRule',
    EventPattern=
    '{
        "source": ["aws.partnercentral-selling"],
        "detail-type": ["Opportunity Created"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
print('Rule ARN:', response['RuleArn'])
```

進一步了解銷售 API 事件

下列各節說明銷售 API 事件類型、觸發它們的案例，以及事件範例。

Event types (事件類型)

以下是事件類型及其觸發條件。

- 機會建立：在建立新機會時觸發。
- 機會已更新：當機會（機會或其對應的AWS機會摘要）更新時觸發。
- 建立的參與邀請：建立AWS推薦邀請時觸發。
- 已接受業務開發邀請：當合作夥伴接受AWS業務開發邀請時觸發，確認他們對與 合作AWS的機會感興趣。
- 拒絕業務開發邀請：當合作夥伴拒絕AWS業務開發邀請時觸發。
- 參與邀請已過期：當合作夥伴拒絕 EngagementInvitation 時，會觸發此事件。它會通知傳送和接收合作夥伴邀請已過期。
- 新增的參與成員：當新成員在接受邀請後加入參與時，會觸發此事件。它會通知所有目前的業務開發成員有關要新增至業務開發的新成員。
- 建立業務開發資源快照：建立新的業務開發相關資源（例如機會）快照修訂版時，會觸發此事件。它會通知業務開發的所有目前成員有關相關資源快照的變更。
- 建立業務開發：在建立新的業務開發時觸發。
- 業務開發已更新：在業務開發更新時觸發。

事件案例

下表說明事件在不同情況下的運作方式。下列假設適用於這些案例：

- AWS也是這些案例中的合作夥伴。
- `ResourceSnapshotJob` 是由合作夥伴正確設定。
- 機會更新的欄位也位於資源快照的機會範本上。

身為合作夥伴的您所採取的動作	收到的事件	參與者類型
您建立機會	已建立機會	
您可以使用 `StartEngagementFromOpportunityTask` 來提交機會	機會已更新、已建立業務開發、已建立業務開發資源快	

身為合作夥伴的您所採取的動作	收到的事件	參與者類型
	照、新增業務開發成員、已建立業務開發邀請	
AWS核准您提交的機會	已接受業務開發邀請、已新增業務開發成員、已更新機會、已建立業務開發資源快照	
AWS拒絕您提交的機會	業務開發邀請遭拒、機會已更新、業務開發資源快照已建立	
您可以將AWS Marketplace優惠與機會建立關聯	機會已更新	
將解決方案與機會建立關聯	機會已更新，已建立業務開發資源快照	
您更新機會	機會已更新、業務開發資源快照已建立（如果 ResourceSnapshotJob 已設定，且欄位更新位於範本上，則為選用）	
AWS更新您的機會	機會已更新，已建立業務開發資源快照	
您邀請其他合作夥伴參與	已建立業務開發邀請	寄件者
合作夥伴已接受您加入參與的邀請	已接受業務開發邀請，已新增業務開發成員	寄件者
合作夥伴拒絕您加入參與的邀請	參與邀請遭拒	寄件者
合作夥伴在 15 天內都不會對您發出加入參與的邀請採取行動	參與邀請已過期	寄件者

身為合作夥伴的您所採取的動作	收到的事件	參與者類型
您收到其他合作夥伴的邀請來加入業務開發	已建立業務開發邀請	接收者
您接受其他合作夥伴透過 加入業務開發的邀請 `StartEngagementByAcceptingInvitationTask`	已接受業務開發邀請、已新增業務開發成員、已建立機會、已更新機會、已建立業務開發資源快照	接收者
您拒絕從其他合作夥伴加入參與的邀請	參與邀請遭拒	接收者
您在 15 天內未對邀請加入其他合作夥伴的參與採取行動	參與邀請已過期	接收者
您建立業務開發	已建立參與	寄件者
您更新業務開發	已更新業務開發	寄件者、收件人

範例事件

下列各節提供上一節先前列出的事件範例。

主題

- 已建立機會
- 機會已更新
- 已建立業務開發邀請
- 已接受業務開發邀請
- 參與邀請遭拒
- 參與邀請已過期
- 已新增業務開發成員
- 已建立業務開發資源快照
- 已建立參與
- 已更新業務開發

已建立機會

合作夥伴使用此事件來：

- 通知使用者已建立新的機會。
- 觸發自動化工作流程，例如更新 CRM 系統或通知銷售團隊有關新機會的建立。

下列範例顯示典型的機會建立事件。

```
{
    "version": "1",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "source": "aws.partnercentral-selling",
    "detail-type": "Opportunity Created",
    "time": "2023-04-16T15:23:45Z",
    "region": "us-east-1",
    "account": "123456789012",
    "detail": {
        "schemaVersion": "<version number>",
        "catalog": "<Sandbox | AWS>",
        "opportunity": {
            "identifier": "String"
        }
    }
}
```

機會已更新

合作夥伴使用此事件來：

- 通知使用者現有機會已更新。
- 觸發自動化工作流程，例如更新 CRM 系統或通知銷售團隊。

下列範例顯示典型的 機會更新事件。

```
{
    "version": "1",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "source": "aws.partnercentral-selling",
    "detail-type": "Opportunity Updated",
    "time": "2023-04-16T15:23:45Z",
```

```
    "region": "us-east-1",
    "account": "123456789012",
    "detail": {
        "schemaVersion": "<version number>",
        "catalog": "<Sandbox | AWS>",
        "opportunity": {
             "identifier": "String"
        }
    }
}
```

已建立業務開發邀請

合作夥伴使用此事件來：

- 通知使用者他們收到的新 EngagementInvitation。
- 觸發自動化工作流程以接受或拒絕邀請，例如更新 CRM 系統或通知銷售團隊。

下列範例顯示典型的建立業務開發邀請事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Invitation Created",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-15T12:34:56Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/engi-
v7p8z56whnauo"
    ],
    "detail": {
      "catalog": "AWS",
      "engagementInvitation": {
        "arn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/
engi-v7p8z56whnauo",
        "id": "engi-v7p8z56whnauo",
        "engagementId": "eng-12345678901234",
        "senderAccountId": "string",
        "receiverAccountId": "string",
        "senderCompanyName": "string",
```

```
        "expirationDate": "string",
        "participantType": "Enum", //Sender/Receiver
  "payloadType": "LeadInvitation" //OpportunityInvitation|LeadInvitation
      }
    }
}
```

已接受業務開發邀請

合作夥伴使用此事件來：

- 通知使用者已接受 EngagementInvitation。
- 更新其內部記錄，以反映業務開發中的新合作夥伴。
- 觸發自動化工作流程以同步資料，或通知其他團隊成員有關新的業務開發成員。

下列範例顯示典型的參與邀請接受事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Invitation Accepted",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-16T15:23:45Z",
    "region": "us-east-1",
    "resources": ["arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/
engi-v7p8z56whnauo"],
    "detail": {
        "catalog": "AWS",
        "engagementInvitation": {
            "arn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-
invitation/engi-v7p8z56whnauo",
            "id": "engi-v7p8z56whnauo",
            "engagementId": "eng-12356whnauo",
            "participantType": "Enum", //Sender/Receiver
            "senderAccountId": "string",
            "receiverAccountId": "string",
            "payloadType": "LeadInvitation" //OpportunityInvitation|LeadInvitation
        }
    }
}
```

參與邀請遭拒

合作夥伴使用此事件來：

- 通知使用者 EngagementInvitation 已被拒絕。
- 更新其內部記錄以反映遭拒的邀請。
- 觸發任何必要的工作流程，例如通知銷售團隊拒絕。

下列範例顯示典型的參與邀請遭拒事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Invitation Rejected",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-17T09:48:27Z",
    "region": "us-east-1",
    "resources": ["arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/
engi-v7p8z56whnauo"],
    "detail": {
        "catalog": "AWS",
        "engagementInvitation": {
            "arn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-
invitation/engi-v7p8z56whnauo",
            "id": "engi-v7p8z56whnauo",
            "engagementId": "eng-12356whnauo",
            "participantType": "Enum", //Sender/Receiver
            "senderAccountId": "string",
            "receiverAccountId": "string",
            "payloadType": "LeadInvitation" //OpportunityInvitation|LeadInvitation
        }
    }
}
```

參與邀請已過期

合作夥伴使用此事件來：

- 通知使用者 EngagementInvitation 已過期且無法再對其採取行動。
- 更新其內部記錄以反映過期的邀請。

- 觸發任何必要的工作流程，例如通知銷售團隊過期的邀請。

下列範例顯示典型的 參與邀請已過期事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Invitation Expired",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/engi-
v7p8z56whnauo"
    ],
    "detail": {
      "catalog" : "AWS",
      "engagementInvitation": {
        "arn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/
engi-v7p8z56whnauo",
        "id": "engi-v7p8z56whnauo",
        "engagementId": "eng-12356whnauo",
        "participantType": "Enum", //Sender/Receiver
        "senderAccountId": "string",
        "receiverAccountId": "string",
        "payloadType": "LeadInvitation" //OpportunityInvitation|LeadInvitation
      }
    }
}
```

已新增業務開發成員

合作夥伴使用此事件來：

- 通知使用者參與成員資格的變更。
- 更新其內部記錄以反映新的業務開發成員。
- 觸發任何必要的工作流程，例如通知團隊成員變更。

下列範例顯示典型的 參與成員新增事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Member Added",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-16T15:23:45Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-v7p8z56whnauo"
    ],
    "detail": {
      "catalog" : "AWS",
      "engagement": {
        "id": "eng-v7p8z56whnauo"
      },
      "engagementMember": {
        "accountId": "string",
        "companyName": "string"
      }
    }
}
```

已建立業務開發資源快照

合作夥伴使用此事件來追蹤與參與相關的資源變更，並觸發自動化工作流程，例如更新 CRM 系統或通知銷售團隊。快照範本會決定更新類型：

`OpportunitySummaryView`

- 通知業務開發的所有成員，業務開發已更新。

`AwsOpportunitySummaryFullView`

- 追蹤 AWS 專門針對業務開發中的機會所做的更新。
- 只有機會擁有者可以看到此通知和快照。
- 包括無法透過`OpportunitySummaryView`範本取得的交易規模調整更新。

下列範例顯示典型的 業務開發資源快照已建立事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
```

```
    "detail-type": "Engagement Resource Snapshot Created",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-v7p8z56whnauo"
    ],
    "detail": {
      "catalog" : "AWS",
      "resourceSnapshot": {
        "arn": "arn:aws:partnercentral-selling:us-east-1::catalog/AWS/engagement/eng-
v7p8z56whnauo/resource/Opportunity/o-12312/template/temp-name/snapshot/snapshot-19232",
        "engagementId": "eng-v7p8z56whnauo",
        "resourceType": "Opportunity",
        "resourceId" : "O123211231",
        "createdBy": "123123123123",
        "targetMemberAccounts": ["123123123123", "aws"],
        "resourceSnapshotTemplateName": "temp-name"
      }
    }
}
```

已建立參與

合作夥伴使用此事件來：

- 通知使用者已建立新的業務開發。
- 觸發自動化工作流程，例如更新 CRM 系統，或通知銷售團隊新的互動建立。

下列範例顯示典型的 Engagement Created 事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Created",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-567a1j5hxp35lo"
```

```
    ],
    "detail": {
        "catalog": "AWS",
        "engagement": {
            "engagementArn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/
eng-567a1j5hxp35lo",
            "engagementId": "eng-567a1j5hxp35lo",
            "CreatedAt": "2023-04-18T18:20:15Z",
            "CreatedBy": "aws",
            "ContextTypes": ["Lead"]

        }
    }
}
```

已更新業務開發

合作夥伴使用此事件來：

- 通知使用者現有業務開發已更新。
- 觸發自動化工作流程，例如更新 CRM 系統或通知銷售團隊。

下列範例顯示典型的 Engagement Updated 事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Updated",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-567a1j5hxp35lo"
    ],
    "detail": {
        "catalog": "AWS",
        "engagement": {
            "engagementArn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/
eng-567a1j5hxp35lo",
            "engagementId": "eng-567a1j5hxp35lo",
            "LastModifiedAt": "2023-04-18T18:20:15Z",
```

```
            "LastModifiedBy": "aws",
            "ContextTypes": ["Lead", "CustomerProject"]
        }
    }
}
```

AWS Partner Central Benefits API 的通知

Benefits Service 為 BenefitApplications 和 BenefitAllocations 提供 EventBridge 事件。這些事件可讓客戶建置全面的事件驅動型架構，以回應從初始應用程式到配置和最終履行的整個資源生命週期的變更。

主題

- 完成監控事件的先決條件
- 設定 Amazon EventBridge 以監控事件
- 進一步了解 優勢 API 事件

完成監控事件的先決條件

使用者需要適當的 IAM 許可，才能存取和管理 利益 API 發佈的事件。如需 EventBridge 可用動作、資源和條件索引鍵的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的在 Amazon EventBridge 中使用 IAM 政策條件。 EventBridge 其中一個條件索引鍵是 `events:detail-type`，可用於將許可範圍限定為特定事件類型。

下列範例政策示範如何自訂和限制提議事件的許可。`AllowPutRuleForPartnercentralBenefitsEvents` 陳述式允許建立規則，但僅適用於來源的事件`aws.partnercentral-benefits`。

如需詳細的 IAM 政策範例，請參閱 EventBridge 許可上的 AWS 文件。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralBenefitsEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
```

```
        "StringEquals": {
          "events:source": "aws.partnercentral-benefits"
        }
      }
    }
  ]
}
```

設定 Amazon EventBridge 以監控事件

若要監控效益 API 事件，您可以建立符合您要擷取之事件的 EventBridge 規則。您可以使用AWS 管理主控台或AWS SDKs來建立和管理規則。以下各節說明如何使用兩種方法建立規則。無論您使用何種方法，都必須在美國東部（維吉尼亞北部）us-east-1區域中建立規則。

AWS 管理主控台設定

若要使用 設定 EventBridge 規則AWS 管理主控台，請遵循《Amazon EventBridge 使用者指南》中的建立對 Amazon EventBridge 中的事件做出反應的規則中的步驟。 EventBridge 建立規則時，您必須將事件匯流排設定為預設值，並在美國東部（維吉尼亞北部）us-east-1區域中建立規則。

以下是事件規則的範例：

```
{
    "source": ["aws.partnercentral-benefits"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS開發套件設定

您可以使用AWS SDKs以程式設計方式建立和管理 EventBridge 規則。如需詳細資訊，請參閱《Amazon EventBridge API 參考》中的 PutRule。

下列範例使用適用於 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
```

```
    Name='MyBenefitApplicationCreatedRule',
    EventPattern=
    '{
        "source": ["aws.partnercentral-benefits"],
        "detail-type": ["Benefit Application Created"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
print('Rule ARN:', response['RuleArn'])
```

進一步了解 優勢 API 事件

下列各節說明 API 事件類型的優點、觸發它們的案例，以及事件範例。

Event types (事件類型)

以下是事件類型及其觸發條件。

利益應用程式

- 已建立利益應用程式：建立利益應用程式時
- 利益應用程式審核中：提交利益應用程式時，或應用程式從必要動作轉換回審核中時，會觸發此事件。
- 需要利益應用程式動作：當內部檢閱者更新應用程式狀態，表示合作夥伴需要提供其他資訊、釐清或修改，才能繼續檢閱時，就會發生此事件。
- 利益應用程式已拒絕：當內部檢閱者更新應用程式狀態以指出應用程式不符合利益條件或要求，且已拒絕時，就會發生此事件
- 利益應用程式已核准：當內部審核者更新應用程式狀態，表示應用程式已核准利益分配時，就會發生此事件。
- 利益應用程式階段已變更：當內部審核者在業務審核、AWS審核、財務審核等審核程序期間更新應用程式的階段時，就會發生此事件。

利益分配

- 利益分配已啟動：當利益分配由內部服務建立或更新回已啟動時。
- 利益分配已停用：當內部檢閱者更新配置狀態或利益分配自動過期時，會發生此事件。
- 利益分配已實現：當內部檢閱者更新分配狀態以表示配置已完全使用時，就會發生此事件。

範例事件

下列各節提供上一節先前列出的事件範例。

主題

- 已建立利益應用程式
- 利益應用程式審核中
- 需要利益應用程式動作
- 利益應用程式遭拒
- 利益應用程式已核准
- 利益應用程式階段已變更
- 已啟用利益分配
- 利益分配已停用
- 已實現利益分配

已建立利益應用程式

觸發內容

當透過 Benefits Service 成功建立新的 Benefit Application 時，Benefit Application Created 事件會發佈到 Amazon EventBridge。當 CreateBenefitApplication API 呼叫成功完成，表示合作夥伴已啟動特定利益的請求時，就會發生此事件。

此事件代表利益應用程式生命週期的開始，並在新應用程式提交至其帳戶時立即通知客戶。

收件人

擁有的帳戶會收到利益分配的事件。

範例

```
{
  "id": "7gh88765-k0jh-d08g-i292-2ff1796m030n",
  "detail-type": "Benefit Application Created",
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-02-10T15:45:00Z",
  "region": "us-east-1",
  "resources": [
```

```
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-application/
benappl-12345abcdef123"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitApplication": {
      "id": "benappl-12345abcdef123",
      "fulfillmentTypes": ["CREDITS"],
      "status": "PENDING_SUBMISSION",
      "revision": "hh7e8400-e29b-41d4-a716-446655440012"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

利益應用程式審核中

觸發內容

當利益應用程式透過 Benefits Service 轉換為 IN_REVIEW 狀態時，利益應用程式審核中事件會發佈至 Amazon EventBridge。此事件會在下列情況下發生：

- 合作夥伴透過 SubmitBenefitApplication API 呼叫提交其應用程式
- 在合作夥伴解決意見回饋後，應用程式會從 ACTION_REQUIRED 轉換回 IN_REVIEW
- 應用程式會重新叫用，然後重新提交

此事件表示應用程式已進入正式審核程序，且正由AWS內部團隊進行評估。

收件人

擁有的帳戶會收到利益分配的事件。

範例

```
{
  "id": "8hi99876-l1ki-e19h-j303-3gg2807n141o",
  "detail-type": "Benefit Application In Review",
  "source": "aws.partnercentral-benefits",
```

```
  "account": "123456789012",
  "time": "2025-02-10T16:00:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-application/
benappl-12345abcdef123"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitApplication": {
      "id": "benappl-12345abcdef123",
      "fulfillmentTypes": ["CREDITS"],
      "status": "IN_REVIEW",
      "stage": "BUSINESS_REVIEW",
      "revision": "ii8f9511-f30c-52e5-b827-557766551123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

需要利益應用程式動作

觸發內容

當利益應用程式透過 Benefits Service 轉換為 ACTION_REQUIRED 狀態時，Benefit Application Action Required 事件會發佈至 Amazon EventBridge。當內部檢閱者透過 UpdateBenefitApplicationInternal API 更新應用程式狀態，表示合作夥伴需要提供其他資訊、釐清或修改，才能繼續檢閱時，就會發生此事件。

此事件代表應用程式生命週期中需要合作夥伴介入才能在審核程序中將應用程式向前移動的關鍵點。

收件人

擁有的帳戶將會收到利益分配的事件

範例

```
{
  "id": "9ij00987-m2lj-f20i-k414-4hh3918o252p",
  "detail-type": "Benefit Application Action Required",
```

```
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-02-12T10:30:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-application/
benappl-12345abcdef123"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitApplication": {
      "id": "benappl-12345abcdef123",
      "fulfillmentTypes": ["CREDITS"],
      "status": "ACTION_REQUIRED",
      "stage": "BUSINESS_REVIEW",
      "revision": "jj9g0622-g41d-63f6-c938-668877662234"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

利益應用程式遭拒

觸發內容

當 Benefit Application 透過 Benefits Service 轉換為 REJECTED 狀態時，Benefit Application 拒絕事件會發佈至 Amazon EventBridge。當內部檢閱者透過 UpdateBenefitApplicationInternal API 更新應用程式狀態，以指出應用程式不符合利益條件或要求且已遭拒時，就會發生此事件。

此事件代表應用程式生命週期中的結束狀態，其中應用程式已正式拒絕，不會繼續受益配置。

收件人

擁有的帳戶會收到利益分配的事件。

範例

```
{
  "id": "0kl11098-n3mk-g31j-l525-5ii4029p363q",
  "detail-type": "Benefit Application Rejected",
```

```
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-02-15T14:20:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-application/
benappl-12345abcdef123"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitApplication": {
      "id": "benappl-12345abcdef123",
      "fulfillmentTypes": ["CREDITS"],
      "status": "REJECTED",
      "revision": "kk0h1733-h52e-74g7-d049-779988773345"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

利益應用程式已核准

觸發內容

當利益應用程式透過 Benefits Service 轉換為核准狀態時，利益應用程式核准事件會發佈至 Amazon EventBridge。當內部檢閱者透過 UpdateBenefitApplicationInternal API 更新應用程式狀態，表示應用程式符合所有利益條件和要求，並已核准利益分配時，就會發生此事件。

此事件代表應用程式檢閱程序成功完成，通常會觸發對應利益分配的建立。

收件人

擁有的帳戶會收到利益分配的事件。

範例

```
{
  "id": "1lm22109-o4nl-h42k-m636-6jj5130q474r",
  "detail-type": "Benefit Application Approved",
  "source": "aws.partnercentral-benefits",
```

```
  "account": "123456789012",
  "time": "2025-02-14T16:45:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-application/
benappl-12345abcdef123"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitApplication": {
      "id": "benappl-12345abcdef123",
      "fulfillmentTypes": ["CREDITS"],
      "status": "APPROVED",
      "revision": "ll1i2844-i63f-85h8-e150-880099884456"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

利益應用程式階段已變更

觸發內容

透過 Benefits Service 更新 Benefit Application 的審核階段時，Benefit Application Stage Changed 事件會發佈至 Amazon EventBridge。當內部檢閱者將應用程式的階段更新為其中一個可能的列舉值 (FINANCE_REVIEW、BUSINESS_REVIEW、TECH_REVIEW 和AWS_REVIEW) 時，就會發生此事件。

與狀態變更不同，階段變更是唯讀資訊更新，可提供內部審核工作流程進展的可見性，而不需要合作夥伴動作。

收件人

擁有的帳戶會收到利益分配的事件。

範例

```
{
  "id": "2mn33210-p5om-i53l-n747-7kk6241r585s",
```

```
  "detail-type": "Benefit Application Stage Changed",
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-02-13T11:15:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-application/
benappl-12345abcdef123"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitApplication": {
      "id": "benappl-12345abcdef123",
      "fulfillmentTypes": ["CREDITS"],
      "status": "IN_REVIEW",
      "stage": "FINANCIAL_REVIEW",
      "revision": "mm2j3955-j74g-96i9-f261-991100995567"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

已啟用利益分配

觸發內容

當利益分配透過 利益服務轉換為 ACTIVE 狀態時，利益分配啟用事件會發佈到 Amazon EventBridge。當內部檢閱者透過 UpdateBenefitAllocationInternal API 更新配置狀態，以重新啟用先前非作用中的配置時，就會發生此事件。

當暫時停用的配置 （由於過期、政策變更或管理原因） 還原為作用中狀態時，通常會發生此事件，以便再次獲得合作夥伴使用率的好處。

收件人

屬於合作夥伴AWS的帳戶會收到訊息。換言之，擁有資源AWS的帳戶。

範例事件

```
{
```

```
  "id": "3no44321-q6pn-j64m-o858-8ll7352s696t",
  "detail-type": "Benefit Allocation Activated",
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-03-01T08:15:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-allocation/benalloc-
abc123def456"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitAllocation": {
      "id": "benalloc-abc123def456",
      "fulfillmentType": "CREDITS",
      "status": "ACTIVE"
    },
    "benefitApplication": {
      "id": "benappl-12345abcdef123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```


利益分配已停用

觸發內容

當利益分配透過 Benefits Service 轉換為 INACTIVE 狀態時，利益分配停用事件會發佈至 Amazon EventBridge。當內部檢閱者透過 UpdateBenefitAllocationInternal API 更新配置狀態，或當利益分配自動過期時，就會發生此事件。

此事件通常發生在暫停配置時（由於過期、政策變更、合規問題或管理原因），使得在重新啟用之前無法使用合作夥伴使用利益。

收件人

屬於合作夥伴AWS的帳戶會收到訊息。換言之，擁有資源AWS的帳戶。

範例事件

```
{
  "id": "4op55432-r7qo-k75n-p969-9mm8463t707u",
  "detail-type": "Benefit Allocation Inactivated",
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-06-30T23:59:59Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-allocation/benalloc-
abc123def456"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitAllocation": {
      "id": "benalloc-abc123def456",
      "fulfillmentType": "CREDITS",
      "status": "INACTIVE"
    },
    "benefitApplication": {
      "id": "benappl-12345abcdef123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

已實現利益分配

觸發內容

當利益分配透過 Benefits Service 轉換為 FULFILLED 狀態時，利益分配履行事件會發佈到 Amazon EventBridge。當內部檢閱者透過 UpdateBenefitAllocationInternal API 更新配置狀態，表示配置已完全使用，或滿足利益條款時，就會發生此事件。

此事件代表利益生命週期的完成，表示合作夥伴已成功利用配置的利益，或配置已達成自然結論。

收件人

屬於合作夥伴AWS的帳戶會收到訊息。換言之，擁有資源AWS的帳戶。

範例事件

```
{
  "id": "5pq66543-s8rp-l86o-q070-0nn9574u818v",
  "detail-type": "Benefit Allocation Fulfilled",
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-12-15T17:30:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-allocation/benalloc-
abc123def456"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitAllocation": {
      "id": "benalloc-abc123def456",
      "fulfillmentType": "CREDITS",
      "status": "FULFILLED"
    },
    "benefitApplication": {
      "id": "benappl-12345abcdef123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

AWS Partner Central Channel API 的通知

頻道 API 的事件提供有關頻道相關活動狀態變更或詳細資訊的即時通知。這些事件有助於讓您的系統與AWS Partner Central 保持同步，並有助於確保及時回應和更新。

先決條件

您需要適當的 IAM 許可，才能從AWS Partner Central Channel API 存取和管理事件。如需 EventBridge 可用動作、資源和條件索引鍵的相關資訊，請參閱《Amazon EventBridge 使用者指南》中的在 Amazon EventBridge 中使用 IAM 政策條件。 EventBridge 其中一個條件索引鍵是 `events:detail-type`，您可以使用它來限制特定事件類型的許可範圍。

下列範例政策示範如何自訂事件的許可和範圍。AllowPutRuleForPartnercentralChannelEvents 陳述式允許建立規則，但僅適用於來源的事件aws.partnercentral-channel。

如需詳細的 IAM 政策範例，請參閱 EventBridge 許可上的 AWS 文件。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralChannelEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-channel"
        }
      }
    }
  ]
}
```

設定 Amazon EventBridge 以監控事件

若要監控頻道 API 事件，您可以建立符合您要擷取之事件的 EventBridge 規則。您可以使用AWS 管理主控台或AWS SDKs來建立和管理規則。以下各節說明如何使用兩種方法建立規則。無論您使用何種方法，都必須在美國東部（維吉尼亞北部）us-east-1區域中建立規則。

AWS 管理主控台設定

若要使用 設定 EventBridge 規則AWS 管理主控台，請遵循《Amazon EventBridge 使用者指南》中的建立對 Amazon EventBridge 中的事件做出反應的規則中的步驟。 EventBridge 建立規則時，您必須將事件匯流排設定為預設值，並在美國東部（維吉尼亞北部）us-east-1區域中建立規則。

以下是事件規則的範例：

```
{
    "source": ["aws.partnercentral-channel"],
    "detail": {
        "catalog": ["AWS"]
    }
```

```
}
```

AWS開發套件設定

您可以使用AWS SDKs以程式設計方式建立和管理 EventBridge 規則。如需詳細資訊，請參閱《Amazon EventBridge API 參考》中的 PutRule。

下列範例使用適用於 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
    Name='ChannelHandshakeCreatedRule',
    EventPattern=
    '{
        "source": ["aws.partnercentral-channel"],
        "detail-type": ["Channel Handshake Created"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
print('Rule ARN:', response['RuleArn'])
```

進一步了解頻道 API 事件

下列各節說明頻道 API 事件類型、觸發它們的案例，以及事件範例。

Event types (事件類型)

以下是事件類型及其觸發條件。

- 頻道交握已建立：在建立新的頻道交握時觸發。
- 頻道交握已接受：接受新頻道交握時觸發。
- 頻道交握遭拒：當新頻道交握遭拒時觸發。

頻道交握已建立

下列範例顯示適用於不同交握類型的典型頻道交握建立事件。

啟動服務期間交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Created",
    "source": "aws.partnercentral-channel",
    "account": "789789789789",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-abc123def456g"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "START_SERVICE_PERIOD",
        "id": "ch-abc123def456g",
        "receiverAccountId": "789789789789",
        "ownerAccountId": "123123123123",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "startServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

撤銷服務期間交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Created",
    "source": "aws.partnercentral-channel",
    "account": "789789789789",
    "time": "2025-02-01T00:00:00Z",
```

```
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-def456ghi789j"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "REVOKE_SERVICE_PERIOD",
        "id": "ch-def456ghi789j",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "789789789789",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "revokeServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

程式管理帳戶交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Created",
    "source": "aws.partnercentral-channel",
    "account": "456456456456",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-ghi789jkl012m"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
```

```
        "catalog": "AWS",
        "associatedResourceIdentifier": "pma-abc123def456g",
        "handshakeType": "PROGRAM_MANAGEMENT_ACCOUNT",
        "id": "ch-ghi789jkl012m",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "456456456456",
        "senderAccountId": "123123123123",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "programManagementAccountHandshakeDetail": {
                "program": "SOLUTION_PROVIDER"
            }
        }
    }
}
```

頻道交握已接受

下列範例顯示不同交握類型的典型頻道交握接受事件。

啟動服務期間交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Accepted",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-abc123def456g"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "START_SERVICE_PERIOD",
        "id": "ch-abc123def456g",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "789789789789",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
```

```
        "handshakeDetail": {
            "startServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

撤銷服務期間交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Accepted",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-def456ghi789j"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "REVOKE_SERVICE_PERIOD",
        "id": "ch-def456ghi789j",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "789789789789",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "revokeServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
```

```
            }
        }
    }
}
```

程式管理帳戶交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Accepted",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-ghi789jkl012m"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "pma-abc123def456g",
        "handshakeType": "PROGRAM_MANAGEMENT_ACCOUNT",
        "id": "ch-ghi789jkl012m",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "456456456456",
        "senderAccountId": "123123123123",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "programManagementAccountHandshakeDetail": {
                "program": "SOLUTION_PROVIDER"
            }
        }
    }
}
```

頻道交握遭拒

下列範例顯示不同交握類型的典型頻道交握遭拒事件。

啟動服務期間交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Rejected",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-abc123def456g"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "START_SERVICE_PERIOD",
        "id": "ch-abc123def456g",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "789789789789",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "startServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

撤銷服務期間交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Rejected",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
```

```
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-def456ghi789j"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "REVOKE_SERVICE_PERIOD",
        "id": "ch-def456ghi789j",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "789789789789",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "revokeServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

程式管理帳戶交握：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Rejected",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-ghi789jkl012m"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
```

```
        "catalog": "AWS",
        "associatedResourceIdentifier": "pma-abc123def456g",
        "handshakeType": "PROGRAM_MANAGEMENT_ACCOUNT",
        "id": "ch-ghi789jkl012m",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "456456456456",
        "senderAccountId": "123123123123",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "programManagementAccountHandshakeDetail": {
                "program": "SOLUTION_PROVIDER"
            }
        }
    }
}
```

在沙盒中測試

合作夥伴可以使用沙盒在安全且隔離的環境中測試和驗證其AWS Partner Central API 互動，確保在將解決方案提升至生產環境之前能順暢操作。AWS會提供動態沙盒給AWS Partner Central API 使用者，其會傳回類似生產環境的回應。AWS不會提供沙盒環境的使用者介面。因此，合作夥伴需要依賴程式設計回應來測試其解決方案。

存取沙盒環境

合作夥伴只要AWS 帳戶將其連結至 Partner Central 帳戶，即可存取測試環境。如需詳細資訊，請參閱將AWS您的帳戶連結至AWS Partner Central 帳戶。每個請求都包含 `catalog` 參數，可決定資料環境。當 `catalog` 設為 時`AWS`，它會參考生產資料，當它設為 時`Sandbox`，它會參考沙盒資料。

沙盒環境的重要詳細資訊

1. 資料重新整理：每年一次，AWS重新整理沙盒環境中的資料（通常在年初）。在此重新整理之後，您可能會遺失測試環境中的部分資料。
2. 測試範圍：沙盒環境通常用於功能測試，而非測試可擴展性或效能。

主題

- 在沙盒中測試AWS Partner Central Account API
- 在沙盒中測試AWS Partner Central Selling API

- 在沙盒中測試AWS Partner Central Benefits API
- 在沙盒中測試AWS Partner Central Channel API
- 在沙盒中測試AWS Partner Central Revenue Measurement API

在沙盒中測試AWS Partner Central Account API

如何使用沙盒

若要使用沙盒，請完成下列步驟：

1. 建立 IAM 角色：

   在與您的AWS Partner Central 帳戶AWS 帳戶連結的 中建立 IAM 角色。
2. 指派政策：

   將下列政策指派給 IAM 角色。如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountManagement",
      "Effect": "Allow",
      "Action": [
        "partnercentral:AcceptConnectionInvitation",
        "partnercentral:AssociateAwsTrainingCertificationEmailDomain",
        "partnercentral:CancelConnection",
        "partnercentral:CancelConnectionInvitation",
        "partnercentral:CancelProfileUpdateTask",
        "partnercentral:CreateConnectionInvitation",
        "partnercentral:CreatePartner",
        "partnercentral:DisassociateAwsTrainingCertificationEmailDomain",
        "partnercentral:GetAllianceLeadContact",
        "partnercentral:GetConnection",
        "partnercentral:GetConnectionInvitation",
        "partnercentral:GetConnectionPreferences",
        "partnercentral:GetPartner",
        "partnercentral:GetProfileUpdateTask",
        "partnercentral:GetProfileVisibility",
  "partnercentral:GetVerification",
        "partnercentral:ListConnectionInvitations",
```

```
        "partnercentral:ListConnections",
        "partnercentral:ListPartners",
        "partnercentral:PutAllianceLeadContact",
        "partnercentral:PutProfileVisibility",
        "partnercentral:RejectConnectionInvitation",
        "partnercentral:SendEmailVerificationCode",
        "partnercentral:StartProfileUpdateTask",
  "partnercentral:StartVerification",
        "partnercentral:UpdateConnectionPreferences"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "Sandbox"
          ]
        }
      }
    },
    {
      "Sid": "TaggingAccess",
      "Effect": "Allow",
      "Action": [
        "partnercentral:TagResource",
        "partnercentral:UntagResource",
        "partnercentral:ListTagsForResource"
      ],
      "Resource": [
        "arn:aws:partnercentral:*:*:catalog/Sandbox/partner/*"
      ],
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "Sandbox"
          ]
        }
      }
    }
  ]
}
```

3. 使用 IAM 角色登入資料：

   在您的解決方案中使用此 IAM 角色的登入資料（私密金鑰和存取金鑰）來執行 API 動作。

測試AWS動作

在測試階段，通常需要模擬AWS動作。此模擬可讓合作夥伴徹底測試其與AWS服務整合的完整end-to-end流程。每個請求都包含目錄參數，可決定資料環境。

模擬合作夥伴的建立

若要模擬合作夥伴的建立，請在 CreatePartner動作的承載中，將 包含在承載"catalog": "Sandbox"中。

例如：

```
{
  "ClientToken": "client-generated-uuid",
  "Catalog": "Sandbox",
  "LegalName": "Your Name",
  "PrimarySolutionType": "SOLUTION_TYPE",
  "AllianceLeadContact": {
    "FirstName": "Example First Name",
    "LastName": "Example Last Name",
    "BusinessTitle": "Example Title",
    "Email": "example@domain.com"
  },
  "EmailVerificationCode": "123456"
}
```

注意：在沙盒中，電子郵件驗證已停用。您可以使用任何六位數的 EmailVerificationCode 進行測試。此外，請不要在請求中使用「標籤」。

其他測試備註

1. 若要測試其他動作，請在承載"catalog": "Sandbox"中設定 。
2. 若要在沙盒中測試合作夥伴帳戶連線，您必須在沙盒目錄中建立合作夥伴之後執行StartProfileUpdateTask動作。
3. 若要移至生產環境，請將目錄值從 Sandbox變更為 AWS。

在沙盒環境中測試事件

合作夥伴可以取用沙盒環境的事件，以協助測試事件型實作。AWS 帳戶使用規則在相同的 中設定 EventBridge，透過在事件詳細資訊catalog: Sandbox中指定 來接聽沙盒事件。如需詳細資訊，請參閱AWS合作夥伴中央帳戶 API 的通知。

範例事件規則：

```
{
  "source": ["aws.partnercentral-account"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

指定 `catalog` 為 的事件規則`Sandbox`只會比對來自沙盒的事件，這些事件是由您在沙盒環境中執行的動作所產生。

在沙盒中測試AWS Partner Central Selling API

如何使用沙盒

1. 建立 IAM 角色：

   在與您的AWS Partner Central 帳戶AWS 帳戶連結的 中建立 IAM 角色。

2. 指派政策：

   將下列政策指派給 IAM 角色。如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "partnercentral:CreateOpportunity",
                "partnercentral:UpdateOpportunity",
                "partnercentral:ListOpportunities",
                "partnercentral:GetOpportunity",
                "partnercentral:GetAwsOpportunitySummary",
                "partnercentral:ListSolutions",
                "partnercentral:AssociateOpportunity",
                "partnercentral:DisassociateOpportunity",
                "partnercentral:AssignOpportunity",
                "partnercentral:SubmitOpportunity",
                "partnercentral:AcceptEngagementInvitation",
```

```
                "partnercentral:CreateEngagementInvitation",
                "partnercentral:RejectEngagementInvitation",
                "partnercentral:GetEngagementInvitation",
                "partnercentral:ListEngagementInvitations",
                "partnercentral:StartEngagementFromOpportunityTask",
                "partnercentral:StartEngagementByAcceptingInvitationTask",
                "partnercentral:CreateResourceSnapshotJob",
                "partnercentral:StartResourceSnapshotJob",
                "partnercentral:CreateEngagement"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": "Sandbox"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:ListEntities",
                "aws-marketplace:DescribeEntity"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:SearchAgreements",
                "aws-marketplace:DescribeAgreement"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws-marketplace:PartyType": "Proposer"
                }
            }
        }
    ]
}
```

3. 使用 IAM 角色登入資料：

   在您的解決方案中使用此 IAM 角色的登入資料（私密金鑰和存取金鑰）來執行 API 動作。

測試AWS動作

在測試階段，通常需要模擬AWS動作。此模擬可讓合作夥伴徹底測試其與AWS服務整合的完整end-to-end流程。

模擬AWS原始機會的建立

若要模擬AWS建立原始機會，請在 CreateOpportunity動作的承載中包含 "Catalog": "Sandbox"和 "Origin": "AWS Referral"。

例如：

```
{
    "Catalog": "Sandbox",
    "Origin": "AWS Referral",
    "OpportunityIdentifier": "O123456",
    "Title": "Test Opportunity",
    ...
}
```

模擬合作夥伴來源機會的更新

若要在合作夥伴發起的機會上模擬更新或其他動作，請在承載“Action Required”中使用 UpdateOpportunity動作搭配 "Catalog": "Sandbox"和 Lifecycle.ReviewStatus: “Approved”或 。

如果您要從 GetOpportunity動作取得承載來執行更新，請確定您將 ID 變更為 Identifier，並移除下列欄位：

- CreatedDate
- OpportunityTeam
- RelatedEntityIdentifiers

例如：

```
{
    "Catalog": "Sandbox",
    "Identifier": "O123456",
    "Title": "Updated Test Opportunity",
    "Lifecycle": {
```

```
        "ReviewStatus": "Approved"
    }
    ...
}
```

在沙盒環境中測試事件

合作夥伴可以從沙盒環境中取用機會事件，以協助測試事件型實作。AWS 帳戶使用規則在相同的 中設定 EventBridge，透過在事件詳細資訊`catalog: sandbox`中指定 來接聽沙盒事件。如需詳細資訊，請參閱AWS Partner Central Selling API 的通知。

範例事件規則：

```
{
    "source": ["aws.partnercentral-selling"],
    "detail": {
        "catalog": ["Sandbox"]
    }
}
```

指定 `catalog` 為 的事件規則`Sandbox`只會比對來自沙盒的事件，這些事件是由您在沙盒環境中執行的動作所產生。

其他測試備註

1. 測試`AssociateOpportunity`動作：
   a. 使用預設解決方案`"S-1234567"`來測試 `AssociateOpportunity`動作。
   b. 若要測試 Marketplace 優惠，請使用您帳戶中的實際優惠 ID。
2. 若要移至生產環境，請將`catalog`值從 `Sandbox`變更為 `AWS`。

取得說明

如果您遇到將 CRM 與 整合的挑戰AWS，或者如果您需要測試此處未涵蓋的特定案例，請透過下列步驟提出案例來尋求支援：

1. 使用您的 AWS Partner Network 登入資料登入 Partner Central。AWS
2. 在 Partner Central 支援中心上，選擇`Open New Case`記錄新案例。如下所示完成欄位：
   a. 支援案例類型：`AWS Partner Central`。

b. 有關 的問題：Partner Central Tools or ACE leads and opportunities。

c. 取得特定：選取最適合的 CRM 整合案例類型。

d. 主旨：包含請求的簡短描述。

e. 描述：提供問題、問題、錯誤和故障診斷步驟的詳細說明。

f. 附件：適用時包含日誌和螢幕擷取畫面。

在沙盒中測試AWS Partner Central Benefits API

如何使用沙盒

若要使用沙盒，請完成下列步驟：

1. 建立 IAM 角色：

   在與您的AWS Partner Central 帳戶AWS 帳戶連結的 中建立 IAM 角色。

2. 指派政策：

   將下列政策指派給 IAM 角色。如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

```
{
  "Version": "2012-10-17",
    "Statement": [{
            "Sid": "BenefitsManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListBenefits",
                "partnercentral:GetBenefit",
                "partnercentral:CreateBenefitApplication",
                "partnercentral:AmendBenefitApplication",
                "partnercentral:UpdateBenefitApplication",
                "partnercentral:SubmitBenefitApplication",
                "partnercentral:GetBenefitApplication",
                "partnercentral:CancelBenefitApplication",
                "partnercentral:RecallBenefitApplication",
                "partnercentral:ListBenefitApplications",
                "partnercentral:AssociateBenefitApplicationResource",
                "partnercentral:DisasssociateBenefitApplicationResource",
                "partnercentral:ListBenefitAllocations",
                "partnercentral:GetBenefitAllocation",
                "partnercentral:TagResource",
```

```
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "AWSPartnerOpportunityAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetAwsOpportunitySummary",
                "partnercentral:GetOpportunity",
                "partnercentral:ListOpportunities"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "ListingAWSMarketplaceEntities",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:ListEntities"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSMarketplaceOffersAccess",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:DescribeEntity"
            ],
```

```
            "Resource": [
                "arn:aws:aws-marketplace:::AWSMarketplace/Offer/*"
            ]
        },
        {
            "Sid": "S3PutObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": ["arn:aws:s3:::aws-partner-central-marketplace-ephemeral-
writeonly-files/*"]
        },
        {
            "Sid": "TaggingAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:TagResource",
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": [
                "arn:aws:partnercentral:us-east-1:*:catalog/Sandbox/benefit-
application/*",
                "arn:aws:partnercentral:us-east-1:*:catalog/Sandbox/benefit-
allocation/*"
            ],
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        }
    ]
}
```

3. 使用 IAM 角色登入資料：

   在您的解決方案中使用此 IAM 角色的登入資料（私密金鑰和存取金鑰）來執行 API 動作。

測試AWS動作

在測試階段，通常需要模擬AWS動作。此模擬可讓合作夥伴徹底測試其與AWS服務整合的完整end-to-end流程。每個請求都包含目錄參數，可決定資料環境。

模擬利益應用程式的建立

若要模擬利益應用程式的建立，請在 CreateBenefitApplication動作的承載中，將 包含在承載"catalog": "Sandbox"中。

例如：

```
{
  "Catalog": "Sandbox",
  "ClientToken": "String",
  "BenefitIdentifier": "String",
  "FulfillmentTypes": ["String"],
  "BenefitApplicationDetails": JsonDocument,
  "Name": "String",
  "Description": "String",
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ],
  "PartnerContacts": [
    {
      "BusinessTitle": "string",
      "Email": "string",
      "FirstName": "string",
      "LastName": "string",
      "Phone": "string"
    }
  ],
  "FileDetails": [
    {
      "FileURI": "String",
      "BusinessUseCase": "String"
    }
  ],
  "AssociatedResources": [
    {
```

```
      "ResourceType": "BENEFIT_ALLOCATION | OPPORTUNITY",
      "ResourceArn": "ARN"
    }
  ]
}
```

其他測試備註

1. 若要測試其他動作，請在承載"catalog": "Sandbox"中設定 。
2. 沙盒目錄和 AWS 目錄中的效益 IDs 不同。使用 進行 GetBenefit API 呼叫"catalog": "Sandbox"，以在沙盒中取得利益識別符。
3. 沙盒目錄和 AWS 目錄中的效益分配不同。
4. 若要移至生產環境，請將目錄值從 Sandbox變更為 AWS。

在沙盒環境中測試事件

合作夥伴可以取用沙盒環境的事件，以協助測試事件型實作。AWS 帳戶使用規則在相同的 中設定 EventBridge，透過在事件詳細資訊catalog: Sandbox中指定 來接聽沙盒事件。如需詳細資訊，請參閱AWS Partner Central Benefits API 的通知。

範例事件規則：

```
{
  "source": ["aws.partnercentral-benefits"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

指定catalog為 的事件規則Sandbox只會比對來自沙盒的事件，這些事件是由您在沙盒環境中執行的動作所產生。

在沙盒中測試AWS Partner Central Channel API

沙盒中的帳戶不符合頻道利益的資格，而且無法針對所有計劃需求進行驗證。

如何使用沙盒

若要使用沙盒，請完成下列步驟：

1. 建立 IAM 角色：

   在與您的AWS Partner Central 帳戶AWS 帳戶連結的 中建立 IAM 角色。

2. 指派政策：

   將下列政策指派給 IAM 角色。如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ChannelManagement",
      "Effect": "Allow",
      "Action": [
        "partnercentral:CreateProgramManagementAccount",
        "partnercentral:UpdateProgramManagementAccount",
        "partnercentral:DeleteProgramManagementAccount",
        "partnercentral:ListProgramManagementAccounts",
        "partnercentral:GetProgramManagementAccount",
        "partnercentral:CreateRelationship",
        "partnercentral:UpdateRelationship",
        "partnercentral:DeleteRelationship",
        "partnercentral:GetRelationship",
        "partnercentral:ListRelationships",
        "partnercentral:CreateChannelHandshake",
        "partnercentral:AcceptChannelHandshake",
        "partnercentral:RejectChannelHandshake",
        "partnercentral:CancelChannelHandshake",
        "partnercentral:ListChannelHandshakes"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "Sandbox"
          ]
        }
      }
    },
    {
      "Sid": "TaggingAccess",
      "Effect": "Allow",
      "Action": [
```

```
        "partnercentral:TagResource",
        "partnercentral:UntagResource",
        "partnercentral:ListTagsForResource"
      ],
      "Resource": [
        "arn:aws:partnercentral:*:*:catalog/Sandbox/program-management-account/*",
        "arn:aws:partnercentral:*:*:catalog/Sandbox/channel-handshake/*"
      ],
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "Sandbox"
          ]
        }
      }
    }
  ]
}
```

3. 使用 IAM 角色登入資料：

   在您的解決方案中使用此 IAM 角色的登入資料（私密金鑰和存取金鑰）來執行 API 動作。

測試AWS動作

在測試階段，通常需要模擬AWS動作。此模擬可讓合作夥伴徹底測試其與AWS服務整合的完整end-to-end流程。每個請求都包含目錄參數，決定資料環境。

模擬建立程式管理帳戶

若要模擬建立程式管理帳戶，請在 `CreateProgramManagementAccount`動作的承載中，將 包含在承載`"catalog": "Sandbox"`中。

例如：

```
{
  "catalog": "Sandbox",
  "accountId": "123456789012",
  "displayName": "Test PMA Name",
  "clientToken": "123abc456def789ghi",
  "program": "SOLUTION_PROVIDER",
  "tags": [
    {
```

```
      "key": "testkey",
      "value": "testvalue"
    }
  ]
}
```

模擬關係的更新

若要模擬關係的更新，請在承載"catalog": "Sandbox"中使用 UpdateRelationship動作搭配 。

例如：

```
{
  "catalog": "Sandbox",
  "displayName": "Updated Test PMA",
  "identifier": "rs-abc123def456g",
  "programManagementAccountIdentifier": "pma-123abc456def7"
}
```

其他測試備註

1. 若要測試其他動作，請在承載"catalog": "Sandbox"中設定 。
2. 若要移至生產環境，請將目錄值從 Sandbox變更為 AWS。

在沙盒環境中測試事件

合作夥伴可以使用沙盒環境的事件，以協助測試事件型實作。AWS 帳戶使用規則在相同的 中設定 EventBridge，透過在事件詳細資訊catalog: Sandbox中指定 來接聽沙盒事件。如需詳細資訊，請參閱AWS Partner Central Channel API 的通知。

範例事件規則：

```
{
  "source": ["aws.partnercentral-channel"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

指定 `catalog` 為 的事件規則Sandbox只會比對來自沙盒的事件，這些事件是由您在沙盒環境中執行的動作所產生。

取得說明

如果您在測試時遇到挑戰，或者如果您需要測試此處未涵蓋的特定案例，請透過下列步驟提出案例來尋求支援：

1. 使用您的 AWS Partner Network 登入資料登入 Partner Central。AWS
2. 在 Partner Central 支援中心上，選擇`Open New Case`記錄新案例。如下所示完成欄位：
   a. 支援案例類型： `Channel Program`
   b. 有關下列項目的問題： `General Program Inquiry`
   c. 取得特定：選取最適合的頻道案例類型。
   d. 主旨：包含請求的簡短描述。
   e. 描述：提供問題、問題、錯誤和故障診斷步驟的詳細說明。
   f. 附件：適用時包含日誌和螢幕擷取畫面。

在沙盒中測試AWS Partner Central Revenue Measurement API

如何使用沙盒

若要使用沙盒，請完成下列步驟：

1. 建立 IAM 角色：

   在與您的AWS Partner Central 帳戶AWS 帳戶連結的 中建立 IAM 角色。
2. 指派政策：

   將下列政策指派給 IAM 角色。如需更多資訊，請參閱《新增和移除 IAM 身分許可》。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RevenueMeasurementCreation",
            "Effect": "Allow",
            "Action": [
                "partnercentral:CreateRevenueAttribution",
                "partnercentral:CreateMarketplaceRevenueShare",
```

```
                "partnercentral:CreateMarketplaceRevenueShareAllocation"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueMeasurementListing",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListRevenueAttributions",
                "partnercentral:ListRevenueAttributionAllocations",
                "partnercentral:ListMarketplaceRevenueShares",
                "partnercentral:ListMarketplaceRevenueShareAllocations"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueAttributionManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetRevenueAttribution",
                "partnercentral:UpdateRevenueAttribution",
                "partnercentral:GetRevenueAttributionAllocation",
                "partnercentral:StartRevenueAttributionAllocationsTask",
                "partnercentral:GetRevenueAttributionAllocationsTask"
            ],
            "Resource": "arn:aws:partnercentral:*:*:catalog/Sandbox/revenue-
attribution/*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
```

```
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "MarketplaceRevenueShareManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetMarketplaceRevenueShare",
                "partnercentral:GetMarketplaceRevenueShareAllocation",
                "partnercentral:UpdateMarketplaceRevenueShareAllocation"
            ],
            "Resource": "arn:aws:partnercentral:*:*:catalog/Sandbox/marketplace-
revenue-share/*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "TaggingAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:TagResource",
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": [
                "arn:aws:partnercentral:*:*:catalog/Sandbox/revenue-attribution/*",
                "arn:aws:partnercentral:*:*:catalog/Sandbox/marketplace-revenue-
share/*"
            ],
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
```

```
        {
            "Sid": "OpportunityAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListOpportunities",
                "partnercentral:GetOpportunity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "ListingAWSMarketplaceEntities",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:ListEntities"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSMarketplaceEntityAccess",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:DescribeEntity"
            ],
            "Resource": [
                "arn:aws:aws-marketplace:*:*:AWSMarketplace*/Offer/*"
            ]
        }
    ]
}
```

3. 使用 IAM 角色登入資料：

   在您的解決方案中使用此 IAM 角色的登入資料（私密金鑰和存取金鑰）來執行 API 動作。

測試AWS動作

在測試階段，通常需要模擬AWS動作。此模擬可讓合作夥伴徹底測試其與AWS服務整合的完整end-to-end流程。每個請求都包含目錄參數，可決定資料環境。

模擬營收歸因 ID 的建立

若要模擬建立營收歸因 ID，請在 CreateRevenueAttribution動作的承載中，將 包含在承載"Catalog": "Sandbox"中。

例如：

```
{
  "Catalog": "Sandbox",
  "ClientToken": "String",
  "Name": "String",
  "Description": "String",
  "TenancyModel": "MULTI_TENANT | SINGLE_TENANT",
  "ProductIdentifier": "String",
  "Tags": [
    {
      "Key": "String",
      "Value": "String"
    }
  ]
}
```

其他測試備註

1. 若要測試其他動作，請在承載"Catalog": "Sandbox"中設定 。
2. 若要移至生產環境，請將目錄值從 Sandbox變更為 AWS。

Partner Central AWS SDKs的程式碼範例

下列程式碼範例示範如何使用 Partner Central 搭配AWS軟體開發套件 (SDK)。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他AWS 服務組合來完成特定任務的程式碼範例。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- Partner Central AWS SDKs的基本範例
  - Partner Central AWS SDKs的動作
    - AssignOpportunity 搭配AWS SDK 使用
    - AssociateOpportunity 搭配AWS SDK 使用
    - CreateOpportunity 搭配AWS SDK 使用
    - DisassociateOpportunity 搭配AWS SDK 使用
    - GetAwsOpportunitySummary 搭配AWS SDK 使用
    - GetEngagementInvitation 搭配AWS SDK 使用
    - GetOpportunity 搭配AWS SDK 使用
    - ListEngagementInvitations 搭配AWS SDK 使用
    - ListOpportunities 搭配AWS SDK 使用
    - ListSolutions 搭配AWS SDK 使用
    - RejectEngagementInvitation 搭配AWS SDK 使用
    - StartEngagementByAcceptingInvitationTask 搭配AWS SDK 使用
    - StartEngagementFromOpportunityTask 搭配AWS SDK 使用
    - UpdateOpportunity 搭配AWS SDK 使用
- Partner Central AWS SDKs的案例
  - 更新機會的關聯實體

Partner Central AWS SDKs的基本範例

下列程式碼範例示範如何搭配AWS SDKs 使用AWS Partner Central 的基本概念。

範例

- Partner Central AWS SDKs的動作
  - AssignOpportunity 搭配AWS SDK 使用
  - AssociateOpportunity 搭配AWS SDK 使用
  - CreateOpportunity 搭配AWS SDK 使用
  - DisassociateOpportunity 搭配AWS SDK 使用
  - GetAwsOpportunitySummary 搭配AWS SDK 使用
  - GetEngagementInvitation 搭配AWS SDK 使用
  - GetOpportunity 搭配AWS SDK 使用
  - ListEngagementInvitations 搭配AWS SDK 使用
  - ListOpportunities 搭配AWS SDK 使用
  - ListSolutions 搭配AWS SDK 使用
  - RejectEngagementInvitation 搭配AWS SDK 使用
  - StartEngagementByAcceptingInvitationTask 搭配AWS SDK 使用
  - StartEngagementFromOpportunityTask 搭配AWS SDK 使用
  - UpdateOpportunity 搭配AWS SDK 使用

Partner Central AWS SDKs的動作

下列程式碼範例示範如何使用AWS SDKs 執行個別的 Partner Central 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 Partner Central API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 Partner Central AWS SDKs的案例 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《AWS Partner Central API 參考》。

範例

- AssignOpportunity 搭配AWS SDK 使用
- AssociateOpportunity 搭配AWS SDK 使用

- CreateOpportunity 搭配AWS SDK 使用
- DisassociateOpportunity 搭配AWS SDK 使用
- GetAwsOpportunitySummary 搭配AWS SDK 使用
- GetEngagementInvitation 搭配AWS SDK 使用
- GetOpportunity 搭配AWS SDK 使用
- ListEngagementInvitations 搭配AWS SDK 使用
- ListOpportunities 搭配AWS SDK 使用
- ListSolutions 搭配AWS SDK 使用
- RejectEngagementInvitation 搭配AWS SDK 使用
- StartEngagementByAcceptingInvitationTask 搭配AWS SDK 使用
- StartEngagementFromOpportunityTask 搭配AWS SDK 使用
- UpdateOpportunity 搭配AWS SDK 使用

`AssignOpportunity` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `AssignOpportunity`。

Java

適用於 Java 2.x 的 SDK

將現有的機會重新指派給另一名使用者。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.AssignOpportunityRequest;
```

```
import
 software.amazon.awssdk.services.partnercentralselling.model.AssignOpportunityResponse;
import
 software.amazon.awssdk.services.partnercentralselling.model.AssigneeContact;

/*
Purpose
PC-API-07 Assigning a new owner
*/

public class AssignOpportunity {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     String assigneeFirstName = "John";

     String assigneeLastName = "Doe";

     String assigneeEmail = "test@test.com";

     String businessTitle = "PartnerAccountManager";

     AssignOpportunityResponse response = getResponse(opportunityId,
 assigneeFirstName, assigneeLastName, assigneeEmail, businessTitle);

     ReferenceCodesUtils.formatOutput(response);
    }

 static AssignOpportunityResponse getResponse(String opportunityId, String
 assigneeFirstName, String assigneeLastName, String assigneeEmail, String
 businessTitle) {

  AssignOpportunityRequest assignOpportunityRequest =
 AssignOpportunityRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
```

```
            .identifier(opportunityId)
            .assignee(AssigneeContact.builder()
              .firstName(assigneeFirstName)
              .lastName(assigneeLastName)
              .email(assigneeEmail)
              .businessTitle(businessTitle)
              .build())
            .build();

        AssignOpportunityResponse response =
 client.assignOpportunity(assignOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 AssignOpportunity。

Python

適用於 Python 的 SDK (Boto3)

將現有的機會重新指派給另一名使用者。

```
#!/usr/bin/env python

"""
Purpose
PC-API-07 Assigning a new owner
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
```

```
        region_name='us-east-1'
)

def assign_opportunity(identifier):
    assign_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "Identifier": identifier,
        "Assignee": {
            "BusinessTitle": "OpportunityOwner",
            "Email": "test@test.com",
            "FirstName": "John",
            "LastName": "Doe"
        }
    }
    try:
        # Perform an API call
        response =
 partner_central_client.assign_opportunity(**assign_opportunity_request)
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    identifier = "O4236468"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Assigning a new owner to an opportunity.")
    print("-" * 88)

    helper.pretty_print_datetime(assign_opportunity(identifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 AssignOpportunity。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`AssociateOpportunity` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `AssociateOpportunity`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 更新機會的關聯實體

Java

適用於 Java 2.x 的 SDK

建立機會與各種相關實體之間的正式關聯。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.AssociateOpportunityRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.AssociateOpportunityResponse

/*
Purpose
PC-API -11 Associating a product
PC-API -12 Associating a solution
PC-API -13 Associating an offer
entity_type = Solutions | AWSProducts | AWSMarketplaceOffers
*/
```

```
public class AssociateOpportunity {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     String entityType = "Solutions";

     String entityIdentifier = "S-0000000";

     AssociateOpportunityResponse response = getResponse(opportunityId,
 entityType, entityIdentifier );

     ReferenceCodesUtils.formatOutput(response);
    }

static AssociateOpportunityResponse getResponse(String opportunityId, String
 entityType, String entityIdentifier) {

        AssociateOpportunityRequest associateOpportunityRequest =
 AssociateOpportunityRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .opportunityIdentifier(opportunityId)
          .relatedEntityType(entityType)
          .relatedEntityIdentifier(entityIdentifier)
          .build();

        AssociateOpportunityResponse response =
 client.associateOpportunity(associateOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 AssociateOpportunity。

Python

適用於 Python 的 SDK (Boto3)

建立機會與各種相關實體之間的正式關聯。

```
#!/usr/bin/env python

"""
Purpose
PC-API -11 Associating a product
PC-API -12 Associating a solution
PC-API -13 Associating an offer
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def associate_opportunity(entity_type, entity_identifier, opportunityIdentifier):
    associate_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "OpportunityIdentifier" : opportunityIdentifier,
        "RelatedEntityType" : entity_type,
        "RelatedEntityIdentifier" : entity_identifier
    }
    try:
        # Perform an API call
        response =
 partner_central_client.associate_opportunity(**associate_opportunity_request)
```

```
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    #entity_type = Solutions | AWSProducts | AWSMarketplaceOffers
    entity_type = "Solutions"
    entity_identifier = "S-0059717"
    opportunityIdentifier = "O5465588"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Associate Opportunity.")
    print("-" * 88)

    helper.pretty_print_datetime(associate_opportunity(entity_type,
 entity_identifier, opportunityIdentifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 AssociateOpportunity。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateOpportunity` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `CreateOpportunity`。

.NET

適用於 .NET 的 SDK

建立機會。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// PDX-License-Identifier: Apache-2.0
```

```
using System;
using Newtonsoft.Json;
using Amazon;
using Amazon.Runtime;
using Amazon.PartnerCentralSelling;
using Amazon.PartnerCentralSelling.Model;

namespace AWSExample
{
    class Program
    {
        static readonly string catalogToUse = "AWS";
        static async Task Main(string[] args)
        {
            // Initialize credentials from .aws/credentials file
            var credentials = new
 Amazon.Runtime.CredentialManagement.SharedCredentialsFile();
            if (credentials.TryGetProfile("default", out var profile))
            {
                AWSCredentials awsCredentials =
 profile.GetAWSCredentials(credentials);

                var client = new
 AmazonPartnerCentralSellingClient(awsCredentials);

                var request = new CreateOpportunityRequest
                {
                    Catalog = catalogToUse,
                    Origin = "Partner Referral",
                    Customer = new Customer
                    {
                        Account = new Account
                        {
                            Address = new Address
                            {
                                CountryCode = "US",
                                PostalCode = "99502",
                                StateOrRegion = "Alaska"
                            },
                            CompanyName = "TestCompanyName",
                            Duns = "123456789",
                            WebsiteUrl = "www.test.io",
                            Industry = "Automotive"
```

```
                        },
                        Contacts = new List<Contact>
                        {
                            new Contact
                            {
                                Email = "test@test.io",
                                FirstName = "John  ",
                                LastName = "Doe",
                                Phone = "+14444444444",
                                BusinessTitle = "test title"
                            }
                        }
                    },
                    LifeCycle = new LifeCycle
                    {
                        ReviewStatus = "Submitted",
                        TargetCloseDate = "2024-12-30"
                    },
                    Marketing = new Marketing
                    {
                        Source = "None"
                    },
                    OpportunityType = "Net New Business",
                    PrimaryNeedsFromAws = new List<string> { "Co-Sell -
Architectural Validation" },
                    Project = new Project
                    {
                        Title = "Moin Test UUID",
                        CustomerBusinessProblem = "Sandbox is not working as
expected",
                        CustomerUseCase = "AI Machine Learning and Analytics",
                        DeliveryModels = new List<string> { "SaaS or PaaS" },
                        ExpectedCustomerSpend = new List<ExpectedCustomerSpend>
                        {
                            new ExpectedCustomerSpend
                            {
                                Amount = "2000.0",
                                CurrencyCode = "USD",
                                Frequency = "Monthly",
                                TargetCompany = "Ibexlabs"
                            }
                        },
                        SalesActivities = new List<string> { "Initialized
discussions with customer" }
```

```
                    }
                };

                try
                {
                    var response = await client.CreateOpportunityAsync(request);
                    Console.WriteLine(response.HttpStatusCode);
                    string formattedJson = JsonConvert.SerializeObject(response,
 Formatting.Indented);
                    Console.WriteLine(formattedJson);
                }
                catch (ValidationException ex)
                {
                    Console.WriteLine("Validation error: " + ex.Message);
                }
                catch (AmazonPartnerCentralSellingException e)
                {
                    Console.WriteLine("Failed:");
                    Console.WriteLine(e.RequestId);
                    Console.WriteLine(e.ErrorCode);
                    Console.WriteLine(e.Message);
                }
            }
            else
            {
                Console.WriteLine("Profile not found.");
            }
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 CreateOpportunity。

Java

適用於 Java 2.x 的 SDK

建立機會。

```
package org.example;

import java.time.Instant;
```

```
import java.util.ArrayList;
import java.util.List;

import static org.example.utils.Constants.*;

import org.example.entity.Root;
import org.example.utils.ReferenceCodesUtils;
import org.example.utils.StringSerializer;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import software.amazon.awssdk.services.partnercentralselling.model.Account;
import software.amazon.awssdk.services.partnercentralselling.model.Address;
import software.amazon.awssdk.services.partnercentralselling.model.Contact;
import
 software.amazon.awssdk.services.partnercentralselling.model.CreateOpportunityRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.CreateOpportunityResponse;
import software.amazon.awssdk.services.partnercentralselling.model.Customer;
import
 software.amazon.awssdk.services.partnercentralselling.model.ExpectedCustomerSpend;
import software.amazon.awssdk.services.partnercentralselling.model.LifeCycle;
import software.amazon.awssdk.services.partnercentralselling.model.Marketing;
import software.amazon.awssdk.services.partnercentralselling.model.MonetaryValue;
import
 software.amazon.awssdk.services.partnercentralselling.model.NextStepsHistory;
import software.amazon.awssdk.services.partnercentralselling.model.Project;
import
 software.amazon.awssdk.services.partnercentralselling.model.SoftwareRevenue;

import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.ToNumberPolicy;

public class CreateOpportunity {

 static final Gson GSON = new GsonBuilder()
   .setObjectToNumberStrategy(ToNumberPolicy.LAZILY_PARSED_NUMBER)
   .registerTypeAdapter(String.class, new StringSerializer())
   .create();
```

```
static PartnerCentralSellingClient client =
PartnerCentralSellingClient.builder()
           .region(Region.US_EAST_1)
           .credentialsProvider(DefaultCredentialsProvider.create())
           .httpClient(ApacheHttpClient.builder().build())
           .build();

public static void main(String[] args) {

 String inputFile = "CreateOpportunity2.json";

 if (args.length > 0)
  inputFile = args[0];

 CreateOpportunityResponse response = createOpportunity(inputFile);

 client.close();
}

static CreateOpportunityResponse createOpportunity(String inputFile) {

 String inputString = ReferenceCodesUtils.readInputFileToString(inputFile);

 Root root = GSON.fromJson(inputString, Root.class);

 List<NextStepsHistory> nextStepsHistories = new ArrayList<NextStepsHistory>();
 if ( root.lifeCycle != null && root.lifeCycle.nextStepsHistories != null) {
  for (org.example.entity.NextStepsHistory nextStepsHistoryJson :
root.lifeCycle.nextStepsHistories) {
   NextStepsHistory nextStepsHistory = NextStepsHistory.builder()
     .time(Instant.parse(nextStepsHistoryJson.time))
     .value(nextStepsHistoryJson.value)
                 .build();
   nextStepsHistories.add(nextStepsHistory);
  }
 }

 LifeCycle lifeCycle = null;
 if ( root.lifeCycle != null ) {
  lifeCycle = LifeCycle.builder()
   .closedLostReason(root.lifeCycle.closedLostReason)
   .nextSteps(root.lifeCycle.nextSteps)
   .nextStepsHistory(nextStepsHistories)
   .reviewComments(root.lifeCycle.reviewComments)
```

```
    .reviewStatus(root.lifeCycle.reviewStatus)
    .reviewStatusReason(root.lifeCycle.reviewStatusReason)
    .stage(root.lifeCycle.stage)
    .targetCloseDate(root.lifeCycle.targetCloseDate)
    .build();
  }

  Marketing marketing = null;
  if ( root.marketing != null ) {
   marketing = Marketing.builder()
     .awsFundingUsed(root.marketing.awsFundingUsed)
     .campaignName(root.marketing.campaignName)
     .channels(root.marketing.channels)
     .source(root.marketing.source)
     .useCases(root.marketing.useCases)
     .build();

  }

  Address address = null;
  if ( root.customer != null && root.customer.account != null &&
root.customer.account.address != null ) {
   address = Address.builder()
    .city(root.customer.account.address.city)
                .postalCode(root.customer.account.address.postalCode)
                .stateOrRegion(root.customer.account.address.stateOrRegion)
                .countryCode(root.customer.account.address.countryCode)
                .streetAddress(root.customer.account.address.streetAddress)
                .build();
  }

  Account account = null;
  if ( root.customer != null && root.customer.account!= null) {
   account = Account.builder()
              .address(address)
              .awsAccountId(root.customer.account.awsAccountId)
                 .duns(root.customer.account.duns)
                 .industry(root.customer.account.industry)
                 .otherIndustry(root.customer.account.otherIndustry)
                 .companyName(root.customer.account.companyName)
                 .websiteUrl(root.customer.account.websiteUrl)
                 .build();
  }
```

```
 List<Contact> contacts = new ArrayList<Contact>();
 if ( root.customer != null && root.customer.contacts != null) {
  for (org.example.entity.Contact jsonContact : root.customer.contacts) {
   Contact contact = Contact.builder()
                .email(jsonContact.email)
                .firstName(jsonContact.firstName)
                .lastName(jsonContact.lastName)
                .phone(jsonContact.phone)
                .businessTitle(jsonContact.businessTitle)
                .build();
   contacts.add(contact);
  }
 }

 Customer customer = Customer.builder()
   .account(account)
   .contacts(contacts)
   .build();

 Contact oportunityTeamContact = null;
 if (root.opportunityTeam != null && root.opportunityTeam.get(0) != null ) {
  oportunityTeamContact = Contact.builder()
              .firstName(root.opportunityTeam.get(0).firstName)
              .lastName(root.opportunityTeam.get(0).lastName)
              .email(root.opportunityTeam.get(0).email)
              .phone(root.opportunityTeam.get(0).phone)
              .businessTitle(root.opportunityTeam.get(0).businessTitle)
              .build();
 }

 List<ExpectedCustomerSpend> expectedCustomerSpends = new
ArrayList<ExpectedCustomerSpend>();
 if ( root.project != null && root.project.expectedCustomerSpend != null) {
  for (org.example.entity.ExpectedCustomerSpend expectedCustomerSpendJson :
root.project.expectedCustomerSpend) {
   ExpectedCustomerSpend expectedCustomerSpend = null;
   expectedCustomerSpend = ExpectedCustomerSpend.builder()
     .amount(expectedCustomerSpendJson.amount)
     .currencyCode(expectedCustomerSpendJson.currencyCode)
     .frequency(expectedCustomerSpendJson.frequency)
     .targetCompany(expectedCustomerSpendJson.targetCompany)
     .build();
   expectedCustomerSpends.add(expectedCustomerSpend);
  }
```

```
        }

        Project project = null;
        if ( root.project != null) {
         project = Project.builder()
                .title(root.project.title)
                .customerBusinessProblem(root.project.customerBusinessProblem)
                .customerUseCase(root.project.customerUseCase)
                .deliveryModels(root.project.deliveryModels)
                .expectedCustomerSpend(expectedCustomerSpends)
                .salesActivities(root.project.salesActivities)
                .competitorName(root.project.competitorName)
                .otherSolutionDescription(root.project.otherSolutionDescription)
                .build();
        }

        SoftwareRevenue softwareRevenue = null;
        if ( root.softwareRevenue != null) {
         MonetaryValue monetaryValue = null;
         if ( root.softwareRevenue.value != null) {
          monetaryValue = MonetaryValue.builder()
             .amount(root.softwareRevenue.value.amount)
             .currencyCode(root.softwareRevenue.value.currencyCode)
             .build();
         }
         softwareRevenue = SoftwareRevenue.builder()
           .deliveryModel(root.softwareRevenue.deliveryModel)
           .effectiveDate(root.softwareRevenue.effectiveDate)
           .expirationDate(root.softwareRevenue.expirationDate)
           .value(monetaryValue)
           .build();
        }

 // Building the Actual CreateOpportunity Request
 CreateOpportunityRequest createOpportunityRequest =
CreateOpportunityRequest.builder()
   .catalog(CATALOG_TO_USE)
   .clientToken(root.clientToken)
   .primaryNeedsFromAwsWithStrings(root.primaryNeedsFromAws)
   .opportunityType(root.opportunityType)
   .lifeCycle(lifeCycle)
   .marketing(marketing)
   .nationalSecurity(root.nationalSecurity)
   .origin(root.origin)
```

```
    .customer(customer)
    .project(project)
    .partnerOpportunityIdentifier(root.partnerOpportunityIdentifier)
    .opportunityTeam(oportunityTeamContact)
    .softwareRevenue(softwareRevenue)
    .build();

  CreateOpportunityResponse response =
 client.createOpportunity(createOpportunityRequest);
  System.out.println("Successfully created: " + response);

  return response;
    }

}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 CreateOpportunity。

Python

適用於 Python 的 SDK (Boto3)

建立機會。

```
#!/usr/bin/env python
import boto3
import logging
import sys
import os
sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
import utils.helpers as helper
import utils.stringify_details as sd
from botocore.client import ClientError
from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

def create_opportunity(partner_central_client):
    create_opportunity_request = helper.remove_nulls(sd.stringify_json("src/
create_opportunity/createOpportunity.json"))
    try:
```

```
        # Perform an API call
        response =
 partner_central_client.create_opportunity(**create_opportunity_request)

        helper.pretty_print_datetime(response)

        # Retrieve the opportunity details
        get_response = partner_central_client.get_opportunity(
            Identifier=response["Id"],
            Catalog=CATALOG_TO_USE
        )
        helper.pretty_print_datetime(get_response)
        return response
    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Create Opportunity.")
    print("-" * 88)

    partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
    )

    create_opportunity(partner_central_client)

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 CreateOpportunity。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DisassociateOpportunity` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `DisassociateOpportunity`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 更新機會的關聯實體

Java

適用於 Java 2.x 的 SDK

移除機會與相關實體之間的現有關聯。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.DisassociateOpportunityReque
import
 software.amazon.awssdk.services.partnercentralselling.model.DisassociateOpportunityRespo

/*
Purpose
PC-API -14 Removing a Solution
PC-API -15 Removing an offer
PC-API -16 Removing a product
entity_type = Solutions | AWSProducts | AWSMarketplaceOffers
*/

public class DisassociateOpportunity {
```

```
 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     String entityType = "Solutions";

     String entityIdentifier = "S-0000000";

     DisassociateOpportunityResponse response = getResponse(opportunityId,
 entityType, entityIdentifier );

     ReferenceCodesUtils.formatOutput(response);
    }

 static DisassociateOpportunityResponse getResponse(String opportunityId, String
 entityType, String entityIdentifier) {

  DisassociateOpportunityRequest disassociateOpportunityRequest =
 DisassociateOpportunityRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .opportunityIdentifier(opportunityId)
          .relatedEntityType(entityType)
          .relatedEntityIdentifier(entityIdentifier)
          .build();

        DisassociateOpportunityResponse response =
 client.disassociateOpportunity(disassociateOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DisassociateOpportunity。

Python

適用於 Python 的 SDK (Boto3)

移除機會與相關實體之間的現有關聯。

```
#!/usr/bin/env python

"""
Purpose
PC-API -14 Removing a Solution
PC-API -15 Removing an offer
PC-API -16 Removing a product
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def disassociate_opportunity(entity_type, entity_identifier,
 opportunityIdentifier):
    disassociate_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "OpportunityIdentifier" : opportunityIdentifier,
        "RelatedEntityType" : entity_type,
        "RelatedEntityIdentifier" : entity_identifier
    }
    try:
        # Perform an API call
        response =
 partner_central_client.disassociate_opportunity(**disassociate_opportunity_request)
        return response

    except ClientError as err:
```

```
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    #entity_type = Solutions | AWSProducts | AWSMarketplaceOffers
    entity_type = "Solutions"
    entity_identifier = "S-0049999"
    opportunityIdentifier = "O4397574"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Get updated Opportunity.")
    print("-" * 88)

    helper.pretty_print_datetime(disassociate_opportunity(entity_type,
 entity_identifier, opportunityIdentifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 DisassociateOpportunity。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetAwsOpportunitySummary` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `GetAwsOpportunitySummary`。

Java

適用於 Java 2.x 的 SDK

擷取AWS機會的摘要。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
```

```
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.GetAwsOpportunitySummaryRequ
import
 software.amazon.awssdk.services.partnercentralselling.model.GetAwsOpportunitySummaryResp

/*
 * Purpose
 * PC-API-25 Retrieves a summary of an AWS Opportunity.
 */

public class GetAwsOpportunitySummary {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     GetAwsOpportunitySummaryResponse response = getResponse(opportunityId);

     ReferenceCodesUtils.formatOutput(response);
    }

 public static GetAwsOpportunitySummaryResponse getResponse(String opportunityId)
 {

  GetAwsOpportunitySummaryRequest getOpportunityRequest =
 GetAwsOpportunitySummaryRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .relatedOpportunityIdentifier(opportunityId)
          .build();
```

```
  GetAwsOpportunitySummaryResponse response =
 client.getAwsOpportunitySummary(getOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 GetAwsOpportunitySummary。

Python

適用於 Python 的 SDK (Boto3)

擷取AWS機會的摘要。

```
#!/usr/bin/env python

"""
Purpose
PC-API-25 Retrieves a summary of an AWS Opportunity.
 LifeCycle.ReviewStatus=Approved
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def get_opportunity(identifier):
    get_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "RelatedOpportunityIdentifier": identifier
```

```
    }
    try:
        # Perform an API call
        response =
 partner_central_client.get_aws_opportunity_summary(**get_opportunity_request)
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    identifier = "O5465588"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Get AWS Opportunity summary.")
    print("-" * 88)

    helper.pretty_print_datetime(get_opportunity(identifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 GetAwsOpportunitySummary。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetEngagementInvitation` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `GetEngagementInvitation`。

Java

適用於 Java 2.x 的 SDK

擷取AWS與合作夥伴共用的參與邀請詳細資訊。

```
package org.example;
```

```
import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.GetEngagementInvitationReque
import
 software.amazon.awssdk.services.partnercentralselling.model.GetEngagementInvitationRespo

/*
 * Purpose
 * PC-API-22 Get engagement invitation opportunity
 */

public class GetEngagementInvitation {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     GetEngagementInvitationResponse response = getResponse(opportunityId);

     ReferenceCodesUtils.formatOutput(response);
    }

 static GetEngagementInvitationResponse getResponse(String opportunityId) {

  GetEngagementInvitationRequest getOpportunityRequest =
 GetEngagementInvitationRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
```

```
            .identifier(opportunityId)
            .build();

   GetEngagementInvitationResponse response =
  client.getEngagementInvitation(getOpportunityRequest);

          return response;
  }
 }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 GetEngagementInvitation。

Python

適用於 Python 的 SDK (Boto3)

擷取AWS與合作夥伴共用的參與邀請詳細資訊。

```
#!/usr/bin/env python

"""
Purpose
PC-API-22  GetOpportunityEngagementInvitation - Retrieves details of a specific
 engagement invitation.
This operation allows partners to view the invitation and its associated
 information,
such as the customer, project, and lifecycle details.
"""
import json
import logging
import boto3
import utils.helpers as helper

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
```

```
)

def get_opportunity_engagement_invitation(identifier):
    get_opportunity_engagement_invitation_request ={
        "Catalog": CATALOG_TO_USE,
     "Identifier": identifier
    }
    try:
        # Perform an API call
        response =
 partner_central_client.get_engagement_invitation(**get_opportunity_engagement_invitation
        return response

    except Exception as err:
        # Catch all client exceptions
        print(json.dumps(err.response))

def usage_demo():
    identifier = "arn:aws:partnercentral-selling:us-east-1:aws:catalog/Sandbox/
engagement-invitation/engi-0000000IS0Qga"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Given the ARN identifier, retrieve details of Opportunity Engagement
 Invitation.")
    print("-" * 88)


 helper.pretty_print_datetime(get_opportunity_engagement_invitation(identifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 GetEngagementInvitation。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetOpportunity 搭配AWS SDK 使用

下列程式碼範例示範如何使用 GetOpportunity。

.NET

適用於 .NET 的 SDK

取得機會。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// PDX-License-Identifier: Apache-2.0

using System;
using Newtonsoft.Json;
using Amazon;
using Amazon.Runtime;
using Amazon.PartnerCentralSelling;
using Amazon.PartnerCentralSelling.Model;

namespace AWSExample
{
    class Program
    {
        static readonly string catalogToUse = "AWS";
        static readonly string identifier = "O1111111";
        static async Task Main(string[] args)
        {
            // Initialize credentials from .aws/credentials file
            var credentials = new
 Amazon.Runtime.CredentialManagement.SharedCredentialsFile();
            if (credentials.TryGetProfile("default", out var profile))
            {
                AWSCredentials awsCredentials =
 profile.GetAWSCredentials(credentials);

                var client = new
 AmazonPartnerCentralSellingClient(awsCredentials);

                var request = new GetOpportunityRequest
                {
                    Catalog = catalogToUse,
                    Identifier = identifier
                };
```

```
                try {
                    var response = await client.GetOpportunityAsync(request);
                    Console.WriteLine(response.HttpStatusCode);
                    string formattedJson = JsonConvert.SerializeObject(response,
 Formatting.Indented);
                    Console.WriteLine(formattedJson);
                } catch(ValidationException ex) {
                    Console.WriteLine("Validation error: " + ex.Message);
                } catch (AmazonPartnerCentralSellingException e) {
                    Console.WriteLine("Failed:");
                    Console.WriteLine(e.RequestId);
                    Console.WriteLine(e.ErrorCode);
                    Console.WriteLine(e.Message);
                }
            }
            else
            {
                Console.WriteLine("Profile not found.");
            }
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 GetOpportunity。

Go

SDK for Go V2

取得機會。

```
package main

import (
 "context"
 "encoding/json"
 "fmt"
 "log"

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

```
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/partnercentralselling"
)

func main() {
 config, err := config.LoadDefaultConfig(context.TODO())

 if err != nil {
  log.Fatal(err)
 }

 config.Region = "us-east-1"

 client := partnercentralselling.NewFromConfig(config)

 output, err := client.GetOpportunity(context.TODO(),
 &partnercentralselling.GetOpportunityInput{
  Identifier: aws.String("O1111111"),
  Catalog:    aws.String("AWS"),
 })

 if err != nil {
  log.Fatal(err)
 }
 log.Println("printing opportuniy...\n")

 jsonOutput, err := json.MarshalIndent(output, "", "    ")

 fmt.Println(string(jsonOutput))
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 GetOpportunity。

Java

適用於 Java 2.x 的 SDK

取得機會。

```
package org.example;

import static org.example.utils.Constants.*;
```

```
import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.GetOpportunityRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.GetOpportunityResponse;

/*
 * Purpose
 * PC-API-08 Get updated Opportunity
 */

public class GetOpportunity {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     GetOpportunityResponse response = getResponse(opportunityId);

     ReferenceCodesUtils.formatOutput(response);
    }

 public static GetOpportunityResponse getResponse(String opportunityId) {

        GetOpportunityRequest getOpportunityRequest =
 GetOpportunityRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .identifier(opportunityId)
          .build();
```

```
        GetOpportunityResponse response =
 client.getOpportunity(getOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 GetOpportunity。

Python

適用於 Python 的 SDK (Boto3)

取得機會。

```
#!/usr/bin/env python

"""
Purpose
PC-API -08 Get updated Opportunity given opportunity id
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def get_opportunity(identifier):
    get_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "Identifier": identifier
    }
```

```
    try:
        # Perform an API call
        response =
 partner_central_client.get_opportunity(**get_opportunity_request)
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    identifier = "O5465588"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Get updated Opportunity.")
    print("-" * 88)

    helper.pretty_print_datetime(get_opportunity(identifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 GetOpportunity。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListEngagementInvitations` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `ListEngagementInvitations`。

Java

適用於 Java 2.x 的 SDK

擷取傳送給合作夥伴的參與邀請清單。

```
package org.example;
```

```
import java.util.ArrayList;
import java.util.List;

import org.example.utils.ReferenceCodesUtils;
import static org.example.utils.Constants.*;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.ListEngagementInvitationsReq
import
 software.amazon.awssdk.services.partnercentralselling.model.ListEngagementInvitationsRes
import
 software.amazon.awssdk.services.partnercentralselling.model.ParticipantType;
import
 software.amazon.awssdk.services.partnercentralselling.model.EngagementInvitationSummary;

public class ListEngagementInvitations {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     List<EngagementInvitationSummary> opportunitySummaries = getResponse();
        ReferenceCodesUtils.formatOutput(opportunitySummaries);
    }

    static List<EngagementInvitationSummary> getResponse() {

  List<EngagementInvitationSummary> opportunitySummaries = new
 ArrayList<EngagementInvitationSummary>();

  ListEngagementInvitationsRequest listOpportunityRequest =
 ListEngagementInvitationsRequest.builder()
                .catalog(CATALOG_TO_USE)
                .participantType(ParticipantType.RECEIVER)
```

```
            .maxResults(5)
            .build();

  ListEngagementInvitationsResponse response =
 client.listEngagementInvitations(listOpportunityRequest);

     opportunitySummaries.addAll(response.engagementInvitationSummaries());

     client.close();

        return opportunitySummaries;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 ListEngagementInvitations。

Python

適用於 Python 的 SDK (Boto3)

擷取傳送給合作夥伴的參與邀請清單。

```
#!/usr/bin/env python

"""
Purpose
PC-API-21 ListEngagementInvitations - Retrieves a list of engagement invitations
 based on specified criteria.
This operation allows partners to view all invitations to engagement.
"""
import json
import logging
import boto3
import utils.helpers as helper

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
```

```
        service_name=serviceName,
        region_name='us-east-1'
)

def list_engagement_invitations():
    list_engagement_invitations_request ={
        "Catalog": CATALOG_TO_USE,
        "MaxResults": 20
    }
    try:
        # Perform an API call
        response =
 partner_central_client.list_engagement_invitations(**list_engagement_invitations_request
        return response

    except Exception as err:
        # Catch all client exceptions
        print(json.dumps(err.response))

def usage_demo():
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Retrieve list of Engagement Invitations.")
    print("-" * 88)

    helper.pretty_print_datetime(list_engagement_invitations())

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 ListEngagementInvitations。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListOpportunities` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `ListOpportunities`。

.NET

適用於 .NET 的 SDK

列出機會。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// PDX-License-Identifier: Apache-2.0

using System;
using Newtonsoft.Json;
using Amazon;
using Amazon.Runtime;
using Amazon.PartnerCentralSelling;
using Amazon.PartnerCentralSelling.Model;

namespace AWSExample
{
    class Program
    {
        static readonly string catalogToUse = "Sandbox";
        static async Task Main(string[] args)
        {
            // Initialize credentials from .aws/credentials file
            var credentials = new
 Amazon.Runtime.CredentialManagement.SharedCredentialsFile();
            if (credentials.TryGetProfile("default", out var profile))
            {
                AWSCredentials awsCredentials =
 profile.GetAWSCredentials(credentials);

                //var config = new AmazonPartnerCentralSellingConfig()
                //{
                //    ServiceURL = "https://partnercentral-selling.us-
east-1.api.aws",
                //};
                //var client = new
 AmazonPartnerCentralSellingClient(awsCredentials, config);
                var client = new
 AmazonPartnerCentralSellingClient(awsCredentials);
                var request = new ListOpportunitiesRequest
                {
                    Catalog = catalogToUse,
                    MaxResults = 2
```

```
                };

                try {
                    var response = await client.ListOpportunitiesAsync(request);
                    Console.WriteLine(response.HttpStatusCode);
                    foreach (var opportunity in response.OpportunitySummaries)
                    {
                        Console.WriteLine("Opportunity id: " + opportunity.Id);
                    }
                    string formattedJson =
 JsonConvert.SerializeObject(response.OpportunitySummaries, Formatting.Indented);
                    Console.WriteLine(formattedJson);
                } catch(ValidationException ex) {
                    Console.WriteLine("Validation error: " + ex.Message);
                } catch (AmazonPartnerCentralSellingException e) {
                    Console.WriteLine("Failed:");
                    Console.WriteLine(e.RequestId);
                    Console.WriteLine(e.ErrorCode);
                    Console.WriteLine(e.Message);
                }
            }
            else
            {
                Console.WriteLine("Profile not found.");
            }
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 ListOpportunities。

Go

SDK for Go V2

列出機會。

```
package main

import (
 "context"
```

```
 "encoding/json"
 "fmt"
 "log"

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

func main() {
 config, err := config.LoadDefaultConfig(context.TODO())

 if err != nil {
  log.Fatal(err)
 }

 config.Region = "us-east-1"

 client := partnercentralselling.NewFromConfig(config)

 output, err := client.ListOpportunities(context.TODO(),
 &partnercentralselling.ListOpportunitiesInput{
  MaxResults: aws.Int32(2),
  Catalog:    aws.String("AWS"),
 })

 if err != nil {
  log.Fatal(err)
 }

 jsonOutput, err := json.MarshalIndent(output, "", "    ")
 fmt.Println(string(jsonOutput))
}
```

- 如需 API 詳細資訊，請參閱《適用於 Go 的 AWS SDK API 參考》中的 ListOpportunities。

Java

適用於 Java 2.x 的 SDK

列出機會。

```
package org.example;

import java.util.ArrayList;
import java.util.List;

import org.example.utils.ReferenceCodesUtils;
import static org.example.utils.Constants.*;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.ListOpportunitiesRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.ListOpportunitiesResponse;
import
 software.amazon.awssdk.services.partnercentralselling.model.OpportunitySummary;

/*
 * Purpose
 * PC-API-18 Getting list of Opportunities
 */

public class ListOpportunititesPaging {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {
     List<OpportunitySummary> opportunitySummaries = getResponse();
        ReferenceCodesUtils.formatOutput(opportunitySummaries);
    }

    private static List<OpportunitySummary> getResponse() {
     List<OpportunitySummary> opportunitySummaries = new
 ArrayList<OpportunitySummary>();
```

```
 ListOpportunitiesRequest listOpportunityRequest =
ListOpportunitiesRequest.builder()
               .catalog(CATALOG_TO_USE)
         .maxResults(5)
         .build();

    ListOpportunitiesResponse response =
client.listOpportunities(listOpportunityRequest);

    opportunitySummaries.addAll(response.opportunitySummaries());

    while (response.nextToken() != null && response.nextToken().length() > 0) {
     listOpportunityRequest = ListOpportunitiesRequest.builder()
                   .catalog(CATALOG_TO_USE)
             .maxResults(5)
             .nextToken(response.nextToken())
             .build();
     response = client.listOpportunities(listOpportunityRequest);
     opportunitySummaries.addAll(response.opportunitySummaries());
    }

    client.close();

       return opportunitySummaries;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 ListOpportunities。

Python

適用於 Python 的 SDK (Boto3)

列出機會。

```
#!/usr/bin/env python

"""
Purpose
PC-API -18 Getting list of Opportunities
"""
```

```
import json
import logging
import boto3
import utils.helpers as helper

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def get_list_of_opportunities():

    opportunity_list = []

    list_opportunities_request ={
        "Catalog": CATALOG_TO_USE,
     "MaxResults": 20
    }
    try:
        # Perform an API call
        response =
 partner_central_client.list_opportunities(**list_opportunities_request)
        opportunity_list.extend(response["OpportunitySummaries"])

        while "NextToken" in response and response["NextToken"] is not None:
            list_opportunities_request["NextToken"] = response["NextToken"]
            response =
 partner_central_client.list_opportunities(**list_opportunities_request)
            opportunity_list.extend(response["OpportunitySummaries"])

        return opportunity_list

    except Exception as err:
        # Catch all client exceptions
        print(json.dumps(err.response))

def usage_demo():
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
```

```
    print("Getting list of Opportunities.")
    print("-" * 88)

    helper.pretty_print_datetime(get_list_of_opportunities())

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的 ListOpportunities。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListSolutions` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `ListSolutions`。

Java

適用於 Java 2.x 的 SDK

擷取合作夥伴在 Partner Central 上註冊的合作夥伴解決方案清單。

```
package org.example;

import java.util.ArrayList;
import java.util.List;

import static org.example.utils.Constants.*;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.ListSolutionsRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.ListSolutionsResponse;
```

```
import software.amazon.awssdk.services.partnercentralselling.model.SolutionBase;

/*
 * Purpose
 * PC-API-10 Getting list of solutions
 */

public class ListSolutions {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {
     List<SolutionBase> solutionSummaries = getResponse();
        ReferenceCodesUtils.formatOutput(solutionSummaries);
    }

    static List<SolutionBase> getResponse() {
  List<SolutionBase> solutionSummaries = new ArrayList<SolutionBase>();

  ListSolutionsRequest listSolutionsRequest = ListSolutionsRequest.builder()
    .catalog(CATALOG_TO_USE)
          .maxResults(5)
          .build();

     ListSolutionsResponse response = client.listSolutions(listSolutionsRequest);

     solutionSummaries.addAll(response.solutionSummaries());

        return solutionSummaries;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 ListSolutions。

Python

適用於 Python 的 SDK (Boto3)

擷取合作夥伴在 Partner Central 上註冊的合作夥伴解決方案清單。

```
#!/usr/bin/env python

"""
Purpose
PC-API-10 Getting list of solutions
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def get_list_of_solutions():
    list_solutions_request ={
        "Catalog": CATALOG_TO_USE,
     "MaxResults": 20
    }
    try:
        # Perform an API call
        response =
 partner_central_client.list_solutions(**list_solutions_request)
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
```

```
    print("-" * 88)
    print("Getting list of solutions.")
    print("-" * 88)

    helper.pretty_print_datetime(get_list_of_solutions())

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 ListSolutions。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`RejectEngagementInvitation` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `RejectEngagementInvitation`。

Java

適用於 Java 2.x 的 SDK

拒絕AWS共用的 EngagementInvitation。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.RejectEngagementInvitationRe
import
 software.amazon.awssdk.services.partnercentralselling.model.RejectEngagementInvitationRe
```

```
/*
 * Purpose
 * PC-API-05 AWS Originated(AO) rejection
 */

public class RejectEngagementInvitation {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

  RejectEngagementInvitationResponse response = getResponse(opportunityId);

     ReferenceCodesUtils.formatOutput(response);
    }

 static RejectEngagementInvitationResponse getResponse(String invitationId) {

        RejectEngagementInvitationRequest rejectOpportunityRequest =
 RejectEngagementInvitationRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .identifier(invitationId)
          .rejectionReason("Unable to support")
          .build();

  RejectEngagementInvitationResponse response =
 client.rejectEngagementInvitation(rejectOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 RejectEngagementInvitation。

Python

適用於 Python 的 SDK (Boto3)

拒絕AWS共用的 EngagementInvitation。

```
#!/usr/bin/env python

"""
Purpose
PC-API-05 AWS Originated AO rejection - RejectOpportunityEngagementInvitation -
 Rejects a engagement invitation.
This action indicates that the partner does not wish to participate in the
 engagement and
provides a reason for the rejection.
Upon rejection, a OpportunityEngagementInvitationRejected event is triggered.
Subsequently, the invitation will no longer be available for the partner to act
 on.
"""
import json
import logging
import boto3
import utils.helpers as helper

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def reject_opportunity_engagement_invitation(identifier, reject_reason):
    reject_opportunity_engagement_invitation_request ={
        "Catalog": CATALOG_TO_USE,
     "Identifier": identifier,
        "RejectionReason": reject_reason
    }
    try:
        # Perform an API call
        response =
 partner_central_client.reject_engagement_invitation(**reject_opportunity_engagement_invi
```

```
        return response

    except Exception as err:
        # Catch all client exceptions
        print(json.dumps(err.response))

def usage_demo():
    identifier = "arn:aws:partnercentral:us-east-1::catalog/Sandbox/engagement-
invitation/engi-0000002isviga"
    reject_reason = "Customer problem unclear"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Given the ARN identifier and reject reason, reject the Opportunity
 Engagement Invitation.")
    print("-" * 88)


 helper.pretty_print_datetime(reject_opportunity_engagement_invitation(identifier,
 reject_reason))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 RejectEngagementInvitation。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`StartEngagementByAcceptingInvitationTask` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `StartEngagementByAcceptingInvitationTask`。

Java

適用於 Java 2.x 的 SDK

接受 EngagementInvitation 即可開始參與。

```
package org.example;
```

```
import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.StartEngagementByAcceptingIn
import
 software.amazon.awssdk.services.partnercentralselling.model.StartEngagementByAcceptingIn
import
 software.amazon.awssdk.services.partnercentralselling.model.GetEngagementInvitationReque
import
 software.amazon.awssdk.services.partnercentralselling.model.GetEngagementInvitationRespo
import
 software.amazon.awssdk.services.partnercentralselling.model.InvitationStatus;

/*
Purpose
PC-API-04: Start Engagement By Accepting InvitationTask for AWS Originated(AO)
 opportunity
*/

public class StartEngagementByAcceptingInvitationTask {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

 static String clientToken = "test-a30d161";

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;
```

```
    StartEngagementByAcceptingInvitationTaskResponse response =
 getResponse(opportunityId);

    if ( response == null) {
     System.out.println("Opportunity is not AWS Originated.");
    } else {
     ReferenceCodesUtils.formatOutput(response);
    }
   }

   private static GetEngagementInvitationResponse getInvitation(String
 invitationId) {

    GetEngagementInvitationRequest getRequest =
 GetEngagementInvitationRequest.builder()
          .catalog(Constants.CATALOG_TO_USE)
          .identifier(invitationId)
          .build();

  GetEngagementInvitationResponse response =
 client.getEngagementInvitation(getRequest);

       return response;
 }

 static StartEngagementByAcceptingInvitationTaskResponse getResponse(String
 invitationId) {

  if ( getInvitation(invitationId).status().equals(InvitationStatus.PENDING)) {
   StartEngagementByAcceptingInvitationTaskRequest acceptOpportunityRequest =
    StartEngagementByAcceptingInvitationTaskRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .identifier(invitationId)
          .clientToken(clientToken)
          .build();

   StartEngagementByAcceptingInvitationTaskResponse response =
 client.startEngagementByAcceptingInvitationTask(acceptOpportunityRequest);
        return response;
  }
  return null;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 StartEngagementByAcceptingInvitationTask。

Python

適用於 Python 的 SDK (Boto3)

接受 EngagementInvitation 即可開始參與。

```
#!/usr/bin/env python

"""
Purpose
PC-API -11 Associating a product
PC-API -12 Associating a solution
PC-API -13 Associating an offer
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def get_opportunity(identifier):
    get_opportunity_request ={
     "Identifier": identifier,
        "Catalog": CATALOG_TO_USE
    }
    try:
        # Perform an API call
        response =
 partner_central_client.get_engagement_invitation(**get_opportunity_request)
```

```
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def start_engagement_by_accepting_invitation_task(identifier):

    response = get_opportunity(identifier)

    if ( response['Status'] == 'PENDING') :
        accept_opportunity_engagement_invitation_request ={
            "Catalog": CATALOG_TO_USE,
         "Identifier" : identifier,
            "ClientToken": "test-123456"
        }
        try:
            # Perform an API call
            response =
 partner_central_client.start_engagement_by_accepting_invitation_task(**accept_opportunit
            return response

        except ClientError as err:
            # Catch all client exceptions
            print(err.response)
            return None
    else:
        return None

def usage_demo():
    identifier = "arn:aws:partnercentral:us-east-1::catalog/Sandbox/engagement-
invitation/engi-0000002isusga"
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Get updated Opportunity.")
    print("-" * 88)


 helper.pretty_print_datetime(start_engagement_by_accepting_invitation_task(identifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 StartEngagementByAcceptingInvitationTask。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`StartEngagementFromOpportunityTask` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `StartEngagementFromOpportunityTask`。

Java

適用於 Java 2.x 的 SDK

透過接受參與邀請，並在合作夥伴的系統中建立對應的機會，即可從現有機會啟動參與程序。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import software.amazon.awssdk.services.partnercentralselling.model.AwsSubmission;
import
 software.amazon.awssdk.services.partnercentralselling.model.SalesInvolvementType;
import
 software.amazon.awssdk.services.partnercentralselling.model.StartEngagementFromOpportuni
import
 software.amazon.awssdk.services.partnercentralselling.model.StartEngagementFromOpportuni
import software.amazon.awssdk.services.partnercentralselling.model.Visibility;

/*
 * Purpose
 * PC-API-01 Partner Originated (PO) opp submission(Start Engagement From
 Opportunity Task for AO Originated Opportunity)
```

```
 */

public class StartEngagementFromOpportunityTask {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     StartEngagementFromOpportunityTaskResponse response =
 getResponse(opportunityId);

     ReferenceCodesUtils.formatOutput(response);
    }

 static StartEngagementFromOpportunityTaskResponse getResponse(String
 opportunityId) {

  StartEngagementFromOpportunityTaskRequest submitOpportunityRequest =
 StartEngagementFromOpportunityTaskRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .identifier(opportunityId)
          .clientToken("test-annjqwesdsd99")

 .awsSubmission(AwsSubmission.builder().involvementType(SalesInvolvementType.CO_SELL).vis
          .build();

  StartEngagementFromOpportunityTaskResponse response =
 client.startEngagementFromOpportunityTask(submitOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 StartEngagementFromOpportunityTask。

Python

適用於 Python 的 SDK (Boto3)

透過接受參與邀請，並在合作夥伴的系統中建立對應的機會，即可從現有機會啟動參與程序。

```
#!/usr/bin/env python

"""
Purpose
PC-API -11 Associating a product
PC-API -12 Associating a solution
PC-API -13 Associating an offer
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def start_engagement_from_opportunity_task(identifier):

    start_engagement_from_opportunity_task_request ={
            "AwsSubmission": {
                "InvolvementType": "Co-Sell",
                "Visibility": "Full"
            },
            "Catalog": CATALOG_TO_USE,
         "Identifier" : identifier,
            "ClientToken": "test-annjqwesdsd99"
    }
    try:
            # Perform an API call
            response =
 partner_central_client.start_engagement_from_opportunity_task(**start_engagement_from_op
```

```
            return response

    except ClientError as err:
            # Catch all client exceptions
            print(err.response)
            return None

def usage_demo():
    identifier = "O5465588"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Start Engagement from Opportunity Task.")
    print("-" * 88)


 helper.pretty_print_datetime(start_engagement_from_opportunity_task(identifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 StartEngagementFromOpportunityTask。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`UpdateOpportunity` 搭配AWS SDK 使用

下列程式碼範例示範如何使用 `UpdateOpportunity`。

Java

適用於 Java 2.x 的 SDK

更新機會。

```
package org.example;

import java.time.Instant;
import java.util.ArrayList;
```

```
import java.util.List;

import static org.example.utils.Constants.*;

import org.example.entity.Root;
import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;
import org.example.utils.StringSerializer;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import software.amazon.awssdk.services.partnercentralselling.model.Account;
import software.amazon.awssdk.services.partnercentralselling.model.Address;
import software.amazon.awssdk.services.partnercentralselling.model.Contact;
import software.amazon.awssdk.services.partnercentralselling.model.Customer;
import
 software.amazon.awssdk.services.partnercentralselling.model.ExpectedCustomerSpend;
import
 software.amazon.awssdk.services.partnercentralselling.model.GetOpportunityRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.GetOpportunityResponse;
import software.amazon.awssdk.services.partnercentralselling.model.LifeCycle;
import software.amazon.awssdk.services.partnercentralselling.model.Marketing;
import
 software.amazon.awssdk.services.partnercentralselling.model.NextStepsHistory;
import software.amazon.awssdk.services.partnercentralselling.model.Project;
import software.amazon.awssdk.services.partnercentralselling.model.ReviewStatus;
import
 software.amazon.awssdk.services.partnercentralselling.model.UpdateOpportunityRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.UpdateOpportunityResponse;

import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.ToNumberPolicy;

/*
 * Purpose
 * PC-API-02/06 Update opportunity when LifeCycle.ReviewStatus is not Submitted
 or In-Review
 */
```

```
public class UpdateOpportunity {

 static final Gson GSON = new GsonBuilder()
   .setObjectToNumberStrategy(ToNumberPolicy.LAZILY_PARSED_NUMBER)
   .registerTypeAdapter(String.class, new StringSerializer())
   .create();

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

 static String OPPORTUNITY_ORIGIN = ORIGIN_PARTNER_ORIGINATED;

 public static void main(String[] args) {

  String inputFile = "updateOpportunity.json";

  if (args.length > 0)
   inputFile = args[0];

  UpdateOpportunityResponse response = updateOpportunity(inputFile);

  client.close();
 }

 public static GetOpportunityResponse getResponse(String opportunityId) {

        GetOpportunityRequest getOpportunityRequest =
 GetOpportunityRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .identifier(opportunityId)
          .build();

        GetOpportunityResponse response =
 client.getOpportunity(getOpportunityRequest);
        System.out.println(opportunityId + ":" + response);
        return response;
 }

 public static UpdateOpportunityResponse updateOpportunity(String inputFile) {
```

```
 String inputString = ReferenceCodesUtils.readInputFileToString(inputFile);

 Root root = GSON.fromJson(inputString, Root.class);
 GetOpportunityResponse response = getResponse(root.identifier);

 if (response != null
   && response.lifeCycle() != null
   && response.lifeCycle().reviewStatus() != null
   && response.lifeCycle().reviewStatus() != ReviewStatus.SUBMITTED
   && response.lifeCycle().reviewStatus() != ReviewStatus.IN_REVIEW) {

  List<NextStepsHistory> nextStepsHistories = new ArrayList<NextStepsHistory>();
  if ( root.lifeCycle != null && root.lifeCycle.nextStepsHistories != null) {
   for (org.example.entity.NextStepsHistory nextStepsHistoryJson :
root.lifeCycle.nextStepsHistories) {
    NextStepsHistory nextStepsHistory = NextStepsHistory.builder()
      .time(Instant.parse(nextStepsHistoryJson.time))
      .value(nextStepsHistoryJson.value)
                  .build();
    nextStepsHistories.add(nextStepsHistory);
   }
  }

  LifeCycle lifeCycle = null;
  if ( root.lifeCycle != null ) {
   lifeCycle = LifeCycle.builder()
    .closedLostReason(root.lifeCycle.closedLostReason)
    .nextSteps(root.lifeCycle.nextSteps)
    .nextStepsHistory(nextStepsHistories)
    .reviewComments(root.lifeCycle.reviewComments)
    .reviewStatus(root.lifeCycle.reviewStatus)
    .reviewStatusReason(root.lifeCycle.reviewStatusReason)
    .stage(root.lifeCycle.stage)
    .targetCloseDate(root.lifeCycle.targetCloseDate)
    .build();
  }

  Marketing marketing = null;
  if ( root.marketing != null ) {
   marketing = Marketing.builder()
     .awsFundingUsed(root.marketing.awsFundingUsed)
     .campaignName(root.marketing.campaignName)
     .channels(root.marketing.channels)
```

```
      .source(root.marketing.source)
      .useCases(root.marketing.useCases)
      .build();

  }

  Address address = null;
  if (root.customer != null && root.customer.account != null &&
root.customer.account.address != null) {
   address =
Address.builder().postalCode(root.customer.account.address.postalCode)
     .stateOrRegion(root.customer.account.address.stateOrRegion)
     .countryCode(root.customer.account.address.countryCode).build();
  }

  Account account = null;
  if (root.customer != null && root.customer.account != null) {
   account = Account.builder().address(address).duns(root.customer.account.duns)

.industry(root.customer.account.industry).companyName(root.customer.account.companyName)
     .websiteUrl(root.customer.account.websiteUrl).build();
  }

  List<Contact> contacts = new ArrayList<Contact>();
  if ( root.customer != null && root.customer.contacts != null) {
   for (org.example.entity.Contact jsonContact : root.customer.contacts) {
    Contact contact = Contact.builder()
                  .email(jsonContact.email)
                  .firstName(jsonContact.firstName)
                  .lastName(jsonContact.lastName)
                  .phone(jsonContact.phone)
                  .businessTitle(jsonContact.businessTitle)
                  .build();
    contacts.add(contact);
   }
  }

  Customer customer =
Customer.builder().account(account).contacts(contacts).build();

  List<ExpectedCustomerSpend> expectedCustomerSpends = new
ArrayList<ExpectedCustomerSpend>();
  if ( root.project != null && root.project.expectedCustomerSpend != null) {
```

```
   for (org.example.entity.ExpectedCustomerSpend expectedCustomerSpendJson :
root.project.expectedCustomerSpend) {
    ExpectedCustomerSpend expectedCustomerSpend = null;
    expectedCustomerSpend = ExpectedCustomerSpend.builder()
      .amount(expectedCustomerSpendJson.amount)
      .currencyCode(expectedCustomerSpendJson.currencyCode)
      .frequency(expectedCustomerSpendJson.frequency)
      .targetCompany(expectedCustomerSpendJson.targetCompany)
      .build();
    expectedCustomerSpends.add(expectedCustomerSpend);
   }
        }

  Project project = null;
  if (root.project != null) {
   project = Project.builder().title(root.project.title)
     .customerBusinessProblem(root.project.customerBusinessProblem)

.customerUseCase(root.project.customerUseCase).deliveryModels(root.project.deliveryModel
     .expectedCustomerSpend(expectedCustomerSpends)

.salesActivities(root.project.salesActivities).competitorName(root.project.competitorNam
     .otherSolutionDescription(root.project.otherSolutionDescription).build();
  }

  // Building the Actual CreateOpportunity Request
  UpdateOpportunityRequest updateOpportunityRequest =
UpdateOpportunityRequest.builder().catalog(root.catalog)

.identifier(root.identifier).lastModifiedDate(Instant.parse(root.lastModifiedDate))

.primaryNeedsFromAwsWithStrings(root.primaryNeedsFromAws).opportunityType(root.opportuni
    .lifeCycle(lifeCycle)
    .customer(customer)
    .project(project)
    .partnerOpportunityIdentifier(root.partnerOpportunityIdentifier)
    .marketing(marketing)
    .nationalSecurity(root.nationalSecurity)
    .opportunityType(root.opportunityType)
    .build();

  UpdateOpportunityResponse updateResponse =
client.updateOpportunity(updateOpportunityRequest);
  System.out.println("Successfully updated opportunity: " + updateResponse);
```

```
   return updateResponse;
  } else {
   System.out.println("Opportunity cannot be updated.");
   return null;
  }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 UpdateOpportunity。

Python

適用於 Python 的 SDK (Boto3)

更新機會。

```
#!/usr/bin/env python

"""
Purpose
PC-API-2  Updating Partner Originated Opportunity
"""
import logging
import boto3
import sys
import os
sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
import utils.helpers as helper
from botocore.client import ClientError
import utils.stringify_details as sd
from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def get_opportunity(identifier):
```

```
    get_opportunity_request ={
     "Identifier": identifier,
        "Catalog": CATALOG_TO_USE
    }
    try:
        # Perform an API call
        response =
 partner_central_client.get_opportunity(**get_opportunity_request)
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def update_opportunity():
    update_opportunity_request_orig = sd.stringify_json("src/update_opportunity/
update_opportunity_technical_validation.json")
    update_opportunity_request =
 helper.remove_nulls(update_opportunity_request_orig)

    try:
        # Perform an API call
        response =
 partner_central_client.update_opportunity(**update_opportunity_request)
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def update_opportunity_if_eligible(identifier):
    response = get_opportunity(identifier)
    if response is not None:
        return update_opportunity()
    else:
        print("Failed to retrieve opportunity details")

def usage_demo():
    identifier = "O5465588"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Updating opportunity.")
```

```
    print("-" * 88)

    helper.pretty_print_datetime(update_opportunity_if_eligible(identifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 UpdateOpportunity。

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Partner Central AWS SDKs的案例

下列程式碼範例說明如何在 Partner Central AWS SDKs中實作常見案例。這些案例會向您示範如何呼叫 Partner Central 中的多個函數，或與其他AWS 服務結合，藉以完成特定任務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- 更新機會的關聯實體

更新機會的關聯實體

下列程式碼範例示範如何：

- 取消舊實體的關聯。
- 關聯新實體。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 Scenarios 儲存庫中設定和執行。

更新機會的關聯實體

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.AssociateOpportunityRequest;
import
 software.amazon.awssdk.services.partnercentralselling.model.AssociateOpportunityResponse
import
 software.amazon.awssdk.services.partnercentralselling.model.DisassociateOpportunityReque
import
 software.amazon.awssdk.services.partnercentralselling.model.DisassociateOpportunityRespo

/*
Purpose
PC-API -17 Replacing a solution
*/

public class ReplaceSolution {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
```

```
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

   public static void main(String[] args) {

    String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

    String entityType = "Solutions";
    String originalEntityIdentifier = "S-0000000";
    String newEntityIdentifier = "S-0011111";

    disassociateOppornititityResponse(opportunityId, entityType,
originalEntityIdentifier );
    AssociateOpportunityResponse associateOpportunityResponse =
associateOpportunityResponse(opportunityId, entityType, newEntityIdentifier );

    ReferenceCodesUtils.formatOutput(associateOpportunityResponse);
   }

private static AssociateOpportunityResponse associateOpportunityResponse(String
opportunityId, String entityType, String entityIdentifier) {

       AssociateOpportunityRequest associateOpportunityRequest =
AssociateOpportunityRequest.builder()
   .catalog(Constants.CATALOG_TO_USE)
         .opportunityIdentifier(opportunityId)
         .relatedEntityType(entityType)
         .relatedEntityIdentifier(entityIdentifier)
         .build();

       AssociateOpportunityResponse response =
client.associateOpportunity(associateOpportunityRequest);

       return response;
}

private static DisassociateOpportunityResponse
disassociateOppornititityResponse(String opportunityId, String entityType, String
entityIdentifier) {
 PartnerCentralSellingClient client = PartnerCentralSellingClient.builder()
           .region(Region.US_EAST_1)
           .credentialsProvider(DefaultCredentialsProvider.create())
           .httpClient(ApacheHttpClient.builder().build())
```

```
            .build();

        DisassociateOpportunityRequest disassociateOpportunityRequest =
 DisassociateOpportunityRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .opportunityIdentifier(opportunityId)
          .relatedEntityType(entityType)
          .relatedEntityIdentifier(entityIdentifier)
          .build();

        DisassociateOpportunityResponse response =
 client.disassociateOpportunity(disassociateOpportunityRequest);

        return response;
 }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
  - AssociateOpportunity
  - DisassociateOpportunity

Python

適用於 Python 的 SDK (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS程式碼範例儲存庫中設定和執行。

更新機會的關聯實體

```
#!/usr/bin/env python

"""
Purpose
PC-API -17 Replacing a solution
"""
```

```
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def replace_solution(original_entity_identifier, new_entity_identifier,
 opportunityIdentifier):
    disassociate_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "OpportunityIdentifier" : opportunityIdentifier,
        "RelatedEntityType" : "Solutions",
        "RelatedEntityIdentifier" : original_entity_identifier
    }

    associate_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "OpportunityIdentifier" : opportunityIdentifier,
        "RelatedEntityType" : "Solutions",
        "RelatedEntityIdentifier" : new_entity_identifier
    }
    try:
        # Perform an API call
        response =
 partner_central_client.disassociate_opportunity(**disassociate_opportunity_request)
        response =
 partner_central_client.associate_opportunity(**associate_opportunity_request)
        return response

    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    original_entity_identifier = "S-0049999"
    new_entity_identifier = "S-0050014"
```

```
    opportunityIdentifier = "O4397574"

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Replacing a solution.")
    print("-" * 88)

    helper.pretty_print_datetime(replace_solution(original_entity_identifier,
 new_entity_identifier, opportunityIdentifier))

if __name__ == "__main__":
    usage_demo()
```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》中的下列主題。
  - AssociateOpportunity
  - DisassociateOpportunity

如需AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配AWS SDK 使用AWS Partner Central API。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

版本備註

此頁面包含AWS Partner Central API 文件的重大變更摘要，從最新的更新開始。

修訂歷史記錄

變更	描述	日期
RC 6.0	• AWS Partner Central 新增了新的合作夥伴收入測量 APIs。	2026-06-30
RC 5.3	• ACE Resonate 更新：GetAwsOpportunitySummary 回應已更新。	2026-06-16
RC 5.2	• 歐元貨幣支援： CreateOpportunity 除了 USD MRR 貨幣代碼之外，UpdateOpportunity 、和 CreateEngagementInvitation 現在接受 EUR 。當 AWS 分割區為 aws-eusc(AWS 歐洲主權雲端）時，貨幣代碼必須為 EUR。 • 讀取操作中的 EUR：ListOpportunities 、 GetOpportunity 和 GetAwsOpportunitySummary 現在EUR會在 CurrencyCode 欄位中傳回aws-eusc分割區中的機會。	2026-03-30
RC 5.1	• 歐元貨幣支援： CreateOpportunity 除了 USD MRR 貨幣代碼之外，	2026-03-30

變更	描述	日期
	UpdateOpportunity 、和 CreateEngagementInvitation 現在接受 EUR。當 AWS 分割區為 aws-eusc(AWS 歐洲主權雲端）時，貨幣代碼必須為 EUR。 • 讀取操作中的 EUR：ListOpportunities 、 GetOpportunity 和 GetAwsOpportunitySummary 現在EUR會在 CurrencyCode 欄位中傳回aws-eusc分割區中的機會。	

變更	描述	日期
RC 5	• 引進的帳戶 API：將新的帳戶 API 新增至AWS Partner Central，讓合作夥伴能夠管理其帳戶資訊、註冊、設定檔、網域管理和合作夥伴連線。 • 帳戶 API 參考文件：新增帳戶 API 的完整 API 參考文件，包括合作夥伴註冊、設定檔管理、網域驗證和帳戶連線工作流程。 • 帳戶 API 記錄：新增帳戶 API 的記錄支援和文件，以協助稽核和追蹤 API 用量。 • 帳戶 API 通知：新增了帳戶 API 的事件通知支援，可讓合作夥伴接收帳戶相關活動的即時更新，包括合作夥伴帳戶連線。 • 帳戶 API 配額：新增帳戶 API 的服務配額文件。 • 帳戶 API 的沙盒測試：新增帳戶 API 的沙盒測試功能，允許合作夥伴在非生產環境中測試帳戶管理工作流程。 • 引進的優勢 API：將新的優勢 API 新增至AWS Partner Central，讓合作夥伴能夠透過集中式界面探索、申請和管理跨AWS合作夥伴計劃的優勢。 • 利益 API 參考文件：新增利益 API 的完整 API 參考文	2025-11-30

變更	描述	日期
	件，涵蓋利益應用程式、配置和履行程序。 • 效益 API 記錄：新增效益 API 的記錄支援和文件，以協助稽核和追蹤 API 使用情況。 • 效益 API 通知：新增了效益 API 的事件通知支援，可讓合作夥伴接收效益應用程式和配置的即時更新。 • Benefits API 配額：新增 Benefits API 的服務配額文件。 • 效益 API 的沙盒測試：新增效益 API 的沙盒測試功能，允許合作夥伴在非生產環境中測試效益管理工作流程。 • 已更新銷售 API：增強銷售 API 參與動作，以支援潛在客戶管理工作流程。 • 銷售 API 文件更新：增強型銷售 API 文件，具有改善的結構和更新的潛在客戶管理工作流程。 • 銷售 API 通知更新：增強參與和參與邀請事件的事件通知文件。 • API 結構擴展：擴展文件結構，以支援四個不同的 APIs（帳戶、優勢、銷售和管道），每個 API 的許可、記錄、通知、配額和沙盒測試都有專用區段。	

變更	描述	日期
	• 增強型 API 用量文件：新增了帳戶和利益 APIs，其中包含詳細的工作流程和整合模式。 • 支援的區域文件：新增帳戶和利益 APIs 支援AWS區域的文件。 • 的新篩選條件`ListOpportunities` ：新增`TargetCloseDate` 篩選條件，以使用 YYYY-MM-DD 格式的 `AfterTargetCloseDate` 和 `BeforeTargetCloseDate` 欄位，依其預期的結束日期啟用篩選機會。	

變更	描述	日期
RC 4	• 引進的頻道 API：將新的頻道 API 新增至AWS Partner Central，讓合作夥伴能夠管理頻道關係和partner-to-partner協同合作。 • 頻道 API 參考文件：新增頻道 API 的完整 API 參考文件，包括所有可用的動作和資料模型。 • 頻道 API 許可：新增了頻道 API 特定的 IAM 許可政策和存取控制文件。 • 通道 API 記錄：新增通道 API 的記錄支援和文件，以協助稽核和追蹤 API 用量。 • 頻道 API 通知：新增了頻道 API 的事件通知支援，可讓合作夥伴接收頻道相關活動的即時更新。 • 頻道 API 配額：已新增頻道 API 的服務配額文件。 • 頻道 API 的沙盒測試：新增頻道 API 的沙盒測試功能，允許合作夥伴在非生產環境中測試頻道工作流程。 • API 結構重組：重組文件，以區分銷售 API 和頻道 API，以及每個 API 許可、記錄、通知和配額的專用區段。	2025-11-19

變更	描述	日期
	• 支援的區域文件：新增銷售和頻道 APIs 支援AWS區域的文件。	

變更	描述	日期

RC 3	• 已更新 API 許可政策：政策已更新，以反映此版本中引入的新動作。請確定您的 IAM 角色和許可已相應地調整。
	• 自訂標頭用量詳細資訊：新增如何使用自訂標頭X-Amzn-User-Agent 以協助稽核和測量 API 用量的詳細資訊。
	• 以非同步動作取代動作：AcceptOpportunityEngagementInvitation 以非同步動作取代StartEngagementByAcceptingInvitationTask ，且承載格式在 RC2 方面已變更。
	• 實體和動作重新命名：變更為 OpportunityEngagementInvitation EngagementInvitation 。這些動作的屬性也已變更，以符合新的實體。已取代下列動作：
	  • GetOpportunityEngagementInvitation 現在是 GetEngagementInvitation 。邀請中的多個屬性已變更、新增或移除。
	  • ListOpportunityEngagementIn
		2024-10-17

變更	描述	日期
	vitations 現在是 ListEngagementInvitations 。 • RejectOpportunityEngagementInvitation 現在為 RejectEngagementInvitation 。 • AssignOpportunity 參數更新： AssignOpportunity 現在需要 Assignee而非 AssigneeEmail 。相應地更新您的實作。 • SubmitOpportunity 功能變更： SubmitOpportunity 現在由非同步動作 執行StartEngagementFromOpportunityTask 。 SubmitOpportunity 現在接受 InvolvementType 和 Visibility 。 • 屬性重新命名和擴展：EstimatedAwsMonthlyRevenue 變更為 ExpectedCustomerSpend 。 ExpectedCustomerSpend 現在包含新的屬性，例如 Frequency 和 TargetCompany 。 • AWS機會摘要的新動作：現在透過名為 的單獨動作提	

變更	描述	日期

	供機會可見性GetAwsOpportunitySummary 。
	• 使用機會的工作流程更新：使用機會的工作流程已更新，以使用此版本中引入的新動作。
	• 使用機會的工作流程更新：使用機會的工作流程已更新為使用 EngagementInvitations ，這改善了 AWS 提供的處理機會。
	• 追蹤更新的工作流程更新：追蹤更新的工作流程現在使用機會更新，提高監控機會狀態的清晰度和效率。
	• 列出動作的新篩選條件：新的篩選條件可用於 ListSolutions 和 ListOpportunities 動作。
	  • 對於 ListSolutions ，新的篩選條件為 FilterCategory 、FilterIdentifier 和 FilterStatus 。
	  • 對於 ListOpportunities ，新的篩選條件為 FilterCustomerCompanyName 、FilterIdentifier 、FilterLastModified Date 、FilterLif	

變更	描述	日期
	eCycleStage 、FilterLifeCycleApprovalStatus 和 FilterOrigin 。 • 移除AccessControl 子物件：已移除機會模型中的AccessControl 子物件。 現在NationalSecurity 是頂層屬性。 • 屬性已重新定位至 GetAwsOpportunitySummary ：AWSOpportunityTeam 、洞見 (EngagementScore 、NextBestAction)、原始伺服器 和 等屬性InvolvementType 現在可透過 GetAwsOpportunitySummary 而非取得GetOpportunity 。 • InvolvementType 欄位簡介：引進 InvolvementType 欄位，以區分 Cosell 和僅限可見性的機會。 • 事件名稱更新：事件已更新為 以反映變更。例如，建立的機會參與邀請現在是建立的參與邀請。 • 回應承載更新：所有回應承載現在都包含 ID 和/或 ARN。	

變更	描述	日期
	• 請求承載更新：所有請求承載現在都使用識別符作為輸入，接受 ID 或 ARN。 • 業務開發實體更新：業務開發實體中的標題屬性已重新命名為 EngagementTitle 。 • 篩選字首移除：所有篩選不再包含篩選字首，簡化篩選機制。 • StartEngagementFromOpportunity 動作更新： 屬性AwsInvolvement 已變更為 AwsSubmission 。 • 邀請專案詳細資訊更新： Invitation.ProjectDetails.TargetCompletionDate 已重新命名為 Invitation.ProjectDetails.EstimatedCompletionDate 。 • 營收預估更新： 現在Invitation.PartnerRevenueEstimate 為 ExpectedCustomerSpend 。資料類型從一個物件變更為包含一個物件的陣列。 • 收件人帳戶更新： 欄位ReceiverAccount 已重新命名為 AccountReceiver 。	

變更	描述	日期
	• 業務開發邀請新屬性：針對業務開發邀請引進 SenderContact 屬性。 • Customer.Contact 現在是陣列，而不是物件。OpportunityOwner 、 PartnerAccountManager 是支援的值 BusinessTitle 。 • PartnerOpportunityTeam 現在稱為 OpportunityTeam 。 • APNPrograms 現在是 ApnPrograms • 已更新參與邀請狀態。 • ListEngagementInvitations 現在需要 ParticipantType 做為 RECEIVER。	

變更	描述	日期
RC 2	• 引進名為 OpportunityEngagementInvitation 的新實體。 • 引進了新的動作：ListOpportunityEngagementInvitations。 • 引進新的動作：GetOpportunityEngagementInvitation。 • 引進新事件：「已建立機會參與邀請」。 • 將 AcceptOpportunity 取代為 AcceptOpportunityEngagementInvitation。 • 將 RejectOpportunity 取代為 RejectOpportunityEngagementInvitation。 • 將 "Opportunity Accepted" 取代為 "Opportunity Engagement Invitation Accepted"。 • 將「機會遭拒」取代為「機會參與邀請遭拒」。 • `PrimaryNeedsFromAWS` 已變更為 PrimaryNeedsFromAws。 • `AWSOpportunityTeam` 已變更為 `AwsOpportunityTeam` 。 • `AWSSalesLifeCycle` 已變更為 `AwsSalesLifeCycle` 。	2024-08-07

變更	描述	日期
	• PrimaryNeedsFromAWSChangeReason 已變更為 PrimaryNeedsFromAwsChangeReason 。 • AWSAccountId 已變更為 AwsAccountId 。 • ExpectedMonthlyAWSRevenue 已變更為 ExpectedMonthlyAwsRevenue 。 • AWSFundingUsed 已變更為 AwsFundingUsed 。 • AWSMarketplaceOffers 已變更為 AwsMarketplaceOffers 。 • Country 已變更為 CountryCode 。 • PartnerOpportunityContact 已變更為 PartnerOpportunityTeam 。 • 將 欄位更新Contact.Title 為 Contact.BusinessTitle 。 • 將 欄位更新ContactInfo 為 OwnerInfo 。 • 將AWS團隊從映射變更為清單。 • 新增有關接受的 Rejection Reasons 清單的詳細資訊。 • 提供如何使用用戶端字符的資訊。	

變更	描述	日期
	• 包含如何使用 UpdateOpportunity 的詳細資訊。 • 新增使用AWS產品和合作夥伴解決方案的指引。 • 提供 OpportunityEngagementInvitation 實體的詳細資訊。 • 更新 StateOrRegion 以包含已修訂的可接受值清單。 • 實作多個錯誤修正，以改善效能和錯誤訊息。	

變更	描述	日期
RC 1	• 【中斷】 將 ApprovalStatus 重新命名為 ReviewStatus：我們已將 ApprovalStatus 欄位重新命名為 ReviewStatus，以更好地反映其用途。此外，我們推出了新的狀態，待提交，以指出草稿的機會。ReviewStatus 現在將接受下列值：待提交、已提交、審核中、已核准、已拒絕和必要動作。之前的草稿狀態將由待提交取代。 • 【突破性】 SubmitOpportunity 動作簡介：已推出 SubmitOpportunity 新動作。與 CreateOpportunity 也提交機會的目前行為不同，您現在可以在等待提交狀態下建立機會，而無需立即連結解決方案或AWS產品。若要完成提交，請使用 AssociateOpportunity 動作來連結解決方案或產品，後面接著 SubmitOpportunity。此區隔在提交之前提供更好的 API 回應效能和靈活的準備階段。 • 【中斷】 目錄參數的需求和沙盒端點的棄用：「gamma」端點 (partnercentral-selling-gamma.us-east-1.amazonaws.com：//) 在 7/30/2024 之後將無法使用。所有 API 動作現在都	2024-06-21

變更	描述	日期
	需要 Catalog 參數來指定操作是在沙盒或AWS目錄上執行。通知規則現在會根據目錄而非 environmentName 進行篩選。 • 【突破性】 將 OpportunityIdentifier 重新命名為識別符：ACTION AssignOpportunity、AcceptOpportunity 和 RejectOpportunity 現在採用識別符，而不是 OpportunityIdentifier，以與 UpdateOpportunity 保持一致。 • 從列舉類型變更為 APNProgram CustomerUseCase 和 UseCase：我們將轉換 Project.APNPrograms、 Marketing.Activity UseCases 和 Project.CustomerUseCase 從列舉類型變更為字串類型，以減少與變更值相關的風險。這些欄位只會接受預先定義清單中的值，但無法在 SDKs 中原生使用。 • 增強型錯誤處理：已大幅改善 APIs中的錯誤處理，以簡化偵錯並更快速解決問題。新的錯誤處理結構錯誤分為兩個階段：1。請求驗證失敗：檢查資料類型和格式，或 2。業務驗證失敗：承載中的所有問題都會列在一個	

變更	描述	日期
	回應中，指定發生錯誤的欄位。此合併意見回饋可讓您更有效率地疑難排解和修正錯誤。錯誤代碼和格式已標準化。 • 用戶端來源識別符：合作夥伴的能力在其請求中包含 X-Amzn-User-Agent 標頭，這有助於識別流量的來源。使用 格式 【Solution Name including solution provider】 | 【Version】 例如：AWS Partner CRM Connector | v2.0.1 • 錯誤修正，已部署，未變更 SDK】 機會建立和AWS帳戶 ID：合作夥伴現在可以啟動機會，同時將AWS帳戶 ID 從 null 變更為有效值，而不會遇到錯誤。先前，錯誤會阻止合作夥伴分別執行這些動作。 • 錯誤修正，已部署，無需變更軟體開發套件】 解決方案與新機會的關聯： 合作夥伴可以將解決方案與新建立的機會建立關聯。之前，當合作夥伴建立新的機會並同時與解決方案建立關聯時，解決方案資訊會遺失。此問題已解決。 • 錯誤修正，已部署，未變更 SDK】 機會擁有者的詳細資訊顯示：已修正某些機會	

變更	描述	日期
	在合併的 "LastName" 欄位中顯示擁有者詳細資訊的問題，而非預期的 "Email"、"FirstName" 和 "LastName" 欄位。 • 錯誤修正，已部署，未變更 SDK】 Customer.Account.WebsiteUrl 欄位選用：當 AccessControls.NationalSecurity 欄位為「是」時，Customer.Account.WebsiteUrl 欄位已設為選用，因此使 API 行為與現有的 UI 行為相符。如果 AccessControls.NationalSecurity 欄位為「否」或 null，則會需要 WebsiteUrl 欄位作為業務驗證。 • 錯誤修正，已部署，未變更 SDK】諮詢合作夥伴AWS的帳戶 ID 需求：UI 中顯示 AWS帳戶 ID 欄位為諮詢合作夥伴選用的不一致已解決。它現在是條件式必要欄位，如 文件所示。 • 錯誤修正，若要在 6 月 19 日部署，SDK 沒有變更】聯合銷售機會的階段關閉遺失：只要提供關閉遺失原因，合作夥伴現在可以成功將聯合銷售機會的階段更新為關閉遺失。這會解決先前指出「遺漏必要欄位：關閉	

變更	描述	日期
	機會時需要關閉遺失原因」的錯誤。 • 文件】 文件的改進：對文件進行了幾項改進，以反映變更。 • 待定修復的已知問題和即將推出的功能清單： • 使用解決方案或產品執行 AssociateOpportunity 動作時，來自優惠的錯誤訊息顯示不正確。 • 能夠在沙盒中遠端建立機會和模擬AWS驗證程序 • 當美國格式為 XXXXX-XXXX 時，郵遞區號會傳回 null。 • 後續步驟和下一步歷史記錄不適用於標記為「不需要支援」的機會AWS。 • 無法取得客戶業務問題描述超過 255 個字元的歷史機會。 • 能夠刪除待提交的機會。 • 資料模型中無法使用 LeadSource。	

變更	描述	日期
Beta 5	• 將 LifeCycle.ApprovalStatus 變更為 LifeCycle.ReviewStatus • 將 Insights.ReviewComments 變更為 LifeCycle.ReviewComments • 新增 LifeCycle.ReviewStatusReason（新屬性） • 已將新值「待定」新增至 PartnerAcceptance.Status • 已發行適用於 dotnet、java、javascript、python、ruby SDKs。 • 已更新 API 參考指南，以反映上述變更。	2024-04-19
Beta 4	• 新增沙盒測試：能夠在沙盒中測試機會事件通知。 • 新增文件：引入機會事件通知範例規則。 • 已新增開發套件版本：已發行適用於 Java (V2)、Javascript (V2)、DotNet (V3) 和 Python (V3) SDKs。 • 更新日期格式：更新以遵循 ISO 標準。	2024-03-04

變更	描述	日期
Beta 3	• 新增變更日誌檔案：引進變更日誌檔案，以更好地追蹤更新和變更。 • 更新 API 參考指南 (PDF)：發行 PDF 格式的 API 參考指南新版本。 • 更新 API 參考指南 (Web 版本）：更新 API 參考指南的 Web 版本。 • 新增了 DotNet 開發套件：發佈了 DotNet 開發套件，擴展了我們對不同程式設計環境的支援。 • 更新 Python Boto 3 開發套件 (V2)：發行 Python Boto 3 開發套件的第 2 版，引入新功能和改進。	2024-01-24
Beta 2	• 新增 API 參考指南 (Web 版本）：發行 API 參考指南 (Web 版本） 的新版本。注意：此版本要求在擷取後執行本機 Web 伺服器，才能正常運作。	2024-01-07

變更	描述	日期
Beta 1	• 新增 Boto3（初始版本）：啟動 Boto3 的初始版本，Boto3 是我們服務的 Python 開發套件。 • 已新增 Beta 版計畫指南：已為 Beta 版測試計畫的參與者推出指南。 • 新增 API 參考指南 (PDF)：以 PDF 格式上傳第一版的 API 參考指南，提供 API 的完整文件。	2023-12-15
Beta 0	此初始版本提供一套功能，可管理和最佳化AWS Partner Central 中的合作夥伴參與和機會。	2023-11-15

文件歷史紀錄

以下是此 API 參考的文件版本清單。

變更	描述	日期
新增合作夥伴收入測量 APIs的文件	AWS合作夥伴中心新增了合作夥伴收入測量 (PRM) 的新 APIs。	2026 年 6 月 30 日
已更新 ACE Resonate 的文件	AWS合作夥伴中央更新 GetAwsOpportunitySummary 動作的 API 回應，以支援 Cosell Motion。	2026 年 6 月 16 日
新增 Partner Central MCP Server 的文件	AWS合作夥伴中央代理程式 MCP 伺服器透過模型內容通訊協定 (MCP) 提供合作夥伴中央工具。	2026 年 3 月 16 日
RC 5 版本	發行AWS Partner Central API 的候選版本 5。將新的帳戶 API 引進AWS Partner Central，讓合作夥伴能夠管理其帳戶資訊、註冊、設定檔、網域管理和合作夥伴連線。將新的 優勢 API 引進AWS Partner Central，讓合作夥伴能夠透過集中式界面探索、申請和管理AWS合作夥伴計劃的優勢。增強型銷售 API 參與動作，以支援潛在客戶管理工作流程。新增了帳戶、利益 APIs 的完整使用指南和 API 文件，其中包含詳細的工作流程和整合模式。使用潛在客戶內容進行業	2025 年 11 月 30 日

	務開發管理的增強型銷售 API 文件。	
RC 4 版本	發行AWS Partner Central API 的候選版本 4。引進了用於管理頻道關係和partner-to-partner協作的頻道 API，讓合作夥伴在使用 Billing Transfer 與最終客戶交易時，符合計劃利益的資格。文件已重新組織，以支援多個AWS Partner Central APIs。	2025 年 11 月 19 日
新增多合作夥伴機會的文件	多合作夥伴機會（預覽）是 AWS Partner Central 的整合體驗，可讓AWS合作夥伴尋找其他合作夥伴並與其連線，並在客戶交易上共同銷售。	2024 年 12 月 4 日
RC 3 版本	Partner AWS Central API 發行的候選版本 3	2024 年 10 月 17 日
RC 2 版本	發行AWS Partner Central API 的候選版本 2	2024 年 8 月 7 日
RC 1 版本	Partner AWS Central API 發行的候選版本 1	2024 年 6 月 21 日
Beta 5 版	已發行AWS Partner Central API 的 Beta 5	2024 年 4 月 19 日
Beta 4 版本	已發行AWS Partner Central API 的 Beta 4	2024 年 3 月 4 日
Beta 3 版	已發行AWS Partner Central API 的 Beta 3	2024 年 1 月 24 日
Beta 2 版本	已發行AWS Partner Central API 的 Beta 2	2024 年 1 月 7 日

Beta 1 版本	已發行AWS Partner Central API 的 Beta 1	2023 年 12 月 15 日
Beta 0 版本	AWS Partner Central API 初始版本	2023 年 11 月 15 日