デベロッパーガイド

AWS Partner Central

AWS Partner Central: デベロッパーガイド

Table of Contents

AWS パートナーセントラル API リファレンスが再構築されました。サポートされている API オペレーションの詳細については、 AWS パートナーセントラル API リファレンスを参照してください。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。

AWS Partner Central デベロッパーガイドへようこそ

このデベロッパーガイドでは、AWSパートナーがサービス APIs を使用してビルド、マーケティング、販売、成長のアクティビティを と統合および管理する方法を説明しますAWS。

トピック

- AWS SDK でのAWS Partner Central API の使用
- セットアップと認証

AWS SDK でのAWS Partner Central API の使用

AWS Software Development Kit (SDKsは、多くの一般的なプログラミング言語で使用できます。各 SDK には、デベロッパーが好みの言語でアプリケーションを簡単に構築できるようになる API、コード例、およびドキュメントが提供されています。

SDK ドキュメント	コード例
AWS SDK for C++	AWS SDK for C++コード例
AWS CLI	AWS CLIコード例
AWS SDK for Go	AWS SDK for Goコード例
AWS SDK for Java	AWS SDK for Javaコード例
AWS SDK for JavaScript	AWS SDK for JavaScriptコード例
AWS SDK for Kotlin	AWS SDK for Kotlinコード例
AWS SDK for .NET	AWS SDK for .NETコード例
AWS SDK for PHP	AWS SDK for PHPコード例
AWS Tools for PowerShell	AWS Tools for PowerShellコード例
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3)コード例
AWS SDK for Ruby	AWS SDK for Rubyコード例

SDK ドキュメント	コード例
AWS SDK for Rust	AWS SDK for Rustコード例
AWS SDK for SAP ABAP	AWS SDK for SAP ABAPコード例
AWS SDK for Swift	AWS SDK for Swiftコード例

> 可用性の例
>
> 必要なものが見つからなかった場合。このページの下側にある [Provide feedback] リンクから、コードの例をリクエストしてください。

セットアップと認証

AWS Partner Central API でのセットアップと認証には 3 つのステップがあります。プロセスの概要を次に示します。

1. Seller AWS Marketplaceアカウントを Partner Central にリンクします。
2. IAM を使用してアクセス許可を設定します。
3. Signature Version 4 (SigV4) を使用して API コールを認証します。

AWS アカウントを Partner Central にリンクする

AWS アカウントを Partner Central にリンクすることは、 API を使用するための前提条件です。詳細については、AWS「Partner Central アカウントとAWS Marketplace販売者アカウントのリンク」を参照してください。アライアンスリーダーまたはクラウド管理者のアクセス許可を持つアカウントで Partner Central にサインインし、 **`Account Linking`**セクションに移動して、プロンプトに従う必要があります。

IAM の設定

AWS Partner Central API を使用するには、(AWS Identity and Access Management IAM) ロールまたは IAM ユーザーが必要です。詳細については、「IAM はいつ使用できますか?」を参照してください。このAWS アカウントガイドの「IAM ロールの作成」および「IAM ユーザーの作成」の手順に

従います。Partner Central にリンクされたAWS Marketplace Seller アカウントで、この IAM ロール/ユーザーを作成する必要があります。IAM ロール/ユーザーの作成にはコストはかかりません。

1. IAM ロール/ユーザーを作成する

   にサインインしAWS Management Console、IAM サービスに移動し、手順に従って IAM ロールまたは IAM ユーザーを作成します。
2. ポリシーの割り当て:

   必要に応じて、 管理ポリシーをアタッチするか、カスタムポリシーを作成します。アクセス許可を変更または拡張するには、 からコンテンツをコピーして他のアクセス許可と組み合わせる代わりに、追加のポリシーを IAM ロールに適用`AWSPartnerCentralOpportunityManagement`します。マネージドポリシーの複製は避けてください。これにより、リリースされた新機能に自動的にアクセスできなくなり、今後ポリシーを手動で更新する必要があります。アクセスポリシーの詳細については、アクセスコントロールのドキュメントを参照してください。

AWS Marketplaceオファーの管理

AWS Marketplaceオファーを管理し、オポチュニティにリンクするには、パートナーは Catalog APIs にアクセスするためのアクセス許可を IAM ロールに付与する必要があります。ロール/ユーザーに `aws-marketplace:ListEntities`や などのアクセス許可があることを確認します`aws-marketplace:SearchAgreements`。

API コールの認証

AWS Partner Central API は、認証に Signature Version 4 (SigV4) を使用します。実装方法は次のとおりです。

AWS SDK の使用

AWS SDKsを自動的に処理します。AWS認証情報を入力すると、SDK が残りを行います。

1. Java については、AWS「 SDK for Java への一時的な認証情報の提供」を参照してください。
2. Python (Boto3) については、「認証情報」を参照してください。
3. JavaScript (Node.js) については、「Node.js での認証情報の設定」を参照してください。
4. .NET については、「認証情報とプロファイルの解決」を参照してください。
5. 他のプログラミング言語やその他の例については、「 で構築するツールAWS」を参照してください。

AWS SDK を使用しない認証

選択したプログラミング言語でAWS SDK を使用できない場合、認証には正規リクエストを手動で作成し、リクエストに署名し、セッショントークンを処理します。SigV4 署名を使用するための包括的なガイダンスAWSを提供します。ただし、手動リクエスト署名を使用すると複雑さが増し、セキュリティトークンを慎重に管理する必要があるため、AWS SDK を使用することをお勧めします。

awsJson1_0 プロトコルのすべてのリクエストは、次のヘッダーを使用して HTTP "POST" メソッドを使用してルート URL (/) に送信する必要があります。

必要なヘッダー

awsJson1_0 プロトコルのすべての API リクエストは、次のヘッダーを使用して HTTP "POST" メソッドを使用してルート URL (/) に送信する必要があります。

ヘッダー	必要	説明
Content-Type	true	このヘッダーの静的値は ですapplication/x-amz-json-1.0 。
Content-Length	true	RFC 9110#section-8.6 で定義される標準の Content-Length ヘッダー。
X-Amz-Target	リクエストの場合は true	たとえば、 サービスの オペレーションCreatePartner の値は PartnerCentralAccount で すPartnerCentralAccount.CreatePartner 。

カスタム user-agent を使用した通話の署名

AWS Partner Central に API リクエストを行うときは、クライアントアプリケーションのソースAWSの特定、使用状況のモニタリング、パフォーマンスの監査に役立つ X-Amzn-User-Agentヘッダーを含めることをお勧めします。 はこのヘッダーAWSを使用して、呼び出しを行うクライアントアプリケーションのタイプを区別し、さまざまなクライアント実装の成功率に関するインサイトを収集します。

カスタムユーザーエージェントヘッダー

ヘッダー名: X-Amzn-User-Agent

目的: API リクエストを行うクライアントのタイプを区別し、インタラクションのソースを分類します。

形式: `{Integrator}|{Product}`

値の例: `AWS|AWS Partner CRM Connector`

すべてのリクエストにこのヘッダーを含めるとAWS、 はリクエストパターンを分析し、統合を追跡し、さまざまなクライアントシステムの API エクスペリエンスを向上させることができます。

SDKs

SDK 呼び出しに `X-Amzn-User-Agent`ヘッダーを含めるには、API 呼び出しを行う前にクライアントリクエストの動作を変更できます。AWS SDK for Python (Boto3) を使用した API の販売例を以下に示します。

```
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
```

この例では、Boto3 SDK にイベントを登録して、すべての API リクエストに `X-Amzn-User-Agent`ヘッダーを自動的に追加する方法を示します。他のAWS SDKs には、それぞれのリクエストインターセプションメカニズムを変更することで、同じアプローチを適用できます。

> **Note**
>
> `X-Amzn-User-Agent` ヘッダー形式が簡素化されました。新しい形式を使用することをお勧めします。以前の形式 (`CompanyName|ProductName|CRMName|ProductVersion`) は、下位互換性のために引き続きサポートされています。既存の統合は変更なしで引き続き機能します。

Partner Central エージェント MCP サーバー

Partner Central エージェント MCP Server は、Model Context Protocol (MCP) を通じて Partner Central ツールを提供し、AI エージェントとツールが自然言語を通じて機会管理、顧客インサイト、資金プログラムを検出して操作できるようにします。

サーバーは、ワークフローを合理化しながら制御を維持しながら、書き込みオペレーションの認証、セッション管理、ヒューhuman-in-the-loop承認を処理します。

概要

Partner Central エージェント MCP Server は、AWS MCP 互換 AI クライアントを Partner Central エージェントに接続するフルマネージド型のホスト型サービスです。SigV4 認証で HTTPS 経由の JSON-RPC 2.0 を使用し、リアルタイムストリーミングレスポンスのサーバー送信イベント (SSE) をサポートします。

サーバーは、マルチターン会話、ドキュメント分析用のファイル添付ファイル、および書き込みオペレーションを実行する前に明示的な同意を必要とする組み込みの承認ワークフローをサポートします。

エージェント機能

- パイプラインインサイト — リスクのある機会、ステージの進行状況、クローズドロスト分析など、セールスパイプラインに関する会話型インテリジェンスを取得します。
- オポチュニティの作成 — 自然言語の会話、会議のメモ、提案、通話のトランスクリプトのアップロード、既存のオポチュニティのクローン作成により、新しいオポチュニティを作成します。
- オポチュニティの概要 — ステージ、支出、終了日などをカバーするすべての取引の簡潔at-a-glanceわかる概要を生成します。
- セールスプレイの生成 — 機会の詳細、業界コンテキスト、AWSソリューションのレコメンデーションを組み合わせて、カスタマイズされたセールス戦略を構築する
- 顧客プロファイルの作成 — 業界、ビジネスモデル、地域、最近の開発に関する公開情報を使用して企業プロファイルを生成します。
- ソリューションのレコメンデーション — 登録済みのソリューションを機会要件と照合して、最適なソリューションを見つける
- 資金に関する推奨事項 — 利用可能なAWS資金プログラムと照らし合わせて機会を評価し、金額を見積もり、資金リクエストを作成します。

- 次のステップの推奨事項 —AWS共同販売基準とステージ進行ガイダンスに基づく優先順位付けされたアクションプランを取得する
- オポチュニティの進行 — パイプラインステージを通じて、サポートドキュメントをアップロードし、関連データを抽出し、オポチュニティを進行させる
- AI 支援製品出品 — 既存のデジタルアセットから高品質のAWS Marketplace製品出品を生成し、AWS Marketplace標準に対するリストの強度をスコア付けし、検出可能性を向上させるためのフィールドレベルのレコメンデーションを受け取ります。詳細については、「 AWS Marketplace 販売者ガイド」の「AI 支援製品リスト」を参照してください。

主な利点

- Partner Central への会話アクセス — 複雑なコンソールワークフローをナビゲートするのではなく、自然言語で質問する
- 販売時間を増やす — 複数ステップのフォームに記入するのではなく、短い会話を通じて機会を作成します。これにより、データ入力が減り、パートナーの営業チームが販売により多くの時間を費やせるようになり、エージェントは改善を推奨して、パートナーが質の高い機会を提供し、パイプラインの衛生状態を改善します。
- Human-in-the-loop safety — すべての書き込みオペレーションには、実行前に明示的な承認が必要です
- マルチターン会話 — コンテキストを繰り返すことなく、セッション内のクエリを絞り込む
- ファイル分析 — エージェントが質問とともに分析するドキュメント (PDF、DOCX、XLSX、CSV、イメージ) をアタッチします。
- 一元化されたアクセス管理 — きめ細かなアクセス許可で IAM ポリシーを通じてアクセスを制御する
- サンドボックステスト — 本番稼働用データにアクセスする前に、分離されたサンドボックス環境でワークフローをテストする
- ストリーミングレスポンス — エージェントがリクエストを処理するときに SSE 経由でフィードバックを取得します。

使用例

AI が生成するすべてのインサイトには、トレーサビリティのためのセッション ID が含まれています。データは、ログインしているパートナー自身の機会に分離されます。すべてのコンテンツには明確な開示ラベルが付いており、AWS責任ある AI ポリシーによって管理されます。

オポチュニティの作成

エージェントは、自然言語の説明またはアップロードされたドキュメント (PDF、DOCX、XLSX、TXT) から新しい機会を作成します。顧客名、プロジェクトスコープ、予定終了日、その他の必須フィールドを抽出し、公開されているウェブデータから顧客の詳細を強化し、AWS の準備状況要件に照らしてドラフトを検証し、承認後に Partner Central Selling API を通じてオポチュニティを作成します。

- 「Acme Corp のオポチュニティを作成する — データウェアハウスを Redshift に移行したい、第 Q3のターゲット終了日、予想される AWS の月額支出 40,000 USD」
- 「昨日の GlobalTech との会議からの通話メモです。このトランスクリプトから機会を作成します」
- 「この提案の PDF を使用して、顧客の SAP 移行の機会を作成します」

オポチュニティのクローン作成

エージェントは既存のオポチュニティから新しいオポチュニティを作成し、指定した新しい顧客またはプロジェクトの詳細にマージし、送信前に少なくとも 1 つの差別化フィールドが変更されていることを検証して、重複が AWS レビューによって拒否されないようにします。

- 「新規顧客のクローンオポチュニティ O1234567890 — Globex、同じワークロード、第 Q4のターゲット終了日」
- O9876543210 のようなオポチュニティを作成するが、サンドボックスではなく顧客の本番環境用」

パイプラインインサイト

販売パイプラインに関する会話型インテリジェンスを取得します。エージェントはステージの進行状況、期限、停滞した取引、閉鎖された失われたパターンを分析して、最も重要なものを明らかにします。

- 「今週はどの機会に注意が必要ですか？」
- 「来月閉鎖される機会はいくつありますか？」
- 「過去 6 か月間に機会が失われた主な理由は何ですか？」

オポチュニティの概要

エージェントは、会社名、業界、ステージ、予想される月額AWS支出、ターゲット終了日、およびその他の主要な詳細を簡潔な概要に合成します。個々のフォームフィールドをスキャンする必要はありません。

- 「機会 O1234567890 の概要を教えてください」
- 「Acme Corp 取引の現在のステータスとキーの詳細は何ですか？」

セールスプレイの生成

エージェントは、オポチュニティの詳細、顧客の業界背景、関連するAWSソリューションのレコメンデーションをready-to-useセールスプレイに結合することで、カスタマイズされたセールス戦略を構築します。

- 「オポチュニティ O1234567890 のセールスプレイを生成する」
- 「この金融サービスの顧客にクラウド移行を販売するための最善のアプローチは何ですか？」
- GlobalTech データ分析取引の販売戦略を構築する」

顧客プロファイルの作成

エージェントは、業界分類、ビジネスモデル (B2B/B2C/ハイブリッド) 、地理的プレゼンス、企業規模、市場フォーカス、最近のビジネス開発など、公開されている情報を使用して企業プロファイルを生成します。プロファイルには「公開されているデータとAWS AI インサイトで生成」というラベルが付けられています。

- 「Acme Corp の顧客プロファイルを作成する」
- 「この顧客の業界とビジネスモデルについて何を知っていますか？」
- GlobalTech との今後の会議の会社概要をまとめる」

ソリューションの推奨事項

エージェントは、登録されたソリューションを機会要件と相互参照し、ソリューション名、説明、およびそれが既に機会にアタッチされているかどうかを示します。

- 「オポチュニティ O1234567890 に最も適したソリューションはどれですか？」

- 「当社のデータ分析ソリューションは、すでにこの取引にアタッチされていますか？」
- 「SAP ワークロードの移行を検討しているお客様にソリューションを提案する」

資金のレコメンデーション

エージェントは、機会の詳細とプログラムの適格性基準に基づいて、利用可能なAWS資金プログラムと照らし合わせて各機会を評価します。一致が見つかると、プログラム名、説明、詳細な推論が表示されます。その後、資金の見積もり、自動入力された資金リクエストのドラフトの作成、プログラムの詳細の会話を行うことができます。SCA (戦略的コラボレーション契約) の予算の可用性は、該当する場合に表示されます。すべてのアクションは IAM アクセス許可を尊重します。

- 「オポチュニティ O6789012345 に関する資金プログラムの対象ですか？」
- 「この顧客と POC の資金額を見積もる」
- 「この機会に MAP メリットアプリケーションを作成する」

次のステップの推奨事項

エージェントは、機会とAWSステージ進行ガイダンスを評価し、現在のデータを適切な機会の基準と比較し、まだ収集する必要がある情報を正確に特定します。その結果、AWS共同販売基準に基づく優先順位付けされたアクションプランが作成されます。

- 「オポチュニティ O1234567890 を前進させるには、次に何をする必要がありますか？」
- 「この機会は送信する準備ができていますか? どのフィールドが欠落していますか？」
- 「この取引を Prospect から Qualified に移行するための要件は何ですか？」

機会の進行

エージェントは、サポートドキュメント (会議のトランスクリプト、通話メモ、E メールの概要) を受け入れ、関連情報を抽出し、オポチュニティフィールドにマッピングし、ステージ要件を評価し、オポチュニティを更新します。ギャップが残っている場合は、満たされた要件と満たされていない要件の内訳を返します。

- 「私の通話メモ — オポチュニティ O1234567890 を関連する詳細で更新する」
- 「会議のトランスクリプトをアタッチしています。説明した内容に基づいて、この機会を進めます。」

- 「この E メールの概要を確認し、満たす機会フィールドを教えてください」

パートナーオンボーディングのエージェントエクスペリエンス

エージェントは、Partner Central へのパートナーのオンボーディングを自動化し、 Marketplace 販売者のセットアップと PRM コンプライアンスに関するガイド付き支援を、すべて自然言語を通じて提供します。

エージェント機能:

- パートナープロファイルの自動化 — ウェブサイトをスキャンし、パートナープロファイルを自動入力し、可視性、提携リードの連絡先、トレーニングドメイン、アカウント接続を管理します。
- 販売者セットアップガイダンス — 納税申告書、銀行業務、コンプライアンス (KYC/BAV/SU)、ESC カタログ、サービスにリンクされたロールを含む Marketplace 販売者登録のStep-by-stepガイダンス
- PRM コンプライアンスガイダンス — PRM の準備状況の評価、収益タグ付け用の製品コードの取得、連結収益属性の子会社アカウント接続の検証

パートナープロファイルの自動化

エージェントは会社ウェブサイトをスキャンしてパートナープロファイルを抽出して自動入力し、残りのギャップをガイドします。プロファイルフィールドの更新、可視性の変更、提携リードの連絡先の管理、トレーニング認定ドメインのリンク、アカウント接続の招待の処理を行うことができます。

- 「ウェブサイトを見てプロファイルを入力できますか？」
- 「AWS のお客様が見つけられるようにプロファイルを公開する」
- 「提携リードの連絡先を [Email] の [name] に更新する」
- 「EMEA 子会社アカウントを接続する」

販売者セットアップガイダンス

エージェントは、マーケットプレイス販売者の登録end-to-endで確認し、国とエンティティタイプに基づいて適切な納税申告書を決定し、リージョン別に銀行業務と支払いの設定を説明し、適用されるコンプライアンス要件 (KYC、銀行口座検証、セカンダリユーザー検証) を特定します。

- 「マーケットプレイスで販売を開始する — どこから始めればいいですか？」

- 「どのような納税申告書が必要ですか? ドイツの会社です」
- 「KYC は必要ですか? フランスの顧客に販売しています」
- 「米国以外で支払いを設定する方法を教えてください。」

PRM コンプライアンスガイダンス

エージェントは、アカウントが PRM 対応であるかどうかをチェックします。アカウントが接続され、リストが存在し、製品コードがタグ付け可能であることを確認します。子会社アカウントを持つパートナーの場合、すべての販売者アカウントが連結収益属性に適切にリンクされていることを確認します。

- 「PRM に準拠していますか？」
- 「タグ付け用の製品コードはどこにありますか？」
- 「PRM に準拠するために実行する必要があるすべてのステップを教えてください」

はじめに

Partner Central エージェント MCP サーバーを設定する準備はできましたか? ステップstep-by-step のセットアップ手順については、「開始方法ガイド」を参照してください。

Partner Central Agent MCP サーバーの開始方法

このガイドでは、カスタム MCP クライアントを使用して Partner Central Agent MCP Server へのプログラムによるアクセスを設定する方法について説明します。サーバーは SigV4 認証で直接 HTTPS を使用します。プロキシや IDE プラグインは必要ありません。

前提条件

開始する前に、以下の準備が整っていることを確認します。

- アクティブな Partner Central アカウント (AWSコンソールに移行)
- Partner Central の IAM アクセス許可を持つAWSアカウント
- AWS CLI がインストールされ、認証情報で設定されている
- us-east-1 (バージニア北部) リージョンへのアクセス
- への HTTPS 接続 `partnercentral-agents-mcp.us-east-1.api.aws`

- HTTP クライアントでの TLS 1.2 以降のサポート
- JSON-RPC 2.0 および SigV4 リクエスト署名をサポートする MCP 互換クライアント

ステップ 1: IAM アクセス許可を設定する

Partner Central エージェント MCP サーバーには、プロトコルアクセス (MCP エンドポイントと通信するため) とデータアクセス (Partner Central オペレーションを実行するため) の 2 つのレベルで IAM アクセス許可が必要です。

IAM ポリシーのアタッチ

AWSマネジメントコンソールを使用して IAM ID にポリシーをアタッチするには:

1. [IAM コンソール] を開きます。
2. 左側のナビゲーションペインで、ポリシーをアタッチするアイデンティティに応じてユーザー、ユーザーグループ、またはロールを選択し、特定のユーザー、グループ、またはロールの名前を選択します。
3. [アクセス許可] タブを選択します。
4. ポリシーをアタッチする (または、初めてアクセス許可を追加する) を選択します。
5. ポリシーリストで、アタッチする管理ポリシー (以下の JSON ブロックから作成したカスタムポリシーなど) を検索して選択します。
6. 確認するには、ポリシーのアタッチ (または次へ、アクセス許可の追加) を選択します。

アクセス許可はすぐに有効になります。同じ ID に複数のポリシーをアタッチできます。

推奨: 管理ポリシーを使用する

MCP プロトコルアクセスを許可する最も簡単な方法は、`AWSMcpServiceActionsFullAccess`マネージドポリシーを IAM アイデンティティにアタッチすることです。このポリシーには、MCP サーバーとやり取りするために必要なすべてのアクセス許可が含まれます。

きめ細かな制御のために、IAM ポリシーの `aws:IsMcpServiceAction`条件キーを使用して、特に MCP サービスアクションに対するアクセス許可の範囲を設定できます。

MCP プロトコルアクセスの最小アクセス許可

少なくとも、IAM ID が MCP サーバーとやり取りするには、このアクションが必要です。

[アクション]	説明
partnercentral:UseSession	会話セッションを作成、更新、取得するために必要です

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "partnercentral:UseSession"
            ],
            "Resource": "*",
   "Condition": {
                "Bool": {
                    "aws:IsMcpServiceAction": "true"
                }
            }
        }
    ]


}
```

データアクセス許可

エージェントを使用して Partner Central オペレーションを実際に実行するには、ユースケースに基づいて追加のアクセス許可が必要です。

オポチュニティ管理:

```
{
    "Effect": "Allow",
    "Action": [
        "partnercentral:List*",
        "partnercentral:Get*",
        "partnercentral:CreateOpportunity",
        "partnercentral:UpdateOpportunity",
        "partnercentral:SubmitOpportunity",
        "partnercentral:AssignOpportunity",
```

```
        "partnercentral:AssociateOpportunity",
        "partnercentral:DisassociateOpportunity"
    ],
    "Resource": "*"
}
```

資金プログラム:

```
{
    "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 アクセス:

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

Partner Central オンボーディング:

```
{
```

```
    "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": "*"
}
```

Marketplace 販売者のセットアップでは、以下も追加します。

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

サービスにリンクされたロール管理 (再販認可/CPPO) の場合:

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

```
        "iam:GetRole",
        "iam:CreateServiceLinkedRole"
    ],
    "Resource": "*"
}
```

フルアクセスポリシー

開発とテストでは、すべてのアクセス許可を 1 つのポリシーにまとめることができます。

```
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": "*"
            }
        ]
    }'
```

読み取り専用ポリシー

本番環境または読み取り専用のユースケースでは、読み取りオペレーションへのアクセス許可を制限します。

```
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": "*"
            }
        ]
    }'
```

ステップ 2: MCP クライアントを接続する

Partner Central Agent MCP Server は、SigV4 リクエスト署名で直接 HTTPS を使用します。プロキシレイヤーはありません。MCP クライアントは JSON-RPC 2.0 リクエストをエンドポイントに直接送信します。

Endpoint

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

認証

すべてのリクエストは、以下を使用してAWS署名バージョン 4 で署名する必要があります。

- サービス名: `partnercentral-agents-mcp`
- リージョン : `us-east-1`

MCP 接続を初期化する

リクエストを送信`initialize`してプロトコルを確立します。

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

予想されるレスポンス

```
{
    "jsonrpc": "2.0",
```

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

使用可能なツールを一覧表示する

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

MCP ヘッダーを使用した通話の署名

Partner Central エージェント MCP にリクエストを行うときは、次の方法を使用してカスタム MCP ヘッダーを含めることをお勧めします。これは、クライアントアプリケーションのソースAWSの特定、使用状況のモニタリング、パフォーマンスの監査に役立ちます。 はこのヘッダーAWSを使用して、呼び出しを行うクライアントアプリケーションのタイプを区別し、さまざまなクライアント実装の成功率に関するインサイトを収集します。

方法 1: _meta フィールド (プログラム/ステートレス MCP)

MCP `tools/call`リクエストを直接構築するコードの場合は、リクエストの `_meta`フィールドを指定します。

```
{
    "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 (セッションベースのカスタムエージェント)

セッションを確立するカスタム MCP クライアントの場合は、 `clientInfo`フィールド内に MCP ヘッダー情報を指定します。

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

のフィールド`clientInfo`:

- `integrator` — パートナーの会社名または「Direct」

  例: `AWS`
- `sourceProduct` — 製品/エージェント名

  例: `AWS CRM Connector`

方法 3: URL パラメータ (ホストされた MCP のみ)

インテグレーターがプロトコルフィールドを変更できないホストされた MCP クライアントのみ。URL パラメータを使用します。

サーバー URL: `https://mcp.partnercentral.aws?appId=<Integrator's Company Name / Direct>`

ステップ 3: セットアップを確認する

簡単なメッセージを送信して、すべてが機能していることを確認します。Sandbox カタログをテストに使用します。

```
{
    "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"
        }
    }
}
```

`"status": "complete"` とエージェントからのテキスト返信を受信した場合、セットアップは正常に動作しています。レスポンスには、フォローアップメッセージ`sessionId`に使用できる も含まれます。

ステップ 4: 最初のタスクを実行する

オポチュニティをクエリする

```
{
    "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"
        }
    }
}
```

資金の適格性を確認する

```
{
    "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"
        }
    }
}
```

セッション履歴を取得する

```
{
```

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

パートナーオンボーディング

```
{
    "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"
        }
    }
}
```

試すその他のオンボーディングタスク:

- 「面接プロセスを案内する」
- 「ウェブサイトを見てパートナープロファイルを入力できますか？」
- 「 Marketplace で販売する準備を整えるには、まだ何をする必要がありますか？」

セキュリティに関する考慮事項

- MCP ツールパラメータを介してAWS認証情報を渡さないでください。認証は、トランスポートレイヤーで SigV4 リクエスト署名によって処理されます。
- テストと開発にはサンドボックスカタログを使用します。"Sandbox" カタログは、本番パートナーデータに影響を与えない分離された環境を提供します。
- 最小特権の IAM ポリシーを本番環境に適用します。ユースケースのモニタリングと報告には、読み取り専用ポリシーを使用します。ユーザーが機会を更新したり、資金調達アプリケーションを送信したりする必要がある場合にのみ、書き込みアクセス許可を付与します。
- 書き込みオペレーションを慎重に確認してください。サーバーは、すべての書き込みオペレーションにhuman-in-the-loop承認を使用します。書き込みアクションが提案された場合は、承認する前にパラメータを確認してください。
- セッションデータは一時的なものです。セッションは作成から 48 時間後に期限切れになります。長期データストレージのセッションに依存しないでください。
- ファイルのアップロードはエフェメラル S3 バケットに送信されます。アップロードされたファイルは一時的に保存され、永続的に保持されません。認証情報、シークレット、またはその他の機密情報を含むファイルをアップロードしないでください。

次の手順

- 設定リファレンス — エンドポイント、IAM アクション、セッション管理、エラーコードの完全なリファレンス
- ツールリファレンス — `sendMessage`および `getSession` ツールの詳細なドキュメント

設定リファレンス

このページでは、 Partner Central Agent MCP Server に接続して設定するための完全なリファレンスを提供します。

Endpoint

プロパティ	値
[URL]	`https://partnercentral-agents-mcp.us-east-1.api.aws/mcp`
リージョン	`us-east-1` (バージニア北部) のみ
プロトコル	HTTPS 経由の JSON-RPC 2.0
ストリーミング	サーバー送信イベント (SSE)
認証	AWS署名バージョン 4
SigV4 サービス名	`partnercentral-agents-mcp`
TLS 要件	TLS 1.2 以降

IAM アクセス許可

特に明記されていない限り、すべての IAM アクションは `partnercentral:` プレフィックスを使用します。

MCP プロトコルアクセス

MCP プロトコルアクセスを許可する最も簡単な方法は、`AWSMcpServiceActionsFullAccess`マネージドポリシーを IAM アイデンティティにアタッチすることです。このポリシーには、MCP サーバーとやり取りするために必要なすべてのアクセス許可が含まれます。きめ細かな制御のために、IAM ポリシーの `aws:IsMcpServiceAction`条件キーを使用して、特に MCP サービスアクションに対するアクセス許可の範囲を設定できます。

[アクション]	説明
`partnercentral:UseSession`	会話セッションの作成、更新、取得

オポチュニティ管理

[アクション]	説明
partnercentral:List*	機会、ソリューション、その他の Partner Central リソースを一覧表示する
partnercentral:Get*	個々の機会やその他のリソースの詳細を取得する
partnercentral:CreateOpportunity	オポチュニティを作成する
partnercentral:UpdateOpportunity	機会フィールド (ステージ、終了日、収益など) を変更する
partnercentral:SubmitOpportunity	AWSレビューの機会を送信する
partnercentral:AssignOpportunity	パートナー担当者にオポチュニティを割り当てる
partnercentral:AssociateOpportunity	オポチュニティを別のリソース (ソリューションなど) にリンクする
partnercentral:DisassociateOpportunity	オポチュニティと別のリソース間のリンクを削除する
partnercentral:StartEngagementFromOpportunityTask	機会からエンゲージメントを開始する機会を送信する

資金プログラム

[アクション]	説明
partnercentral:ListBenefitAllocations	アカウントの利用可能なメリット配分を一覧表示する
partnercentral:ListBenefitApplications	送信された利点アプリケーションを一覧表示する

[アクション]	説明
partnercentral:CreateBenefitApplication	新しいメリットアプリケーションを作成する (MAP、POC、WMP)
partnercentral:GetBenefitApplication	特定の利点アプリケーションの詳細を取得する
partnercentral:UpdateBenefitApplication	保留中の特典アプリケーションを更新する
partnercentral:AssociateBenefitApplicationResource	リソースをメリットアプリケーションにリンクする
partnercentral:DisassociateBenefitApplicationResource	メリットアプリケーションからリソースリンクを削除する

Marketplace アクセス

[アクション]	説明
aws-marketplace:DescribeEntity	マーケットプレイスエンティティの詳細を取得する
aws-marketplace:SearchAgreements	マーケットプレイス契約を検索する
aws-marketplace:ListEntities	マーケットプレイスエンティティを一覧表示する

オンボーディングエージェント

パートナーアカウント管理 (**partnercentral**):

[アクション]	説明
partnercentral:ListPartners	アカウントのパートナー登録を一覧表示する

[アクション]	説明
partnercentral:GetPartner	パートナー登録とプロファイルの詳細を取得する
partnercentral:GetProfileVisibility	現在のプロファイルの可視性を取得する (PUBLIC/PRIVATE)
partnercentral:GetAllianceLeadContact	提携リードの連絡先情報を取得する
partnercentral:GetProfileUpdateTask	保留中の非同期プロファイルの更新のステータスを確認する
partnercentral:StartProfileUpdateTask	パートナープロファイルの更新を送信する (非同期)
partnercentral:CancelProfileUpdateTask	保留中のプロファイル更新タスクをキャンセルする
partnercentral:PutProfileVisibility	プロファイルの可視性を PUBLIC または PRIVATE に変更する
partnercentral:PutAllianceLeadContact	提携リードの連絡先 (名前、タイトル、E メール) を更新する
partnercentral:SendEmailVerificationCode	E メールベースのドメイン/連絡先の変更の検証コードを送信する
partnercentral:AssociateAwsTrainingCertificationEmailDomain	E メールドメインをリンクしてトレーニング証明書を追跡する
partnercentral:DisassociateAwsTrainingCertificationEmailDomain	リンクされたトレーニング証明書 E メールドメインを削除する
partnercentral:GetAccountConnections	アクティブなアカウント接続を一覧表示する (例: 子会社リンク)

[アクション]	説明
partnercentral:GetConnectionInvitations	保留中の送受信された接続の招待を一覧表示する
partnercentral:CreateConnectionInvitation	接続の招待を別のAWSアカウントに送信する
partnercentral:CancelConnectionInvitation	保留中の送信接続の招待をキャンセルする
partnercentral:RespondConnectionInvitation	受信接続の招待を承諾または拒否する
partnercentral:CancelConnection	アクティブなアカウント接続をキャンセルする
partnercentral:ManageConnectionPreferences	接続されたアカウント間の共有設定を取得または更新する

Marketplace 販売者セットアップ (**aws-marketplace**):

[アクション]	説明
aws-marketplace:ListEntities	Marketplace エンティティ (販売者エンティティなど) を一覧表示する
aws-marketplace:DescribeEntity	Marketplace エンティティの完全な詳細を取得する
aws-marketplace:DescribeChangeSet	送信された変更セットのステータスを確認する
aws-marketplace:StartChangeSet	Marketplace 変更セットを送信する (例: 販売者プロファイルの更新)

IAM — サービスにリンクされたロール (**iam**):

[アクション]	説明
iam:GetRole	Marketplace 再販認可サービスにリンクされたロールが存在するかどうかを確認する
iam:CreateServiceLinkedRole	AWSServiceRoleForMarketplaceResaleAuthorization ロールを作成する

カタログ環境

カタログ	説明
"AWS"	本番環境。すべてのオペレーションはライブパートナーデータに影響します。
"Sandbox"	分離されたテスト環境。本番データには影響しません。開発と検証に使用します。

セッション管理

プロパティ	値
セッション ID 形式	session- プレフィックス付き UUID v4 (例: session-550e8400-e29b-41d4-a716-446655440000)
セッションの作成	を使用しない最初のsendMessage 呼び出し時に自動 sessionId
セッションの有効期限	セッション作成から 48 時間 (非アクティブベースではなく絶対有効期限)

ファイルのアップロード

プロパティ	値
メッセージあたりの最大ファイル数	3
イメージサイズ制限	3.75 MB
ドキュメントサイズの制限	4.5 MB
許可された拡張機能	doc, docx, pdf, png, jpeg, xlsx, csv, txt
S3 バケット (本番稼働用)	aws-partner-central-marketplace-ephemeral-writeonly-files
アップロードパス	s3://{bucket}/{aws-account-id}/
S3 URI 要件	versionId クエリパラメータを含める必要があります

ワークフローをアップロードする

1. AWSアカウント ID プレフィックスの下にある適切な S3 バケットにファイルをアップロードします。
2. アップロードによってversionId返された を含む S3 URI に注意してください。
3. sendMessage 呼び出しに ファイルをdocumentコンテンツブロックとして含めます。

S3 URI 形式の例:

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

エラーコード

コード	名前	説明
-32001	AUTHENTICATION_FAILURE	SigV4 署名が無効であるか、認証情報の有効期限が切れています
-31004	TOOL_PERMISSION_DENIED	IAM ID にこのオペレーションに必要なpartnercentral: アクションがない
-32002	ACCESS_DENIED	一般アクセス拒否 (アカウント未登録、リージョンの不一致など)
-32004	LIMIT_EXCEEDED	レート制限またはクォータを超えました。エクスポネンシャルバックオフで再試行します。
-30001	RESOURCE_NOT_FOUND	リクエストされたリソース (セッション、オポチュニティなど) が存在しない
-32600	INVALID_REQUEST	不正な形式の JSON-RPC リクエストまたは無効なパラメータ
-32603	INTERNAL_ERROR	サーバー側のエラー。リクエストを再試行します。

レート制限

サーバーはアカウントごとのレート制限を適用します。

運用	持続率	[Burst] (バースト)
sendMessage	1 分あたり 2 個のリクエスト	10
その他すべてのオペレーション	1 分あたり 10 リクエスト	20

これらの制限を超えると、サーバーはエラーコード -32004 (LIMIT_EXCEEDED) を返します。クライアントにジッターを含むエクスポネンシャルバックオフを実装して、レート制限を適切に処理します。

ストリーミング (SSE) イベントタイプ

sendMessage 呼び出しtrueで が に設定されている場合、サーバーstreamは次のイベントタイプを持つサーバー送信イベントを返します。

イベントタイプ	説明
stream_start	ストリーム接続が確立されました
assistant-response.start	エージェントがレスポンスの生成を開始しました
assistant-response.delta	エージェントのレスポンスの増分テキストチャンク
assistant-response.completed	エージェントがレスポンスの生成を終了しました
server-tool-use	エージェントが内部ツールを呼び出しています (読み取りオペレーション)
server-tool-response	内部ツール呼び出しの結果
tool_approval_request	エージェントが書き込みオペレーションの人間による承認をリクエストしている
stream_end	ストリーム接続の終了

イベントタイプ	説明
done	SSE ストリームが完了したことを示す最終イベント

次の手順

- ツールリファレンス — sendMessageおよび getSession ツールの詳細なドキュメント

ツールリファレンス

Partner Central Agent MCP Server は、sendMessageすべてのエージェントインタラクションとセッション状態の取得getSessionの 2 つの MCP ツールを公開します。Partner Central のすべてのオペレーション — オポチュニティクエリ、資金調達アプリケーション、ドキュメント分析 — は、を介して自然言語で処理されますsendMessage。

ツールの概要

ツール	説明	カテゴリ
sendMessage	Partner Central AI エージェントにメッセージを送信します。テキスト、ファイル添付ファイル、ヒューhuman-in-the-loop承認レスポンスをサポートします。	読み取り/書き込み
getSession	会話履歴、イベント、メタデータを含むセッション状態を取得します。	[Read-only]

sendMessage

すべての Partner Central AI エージェントとのやり取りのためのプライマリツール。このツールを使用して、質問をしたり、アクションをリクエストしたり、分析用のドキュメントをアタッチしたり、書き込みオペレーションの承認リクエストに応答したりできます。

エージェントはセッション内で会話コンテキストを保持するため、以前のコンテキストを繰り返すことなくフォローアップの質問をすることができます。

パラメータ

- content (必須) — コンテンツブロックの配列。各ブロックには、ブロック構造を決定するtypeフィールドを含める必要があります。1 つのメッセージ (テキストとドキュメントの添付ファイルなど) に複数のブロックを含めることができます。

  コンテンツブロックタイプ:

タイプ	フィールド	説明
text	type (必須) 、 text (必須)	エージェントに送信されるユーザーメッセージテキスト
document	type (必須) 、 filename (必須) 、 s3Uri (必須)	分析するエージェントのファイルアタッチメント。には versionId パラメータを含めるs3Uri必要があります。
tool_approval_response	type (必須) 、 toolUseId (必須) 、 decision (必須) 、 message (オプション)	ヒューhuman-in-the-loop承認リクエストへの応答

- catalog (必須) — オペレーションのターゲット環境。

  有効な値: "AWS" (本番稼働用) 、 "Sandbox" (テスト用)

- sessionId (オプション) — 続行する既存のセッションを識別する UUID v4。を省略して新しいセッションを作成します。形式: session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

  デフォルト: 新しいセッションが自動的に作成されます。

- `stream` (オプション) — リアルタイムレスポンス配信のサーバー送信イベント (SSE) ストリーミングを有効にします。

  有効な値: `true`、`false`

  デフォルト: `false`

[応答]

の応答には以下が含まれます。

フィールド	説明
`sessionId`	フォローアップメッセージのセッション識別子
`status`	レスポンスステータス: `"complete"` 、`"requires_approval"` 、または `"error"`
`content`	エージェントからのレスポンスコンテンツブロックの配列

例

基本的なテキストメッセージ (新しいセッション)

リクエスト:

```
{
    "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"
        }
    }
}
```

レスポンス:

```
{
    "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"
    }
}
```

フォローアップメッセージ (既存のセッション)

リクエスト:

```
{
    "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"
        }
    }
}
```

ファイルアタッチメント

まずドキュメントを S3 にアップロードし、メッセージで参照します。

```
{
    "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"
        }
    }
}
```

ファイルのアップロードに関する制約:

- メッセージあたり最大 3 ファイル
- イメージサイズ制限: 3.75 MB

- ドキュメントサイズ制限: 4.5 MB
- 許可される拡張機能: doc、docx、、pdf、png、jpeg、xlsxcsv、 txt
- ファイルは にアップロードする必要があります s3://{bucket}/{your-aws-account-id}/
- S3 URI にはversionIdクエリパラメータを含める必要があります

ヒューHuman-in-the-loop承認ワークフロー

エージェントが書き込みオペレーションを実行する必要がある場合 (オポチュニティの更新、資金調達アプリケーションの送信など) 、提案されたアクションの詳細を含む"requires_approval"ステータスを返します。tool_approval_response コンテンツブロックで応答する必要があります。

ステップ 1 — エージェントは承認をリクエストします。

```
{
    "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"
    }
```

```
}
```

ステップ 2 — アクションを承認します。

```
{
    "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"
        }
    }
}
```

ステップ 2 (代替) — アクションを拒否します。

```
{
    "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"
```

```
        }
    }
}
```

ステップ 2 (代替) — カスタムレスポンスで上書きします。

```
{
    "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"
        }
    }
}
```

承認決定値:

決定	動作
"approve"	提案されたパラメータを使用してツールを実行する
"reject"	ツールを実行しないでください。オプションでその理由messageを説明します。
"override"	経由でカスタムレスポンスまたは変更された指示を提供する message

SSE を使用したストリーミング

ストリーミングを有効にして、エージェントがリクエストを処理するときに増分レスポンスチャンクを受信します。

リクエスト:

```
{
    "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
        }
    }
}
```

サーバーは SSE イベントのストリームで応答します。

```
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

完全な会話履歴、イベント、メタデータなど、会話セッションの現在の状態を取得します。これを使用して、セッションの状態を検査したり、過去のインタラクションを確認したり、会話を再開したりできます。

パラメータ

- sessionId (必須) — 取得するセッションの UUID。形式: session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。
- catalog (必須) — セッションが属する環境。

  有効な値: "AWS"、"Sandbox"

[応答]

フィールド	タイプ	説明
sessionId	string	セッション識別子
createdAt	string	セッション作成の ISO 8601 タイムスタンプ
lastActivity	string	最後のアクティビティの ISO 8601 タイムスタンプ

フィールド	タイプ	説明
sequenceNumber	整数	現在のイベントシーケンス番号
stateType	string	現在のセッション状態
events	array	完全な会話履歴 (ユーザーメッセージ、エージェントレスポンス、ツールの使用)
variables	オブジェクト	セッション変数とメタデータ
eventCount	整数	セッション内のイベントの合計数

例

リクエスト:

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

レスポンス:

```
{
    "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\":{}}"
            }
        ]
    }
}
```

エラー処理

すべてのエラーは、JSON-RPC 2.0 エラー形式に従います。

```
{
    "jsonrpc": "2.0",
    "id": 1,
    "error": {
        "code": -32001,
        "message": "Authentication failed. Verify your SigV4 credentials and ensure
 they have not expired."
    }
}
```

エラーコードとその意味の完全なリストthe section called “エラーコード”については、「」を参照してください。

推奨される再試行戦略

- -32004 (LIMIT_EXCEEDED): 1 秒から指数バックオフで再試行する
- -32603 (INTERNAL_ERROR): エクスポネンシャルバックオフで最大 3 回再試行する
- -32001 (AUTHENTICATION_FAILURE): 認証情報を更新して再試行する
- その他のすべてのエラーの場合: 自動的に再試行しないでください — エラーメッセージを調べてリクエストを修正します

AWS Partner Central APIs

以下のセクションでは、 APIs。

トピック

AWS Partner Central API の使用

以下のセクションでは、AWS Partner Central APIsの使用について説明します。

トピック

AWS Partner Central Account API の使用

パートナーアカウントの管理

パートナーアカウントは登録され、既存のAWSアカウントにリンクされ、IAM ベースの完全な工クスペリエンスを提供します。このアプローチにより、ユーザーレベルの認証情報が不要になり、すべての登録とオペレーションがAWSアカウント内の IAM エンティティを通じて管理されます。AWS Partner Central はAWSサービスとのシームレスな統合を可能にし、このモデルはパートナーエンティティの管理を提供し、複数のカタログをサポートし、AWSアカウントレベルの使用権限システムを確立します。

パートナーは、利用可能なパートナーアカウント APIs を使用して、アカウントを登録、管理、更新できます。

パートナー登録 API

パートナーは、パートナーエンゲージメントのアカウントを使用してAWSアカウントを登録できます。Create API を使用すると、パートナーは必要な提携リード情報を提供し、APN 条件を受け入れ、一意の正式名称を提供します。

パートナープロファイル API

パートナーは、会社の詳細 (名前、説明、ウェブサイト、ロゴ) を含むプロファイルを、パブリック/プライベート可視性コントロール、非同期検証、およびステータス追跡を使用して管理し、一度に 1 つの更新を許可します。

ドメイン管理 API

パートナーは、E メール検証を通じてビジネスドメインを登録および検証し、組織アイデンティティを確立し、従業員のトレーニングと認定を関連付けることができます。

パートナー接続 API

パートナーはAWS、他のパートナーとの共同販売、アカウント間でのAWS Marketplace オファーデータの共有、ディストリビューター/チャネルパートナーによる製品の配布/再販の承認など、AWS さまざまな目的で自分のアカウントを他のアカウントに接続します。

パートナー検証 API

パートナー検証は、パートナーアカウント登録の必須前提条件です。パートナーは、パートナーアカウントを作成する前に、ビジネス検証と ID 検証のプロセスを完了する必要があります。これらの検証により、ビジネスが法的に登録されていることを検証し、AWSアカウントを登録している人物の身元を確認します。

トピック

- パートナー登録の使用
- パートナープロファイルの使用
- ドメイン管理の使用
- パートナーアカウント接続の使用
- パートナー検証の使用

パートナー登録の使用

パートナー登録

パートナーは、パートナーエンゲージメントのアカウントを使用してAWSアカウントを登録できます。Create API を使用すると、パートナーは必要な提携リード情報を提供し、APN 条件を受け入れ、一意の正式名称を提供します。

登録中

1. 同期パートナーエンティティの作成 – お客様は Create API を呼び出してパートナー登録プロセスを開始します。これにより、入力検証が実行され、同じリクエスト内にパートナーエンティティが作成されます。
2. Alliance Lead Information Requirement – パートナー登録中、お客様はAWS Partner Network (APN) 関連のコミュニケーションのために Alliance Lead の連絡先情報を提供する必要があります。
3. 利用規約の適用 – 顧客は登録時に APN 利用規約に同意する必要があります。登録を完了し、機能にアクセスするには、承諾が必要です。この条件は、すべての APN プログラムの利点と機能に適用されます。
4. Tag-On-Createのサポート –AWS一貫した認可エクスペリエンス (CAE) とタグベースの認可の一環として、お客様はパートナーエンティティの作成時にタグを追加できます。
5. 正式名称ベースの検証 – 登録では、パートナーの正式名称に基づいて一意性が適用され、同じエンティティで重複した登録がなくなります。

登録後

1. タグ管理のサポート – 登録後、パートナーは既存のパートナーエンティティのタグのタグ付け、タグ解除、および一覧表示を行うことができます。
2. パートナーエンティティの取得 - パートナーエンティティが作成されると、顧客は List API を使用してパートナー ID を取得し、Get API を使用して特定のパートナーエンティティの詳細を取得できます。

API の要約

1. CreatePartner API: お客様は CreatePartner API を呼び出してパートナー登録プロセスを開始します。これにより、入力検証が実行され、同じリクエスト内にパートナーエンティティが作成されます。

2. ListPartners API: 登録後、顧客は List API を使用してパートナーエンティティのリストを取得できます。
3. GetPartner API: 登録後、Get API を使用して特定のパートナーエンティティの詳細を取得できます。
4. PutAllianceLeadContact API: プライマリ提携リードの連絡先情報を更新します。このオペレーションにより、パートナーは組織の継続性を維持しながら、主要な提携リードの指定を移管したり、連絡先の詳細を変更したりできます。
5. GetAllianceLeadContact API: プライマリ提携リードの連絡先情報を更新します。このオペレーションにより、パートナーは組織の継続性を維持しながら、主要な提携リードの指定を移管したり、連絡先の詳細を変更したりできます。

パートナープロファイルの使用

プロファイル管理

パートナーは、会社の詳細 (名前、説明、ウェブサイト、ロゴ) を含むプロファイルを、パブリック/プライベート可視性コントロール、非同期検証、およびステータス追跡を使用して管理し、一度に 1 つの更新を許可します。

1. プロファイル情報 – パートナープロファイルには、表示名、説明、ウェブサイト URL、ロゴ、IndustrySegments、PrimarySolutionType などの重要な企業の詳細を含める必要があります。
2. 多言語サポート – パートナープロファイルは複数の言語をサポートしています。プロファイルの各ローカライズされたバージョンには、ロケールの 属性が含まれています。
3. プロファイルの可視性コントロール – パートナーはプロファイルの可視性 (パブリックまたはプライベート) を設定できます。
4. プロファイルストレージ – プロファイル情報はパートナーアカウントオブジェクト内に保存されます。
5. プロファイルステータス管理 – プロファイルの更新は、公開前に非同期的に検証されます。パートナーは、最新のプロファイル更新ステータス (IN_PROGRESS、SUCCEED、FAILED、CANCELLED) を確認できます。承認されたプロファイルのみが公開されます。
6. リクエストの同時実行制御 – 一度に実行できるプロファイル更新リクエストは 1 つだけです。プロファイルの更新がすでに進行中の場合は、新しい更新を開始する前に、CancelPartnerProfileUpdateTask API を使用して進行中の更新を明示的にキャンセルする必要があります。

7. 更新タスクのキャンセル – パートナーは、CancelPartnerProfileUpdateTask API を呼び出すことで、進行中のプロファイルの更新をキャンセルできます。キャンセルは、更新が IN_PROGRESS ステータスの場合にのみ許可され、新しい更新を開始する前に完了する必要があります。

API の要約

1. StartPutProfileTask API: この API はパートナープロファイルの更新を受け入れ、基本的な同期入力検証を実行します。
2. GetProfileUpdateTask API: この API により、パートナーは現在のプロファイル更新ステータスを取得できます。これは、お客様が StartPutPartnerProfile API を介してプロファイル更新を開始した後に使用することを目的としています。これにより、パートナーはリクエストが IN_PROGRESS、FAILED、SUCCEEDED、CANCELLED のいずれであるかを確認できます。
3. CancelProfileTask API: パートナーは、現在 IN_PROGRESS ステータスにあるプロファイル更新タスクを明示的にキャンセルできます。これには、自動審査フェーズと手動レビューフェーズの両方が含まれます。リクエストをキャンセルすると、パートナーは現在の審査プロセスが完了するまで待たずに新しいプロファイルの更新を開始できます。
4. PutProfileVisibility API: この API を呼び出すことで、パートナーはロケールに関係なくプロファイルの可視性を制御できます。これにより、パートナーはコンテンツを変更せずにプロファイルをパブリックまたはプライベートにすることができます。
5. GetProfileVisibility API: UpdatePartnerProfileVisibility API を呼び出すことで、パートナーはロケールに関係なくプロファイルの可視性を制御できます。これにより、パートナーはコンテンツを変更せずにプロファイルをパブリックまたはプライベートにすることができます。

ドメイン管理の使用

ドメイン管理

パートナーは、E メール検証を通じてビジネスドメインを登録および検証し、組織アイデンティティを確立し、従業員のトレーニングと認定を関連付けることができます。

API の要約

1. SendEmailVerificationCode API: 指定された E メールアドレスに検証コードを送信して、E メール検証プロセスを開始します。新しい連絡先の作成と E メールアドレスの更新の両方に使用されます。
2. AssociateAwsTrainingCertificationEmailDomain API: E メールドメインをパートナーアカウントの AWS トレーニングと認定に関連付け、従業員認定の自動検証を可能にします。

3. DisassociateAwsTrainingCertificationEmailDomain API: E メールドメインとパートナーアカウントの AWS トレーニングおよび証明書との関連付けを削除します。

パートナーアカウント接続の使用

パートナーは、 Partner Connection API を使用してAWS、自分のアカウント間の接続または他のパートナーのAWSアカウントへの接続を管理できます。このプロセスには 2 つのフェーズがあります。

1. 接続招待フェーズ: 接続リクエストの開始と応答
2. アクティブ接続フェーズ: 確立された接続とそれに関連するビジネスアクティビティの管理

接続招待の作成と送信

ステップ 1: 接続招待を作成する

パートナー接続を確立する最初のステップは、 `CreateConnectionInvitation`アクションを使用して接続招待を作成して送信することです。招待を作成するときは、以下を指定する必要があります。

- カタログ:AWS接続が管理されるカタログ (AWSまたはサンドボックス)
- ConnectionType: 確立する関係のタイプ。次のいずれかを選択します。
  - **`OPPORTUNITY_COLLABORATION`**: 別のパートナーのアカウントに接続リクエストを送信して、オポチュニティコラボレーションのためのオポチュニティエンゲージメントの招待を送信できるようにする場合は、このタイプを使用します。
  - **`SUBSIDIARY`**: このタイプは、所有している複数のAWS Marketplace 販売者アカウントを、APN にリンクした「プライマリ」アカウントまたはパートナーとして登録されている「プライマリ」アカウントに接続する場合に使用します。
- ReceiverIdentifier: レシーバーのパブリックパートナープロファイル ID (オポチュニティコラボレーション接続タイプの場合) またはAWSアカウント ID (子会社接続タイプの場合)
- E メール: 通信用の連絡先 E メールアドレス
- 名前: 送信者としての名前
- メッセージ: 接続リクエストの目的を説明する説明

作成されると、招待は保留中状態になり、受信者からのアクションを待ちます。

> **Important**
>
> 接続の招待を送信することで、受信者が招待を承諾した場合に、受信者にAWSアカウントID を公開することに同意したものとみなされます。この開示は、アカウントレベルの接続を確立するために必要です。

ステップ 2: 送信された招待をモニタリングする

ParticipantType パラメータを に設定して、 ListConnectionInvitationsアクションを使用して送信した招待のステータスを追跡できますSENDER。これにより、次のことが可能になります。

- すべての送信招待を表示する
- ステータス (PENDING、ACCEPTED、、EXPIRED) でフィルタリングREJECTEDする
- 接続タイプでフィルタリングする
- 他のパートナーへの特定の招待を確認する

ステップ 3: 招待をキャンセルする (オプション)

承諾される前に保留中の招待を取り消す必要がある場合は、 CancelConnectionInvitationアクションを使用できます。以下の点に注意してください。

- 保留中状態の招待のみをキャンセルできます
- 招待が承諾され、接続が確立されると、キャンセルすることはできません。
- 確立された接続を終了するには、代わりに CancelConnectionアクションを使用する必要があります。

接続招待の受信と応答

ステップ 1: 着信招待を一覧表示する

受信した接続の招待を表示するには、 ParticipantType パラメータを に設定して ListConnectionInvitationsアクションを使用しますRECEIVER。次の方法でフィルタリングできます。

- [接続タイプ]
- ステータス (PENDING、ACCEPTED、REJECTED)

- 送信者の識別子

ステップ 2: 招待の詳細を確認する

GetConnectionInvitation アクションを使用して、次のような特定の招待の完全な詳細を表示します。

- 送信者の名前と連絡先 E メール
- 目的を説明する招待メッセージ
- リクエストされている接続タイプ
- 有効期限日
- 送信者のパブリックプロファイル情報

ステップ 3: 招待を承諾または拒否する

招待の詳細を確認したら、次の 2 つのオプションがあります。

オプション A: 招待を受け入れる

AcceptConnectionInvitation アクションを使用して接続を確立します。承諾する場合:

- 新しい接続がアクティブ状態で作成されます
- 招待のステータスが Accepted に変わります
- AWSアカウント ID が送信者に公開されます
- 接続タイプで有効になっているビジネスアクティビティの実行を開始できます。

> ⚠ Important
>
> 重要な同意: 接続の招待を受け入れることで、送信者にAWSアカウント ID を公開することに同意します。この開示は、アカウントレベルの接続を確立するために必要です。

オプション B: 招待を拒否する

接続を確立しない場合は、 RejectConnectionInvitationアクションを使用します。オプションで、拒否の理由を指定できます。拒否された場合:

- 招待のステータスが拒否済みに変わります
- 接続が確立されていない
- 保留中の招待のリストに招待が表示されなくなります。

自動有効期限

有効期限内にアクションが実行されない場合、招待は自動的に期限切れ状態に移行します。期限切れの招待を承諾または拒否することはできません。

アクティブな接続の管理

接続の表示

接続が確立されると、両者は次のアクションを使用して接続を表示および管理できます。

`ListConnections`: フィルタリングオプションを使用して接続のリストを取得します。

- 接続タイプとステータスでフィルタリングする (例: `OPPORTUNITY_COLLABORATION:ACTIVE`)
- 他のユーザーの識別子でフィルタリングする
- 各接続の概要情報を表示する

`GetConnection`: 以下を含む、特定の接続の完全な詳細を取得します。

- このパートナー関係に関連付けられているすべての接続タイプ
- 各接続タイプのステータス (アクティブまたは終了)
- 元の招待の連絡先情報
- AWS両方の アカウント IDs とパブリックプロファイル IDs
- 各接続タイプの作成と終了のタイムスタンプ

既存の接続への接続タイプの追加

パートナーとのアクティブな接続がすでにあり、新しいタイプの関係を確立する場合は、別の接続タイプの新しい接続招待を送信できます。承諾時:

- 新しい接続タイプが既存の接続に追加されます。
- どちらの接続タイプも個別に管理できます。

- 関係に対して同じ接続 ID が維持されます

> **Note**
>
> 特定のカタログ内の 2 つのアカウント間では、任意のタイプのアクティブな接続が常に 1 つしか存在できません。

接続タイプの終了

どちらの当事者も、 `CancelConnection`アクションを使用して特定の接続タイプを終了できます。終了する場合:

- 終了する接続 ID と接続タイプを指定する
- 終了の理由を入力します。
- 接続タイプのステータスが に変わります。 `CANCELED`
- 同じ接続内の他の接続タイプは影響を受けません

接続終了を完了する

接続内のすべての接続タイプが終了すると、接続自体は終了状態に移行します。完全に終了した接続:

- 再アクティブ化することはできませんが、そのアカウントとのビジネスを継続するために新しい接続リクエストを送信できます
- 履歴レコード保持のためにシステム内にとどまる
- `GetConnection` API を使用して表示可能
- 関係を再確立するには、新しい招待が必要です

接続イベントのモニタリング

パートナーは、Amazon EventBridge を使用して接続関連のイベントをモニタリングし、以下の情報を常に把握する必要があります。

- 新しい受信接続の招待
- 送信された招待のステータスの変更 (`ACCEPTED`、`REJECTED`、`EXPIRED`)

- 相手側によって開始された接続終了

イベントを受信したら、適切な Get API アクションを使用して最新の詳細を取得します。

- `GetConnectionInvitation` 招待の更新用
- `GetConnection` 接続更新用

考えられるイベント

- Connection Invitation Created: 受信接続の招待を受信者アカウントに通知します。
- Connection Invitation Cancelled: 送信者が受信接続の招待をキャンセルすると、受信者アカウントに通知します。
- 接続招待の有効期限切れ: 接続招待がアクションなしで期限切れになると、送信者アカウントと受信者アカウントの両方に通知します。
- Connection Invitation Rejected: 接続招待が受信者によって拒否されたときに送信者アカウントに通知します。
- Connection Invitation Accepted: 接続の招待が受け入れられ、接続が確立されると、送信者アカウントに通知します。
- Connection Cancelled: すべての接続が完全に終了すると、接続された両方の当事者に通知します。

接続状態について

接続招待の状態

State	説明	に移行可能
`PENDING`	作成時の初期状態、受信者アクションを待機中	`ACCEPTED`, `REJECTED`, `EXPIRED`, `CANCELED`
`ACCEPTED`	受信者が受け入れられました。接続が作成されました	なし (終了状態)
`REJECTED`	レシーバーが拒否されました。オプションの拒否理由が含まれています	なし (終了状態)

State	説明	に移行可能
EXPIRED	有効期限内にアクションがないためにシステム期限切れになる	なし (終了状態)
CANCELED	送信者が承諾前に招待を取り消した	なし (終了状態)

接続タイプの状態

State	説明
ACCEPTED	接続タイプが動作しており、関連するビジネスアクティビティが有効になっている
CANCELED	いずれかの当事者によって終了した接続タイプ

ベストプラクティス

1. 明確なコミュニケーション: 目的と期待されるコラボレーションを説明する詳細な明確なメッセージを接続の招待に入力します。
2. タイムリーな応答: 自動期限切れを避けるため、接続の招待をモニタリングして迅速に応答します。
3. 定期的なレビュー: アクティブな接続を定期的に見直して、現在のビジネスニーズに関連性があることを確認します。
4. 適切な終了: ビジネス関係を終了するときは、CancelConnection アクションを明確な理由とともに使用して、専門的なコミュニケーションを維持します。
5. Event Monitoring: EventBridge ルールを設定して、接続ステータスの変更の通知を自動的に受信し、重要な更新を常に把握できるようにします。
6. 接続タイプ管理: 接続を確立する前に必要な接続タイプを検討します。タイプごとに異なるビジネス機能が有効になるためです。
7. セキュリティ意識: 接続を確立すると、当事者間のAWSアカウント IDsが明らかになることに注意してください。信頼できるパートナーとのみ接続します。

接続タイプとその目的

接続タイプ	目的	ユースケース
OPPORTUNITY_COLLABORATION	他のパートナーとの共同販売の機会に関するコラボレーションを可能にする	他のパートナーがオポチュニティにアクセスできるようにする場合、またはその逆の場合は、この接続タイプを使用します。この接続により、接続されたパートナーにオポチュニティエンゲージメントの招待を送信できます。
SUBSIDIARY	複数のAWS Marketplace 販売者アカウントとプライマリパートナーアカウント間の接続を確立します	パートナーおよび販売者として登録されているプライマリアカウントに接続する複数のAWS Marketplace 販売者アカウントがある場合に使用します。

パートナー検証の使用

パートナー検証の管理

パートナー検証は、パートナーアカウント登録の必須前提条件です。パートナーは、パートナーアカウントを作成する前に、ビジネス検証と ID 検証のプロセスを完了する必要があります。これらの検証により、ビジネスが法的に登録されていることを検証し、AWSアカウントを登録している人物の身元を確認します。

パートナーは、利用可能な検証 APIs を使用して、検証プロセスを開始およびモニタリングできます。

検証中

1. ビジネス検証の開始 – パートナーは、正式名称、登録 ID、国コード、設立管轄区域など、事業者登録の詳細を記載した `StartVerification` API を呼び出して、ビジネス検証を開始します。

2. ID 検証の開始 – パートナーは、登録者の情報を使用して StartVerification API を呼び出して ID 検証を開始します。API は、登録者が政府発行の ID を使用して ID 検証ワークフローを完了する安全な完了 URL を返します。
3. 検証ステータスのモニタリング – パートナーは GetVerification API を使用して、検証プロセスの現在のステータスと詳細を取得します。API は、検証タイプ、ステータス、タイムスタンプ、および検証固有の詳細を返します。
4. 検証の完了 – パートナーアカウントの作成に進むには、ビジネス検証と ID 検証の両方が SUCCEEDED ステータスに達する必要があります。失敗した検証または期限切れの検証は、アカウント登録前に解決する必要があります。

検証後

1. パートナーアカウントの作成 – 検証に成功すると、パートナーはパートナーアカウントのドキュメントに記載されている CreatePartner API を使用してパートナーアカウントの作成に進みます。
2. 検証レコードの取得 – パートナーは、開始時に返された検証 ID を持つ GetVerification API を使用して、いつでも検証の詳細を取得できます。

API の要約

1. **StartVerification** API: パートナーは StartVerification API を呼び出して検証プロセスを開始します。ビジネス検証のために、API はビジネス登録の詳細を検証します。ID 検証の場合、API は登録者が ID 検証を完了するための時間制限付き完了 URL を生成します。
2. **GetVerification** API: 検証を開始した後、パートナーは GetVerification API を使用して検証ステータスと詳細を取得します。API は、現在のステータス、タイムスタンプ、および検証固有の情報を返します。

AWS Partner Central Selling API の使用

このAWS Partner Central API リファレンスは、 AWSパートナーがカスタマーリレーションシップ管理 (CRM) システムを Partner Central と統合するのに役立つように設計されています。API は Partner Central とのやり取りを自動化し、共同事業活動への効果的な関与を確保するのに役立ちます。

API は、標準のAWS API 機能を提供します。API アクションを使用するか、プログラミング言語またはプラットフォームに合わせたAWS SDK を使用してアクセスします。詳細については、「 の開始方法AWS」および「構築するツールAWS」を参照してください。

AWS Partner Central API が提供する機能

1. オポチュニティ管理: 、CreateOpportunity、、UpdateOpportunity、ListOpportunitiesなどの API アクションAWSを使用してGetOpportunity、 との共同販売オポチュニティの管理を容易にしますAssignOpportunity。
2. AWS紹介管理: ListEngagementInvitations、、GetEngagementInvitation、 などのアクションAWSを使用してStartEngagementByAcceptingInvitation、共有された紹介の受信を容易にしますRejectEngagementInvitation。
3. エンティティの関連付け: アクション AssociateOpportunityおよび を使用して、AWS 製品、パートナーソリューション、AWS Marketplaceソリューション、AWS Marketplace製品、AWS Marketplaceプライベートオファーなどの関連エンティティを機会に関連付けますDisassociateOpportunity。
4. AWSオポチュニティの詳細を表示する: GetAWSOpportunitySummaryアクションを使用して、オポチュニティにリンクされたオポチュニティのAWSリアルタイム概要を取得します。
5. ソリューションを一覧表示する: パートナーが を使用して提供するソリューションを一覧表示するためのリスト APIs を提供しますListSolutions。
6. イベントサブスクリプション: パートナーは、Amazon EventBridge を使用して作成されたオポチュニティの作成、オポチュニティの更新、エンゲージメントの招待の承諾、エンゲージメントの招待の拒否、エンゲージメントの招待などのイベントをリッスンすることで、オポチュニティのリアルタイム更新にサブスクライブできます。

サポートされているリージョンおよびエンドポイント

AWS Partner Central API は、米国東部 (バージニア北部) リージョンで利用できます。

リージョン	Endpoint
us-east-1	partnercentral-selling.us-east-1.api.aws

パートナーは、安全なサンドボックス環境で API 統合をテストおよび検証できます。これにより、ライブデータに影響を与えずに API アクションをテストできます。詳細については、「AWS Partner Central Selling API のサンドボックスでのテスト」を参照してください。

API エンティティの販売

AWS Partner Central エンティティは、パートナーと の間の共同販売エンゲージメントに使用される主要なビジネスコンポーネントを表しますAWS。これらのエンティティは、機会、ソリューション、製品、オファーに関連する情報をカプセル化し、共同販売活動のスムーズなコラボレーションと管理を可能にします。以下のセクションでは、 Partner Central Selling API のコアエンティティについて説明します。

機会

オポチュニティとは、ビジネスが特定して積極的に追及する潜在的な販売または取引です。これは、会社の製品またはサービスが対処できる特定のニーズを持つ適格な見込み客またはリードです。オポチュニティは通常、販売パイプラインまたは CRM システムで追跡され、将来の収益の基盤を形成します。効果的な機会管理には、最初の認定から閉鎖まで、販売プロセスを通じてリードを育成することが含まれます。詳細については、「機会の使用」と「データ型」を参照してください。

パートナーソリューション

パートナーソリューション (AWS Partner Central ではサービスと呼ばれます) を表します。これは、AWSパートナーによって作成および提供されるソフトウェア製品またはコンサルティングプラクティスです。パートナーソリューションは、お客様がAWSサービスを使用して特定のビジネス課題に対処したり、特定の目標を達成したりするのに役立ちます。詳細については、「ソリューションとは」を参照してください。

AWS製品

特定のAWSサービスまたは製品を表します。 は、スケーラブルで信頼性が高く、費用対効果の高いインフラストラクチャソリューションを提供するように設計された幅広い製品とサービスAWSを提供します。パートナーは、Partner Central の一括インポートページ (インポートの開始 >AWS製品とソリューション) からAWS製品の最新のリストを取得できます。詳細については、AWS製品について確認したり、すべてのAWS製品のリストを表示したりできます。

AWS Marketplaceプライベートオファー

AWS Marketplaceプライベートオファーは、AWS Marketplace販売者が個々のAWS顧客に特定の料金と条件を提供できるようにする機能です。プライベートオファーを通じて、販売者はカスタム料

金、支払いスケジュール、エンドユーザーライセンス条件を交渉できます。 のAWSお客様は、特定の要件を満たすソフトウェアソリューションを取得できますが、標準サービスよりも有利な条件や料金の恩恵を受けることもできます。プライベートオファープロセスでは、販売者がカスタマイズされた条件で一意のオファーを作成し、それを指定されたAWS顧客とプライベートに共有して確認と承認を行います。詳細については、「」の「プライベートオファーAWS Marketplace」を参照してください。

エンゲージメントの招待

エンゲージメントの招待とは、AWSパートナーが特定の紹介について共同作業するための からの正式なリクエストを指します。これにより、AWSとパートナーは協力して機会を推進できます。招待は、パートナーによって承諾または拒否できます。

トピック

- オポチュニティの使用
- からの機会の使用AWS
- オポチュニティの更新の操作
- マルチパートナーの機会の使用
- 機会の関連付け、関連付け解除、割り当て
- リードの操作
- 取引規模に関するインサイトの使用
- ベストプラクティス

オポチュニティの使用

オポチュニティとは

販売プロセスの初期段階で、販売担当者は、関心のある個人 (リードと呼ばれる) が顧客になる可能性があるかどうかを評価します。この評価および検証フェーズは、認定と呼ばれます。リードが認定済みと見なされ、顧客に変換される可能性が高いと見なされると、そのリードはオポチュニティとして分類されます。

オポチュニティの使用

パートナーは、AWS Partner Central Selling API を使用して、CRM システム内で作成された機会を管理し、AWS Partner Central と同期できます。これにより、パートナーは開始から終了までの機会を追跡および管理できます。

オポチュニティの作成

オポチュニティを管理する最初のステップは、 CreateOpportunityアクションを使用してオポチュニティを作成することです。これにより、 を Lifecycle.ReviewStatusに設定してオポチュニティが作成されますPending Submission。ただし、オポチュニティはまだ検証AWSのために送信されていません。

オポチュニティを作成したら、 AssociateOpportunityアクションを使用して、少なくとも 1 つのパートナーソリューション、AWS Marketplaceソリューション、またはAWS Marketplace製品をオポチュニティに関連付ける必要があります。このアクションは、オポチュニティが何を販売しようとしているかを明確に定義します。ListSolutions API を使用して、アカウントで使用可能なソリューションの完全なリストを表示し、1 ~ 10 のソリューションを関連付けることができます。

オプションで、 AssociateOpportunityアクションを使用して、関連するAWS製品をオポチュニティに関連付けることもできます。このステップは、AWS販売チームが機会と組み合わせて販売される予定のAWS製品を理解するのに役立ちます。

必要なソリューションまたは製品が関連付けられ、オプションAWSで製品がリンクされたら、StartEngagementFromOpportunityTaskアクションを使用してオポチュニティへのエンゲージメントを開始できます。これは、エンゲージメントを開始する機会を送信する場合です。

レビュープロセス

StartEngagementFromOpportunityTask アクションを使用してオポチュニティのエンゲージメントを開始すると、オポチュニティLifecycle.ReviewStatusはAWS検証フェーズに入り、 に設定されますSubmitted。レビュープロセスが完了するまで、機会に変更を加えることはできません。

この検証フェーズでは、 は機会の詳細が正確で完全AWSであることを確認します。検証の進行中、Lifecycle.ReviewStatusは に設定されますIn-Review。

パートナーからの変更や追加の詳細が必要な場合、AWSは Lifecycle.ReviewStatusを に設定しAction Required、必要な更新は Lifecycle.ReviewCommentsフィールドを介して通知されます。

機会が検証に合格すると、 Lifecycle.ReviewStatusは に変更されApproved、共同販売アクティビティの準備が整います。

パートナーは Amazon EventBridge を使用してOpportunity Updatedイベントをモニタリングする必要があります。これにより、ステータスの変更やフィードバックが通知されますAWS。イベン

トを受信すると、パートナーは GetOpportunity API アクションを使用して最新のオポチュニティの詳細を取得し、 Lifecycle.ReviewStatusフィールドを確認できます。

検証の問題の解決

Lifecycle.ReviewStatus が に設定されている場合Action Required、パートナーは によって強調表示された問題に対処する必要がありますAWS。これらを解決するために、パートナーは UpdateOpportunity API アクションを使用してオポチュニティを更新できます。

Action Required 状態中は、特定のフィールドのみが編集可能です。これらのフィールドには、以下が含まれます。

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

必要な変更を行うと、オポチュニティは検証フェーズに再入り、オポチュニティの Lifecycle.ReviewStatusが Approvedまたは に設定されるまでプロセスが繰り返されますDisqualified。

承認後の更新

オポチュニティが になるとApproved、パートナーは UpdateOpportunityアクションを使用して必要に応じてオポチュニティを更新し続け、シームレスな共同販売アクティビティを促進できます。

パートナーは Amazon EventBridge を通じてOpportunity Updatedイベントを引き続きモニタリングし、変更について最新の状態に保つ必要があります。AWS更新の追跡の詳細については、「機会の更新の操作」セクションを参照してください。パートナーは、ビジネス検証ルールに基づいて選択フィールドを更新することもできます。

からの機会の使用AWS

1.AWSオポチュニティの受け取り

営業AWS担当がパートナーをAWS CRM システムのオポチュニティにアタッチすると、オポチュニティはパートナーと共有されます。これらはオポチュニティとAWS呼ばれ、パートナー自身のアカウントで作成されたオポチュニティとは異なります。

AWSオポチュニティにパートナーがアタッチされている場合、 はオポAWSチュニティからのデータのサブセットを含むエンゲージメント招待AWSを作成します。パートナーはエンゲージメント招待作成イベントを受け取ります。

2. エンゲージメントの招待の確認

エンゲージメントの招待に
は、Project.Title、Project.CustomerUseCase、、Lifecycle.Stage、
Project.CustomerBusinessProblemなどの重要な情報が含まれています。パートナーは、このデータを使用して、機会を追求するかどうかを決定できます。

ただし、オAWSポチュニティの以下のフィールドはエンゲージメントの招待に含まれません。

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

3. エンゲージメントの招待の処理

エンゲージメント招待作成イベントを受信すると、パートナーは GetEngagementInvitationアクションを使用してオポチュニティの詳細AWSを取得できます。

パートナーが機会を追求しないことを決定した場合、 RejectEngagementInvitationアクションと必要な を使用して招待を拒否できますRejectionReason。拒否されると、オポチュニティへのアクセスは失われます。

招待を受け入れて続行するには、パートナーは
StartEngagementByAcceptingInvitationTaskアクションを使用する必要があります。これは、次のタスクを順番に実行する非同期アクションです。

1. エンゲージメントの招待を受け入れます。
2. オポチュニティのデータAWSを使用して、パートナーのアカウントに新しいオポチュニティを作成します。
3. パートナーのアカウントで顧客を識別するために必要な追加の詳細が含まれます。

完了すると、オポチュニティ作成イベントが、対応するオポチュニティ ID でトリガーされます。その後、パートナーはこの ID を GetOpportunityアクションで使用して、オポチュニティの完全な詳細を取得できます。

パートナーがイベントを使用していない場合は、同じペイロードで StartEngagementByAcceptingInvitationTask を再度呼び出して、最新のステータスを確認できます。

4. 承諾後のAWSオポチュニティの管理

オポチュニティがパートナーのアカウントに入ると、システム内の他のオポチュニティと同様に管理および更新できます。パートナーは、 などのアクションを使用して、必要な変更UpdateOpportunityを行うことができます。

AWS更新を追跡し、機会の更新を管理する方法の詳細については、オポチュニティの更新の操作「」セクションを参照してください。

オポチュニティの更新の操作

機会の更新

パートナーは UpdateOpportunityアクションを使用して、更新されるフィールドと更新時期を規定する特定のルールで機会を更新できます。

1. Lifecycle.ReviewStatus が Submittedまたは の場合、更新を行うことはできませんIn-Review。
2. 送信前に、パートナーは更新を行うことができますが、StartEngagementFromOpportunityTaskアクションが呼び出されない限り、更新AWSは処理されません。
3. オポチュニティが Submittedまたは In-reviewステータスになると、すべての更新がブロックされます。
4. オポチュニティが Action Requiredステータスの場合、 は更新の選択フィールドAWSを開きます。これらのフィールドには、以下が含まれます。

- 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. レビュープロセス後 (Lifecycle.ReviewStatusが に設定されている場合Approved)、次のフィールドを更新することはできません。
   - Customer.Account.Address.Country
   - Customer.Account.Address.PostalCode
   - Customer.Account.Industry
   - Customer.Account.WebsiteUrl
   - Project.CustomerBusinessProblem
   - PartnerOpportunityIdentifier
   - Project.Title
6. 他のすべてのフィールドでは、 UpdateOpportunityアクションを使用して更新を行うことができます。ただし、特定のプログラムまたは機会タイプのビジネスルールに基づいて、追加の制限が適用される場合があります。詳細については、「フィールドレベルの検証」を参照してください。
7. UI と API の両方を通じて行われたすべての更新について、オポチュニティ更新イベントが生成されます。

オポチュニティAWSに関する からの更新の受信

AWS通常、 はAWSオポチュニティを更新し、更新が行われるたびに、オポチュニティ更新イベントが生成されます。

から最新の更新を取得するにはAWS、パートナーは 2 つの個別のアクションを呼び出す必要があります。

1. `GetOpportunity` パートナーのオポチュニティの詳細を取得します。
2. `GetAWSOpportunitySummary` は、AWSオポチュニティデータのリアルタイム概要を取得します。

からのほとんどの定期的な更新AWSは、`GetAWSOpportunitySummary`レスポンスを通じて利用できます。ただし、パートナーのオポチュニティの属性を直接更新AWSすることがあります。

から更新を使用するにはAWS

1. `GetAWSOpportunitySummary` アクションを呼び出して、AWSオポチュニティの最新の詳細を取得します。
2. 変更をパートナーのオポチュニティに反映する必要がある場合は、 `UpdateOpportunity`アクションを使用して、関連するデータをパートナーのオポチュニティにコピーします。

パートナーは、このプロセスを直接更新メカニズムとして自動化するか、データを検証して更新するための手動レビュープロセスを実装するかを選択できます。

マルチパートナーの機会の使用

AWS Partnerは、アクティブな参加者AWSとして と機会を連携させることができます。

トピック

- エンゲージメントとスナップショット
- のカスタムポリシーの作成 ResourceSnapshotJobRole
- パートナーをオポチュニティに招待する
- エンゲージメントの招待の詳細の取得
- エンゲージメントの招待への応答
- マルチパートナーの機会の確認と更新

- スナップショットを使用してパートナーの更新を受け取る
- エンゲージメントメンバーの表示
- リソーススナップショットジョブのステータスのモニタリング

エンゲージメントとスナップショット

エンゲージメントは、複数のAWS Partnerアカウント間および顧客機会AWSでのコラボレーションAWSのために が所有するリソースです。エンゲージメントにより、個々の機会の所有権と管理を維持しながら、パートナー間の安全な情報共有とコラボレーションが容易になります。エンゲージメントには、アクティブな参加者AWSとしてAWSと パートナーの両方から発生する機会が含まれます。パートナーは、AWSまたは他のパートナーを招待して、 を使用してエンゲージメントに参加させることができますEngagementInvitations。招待されたパートナーが承諾すると、同じエンゲージメント内で独自の機会を受け取ります。

エンゲージメントメンバーは、スナップショットを通じて進捗状況を共有します。つまり、オポチュニティなどの基盤となるリソースから特定のフィールドのポイントインタイムのイミュータブルコピーです。スナップショットはエンゲージメント内で作成され、メンバーと共有されます。基盤となるリソースが変更されると、所有者は更新を反映する新しいスナップショットリビジョンを作成できます。AWSは、リソースが変更されると新しいリビジョンを自動的に作成するスナップショットジョブを提供します。共有されると、スナップショットはエンゲージメントメンバーが永続的にアクセスできるようになります。

のカスタムポリシーの作成 ResourceSnapshotJobRole

マルチパートナーの機会に協力するには、エンゲージメントで他のパートナーと機会のスナップショットを共有する必要があります。最新のオポチュニティの詳細へのアクセスを維持するには、というカスタムロールを作成する必要がありますResourceSnapshotJobRole。このロールにより、システムはユーザーに代わってオポチュニティのスナップショットを作成し、同じエンゲージメントの他のパートナーからスナップショットを取得できます。このロールがない場合、オポチュニティに加えた更新は、エンゲージメント内の他のパートナーには表示されず、オポチュニティデータの古いスナップショットが引き続き表示されます。

> **Note**
>
> 以外の名前を使用できますResourceSnapshotJobRole。別の名前を使用する場合は、次のポリシーResourceSnapshotJobRoleの のすべてのインスタンスをポリシー名に置き換えます。

このロールを作成するには、IAM コンソールに移動し、次のカスタム信頼ポリシーを使用してResourceSnapshotJobRoleロールを作成します。

JSON

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

このロールを作成したら、AWSPartnerCentralSellingResourceSnapshotJobExecutionRolePolicyAWS管理ポリシーをアタッチするか、次のアクセス許可ポリシーをアタッチする必要があります。

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/*"
            ]
        }
    ]
}
```

{account} は自分の AWS アカウント ID に置き換えます。

ResourceSnapshotJobRole ロールを作成してアクセス許可をアタッチしたら、組織の ResourceSnapshotJobRole を設定する必要があります。PutSellingSystemSettings アクションを使用して、会社の (AWSアカウントによって識別される) ResourceSnapshotJobRole として作成したロールを設定します。

```
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"
}
```

をAWSアカウント ID {account}に、 を作成したロールの名前ResourceSnapshotJobRoleに置き換えます。このロールは、リソーススナップショットジョブによって暗黙的に引き受けられ、機会にアクセスし、エンゲージメント内の他のパートナーと共有するためのスナップショットを作成します。

パートナーをオポチュニティに招待する

パートナーは、エンゲージメントの招待を開始することで、パートナーとAWSの両方から生じる機会にコラボレーションできます。最大 9 つのパートナーを 1 つのオポチュニティに招待できます。

パートナーをオポチュニティに招待するには

1. Partner AWS Central 販売ガイドの「パートナーの検索と接続」の手順に従います。パートナーを機会に招待する前に、パートナーに接続する必要があります。この接続により、互いのアカウント IDs への相互アクセスが許可され、招待が目的のパートナーアカウントに到達できるようになります。

2. StartEngagementFromOpportunityTask アクションを使用してエンゲージメントを作成し、そのエンゲージメントにオポチュニティを関連付けます。この非同期アクションは、次のタスクを順番に実行します。

    1. エンゲージメントを作成します。
    2. 機会をエンゲージメントに関連付けます。
    3. CreateEngagementInvitation アクションを に呼び出しますAWS。
    4. オポチュニティを に送信しますAWS。
    5. を起動ResourceSnapshotJobしてスナップショットを作成します。
3. 他のパートナーを招待して参加させます。

    a. オポチュニティに関連付けられたエンゲージメント ID を取得するには、前のステップでTaskIdentifier返された でフィルタリングされた ListEngagementFromOpportunityTasksアクションを使用します。
    b. で招待を送信するには、受信パートナーのエンゲージメント ID とアカウント ID を使用しますCreateEngagementInvitation。

招待が成功すると、エンゲージメント招待作成イベントが受信側パートナーに送信されます。

エンゲージメントの招待の詳細の取得

エンゲージメント招待作成イベントを受け取ったら、 GetEngagementInvitationアクションを使用して招待の詳細を取得します。レスポンスには、エンゲージメントに関連する顧客機会に関する重要な情報が含まれます。

招待の詳細、特に InvitationMessage、、Project.CustomerUseCase、および Project.TitleProject.CustomerBusinessProblemフィールドを確認します。この情報は、顧客の機会と招待パートナーのコラボレーションへの期待に関するコンテキストを提供します。

これらの詳細を使用して、エンゲージメントの招待を承諾または拒否して機会を追求するかどうかを評価します。

エンゲージメントの招待への応答

パートナーがエンゲージメント招待を送信すると、招待を通知するエンゲージメント招待作成イベントが作成されます。招待を受け入れるか拒否するかを選択できます。この決定により、マルチパートナーの機会への関与が決まります。

エンゲージメントの招待を拒否するには:

RejectEngagementInvitation アクションを使用して招待を拒否します。決定を説明するRejectionReasonパラメータを指定する必要があります。拒否されると、招待の詳細にアクセスできなくなり、エンゲージメント招待拒否イベントは、招待を拒否したことを送信側パートナーに通知します。

エンゲージメントの招待を受け入れるには:

招待を受け入れ、マルチパートナーオポチュニティに進むには、StartEngagementByAcceptingInvitationTaskアクションを使用します。この非同期アクションは、次のタスクを順番に実行します。

1. エンゲージメントの招待を受け入れます。
2. 送信パートナーのオポチュニティのデータを使用して、パートナーアカウントに新しいオポチュニティを作成します。オポチュニティ作成イベントを受け取ります。
3. アカウントの顧客を識別するために必要な追加の詳細が含まれます。
4. 招待を承諾し、エンゲージメントに追加されたことをパートナーに通知するエンゲージメント招待承諾イベントを発行します。

期限切れのエンゲージメントの招待

エンゲージメントの招待に 15 日以内に応答しない場合、その招待は期限切れになり、マルチパートナー機会に参加することはできません。エンゲージメント招待の有効期限が切れたイベントは、招待の有効期限が切れたことを送信パートナーに通知します。

> **Note**
>
> それでもコラボレーションする場合は、送信側パートナーが新しいエンゲージメントの招待を開始する必要があります。

マルチパートナーの機会の確認と更新

パートナーがオポチュニティでエンゲージメントを開始すると、受信パートナーのアカウントで新しく作成されたオポチュニティのレビューステータスは、送信パートナーのオポチュニティステータスによって決まります。

レビューステータスが送信されたオポチュニティ:

送信パートナーのオポチュニティが にLifecycle.ReviewStatus設定されている場合Submitted、次のプロセスが発生します。

1. 受信パートナーのアカウントで作成された新しいオポチュニティは にLifecycle.ReviewStatus設定されますSubmitted。
2. Submitted ステータスは、送信パートナーのオポチュニティが になるまで残りますApproved。
3. 送信パートナーのオポチュニティが検証され、 Lifecycle.ReviewStatusが に設定されるとApproved、他のパートナーのオポチュニティは自動的にApprovedステータスを継承します。

このプロセスにより、マルチパートナーの機会全体で一貫した機会の詳細が確保され、複数のレビューが不要になります。

完了すると、オポチュニティ作成イベントが、対応するオポチュニティ ID でトリガーされます。パートナーは、この ID を GetOpportunityアクションとともに使用して、オポチュニティの詳細全体を取得できます。

承認されたレビューステータスのオポチュニティ:

送信側パートナーが を Lifecycle.ReviewStatusに設定してオポチュニティでエンゲージメントを開始するとApproved、次のプロセスが発生します。

1. 受信パートナーのアカウントで作成された新しいオポチュニティは にLifecycle.ReviewStatus設定されますApproved。
2. オポチュニティ作成イベントは、対応するオポチュニティ ID でトリガーされます。
3. パートナーは、オポチュニティ ID で GetOpportunityアクションを使用して、オポチュニティの詳細全体を取得できます。

ただし、承認されたオポチュニティには、送信側パートナーのオポチュニティからコピーされなかったパートナー固有の詳細がない場合があります。受信側パートナーは、オポチュニティステージがQualified以降のステージに更新されたときに、 UpdateOpportunityアクションを使用してこれらの詳細を更新する必要があります。

- Project.CustomerUseCase
- Project.DeliveryModels
- Solution

- PrimaryNeedsFromAws

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

オポチュニティが受信パートナーのアカウントで作成されると、パートナーのシステム内の他のオポチュニティと同様に管理および更新できます。パートナーは、 UpdateOpportunityアクションを使用して変更を行ったり、機会への関与に関する詳細情報を提供したりできます。

スナップショットを使用してパートナーの更新を受け取る

エンゲージメント内では、パートナーは機会を個別に維持および更新します。オポチュニティの追加など、エンゲージメントのリソースを変更すると、エンゲージメントリソーススナップショット作成イベントが公開されます。

変更を常に把握し、他のパートナーの機会から最新の情報にアクセスするには、次のアクションを使用できます。

- ListEngagementResourceAssociations – このアクションを使用して、オポチュニティに関連付けられたエンゲージメント ID を取得します。
- ListResourceSnapshots – このアクションを使用して、エンゲージメントに関連するすべてのオポチュニティスナップショットの包括的なリストを取得し、すべてのパートナーで利用可能なスナップショットの概要を提供します。
- GetResourceSnapshot – このアクションを使用して、特定のオポチュニティスナップショットのリアルタイムの概要を取得し、別のパートナーのオポチュニティに直接アクセスすることなくup-to-date最新の情報を表示できます。

別のパートナーのオポチュニティから、自身に反映すべき関連する変更や更新を特定した場合は、UpdateOpportunityアクションを使用します。このアクションにより、関連するデータをオポチュニティに選択的に組み込むことが容易になり、エンゲージメント全体で整合性と一貫性が確保されます。

エンゲージメントメンバーの表示

マルチパートナーオポチュニティへのエンゲージメントには、最大 10 のパートナーを含めることができます。新しいパートナーがエンゲージメントの招待を受け入れるたびに、エンゲージメントメンバーが追加したイベントが公開され、現在のすべてのメンバーに変更が通知されます。

エンゲージメント内でコラボレーションしているすべてのメンバーを表示するには、次の手順に従います。

1. ListEngagementResourceAssociations アクションを使用して、オポチュニティに関連付けられたエンゲージメント ID を取得します。
2. ステップ 1 の ID を ListEngagementMembersアクションに指定して、エンゲージメントメンバーのパートナーの詳細を取得します。

> **Note**
>
> エンゲージメントのメンバーのみが `ListEngagementMembers`アクションを呼び出すことができます。

リソーススナップショットジョブのステータスのモニタリング

マルチパートナーオポチュニティを使用する場合は、他のパートナーのオポチュニティのスナップショットを作成し、エンゲージメント内の他のパートナーからオポチュニティスナップショットを取得できるロールを作成する必要があります。このロールがない場合、オポチュニティに加えた更新は、エンゲージメントの他のパートナーには利用できず、オポチュニティの古いスナップショットが引き続き表示されます。

エンゲージメントのマルチパートナーオポチュニティに関連付けられたリソーススナップショットジョブのステータスを追跡するには、次の手順に従います。

1. ListEngagementResourceAssociations アクションを使用して、オポチュニティに関連付けられたエンゲージメント ID を取得します。
2. ステップ 1 の ID を ListResourceSnapshotJobsアクションに提供して、エンゲージメントで発信者が所有するすべてのスナップショットジョブのリストを生成します。レスポンスからジョブ ID を取得します。
3. GetResourceSnapshotJob アクションにジョブ ID を指定して、ジョブのステータスを追跡し、実行中かどうかを確認します。

ResourceSnapshotJob オペレーションは、非同期オペレーションのメトリクスを Amazon CloudWatch に発行します。CloudWatch は、データを読み取り可能なほぼリアルタイムのメトリクスに処理し、サービスのパフォーマンスとヘルスをモニタリングするのに役立ちます。

以下のメトリクスが利用可能です。

- 障害 – ジョブ実行中の内部問題をカウントします。このメトリクスを使用して、サービスの安定性をモニタリングし、障害しきい値のアラームを設定します。
- エラー – ジョブ処理中の顧客関連の問題を追跡します。このメトリクスは、顧客の入力または設定に関する問題を特定するのに役立ちます。
- 呼び出し – 成功した試行と失敗した試行の両方を含む、ジョブ呼び出しの合計数を表します。これを使用して、時間の経過に伴うサービス使用状況の傾向を追跡します。

> **Note**
>
> メトリクスは、ResourceSnapshotJobサービスにアクティビティがある場合にのみ CloudWatch に報告されます。特定のメトリクスのジョブまたはデータがない場合、そのメトリクスは報告されません。

オペレーションをスムーズにResourceSnapshotJob操作するには:

- メトリクスを使用して、サービスが期待どおりに動作することを確認します。
- CloudWatch アラームを作成して、メトリクスをモニタリングし、メトリクスが許容範囲を超えたときにアクション (通知の送信など) をトリガーします。
- プロアクティブモニタリングを設定して、問題を特定して解決します。

CloudWatch メトリクスの使用の詳細については、Amazon CloudWatch ユーザーガイド」を参照してください。

機会の関連付け、関連付け解除、割り当て

オポチュニティは、オポチュニティライフサイクルのどの段階でも、パートナーソリューション、AWS製品、AWS Marketplaceソリューション、AWS Marketplace製品、AWS Marketplaceオファー、AWS Marketplaceオファーセットと関連付けたり、関連付けを解除したりできます。

他のエンティティとの機会の関連付け

関連付けられたエンティティは、 RelatedEntityIdentifiers オブジェクト内の GetOpportunityメソッドから取得されます。は を使用して更新RelatedEntityIdentifiersできますAssociateOpportunity。このフィールドは、UpdateOpportunity または CreateOpportunityメソッドでは更新できないことに注意してください。

ソリューション

StartEngagementFromOpportunityTask アクションを使用して とのエンゲージメントAWSを開始する前に、少なくとも 1 つ、最大 10 個の Partner Solutions を関連付ける必要があります。AWS紹介には、パートナーソリューションが含まれる場合と含まれない場合があります。

既存のソリューションを表示するには、 ListSolutionsアクションを使用します。

パートナーは、 AWS Partner Central の Buildセクションでソリューションを作成、更新、管理できます。

> **Important**
>
> オポチュニティステージをコミット済みまたは起動済みに進めるには、有効なソリューションまたはAWS Marketplace製品リストをオポチュニティに関連付ける必要があります。

> **Note**
>
> Solutions エンティティタイプを使用するレガシーパートナーソリューション (S-XXXX 識別子) は引き続きサポートされています。ただし、AwsMarketplaceSolutionsエンティティタイプは今後廃止されるため、Solutionエンティティタイプをオポチュニティに関連付けることをお勧めします。

AWS製品

最大 20 個のAWS製品を 1 つのオポチュニティに関連付けることができます。利用可能なAWS製品のリストを表示するには、AWS GitHub でホストされている製品のリストを使用します。AWS製品との関連付けは、 AssociateOpportunityアクションを使用してのみ行われます。製品を置き換えまたは削除するには、 DisassociateOpportunityアクションを使用します。

ソリューション

機会は、販売者アカウントのソリューションに関連付けることができます。を RelatedEntityTypeに設定して AssociateOpportunityアクションを使用しますAwsMarketplaceSolutions。

エンティティはパブリックまたは制限付きステータスである必要があります。サービスは、下書きエンティティまたは非アクティブなエンティティを拒否します。

ソリューションを関連付けるには、ARN が必要です。次の例は、ソリューションの ARN 形式を示しています。

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

パートナーAWS Marketplaceアカウント接続を使用して、プライマリアカウントに接続された子会社アカウントからソリューションを関連付けることもできます。子会社アカウント ID を含む完全な ARN を入力します。

ListSolutions アクションは各ソリューションの概要AwsMarketplaceSolutionArnを返し、ARN によるフィルタリングをサポートします。

AWS Marketplace製品

オポチュニティは、販売者アカウントからAWS Marketplace製品に関連付けることができます。AssociateOpportunity アクションを RelatedEntityTypeに設定して使用しますAwsMarketplaceProducts。

エンティティはパブリックまたは制限付きステータスである必要があります。サービスは、下書きエンティティまたは非アクティブなエンティティを拒否します。

製品を関連付けるには、ARN が必要です。ARN には、製品タイプ (、、 などContainerProduct) AmiProduct SaaSProductが含まれます。次の例は、AMI 製品の ARN 形式を示しています。

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

パートナーAWS Marketplaceアカウント接続を使用して、プライマリアカウントに接続された子会社アカウントの製品を関連付けることもできます。子会社アカウント ID を含む完全な ARN を入力します。

AWS Marketplaceオファーとオファーセット

AWS Marketplaceオファー

機会はAWS Marketplaceプライベートオファーに関連付けることができます。利用可能なオファーを表示するには、AWS Marketplace Catalog API の ListEntities を使用します。現在、AWS Partner Central にリンクされたAWS Marketplace Seller アカウントのオファーのみを関連付けることができます。

プライベートオファーの ARN を関連付けるには、 が必要です。サンプル:

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

AWS Marketplaceオファーセット

オポチュニティはオファーAWS Marketplaceセットに関連付けることもできます。オファーセットを使用すると、複数の関連マーケットプレイスオファーをグループ化して包括的なソリューションパッケージ化できます。利用可能なオファーセットを表示するには、AWS Marketplace Catalog API の ListEntities を使用します。

オファーセットを関連付けるには、ARN が必要です。サンプル:

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

重要: オポチュニティは 1 つのオファーまたは 1 つのオファーセットに関連付けることができますが、両方に関連付けることはできません。一度にオポチュニティに関連付けることができるエンティティタイプは 1 つだけです。

他のエンティティから機会の関連付けを解除する

`DisassociateOpportunity` アクションを使用して、次のエンティティタイプからオポチュニティのリンクを解除します。

- ソリューション
- AWS製品
- AWS Marketplaceソリューション
- AWS Marketplace製品

- AWS Marketplaceオファー
- AWS Marketplaceオファーセット

オポチュニティの状態に応じて、これらの関連オブジェクトのリンクを解除すると、異なる検証ルールが適用されます。

機会の割り当て

`AssignOpportunity` を使用して、オポチュニティの所有者を変更します。Partner Central ユーザーをオポチュニティ所有者として設定できます。

リードの操作

リードとは

business-to-business (B2B) 販売では、リードは、企業の製品やサービスに関心を示しているが、まだ販売機会として完全に認定されていない潜在的な顧客を表します。リードは販売パイプラインの初期段階であり、アクティブな共同販売の機会に変換する前に、育成と認定が必要ですAWS。

がパートナーAWSと共有しているリードは、ウェビナーへの参加、キャンペーンへの参加、パートナーソリューションの問い合わせなどのカスタマーエンゲージメントシグナルに基づいています。これらのリードには、顧客の企業情報、ビジネス上の問題、エンゲージメントの詳細などの貴重なコンテキストが含まれており、パートナーが潜在的な機会に優先順位を付け、認定するのに役立ちます。

リード招待の使用

パートナーは、潜在的な顧客がパートナーソリューションまたはサービスに関心を示すAWSと、 からリード招待を受け取ります。リード招待プロセスにより、パートナーはエンゲージメントにコミットする前にリードを評価し、効率的なリソース割り当てと変換率の向上を実現できます。

リード招待の受信

AWSはリード招待を作成し、Selling API を通じてパートナーと共有します。新しいリード招待が利用可能になると、 はリードコンテキストを含むエンゲージメント招待を適格なパートナーAWSに送信します。

パートナーは Amazon EventBridge を使用して`Engagement Invitation Created`イベントをモニタリングする必要があります。このイベント通知には、 に設定された `payloadType`フィールドが含まれており`LeadInvitation`、パートナーはリードの招待を機会の招待と区別できます。イベントを受信すると、パートナーは招待の詳細を取得してリードを評価できます。

リード招待の一覧表示

パートナーは、 ListEngagementInvitations API アクションを使用してすべてのリード招待を表示できます。このアクションは、呼び出し元のプリンシパルが送信者または受信者のいずれかである招待を取得します。

リードの招待を特にフィルタリングするには、パートナーは値 のpayloadTypeフィルターを使用する必要がありますLeadInvitation。このフィルターにより、結果からの機会の招待を除き、リードの招待のみが返されます。

リードの招待は、次の状態になります。

- 保留中: パートナーの承諾または拒否待ち
- 承諾: パートナーは招待を承諾し、完全なリード詳細へのアクセスを取得しました
- 拒否: パートナーは招待を拒否しました
- 有効期限切れ: 招待がアクションなしで有効期限を過ぎました

リード招待の評価

リード招待を受け入れる前に、パートナーは GetEngagementInvitation API アクションを使用して詳細な招待情報を取得する必要があります。これにより、情報に基づいた承認または拒否の決定を行うための重要なコンテキストが提供されます。

リード招待ペイロードには以下が含まれます。

顧客情報 (事前承諾可能):

- 会社名、業界、国
- ウェブサイトの URL
- 市場セグメント (エンタープライズ、ラージ、ミディアム、スモール、マイクロ)
- AWS成熟度レベル (評価、単一アカウント、マルチアカウント)

顧客とのやりとり:

- ソースタイプとキャンペーン情報
- カスタマーアクションの実行 (フォームの完了、ウェビナーへの参加など)
- 顧客が提供するビジネス上の問題の説明
- ユースケースカテゴリ

- 問い合わせタイトル (BANT 認定の権限コンテキストを提供します)

> **Important**
>
> 顧客の連絡先情報 (名前、E メール、電話番号) は、招待段階では表示されません。連絡先情報は、招待を承諾した後にのみ利用可能になり、顧客のプライバシーを確保し、リードファームの動作を防止します。

リード招待の承諾

パートナーがリードを追及することを決定した場合、AcceptEngagementInvitationAPI アクションを使用して招待を受け入れる必要があります。このアクションは、パートナーをエンゲージメントに追加し、顧客の連絡先情報を含むリードの詳細全体へのアクセスを許可します。

承諾後:

- パートナーはエンゲージメントのメンバーとして追加されます
- 顧客の連絡先情報がエンゲージメントリードコンテキストを通じて表示される
- リードは、AWSシステムでパートナー認定リード (PQL) に分類されます。
- リードがパートナーのアクティブなリードリストに表示される
- パートナーはリードの育成と顧客からの問い合わせの確立を開始できます

パートナーは、招待AcceptEngagementInvitationごとに を呼び出すことで、複数のリード招待をプログラムで受け入れることができるため、高品質のリードを効率的に一括処理できます。

リード招待の拒否

リードがパートナーの能力、キャパシティ、またはビジネスフォーカスと一致しない場合、パートナーは RejectEngagementInvitation API アクションを使用して招待を拒否できます。リードの招待を拒否する場合、パートナーはリードのルーティングとマッチングAWSの改善に役立つ拒否理由を提供する必要があります。一般的な拒否理由は次のとおりです。

- 地理的カバレッジの欠如
- ユースケースの技術的な専門知識が不十分
- 現在の容量の制約
- 顧客業界の不一致

- 予算のずれ

拒否後:

- リードがパートナーの保留中の招待キューから削除される
- 顧客の連絡先情報がパートナーと共有されることはありません。
- リードは、AWSシステムでパートナー拒否リード (PRL) として分類されます。
- 招待は、パートナーの拒否された招待リストに表示されたままになり、参照できます。

拒否されたリードは、顧客の同意要件が満たされていれば、他のパートナーへの再割り当ての対象となる場合があります。

承認されたリードの管理

リード招待を承諾すると、パートナーは完全なリード情報にアクセスし、顧客関係を育み、認定段階を通じてリードを追跡できます。

リードの詳細の表示

パートナーはGetEngagement、 API アクションを使用して完全なリードの詳細にアクセスします。エンゲージメントには、顧客とのインタラクションに関する包括的な情報を含むリードコンテキストが含まれていますAWS。

リードコンテキストには以下が含まれます。

Customer Profile を完了する:

- 元の招待のすべての会社情報
- すべての顧客とのやりとりの完全な連絡先情報 (名前、E メール、電話、役職)
- AWS成熟度レベルと市場セグメント

インタラクション履歴:

- 複数のカスタマータッチポイントを 1 つのエンゲージメントに統合
- 各インタラクションには、ソース情報、顧客アクション、ビジネス上の問題、連絡先の詳細が含まれます。
- インタラクションは時系列で一覧表示され、完全なカスタマージャーニービューを提供します。

資格ステータス:

- リード認定の現在の状態 (非認定、認定、非認定、またはカスタムステータス)
- パートナーは、認定プロセスを進めるにつれてこのステータスを更新できます。

この包括的なビューにより、パートナーは各やり取りを個別のリードとして扱うのではなく、複数のAWSキャンペーンやタッチポイントにわたる顧客のエンゲージメント履歴全体を把握できます。

アクティブなリードの一覧表示

パートナーは、contextTypeフィルターを に設定して ListEngagements API アクションを使用して、すべてのアクティブなリードを取得できますLead。これにより、パートナーがメンバーであり、エンゲージメントにリードコンテキストが含まれているエンゲージメントが返されます。

リストレスポンスには、次のような重要な情報が含まれます。

- エンゲージメント ID とタイトル
- 顧客の会社名
- 現在のエンゲージメントスコア
- 適格ステータス
- インタラクションの数
- 作成と変更のタイムスタンプ

パートナーはこのリストを使用して、ダッシュボードの構築、リードパイプラインの状態の追跡、注意が必要なリードの特定を行うことができます。

Prospecting Insights によるリードの強化

パートナーは、受け入れられたリードをAWSインサイトで強化し、アウトリーチに投資する前に優先順位を付けて認定することができます。エンリッチメントは、プロスペクティングタスクを通じて非同期的に実行されます。これにより、AWS派生シグナルとのリードエンゲージメントが強化され、認定の決定をサポートします。

プロスペクティングタスクの開始

パートナーは StartProspectingFromEngagementTask API アクションを使用してエンリッチメントを開始します。このアクションは、1 回のリクエストで最大 100 個のエンゲージメント識別子

を受け入れるため、パートナーはリードをバッチで強化できます。タスクは非同期的に実行され、パートナーが進行状況を追跡するために使用する TaskIdと TaskArn をすぐに返します。最初の TaskStatusは、PENDING、、IN_PROGRESSCOMPLETED、または のいずれかですFAILED。

パートナーはオプションで、タスクTaskNameにラベルを付ける とClientToken、再試行全体でべき等性を確保するための を提供できます。

結果のポーリング

プロスペクティングは非同期であるため、パートナーは API GetProspectingFromEngagementTaskアクションを使用して完了をポーリングし、開始時に TaskIdが返されます。レスポンスは、送信された各識別子の全体的なタスクステータスとエンゲージメントごとの結果をレポートします。

各エンゲージメントは個別に処理されるため、個々のエンゲージメントは同じタスク内の他のエンゲージメントに影響を与えずに成功または失敗する可能性があります。エンゲージメントごとに、結果には以下が含まれます。

- エンゲージメント識別子: 処理されたエンゲージメント。
- ステータス: PENDING、IN_PROGRESS、COMPLETED、または FAILED。
- コンテキスト識別子のプロスペクティング: エンゲージメントが正常に処理されたときに入力されます。この識別子は、エンゲージメントに追加された拡張されたプロスペクティングコンテキストを参照し、後続のオペレーションで使用できます。
- 理由コードとメッセージ: エンゲージメントが失敗した場合にのみ入力され、列挙された障害コードと、推奨される復旧ステップを含む人間が読める説明を提供します。

複数のタスクのモニタリング

多くのリードのエンリッチメントを追跡するために、パートナーは ListProspectingFromEngagementTasks API アクションを使用します。このアクションは、発信者のアカウントによって開始されたすべてのプロスペクティングタスクを一覧表示し、設定可能なソートを使用して、タスク識別子、タスク名、または開始時間範囲によるオプションのフィルターをサポートします。レスポンスはページ分割されます。各レスポンスNextTokenの値を使用して後続のページを取得します。

エンリッチメントが正常に完了すると、AWSインサイトがプロスペクティングコンテキストとしてエンゲージメントに追加されます。パートナーは で強化された詳細を取得し、それGetEngagementを使用してリードの資格ステータスを設定し、変換に進む原因を決定できます。

リード情報の更新

パートナーはリードを育成し、追加情報を収集すると、 UpdateEngagementContext API アクションを使用してリードの詳細を更新できます。このアクションにより、パートナーはエンゲージメント内のリードコンテキストを変更できます。

パートナーは以下を更新できます。

認定ステータス: パートナーは、内部認定プロセスを進めるにつれて認定ステータスを更新する必要があります。

- 非修飾: リードを受け入れた後のデフォルトステータス。リードが評価を必要とすることを示します
- 認定済み: リードが認定基準を満たし、オポチュニティへの変換の可能性が高い
- 失格: リードが基準を満たしていないか、パートナーのソリューションに適していない
- カスタム状態: パートナーは、内部プロセスに合わせて独自の認定状態を定義できます。

リードメタデータ: パートナーは、認定および育成プロセスに関連する追加情報を追加または更新できます。

認定ステータスを更新すると、パートナーはリードの進行状況を追跡し、正確なパイプラインレポートを生成し、オポチュニティに変換できるリードを特定するのに役立ちます。

AWS更新のモニタリング

AWSは、新しいカスタマーエンゲージメントシグナルまたは洗練されたエンゲージメントスコアに基づいてリード情報を更新する場合があります。がエンゲージメントAWSを更新すると、パートナーは Amazon EventBridge 経由でEngagement Updatedイベントを受け取ります。

これらの更新には以下が含まれます。

- 新しい顧客アクティビティに基づいてエンゲージメントスコアを改善
- 新しいキャンペーンやタッチポイントからの追加のカスタマーインタラクション
- 更新された顧客情報

パートナーはこれらのイベントをモニタリングし、 GetEngagementを呼び出して最新のリードの詳細を取得する必要があります。これにより、パートナーはリードの優先順位付けと育成に関する最新情報を得ることができます。

Engagement Updated イベントには contextTypesフィールドが含まれており、パートナーは特にリード (リードコンテキストを使用したエンゲージメント) をフィルタリングできます。

リードをオポチュニティに変換する

リードが認定され、真剣な購入の意図を示すと、パートナーはリードを正式な共同販売コラボレーションの機会に変換できますAWS。この変換は、リード育成 (主にパートナー主導) からオポチュニティ管理 (コラボレーション) への移行を示しますAWS。

推奨されるアプローチ: Convenience API

パートナーは、StartOpportunityFromEngagementTask便利な API アクションを使用して、機会の作成とリンクプロセスを合理化できます。このタスク API は、複数のアクションを自動的にオーケストレーションします。

1. リードコンテキストからの予備情報を含むドラフトオポチュニティを作成します。
2. リソーススナップショットを介してソースエンゲージメントにオポチュニティをリンクします
3. CustomerProject コンテキストをエンゲージメントに追加します

この便利な方法により、必要な API コールの数が減少し、適切な属性追跡が保証されます。このアプローチを使用するパートナーは、引き続き以下を行う必要があります。

- を使用してオポチュニティの詳細を入力する UpdateOpportunity
- を使用したパートナーソリューションの関連付け AssociateOpportunity
- 準備StartEngagementFromOpportunityTaskができたら、 を使用してオポチュニティを送信する

コンビニエンス API は、統合の複雑さを最小限に抑えたい、自動化されたlead-to-opportunityワークフローを持つパートナーに特に役立ちます。

代替アプローチ: Step-by-step API

ドラフトオポチュニティの作成

リードを変換する最初のステップは、 CreateOpportunity API アクションを使用してドラフトオポチュニティを作成することです。これにより、 を Lifecycle.ReviewStatusに設定してオポチュニティが作成されますPending Submission。この段階では、オポチュニティはまだ検証AWSのために に送信されていません。

パートナーは、リードの育成中に収集された情報を機会に入力する必要があります。

- 顧客アカウントの詳細
- プロジェクト情報とビジネス上の問題
- 予想される顧客支出
- ターゲット終了日
- 認定中に収集されたその他の関連する詳細

ドラフトオポチュニティにより、パートナーは送信前に完全な情報を準備できるためAWS、承認率が高く、検証が迅速になります。

オポチュニティをリードエンゲージメントにリンクする

ドラフトオポチュニティを作成した後、パートナーは `CreateResourceSnapshot` API アクションを使用してソースリードエンゲージメントにリンクする必要があります。このステップは、以下にとって重要です。

- 帰属の追跡: リードの招待からオポチュニティクローズまでの出所チェーンを確立し、マーケティングキャンペーンとリードソースの正確な ROI 計算を可能にします。
- 変換メトリクス:AWSとパートナーはlead-to-opportunity変換率を測定し、成功したリードソースを特定できます。
- コンテキストの保存: 元のインタラクションやエンゲージメントシグナルを含む、カスタマージャーニーの完全な履歴を維持します。

リソーススナップショットを作成するときに、パートナーは以下を指定します。

- エンゲージメント ID (ソースリードエンゲージメント)
- オポチュニティ識別子
- リソースタイプ (オポチュニティ)

このアクションは、リードがアクティブなオポチュニティに変換されたことを示す CustomerProject コンテキストをエンゲージメントに自動的に追加します。エンゲージメントに、元のリードコンテキスト (保存履歴) と新しい CustomerProject コンテキスト (アクティブなオポチュニティを示す) の両方が含まれるようになりました。

オポチュニティの詳細の完了

オポチュニティが作成されてリンクされると、パートナーは送信前に追加のオポチュニティ要件を完了する必要があります。詳細については、 AssociateOpportunity API アクションを参照してください。

プロジェクトの詳細の更新: パートナーは UpdateOpportunity API アクションを使用して、プロジェクト情報、顧客のビジネス上の問題、予想される支出、およびリード認定中に収集されたその他の関連の詳細を絞り込むことができます。

オポチュニティの送信

必要な情報がすべて完了すると、パートナーはStartEngagementFromOpportunityTaskコンビニエンス API を使用してAWS検証の機会を送信します。これにより、オポチュニティがドラフトステータスからAWSレビュープロセスに移行されます。

検証要件: エンゲージメントには、送信前に CustomerProject コンテキストが必要です。このコンテキストは、 CreateResourceSnapshot を使用してオポチュニティをエンゲージメントにリンクするときに自動的に作成されます。このコンテキストがない場合、送信は失敗します。

送信後:

- オポチュニティの へのLifecycle.ReviewStatus変更 Submitted
- リードは、AWSシステムのパートナー販売認定リード (PSQL) に分類されます。
- AWSは、機会の詳細が正確で完全であることを確認するために検証を開始します
- レビュープロセスが完了するまで、オポチュニティに変更を加えることはできません

その後、オポチュニティは「オポチュニティの操作」ドキュメントで説明されている標準検証ワークフローに従い、レビュー中、アクションが必要、承認済み、または失格などの状態に進みます。

取引規模に関するインサイトの使用

ディールサイジングとは

ディールサイジングは、AWS Partner Central の APN Customer Engagements (ACE) Opportunities 内の機能であり、AI を使用して、パートナーがオポチュニティを作成または更新するときに、自動化されたディールサイズの見積もりとAWSサービスのレコメンデーションを提供します。

AI 予測 MRR の見積もり

ディールサイジングインサイトは、パートナーの過去のAWS機会、ユースケース、ビジネスの説明に基づいて、予測月額経常収益 (MRR) 範囲の中央値を提供します。パートナーは、取引に関する詳

細情報を収集し、販売サイクルの進行状況を把握するため、MRR の合計を確認して更新する必要があります。

AI 推奨AWS製品

AI 推奨製品は、顧客のビジネス上の問題の説明と機会の詳細に基づいて特定されます。AI は、技術要件とユースケースに沿った関連AWS製品を推奨します。パートナーはこれらの提案を確認し、顧客の特定のニーズに合わせて製品の選択をカスタマイズする必要があります。

料金計算ツール URLs を使用した拡張インサイト

パートナーは、AWS料金計算ツール URLs をインポートして、サービスの選択を自動的に入力し、移行促進プログラム (MAP) の適格性指標、最適化の推奨事項、潜在的なコスト削減分析などの強化されたインサイトを取得することもできます。

ソース区切りデータについて

ディールサイジングは、2 つの異なるソースからのインサイトを提供し、オポチュニティ価値を包括的に可視化します。

- AWS出典: 同様の過去の機会からのパターンに基づく AI 推奨製品
- パートナーソース:AWS料金計算ツールからインポートされた独自の見積り

ソース間の差異の処理

パートナーが提供する見積りがAWSレコメンデーションと大きく異なる場合、パートナーは以下を行う必要があります。

1. 機会の詳細を確認して、すべての関連情報がキャプチャされていることを確認します。
2. 料金計算ツールの URL 設定が実際の要件と一致することを確認する
3. 同様の過去の機会から情報を得たデータポイントとしてAWSレコメンデーションを検討する
4. 特定の顧客のコンテキストと要件に基づいて、情報に基づいた意思決定を行う

ディールサイジングインサイトへのアクセス

パートナーは `GetAwsOpportunitySummary` API アクションを通じてディールサイジングデータにアクセスします。これにより、ディールサイジング予測、製品レコメンデーション、最適化インサイトを含む拡張`AwsOpportunitySummary`オブジェクトが返されます。

ディールサイジングデータがいつ利用可能になるか

ディールサイジングインサイトは、十分な顧客およびプロジェクトの詳細でオポチュニティが作成されると利用可能になります。システムは機会属性を分析し、以下を生成します。

1. AI 予測 MRR: 機会の特性に基づいて自動的に計算
2. AWS製品のレコメンデーション: 顧客のビジネス上の問題と技術要件に関連するサービス
3. インサイトの強化 (料金計算ツール URL が指定されている場合): MAP 適格性、最適化の推奨事項、コスト削減分析

> **Note**
>
> ディールサイジングインサイトは、すべてのオポチュニティで利用できるわけではありません。適格性は、パートナー履歴や提供される機会の詳細の完全性などの要因によって異なります。

ディールサイジング入力の提供

パートナーは、 `CreateOpportunity`および `UpdateOpportunity` API アクションを通じて、ディールサイジングの入力を提供します。システムはオポチュニティ属性を使用してレコメンデーションを生成します。

契約総額 (TCV) の提供

毎月の経常収益ではなく合計契約価値を見積もるパートナーは、取引価値を直接提供できます。AWS は、AI を活用した変換モデルを使用して TCV を AWS MRR に自動的に変換します。

TCV でのディールのサイズ設定では、 `ExpectedContractDuration` を `Months` (有効な値の範囲は 1～144 か月) `Frequency`として、 を `TargetCompany`として指定`Self`します`None`。

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

```
      }
    ]
  }
}
```

AWS は AWS MRR の見積もりを取得し、 で両方のエントリを返します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"
      }
    ]
  }
}
```

上記の例では、ExpectedCustomerSpend最初の配列には AWS MRR の見積もり (TargetCompany: "AWS"、Frequency: "Monthly") が含まれ、2 番目の配列はパートナーの合計契約値 (TargetCompany: "Self"、) ですFrequency: "None"。

TCV 検証ルール

すべての TCV 検証エラーは、 INVALID_VALUEでエラーコードを返しますReason: "BUSINESS_VALIDATION_FAILED"。識別係数は FieldNameと ですMessage。

Condition	フィールド	メッセージ
なしの自己入力 ExpectedContractDuration	project.expectedContractDuration	自己エントリが存在する場合は ExpectedContractDuration が必要です

Condition	フィールド	メッセージ
ExpectedContractDuration 自己入力なし (作成のみ)	project.expectedCustomerSpend	ExpectedContractDuration が存在する場合、自己入力が必要です
複数の自己エントリ	project.expectedCustomerSpend	1 つの自己入力のみが許可されます
期間が存在する場合の無効な TargetCompany	project.expectedCustomerSpend.targetCompany	TargetCompany は「AWS」または「Self」である必要があります
AWS とセルフ間の通貨の不一致	project.expectedCustomerSpend.currencyCode	CurrencyCode はエントリ間で一致する必要があります
自己入力 + EstimationUrl を一緒にする	project.expectedCustomerSpend.estimationUrl	自己入力と EstimationUrl は共存できません

> **Note**
>
> ではUpdateOpportunity、 ExpectedContractDurationがセルフエントリなしで存在し、パートナーディール値がストレージに存在する場合、システムはセルフエントリを自動的に再構築します。エラーはスローされません。

TCV と手動 MRR の切り替え

TCV モードから手動 MRR に戻すには、 ExpectedContractDuration を明示的に に設定nullし、 AWS エントリのみを指定します。

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

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

> **Note**
>
> ペイロードExpectedContractDurationから を省略 (フィールドを送信しない) すると、既存の TCV データは保持されます。明示的な送信とは、「クリア TCV モードnull」を意味します。

TCV と料金計算ツールURLs

パートナーがSelfエントリ (TCV モード) と の両方を提供する場合EstimationUrl、API は検証エラーを返します。自己入力と EstimationUrl が同じリクエストに共存することはできません。

手動 MRR 見積りの提供

パートナーは、 Project.ExpectedCustomerSpend[].Amountフィールドを使用して MRR 見積りを手動で入力できます。

> **Note**
>
> CurrencyCode フィールドは USDと を受け入れますEUR。AWS パーティションが aws-eusc (AWS 欧州主権クラウド) の場合、通貨コードは である必要がありますEUR。

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

```
    ]
  }
}
```

このフィールドはすべてのパートナーが使用でき、同じ値を設定して AI 予測を受け入れるか、パートナーが指定した値で上書きするために使用できます。

料金計算ツール URLs

パートナーはオプションで、 `Project.ExpectedCustomerSpend[].EstimationUrl`フィールドを介してAWS料金計算ツール URLs を提供して、サービス設定を自動的にインポートし、MAP 適格性指標、最適化の推奨事項、コスト削減分析などの強化されたインサイトを得ることができます。

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

有効な料金計算ツール URL を指定すると、システムは計算ツール設定から MRR 金額を自動的に計算し、金額フィールドと詳細な製品レベルの支出データの両方を のパートナーソースに入力しますAwsProductsSpendInsightsBySource。パートナーは、EstimationUrl を提供するときに Amount を null に設定する必要があります。システムはそれを自動的に計算します。

無効な形式の URLs とURLs は、 で拒否されます`ValidationException`。

Amount と EstimationUrl が連携する方法

機会に対して毎月の経常収益 (MRR) の見積もりを提供する方法に柔軟性があります。API は、手動金額とAWS料金計算ツール URLs のさまざまな組み合わせをインテリジェントに処理し、機会を常に正常に作成できるようにします。

手動量のみを使用する

Amount フィールドのみを指定すると、API は手動見積りを保存し、EstimationUrl を null に設定します。このアプローチは、顧客との会話または独自の計算から特定の MRR 見積もりがある場合に便利です。

リクエストの例:

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

AWS料金計算ツール URL のみを使用する

EstimationUrl フィールドのみを指定する場合:

- 有効な URL – API は、計算ツール設定から MRR 量を計算し、URL と計算された量の両方を保存します。
- 無効な URL – API は、URL が無効である理由の詳細を含む ValidationException を返します。

リクエストの例:

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

エラーレスポンスの例 (無効な 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"
}
```

Amount と EstimationUrl の両方を提供する

両方のフィールドを指定すると、API はオポチュニティが正常に作成されることを優先します。

1. 両方の値が変更されていない場合 – これらのフィールドでは更新は実行されません
2. URL が無効です – API は指定された量を保存し、EstimationUrl を null に設定して、オポチュニティの作成を続行できるようにします。
3. URL が有効で、計算された金額が指定された金額と一致する – 両方の値が保存されます。
4. URL は有効ですが、計算された金額が指定された金額と一致しません – API は指定された金額を保存し、エラーを返すことなく EstimationUrl を null に設定します。

> **Note**
>
> 両方の値が既存の値と一致しない場合、API はまず URL から量を計算しようとします。不一致や検証の問題が発生した場合は、手動で指定した金額が常に優先されます。

主な利点: 無効な URLsオポチュニティの作成をブロックしない

このアプローチにより、無効な料金計算ツール URL またはアクセスできないAWS料金計算ツール URLs、オポチュニティの作成または更新が妨げられることはありません。手動で提供された金額は、競合や検証の問題が発生した場合に常に優先され、オポチュニティデータを完全に制御できます。

拡張 AwsOpportunitySummary データモデルについて

このセクションでは、 `GetAwsOpportunitySummary` API アクションによって返されるレスポンス構造について説明します。

> **Note**
>
> AI 予測 MRR は で返されます `Project.ExpectedCustomerSpend[].Amount`

`GetAwsOpportunitySummary` API アクションは、ソース属性による包括的な支出分析を提供する新しい`Insights.AwsProductsSpendInsightsBySource`構造を通じて、ディールサイジングのインサイトを返します。

ソース分離:AWSパートナーデータ

ディールサイジングは、透明性を提供し、二重カウントを防ぐために、インサイトをソースごとに分離します。

- AWS出典: からの AI 製品のレコメンデーションAWS
- パートナーソース: パートナーが提供する料金計算ツール URLs からインポートされた支出データと製品の詳細

この分離により、パートナーは次のことが可能になります。

1. 見積りとAWSレコメンデーションを比較する
2. サイジングの前提を検証する
3. 両方のソースを誤って組み合わせて合計を膨張させないようにする
4. データ出所の可視性を維持する

構造の概要

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

```
    }
  },
  "Project": {
    "ExpectedCustomerSpend": [ ]
  },
  "RelatedEntityIds": {
    "AwsProducts": [ ]
  }
}
```

フィールドリファレンス

AwsProductsSpendInsightsBySource マップ

AWSレコメンデーションとパートナー見積りの独立した分析を提供するソース区切りの支出インサイト。

`Insights.AwsProductsSpendInsightsBySource` フィールドは、最大 2 つのキーを含むマップです。

キー	型	説明
AWS	AwsProductsSpendInsights	の推奨製品を含む AI 生成のインサイトAWS
Partner	AwsProductsSpendInsights	詳細なサービスコストや最適化など、料金計算ツール URLs から派生したパートナーソースのインサイト

> **Note**
>
> 利用可能なデータを持つソースのみがレスポンスに含まれます。新しいオポチュニティは通常、入力されたAWSソースのみで始まります。

AwsProductInsights オブジェクト

合計金額、最適化による削減額、プログラムカテゴリの内訳、詳細な製品レベルのインサイトなど、単一のソース (AWSまたはパートナー) の包括的な支出分析。

各ソースオブジェクトには、次のフィールドが含まれます。

フィールド	タイプ	説明
CurrencyCode	string	ISO 4217 通貨コード。サポートされている値は「USD」と「EUR」です。
Frequency	string	顧客が予測額を費やすことが予想される頻度を示します。頻度フィールドでは、毎月の値のみが許可され、定期的な月額支出を表します。
TotalAmount	string	最適化前のこのソースの推定支出の合計
TotalOptimizedAmount	string	推奨最適化を適用した後の合計推定支出
TotalPotentialSavingsAmount	string	最適化を実装することで実現可能なコスト削減の定量化
TotalAmountByCategory	オブジェクト	AWSプログラムとモダナイゼーションパスにマッピングされた支出額
AwsProducts	array	コストや最適化の推奨事項を含む製品レベルの詳細

現在の状態の制限:AWSソースについては、製品ごとの支出のレコメンデーションが利用できない場合、TotalAmount、TotalOptimizedAmount、および TotalPotentialSavingsAmount を省略できます。このような場合、パートナーは合計AWS MRR 見積り`Project.ExpectedCustomerSpend[].Amount`について を参照する必要があります。

TotalAmountByCategory オブジェクト

支出額をAWSプログラムと戦略的イニシアチブにマッピングします。

カテゴリキー	説明
MAP 対応	Migration Acceleration Program の資金の対象となるサービスへの支出
CAT 対応	コスト配分と追跡をサポートするサービス
パスウェイ - クラウドネイティブへの移行	クラウドネイティブのモダナイゼーションに沿ったサービス
パスウェイ - マネージドサービスへの移行	マネージドサービスの導入をサポートするサービス
パスウェイ - オープンソースに移動する	オープンソースサービスの代替案
パスウェイ - コンテナに移動する	コンテナベースのサービスオプション
パスウェイ - サーバーレスへの移行	サーバーレスコンピューティングサービス
パスウェイ - データベースに移動する	データベースモダナイゼーションサービス
パスウェイ - 分析に移動する	分析およびデータ処理サービス

> **Note**
>
> 個々の製品が複数のカテゴリに属する可能性があるため、カテゴリの合計金額が TotalAmount を超える可能性があります。

AwsProducts 配列

プログラム適格性指標 (MAP、モダナイゼーションパスウェイ) 、コスト見積もり、最適化に関する推奨事項を含むAWSサービスのリスト。

各製品オブジェクトには以下が含まれます。

フィールド	Type	必須	説明
ProductCode	string	はい	AWS機会の関連付けに使用される Partner Central 製品識別子
ServiceCode	string	いいえ	料金計算ツールサービスコード (元の計算ツール URL へのリンク)
カテゴリ	array	はい	この製品が対象となるプログラムおよびパスカテゴリのリスト
Amount	string	いいえ	最適化前のベースラインサービスコスト (ソースレコメンデーションでは null AWS になる場合があります)
OptimizedAmount	string	いいえ	最適化を適用した後のサービスコスト (ソースレコメンデーションでは null AWS になる場合があります)
PotentialSavingsAmount	string	いいえ	最適化によるサービス固有のコスト削減 (ソースレコメンデーションでは null AWS になる場合があります)

フィールド	Type	必須	説明
最適化	array	いいえ	この製品の特定の最適化レコメンデーションのリスト

Null Handling: For AWS-sourced products、Amount、OptimizedAmount、および PotentialSavingsAmount フィールドは、製品ごとの支出のレコメンデーションが利用できない場合、null になることがあります。システムは`Project.ExpectedCustomerSpend[0].Amount`代わりに を介して合計 MRR を提供します。

最適化オブジェクト

各最適化レコメンデーションには以下が含まれます。

フィールド	タイプ	説明
説明	string	最適化戦略の人間が読める説明
SavingsAmount	string	この最適化を実装することで実現可能な定量化されたコスト削減

例

例 1:AWS推奨事項のみ (現在の状態)

この例では、AI レコメンデーションのみが利用可能で、製品あたりの支出額がまだ推奨できない場合の一般的なレスポンスを示しています。

```
{
  "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 には、カテゴリはあるが製品あたりの金額がない推奨製品が表示されます。経由で利用可能な合計 MRR 見積り`Project.ExpectedCustomerSpend[].Amount`。

例 2: パートナー料金計算ツールのインポート

この例では、パートナーが有効な料金計算ツール URL を提供した場合のインサイトが強化されています。

```
{
  "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": []
  }
}
```

キーポイント: パートナーソースには、製品あたりの金額を含む完全な支出の詳細が含まれています。最適化に関する推奨事項は、実用的なコスト削減戦略を提供します。カテゴリの内訳は、MAP 資格 (合計 32,500 USD の 22,500 USD) を示しています。製品には、料金計算ツール設定への ServiceCode リンクバックが含まれます。

例 3: 両方のソースが存在する (比較シナリオ)

この例では、AWSレコメンデーションとパートナー見積りの両方がどのように表示されるかを示し、比較を可能にします。

```
{
  "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"
    ]
  }
}
```

キーポイント: 両方のソース present:AWS recommendations (合計 28,000 USD) と Partner Calculator (合計 32.5,000 USD)。AWS recommendations には 4 つの製品が表示され、 Partner Calculator には 4 つの製品 (3 つの重複) が表示されます。パートナーは、レAWSコメンデーションと詳細なestimates.AWS total を比較できますProject.ExpectedCustomerSpend[].Amount。

ベストプラクティス

Null 値の処理

1. EstimationUrl フィールド: ほとんどのパートナーで EstimationUrl が null であることが必要です。これは標準的な動作であり、エラーとして扱うべきではありません。手動 MRR エントリオプションをプライマリパスとして表示します。
2. 欠落しているカテゴリ:AWSソースに TotalAmountByCategory が存在しない場合、これは製品ごとのレコメンデーションが利用できないことを示します。カテゴリの内訳は、料金計算ツール URLsが指定されている場合、パートナーソースデータで使用できます。

最適化に関する推奨事項

1. 実用的なインサイトを表示する: 最適化の説明と削減額を目立つように表示し、パートナーがコスト削減の機会を理解できるようにします。
2. Aggregate Savings: Sum PotentialSavingsAmount 製品全体で合計最適化の可能性を示します。
3. 影響による優先順位付け: SavingsAmount で最適化をソートして、パートナーが最も影響の大きいレコメンデーションに集中できるようにします。

更新のモニタリング

1. 定期的なポーリング: 機会作成ワークフロー中に`GetAwsOpportunitySummary`定期的に (推奨: 5 分ごと) ポーリングします。
2. 非同期生成の処理: ディールサイジングインサイトは、オポチュニティの作成後すぐには利用できない場合があります。適切なロード状態を実装し、ロジックを再試行します。

API 統合パターン

1. 既存の統合を活用する: を既に呼び出しているパートナーは、レスポンスの追加フィールド`GetAwsOpportunitySummary`を使用するだけで済み、新しい API コールは必要ありません。
2. グレースフルデグレード: ディールサイジングデータがまだ利用できない、または不完全であるシナリオを処理するように UIs を設計します。

データの品質と精度

1. 料金計算ツール URLs の検証: null EstimationUrl レスポンスを回避するために、送信前に URLs が有効であることを確認します。
2. 豊富な機会の詳細を提供する: より完全な機会情報 (顧客の詳細、プロジェクトの説明、技術要件) により、より正確な AI レコメンデーションが得られます。
3. 差異の確認: パートナーの見積りがAWSレコメンデーションと大きく異なる場合は、機会の詳細を確認して、関連するすべてのコンテキストがキャプチャされていることを確認します。

ベストプラクティス

イベントへの対応

AWS Partner Central API からイベントを処理するときは、処理ロジックが重複イベントを処理するべき等であることを確認します。各イベントに対して GetOpportunity をすぐに呼び出す代わりに、アプリケーションのニーズに基づいて詳細をバッチ処理または選択的に取得することを検討してください。中断のないオペレーションの場合は、クォータに注意してください。

オプティミスティックロックの実装

オプティミスティックロックは、同時更新中の意図しないデータ上書きを防ぎます。一般的なシナリオを次に示します。

1. パートナーは CRM システムからオポチュニティを取得します。

2. ユーザーはAWS Partner Central でオポチュニティAを更新します。
3. ユーザーは CRM 統合を通じて同じ機会を同時にB更新します。
4. データが変更されると、CRM システムはデータのアップロードを試みますが、 を返しますConflictException。
5. ユーザーはエラーを確認し、競合するデータを手動で解決します。

このシナリオを回避するには、すべての UpdateOpportunity リクエストに、以前の CreateOpportunity、および GetOpportunity アクションから取得できる LastModifiedDateパラメータが含まれている必要があります。 UpdateOpportunity 更新は、 がシステムLastModifiedDateと一致する場合にのみ成功します。そうでない場合は、GetOpportunity LastModifiedDateを使用して最新の を取得し、更新を再試行する必要があります。

CRM とAWS Partner Central 間のデータの同期

システムを Partner Central の最新データと同期させることが重要です。以下は、システムが最新のデータを確実に反映するための 2 つの戦略です。

イベントの使用 (推奨)

1. ListOpportunities を使用してデータをロードします。
2. オポチュニティイベントにサブスクライブします。
3. 新しい機会や変化に対応します。
   - Opportunity Created、Opportunity Updated、または Engagement Invitation Acceptedイベントを受け取ったら、 GetOpportunityを使用して最新のデータを取得します。
   - Engagement Invitation Rejected イベントを受け取ったら、対応するオポチュニティを削除します。

ListOpportunities ポーリングの使用

1. ListOpportunities を使用してデータをロードします。
2. 毎日の読み取りクォータを使い果たさないように、ポーリング頻度を選択します。
3. 保存されたデータLastModifiedDateから最新のデータを特定し、データの送信元を確認しますAWS。
4. ListOpportunities を呼び出すときは、AfterLastModifiedDateフィルターのタイムスタンプを使用します。

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

5. AWSは、タイムスタンプに示された値の後に作成または更新されたオポチュニティを返します。
6. を使用して返されたすべてのページを反復処理しNextToken、GetOpportunity を使用してシステムのデータを更新します。

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

AWS Partner Central Benefits API の使用

AWS Partner Central Benefits API は、パートナーがすべてのAWSパートナープログラムでメリットを検出、申請、管理する方法を合理化します。メリット管理を 1 つの直感的なインターフェイスに一元化することで、このサービスは、運用を簡素化しながらパートナーに報酬を与える透過的な能力向上を実現します。

API は、標準のAWS API 機能を提供します。API アクションを使用するか、プログラミング言語またはプラットフォームに合わせたAWS SDK を使用してアクセスします。詳細については、「 の開始方法AWS」および「構築するツールAWS」を参照してください。

利点とは

AWSパートナーネットワーク (APN) では、メリットとは、ビジネスの成長と顧客の成功をサポートするためにパートナーがAWSから得ることができる利点または能力を表します。メリットは、資格と貢献度に基づいてパートナーに報酬を与えると同時に、パートナーがAWSプラクティスをスケールするのに役立つ具体的な価値を提供するように設計されています。

利点には、次のようなさまざまな形式があります。

- 財務上のインセンティブ - クレジット、現金の支払い、移行促進プログラム (MAP) やマーケティング開発資金 (MDF) などの資金プログラム
- アクセスの利点 - 新しいサービス、ツール、または特殊なリソースへのアクセスをプレビューする
- 認識 - デジタルバッジ、認定、コンピテンシーの指定、パートナー階層のステータス
- テクニカルリソース - ソリューションアーキテクチャのサポート、テクニカルサポート、ProServe エンゲージメント
- マーケティングサポート - 共同マーケティングの機会、注目のパートナーステータス、成功事例の出版物

利用可能な利点の発見

パートナーは、どのようなメリットが得られるかを発見し、資格要件を理解することで、メリットジャーニーを開始します。Benefits API は、Benefit Discovery APIsを通じて包括的な検出機能を提供します。

利用可能な利点の一覧表示

パートナーは API `ListBenefits`アクションを使用して、利用可能なすべての利点を表示できます。このアクションは、パートナーが関連する機会をすばやく見つけることができるように、オプションのフィルタリング機能を備えた利点のカタログを取得します。

メリットを効率的に発見するために、パートナーは以下をフィルタリングできます。

- プログラム - MAP (移行促進プログラム）、MDF (マーケティング開発資金）、ISV ワークロード移行、イノベーションサンドボックス、概念実証、パートナーイニシアチブ資金などの特定のパートナーAWSプログラムでフィルタリングします
- フルフィルメントタイプ - メリットの提供方法でフィルタリングする: `CREDITS`、`CASH`、`DISCOUNT`、`ACCESSRECOGNITION`、、または `RESOURCE`
- ステータス - 特典ステータスでフィルタリング: `ACTIVE`または `DEPRECATED`

ListBenefits API は、利点 ID、名前、説明、関連するプログラム、フルフィルメントタイプなどの重要な情報を含む利点の概要を返します。この概要ビューを使用すると、パートナーは利用可能な利点をすばやくスキャンし、ビジネス目標に最も関連性の高い利点を特定できます。

フィルタリングアプローチの例:

財務インセンティブに重点を置いたパートナーは、フルフィルメントタイプでフィルタリングCREDITSしCASH、資金調達の機会を確認する場合があります。技術的な有効化を求めるパートナーは、フルフィルメントタイプでフィルタリングRESOURCEしたりACCESS、テクニカルサポートの利点やツールへのアクセスを見つけたりする場合があります。

詳細なメリット情報の表示

パートナーが関心のある利点を特定したら、 GetBenefit API アクションを使用して完全な詳細を取得できます。これにより、次のような包括的な情報が提供されます。

- 適格性基準 - 特典を受けるためにパートナーが満たす必要がある詳細な要件
- メリットリクエストスキーマ - 適用時にパートナーが提供する必要がある特定の情報とドキュメント
- グラントの詳細 - 承認時にパートナーが受け取るもの
- 関連プログラム - 特典がサポートするAWSパートナープログラム

メリットリクエストスキーマは、メリットアプリケーションの構造と必須フィールドを定義するため、特に重要です。パートナーは、利点アプリケーションを作成する前にこのスキーマを注意深く確認し、必要な情報をすべて事前に収集する必要があります。

> **Important**
>
> 各利点の適格性の判断は、クライアントまたは UI レイヤーによって処理されます。資金関連の利点については、資格は通常、パートナーのスコアカードのパフォーマンスとプログラムへの参加に基づいています。

トピック

- メリットアプリケーションの使用
- メリット配分の使用

メリットアプリケーションの使用

メリットアプリケーションは、特定のメリットに対するパートナーのリクエストをモデル化します。メリット固有の条件に従ってリクエストを評価および処理するために必要なすべての情報をキャプチャします。利点アプリケーションのライフサイクルは、ドラフトの作成から最終的な承認または拒否まで、複数の状態を経ます。

メリットアプリケーションの作成

パートナーは、 CreateBenefitApplication API アクションを使用してメリットアプリケーションを作成することで、メリットリクエストプロセスを開始します。作成時にアプリケーションは PENDING_SUBMISSION ステータスになり、パートナーはレビューのために送信する前に完全な情報を準備することができます。

メリットアプリケーションを作成する場合、パートナーは以下を提供する必要があります。

- メリット識別子 - リクエストされているメリットの ID または ARN
- フルフィルメントタイプ - パートナーがメリットを期待する方法 (CREDITS、CASH、DISCOUNT、ACCESSRECOGNITION、、または RESOURCE)
- メリットアプリケーションの詳細 - メリットのリクエストスキーマで定義されたメリット固有の情報を含む JSON ドキュメント
- クライアントトークン - 重複した送信を防ぐための一意のべき等性トークン

パートナーはオプションで以下を提供できます。

- 名前と説明 - アプリケーションの人間が読み取れる識別子
- パートナー連絡先 - この特典リクエストを管理するユーザーの連絡先情報 (最大 1 件の連絡先)
- ファイル添付ファイル - プロジェクトプラン、SOWs、コスト証明、顧客満足度調査などのサポートドキュメント (最大 10 ファイル)
- 関連リソース - 関連する機会または既存のメリット配分へのリンク (最大 10 リソース)
- タグ - リソースの整理と追跡のためのキーと値のペア (最大 200 タグ)

CreateBenefitApplication API はソフト検証を実行し、基本的なフィールドタイプとパターンのみをチェックします。完全なビジネスロジックの検証は送信中に行われるため、パートナーは不完全なアプリケーションを保存し、後で戻って完了できます。

ベストプラクティス: パートナーは の特典リクエストスキーマを使用してGetBenefit、アプリケーションを作成する前に必要な情報を正確に理解する必要があります。これにより、データの欠落や無効なデータによる送信失敗の可能性が低くなります。

メリットアプリケーションの更新

パートナーは、 UpdateBenefitApplication API アクションを使用して、ドラフトメリットアプリケーションを変更できます。これにより、パートナーは送信前に情報を絞り込んだり、ドキュメントを追加したり、エラーを修正したりできます。

メリットアプリケーションを更新する場合、パートナーは以下を提供する必要があります。

- 識別子 - 更新するメリットアプリケーションの ID または ARN
- リビジョン - オプティミスティックロックの現在のリビジョン番号
- 利点アプリケーションの詳細 - 完全に更新された JSON ドキュメント

特定のフィールドのみが変更されていても、パートナーは完全なメリットアプリケーションオブジェクトを送信する必要があります。ベストプラクティスは、まず を使用して最新のアプリケーションの詳細を取得しGetBenefitApplication、必要なフィールドを変更してから、更新されたペイロード全体を に送信することですUpdateBenefitApplication。

オプティミスティックロック: リビジョンフィールドは、アプリケーションが最後に取得されてから変更されていない場合にのみ更新が適用されるようにします。リビジョンが現在のデータベース値と一致しない場合、更新は競合エラーで拒否されます。これにより、パートナーが他のプロセスやシステムによって行われた変更を誤って上書きすることを防ぎます。

更新は、アプリケーションが PENDING_SUBMISSIONステータスの場合にのみ実行できます。送信されると、アプリケーションはレビュープロセスに入り、この API を通じて更新できなくなります。

リソースとメリットアプリケーションの関連付け

パートナーは、 AssociateBenefitApplicationResource API アクションを使用して、メリットアプリケーションを関連リソースにリンクできます。これにより、利点とそれらが使用されているビジネスコンテキストとの間に貴重なつながりが生まれます。

サポートされているリソースタイプは以下のとおりです。

- 機会 - メリットアプリケーションを の特定の顧客機会にリンクします。これは、MAP 資金やカスタマーエンゲージメントに関連する POC クレジットなどの機会固有の利点に特に役立ちます。

- BENEFIT_ALLOCATION - 既存のメリット配分へのリンク。パートナーがメリットを連鎖させたり、以前のメリットがどのように活用されているかを示すことができます。

リソースの関連付けは、送信前にのみ行うことができます。メリットアプリケーションが送信されると (ステータスが に変更されるとIN_REVIEW)、それ以上の関連付けは許可されません。パートナーは、各利点アプリケーションに最大 10 個のリソースを関連付けることができます。

リソースの関連付けを解除するには、パートナーは DisassociateBenefitApplicationResource API アクションを使用します。関連付けと同様に、関連付け解除は PENDING_SUBMISSIONステータスでのみ実行できます。

> **Important**
>
> パートナーは、メリットアプリケーションにアクセスし、関連付けられている特定のリソースを読み取るための適切なアクセス許可を持っている必要があります。API は、関連付けオペレーション中にこれらのアクセス許可を検証します。

メリットアプリケーションの送信

メリットアプリケーションが完了し、AWSレビューの準備ができると、パートナーは SubmitBenefitApplication API アクションを使用してそれを送信します。送信により、 から PENDING_SUBMISSION への状態遷移がトリガーIN_REVIEWされ、利点固有の承認ワークフローが開始されます。

送信時に、API は以下を含む包括的な検証を実行します。

- 必須フィールドの検証 - 利点アプリケーションの詳細のすべての必須フィールドが存在することを確認します
- フィールド形式の検証 - フィールド値が予想されるパターンとデータ型と一致することを確認します
- ビジネスルールの検証 - 利点固有のビジネスロジックと制約を適用します
- リソースの検証 - 関連するすべてのリソースが有効でアクセス可能であることを確認します
- ファイル検証 - すべてのファイルアタッチメントが正常に処理を完了したことを確認します

検証が失敗した場合、API は修正が必要なフィールドを示す詳細なエラーコードとメッセージを含む ValidationException を返します。パートナーは、送信を再試行UpdateBenefitApplicationする前に、 を使用してこれらの問題に対処する必要があります。

ファイル処理要件: アタッチされたファイルがステータスのままの場合、メリットアプリケーションを送信できませんPENDING。パートナーは、ファイル処理が完了する (ステータスが に変更されるSUCCEEDED) のを待ってから送信する必要があります。ファイルの処理に失敗した場合 (ステータス FAILED)、パートナーは修正されたバージョンをアップロードする必要があります。

送信が成功した後:

- アプリケーションのステータスが に変わります。 IN_REVIEW
- 特典所有者のチームに新しい送信が通知されます。
- パートナーは標準更新 APIs
- アプリケーションは、ビジネス承認、技術承認、財務承認の各段階を含む、定義された承認ワークフローに入ります。

パートナーは、アプリケーションを取得し、現在の承認ステージ (「ビジネス承認」、「技術承認」、「財務承認」など) を示すステージフィールドをモニタリングすることで、送信の進行状況を追跡できます。

送信済みアプリケーションの管理

送信後、パートナーには利点アプリケーションを管理するためのオプションが限られていますが、API は一般的なシナリオに対して特定のアクションを提供します。

メリットアプリケーションの呼び出し

パートナーがエラーを検出した場合、または送信後に変更を加える必要がある場合は、RecallBenefitApplication API アクションを使用してアプリケーションを呼び出すことができます。リコールはアプリケーションを PENDING_SUBMISSIONステータスに移行し、更新と再送信を可能にします。

アプリケーションを呼び出す場合、パートナーは以下を提供する必要があります。

- 識別子 - リコールするメリットアプリケーションの ID または ARN
- 理由 - リコールに関するオプションの説明 (最大 1000 文字)

理由フィールドを使用すると、トレーサビリティが向上し、リコールにつながる一般的な問題AWSを理解し、メリットアプリケーションプロセスの今後の改善に役立てることができます。

リコール後、パートナーは UpdateBenefitApplicationを使用して必要な変更を行い、準備SubmitBenefitApplicationができたら を使用して再送信できます。

> **Important**
>
> リコールは通常、早期レビュー段階でのみ使用できます。後の承認段階に進んだアプリケーションや承認されたアプリケーションは、リコールの対象とならない場合があります。

メリットアプリケーションの修正

送信後の軽微な修正の場合、パートナーは AmendBenefitApplication API アクションを使用して、アプリケーション全体を呼び出すことなく特定のフィールドを更新できます。これは、AWSレビュープロセス中にレビュアーが明確化または修正をリクエストする場合に特に便利です。

修正では、パートナーが以下を指定する JSON パッチ形式のアプローチを使用します。

- パス - 更新するフィールドを識別する JSONPath 式 (例: `$.CreditDisbursementDetails.AwsAccountIdForCredits`)
- 値 - フィールドの新しい値
- オペレーション - 実行するオペレーション (現在REPLACEは のみサポートされています)

パートナーは、1 回の API コールで最大 10 件の修正を送信できます。各修正には、オプティミスティックロック用に更新されたリビジョン番号を含める必要があります。

軽微な修正には修正を使用する必要があります。大幅な変更を行う場合は、RecallBenefitApplicationを使用してアプリケーションをドラフトステータスに戻し、包括的な更新を行う必要があります。

メリットアプリケーションのキャンセル

パートナーに利点がなくなった場合、またはアプリケーションの取り消しを希望する場合は、CancelBenefitApplication API アクションを使用してキャンセルできます。キャンセルすると、アプリケーションは CANCELED ステータスに移行し、アプリケーションプロセスが完全に終了します。

アプリケーションをキャンセルする場合、パートナーは以下を提供する必要があります。

- 識別子 - キャンセルする特典アプリケーションの ID または ARN
- 理由 - キャンセルのオプションの説明 (最大 1000 文字)

キャンセルされると、アプリケーションを再アクティブ化することはできません。パートナーが後でメリットが必要と判断した場合は、新しいメリットアプリケーションを作成する必要があります。

キャンセルの一般的な理由は次のとおりです。

- 顧客の機会が失われた、または遅延した
- パートナー容量の制約が変更されました
- ビジネスの優先順位の移行
- 意図した目的にメリットが不要になった

メリットアプリケーションの詳細の表示

パートナーは、 `GetBenefitApplication` API アクションを使用してメリットアプリケーションに関する完全な情報を取得できます。これにより、以下を含むアプリケーションの包括的なビューが提供されます。

アプリケーションメタデータ:

- 一意の識別子 (ID) と Amazon リソースネーム (ARN)
- 関連するメリット識別子
- アプリケーション名と説明
- 現在のステータス (`PENDING_SUBMISSION`、`IN_REVIEW`、`ACTION_REQUIRED`、`APPROVED`、`REJECTED`、または `CANCELED`)
- 現在の処理ステージ

ステータス情報:

- ステータスの理由 - 現在のステータスの人間が読み取れる説明
- ステータス理由コード - 特定の問題または要件を示す構造化コード (「Checklist Incomplete」、「Project Plan Missing」、「Design Win Missing」など)

アプリケーションコンテンツ:

- 完全なメリットアプリケーションの詳細 JSON ドキュメント
- パートナーの連絡先情報
- 処理ステータスの添付ファイル
- 関連付けられたリソース (機会または割り当て)
- リソースタグ

監査情報:

- 作成タイムスタンプ
- 最終変更タイムスタンプ
- 現在のリビジョン番号

ステータス理由コードは、アプリケーションが ACTION_REQUIREDステータスの場合に特に役立ちます。これらのコードは、アプリケーションを進めるためにパートナーが対処する必要があることに関する具体的で実用的なガイダンスを提供します。

メリットアプリケーションの一覧表示

パートナーは、 ListBenefitApplications API アクションを使用してすべての利点アプリケーションを表示できます。これにより、強力なフィルタリング機能を備えたアプリケーション概要のページ分割されたリストが返されます。

パートナーは、次の方法でアプリケーションをフィルタリングできます。

- プログラム - 特定のプログラム (MAP、MDF、サンドボックス、POC など) のアプリケーションを表示します。
- フルフィルメントタイプ - 配信方法 (CREDITS、CASH、 ACCESSなど) でフィルタリングする
- メリット識別子 - 特定のメリットのアプリケーションを表示する
- ステータス - アプリケーションステータス (PENDING_SUBMISSION、IN_REVIEW、、REJECTED、CANCELED) でフィルタリングAPPROVEDします。
- ステージ - 承認ステージ (ビジネス承認、技術承認、財務承認) でフィルタリングする
- 関連付けられたリソース ARNs - 特定の機会または割り当てにリンクされたアプリケーションを検索する

リストレスポンスには、次のような重要な概要情報が含まれます。

- アプリケーション ID、ARN、および名前
- 関連付けられた利点 ID
- プログラムとフルフィルメントタイプ
- 現在のステータスとステージ
- 作成と変更のタイムスタンプ
- 関連リソース
- 現在のリビジョン番号
- 特典アプリケーションの詳細から選択されたフィールド (特典所有者が定義)

パートナーは、Sort パラメータを使用して結果のソートを設定できます。これには、作成日、ステータス、プログラム、または利点識別子で昇順または降順でソートするオプションがあります。

ダッシュボードの構築: `ListBenefitApplications` API は、パートナーダッシュボードの作成をサポートするように設計されています。アプリケーションをフィルタリングおよびソートすることで、パートナーは次のようなビューを構築できます。

- アクションが必要なアプリケーション (`ACTION_REQUIRED` ステータス)
- 最近送信されたアプリケーション (作成日、`IN_REVIEW`ステータスでソート)
- 割り当て保留中の承認済みアプリケーション (`APPROVED` ステータス)
- プログラムによるアプリケーション (特定のプログラムでフィルタリング)

メリット配分の使用

メリット配分は、パートナーに付与されたメリットを表します。メリットアプリケーションが承認されると、 は、実際のクレジット、資金、アクセス、またはその他のメリットフルフィルメントをパートナーに提供する 1 つ以上のメリット配分AWSを作成します。

メリット配分について

メリット配分は、さまざまなタイプのフルフィルメントを表すことができます。

- 財務支払い (CASH) - 支払い追跡機能を備えたパートナーへの直接財務支払い
- AWSクレジット (CREDITS) - 使用状況を追跡するAWSアカウントに適用されるクレジットコード
- 消費可能な配分 - 使用率を追跡する事前承認済みの資金金額またはウォレット

- Access Grants (ACCESS) - システム、ツール、またはリソースへのアクセス
- 認識 (RECOGNITION) - デジタルバッジ、証明書、またはステータス指定
- リソース (リソース) - テクニカルサポート、アドバイザリサービス、または ProServe エンゲージメント

各利点の割り当てには、ステータスフィールドを通じて追跡されるライフサイクルがあります。

- ACTIVE - 割り当てを使用できます
- INACTIVE - 割り当ては一時的に使用できません
- FULFILLED - 割り当てが完全に使用または配信されました

メリット配分には、いつ有効になるか (StartsAt) といつ期限切れになるか (ExpiresAt) を定義する時間情報も含まれます。これにより、パートナーはメリットを活用するための時間枠を理解できます。

メリット配分の一覧表示

パートナーは、 `ListBenefitAllocations` API アクションを使用してすべてのメリット配分を表示できます。これにより、包括的なフィルタリング機能を備えた割り当ての概要のページ分割されたリストが返されます。

パートナーは、以下によって割り当てをフィルタリングできます。

- メリット識別子 - 特定のメリットの割り当てを表示する
- メリットアプリケーション識別子 - 特定のアプリケーションから生じる割り当てを表示する
- フルフィルメントタイプ - 割り当てタイプ (`CREDITS`、`CASH`、 `ACCESS`など) でフィルタリングする
- ステータス - 割り当てステータス (`ACTIVE`、`INACTIVE`、`FULFILLED`) でフィルタリングする

リストレスポンスには、以下を含む割り当ての概要が含まれます。

- 割り当て ID、ARN、および名前
- 関連する特典 ID と特典アプリケーション ID (該当する場合)
- フルフィルメントタイプ
- 現在のステータス
- 作成タイムスタンプ

- [開始日]

このリストビューを使用すると、パートナーは配分追跡ダッシュボードを構築し、以下を特定できます。

- すぐに使用できるアクティブな割り当て
- 割り当ての有効期限が近づいています
- 履歴追跡の達成された割り当て
- プログラムまたはフルフィルメントタイプ別の合計割り当て値

詳細な配分情報の表示

パートナーは GetBenefitAllocation API アクションを使用して、完全な割り当ての詳細を取得できます。これにより、割り当てタイプによって異なるフルフィルメント固有の詳細など、割り当てに関する包括的な情報が提供されます。

コア割り当て情報:

- 一意の識別子 (ID) と Amazon リソースネーム (ARN)
- 割り当て名とステータス
- ステータスの理由 (現在のステータスの説明)
- 関連するメリット ID とメリットアプリケーション ID
- フルフィルメントタイプ
- 作成、更新、開始、有効期限のタイムスタンプ

フルフィルメントの詳細 (タイプ固有):

FulfillmentDetail フィールドには、フルフィルメントタイプに基づいて異なる情報構造が含まれています。

資金支払いの詳細 (CASH タイプ)

直接現金支払いの場合、フルフィルメントの詳細には以下が含まれます。

- 支払い金額 - 金額の値と通貨コードを含む、支払い済みの資金の合計金額
- 発行の詳細 (該当する場合) - 発注書または発行イベントに関する情報:
  - 発行 ID - 支払いの一意の識別子

- 発行金額 - 現地通貨での金額
- 発行日時 - 支払いのタイムスタンプ

この構造は、1 回の支払いと、1 回の配分に対して複数の発行が発生する可能性がある事前承認済みの資金シナリオの両方をサポートします。

クレジットの詳細 (クレジットタイプ)

AWSクレジットの場合、フルフィルメントの詳細は次のとおりです。

- 配分額 - 配分された合計クレジット額
- 発行額 - 発行された合計クレジット額 (段階的な発行に割り当てられた額を下回る場合があります)
- クレジットコード - 個々のクレジットコードの詳細の配列:
  - AWSアカウント ID - クレジットが適用されるアカウント
  - 値 - クレジットコードの金額
  - AWSクレジットコード - 実際のクレジットコード文字列
  - ステータス - クレジットコードのステータス (ACTIVE、INACTIVE、FULFILLED)
  - 発行日時 - クレジットコードが発行された日時
  - 有効期限 - クレジットコードの有効期限

この詳細なクレジット追跡により、パートナーは複数のアカウントのクレジット使用率をモニタリングし、利用可能なクレジット、部分的に使用されたクレジット、または完全に消費されたクレジットを把握できます。

消費可能な配分の詳細 (事前承認された金額/ウォレット)

事前承認された資金プールの場合、フルフィルメントの詳細には以下が含まれます。

- 配分額 - 配分の合計額
- 残りの金額 - 未使用の金額はまだ利用可能
- 使用金額 - 既に消費されている金額
- 発行の詳細 (該当する場合) - 割り当ての発注書情報

消費可能な配分により、パートナーは対象となるアクティビティに必要な資金を引き出し、残高をリアルタイムで追跡できます。

アクセスの詳細 (ACCESS タイプ)

アクセス許可の場合、フルフィルメントの詳細には以下が含まれます。

- 説明 - 付与されるアクセスとその使用方法の詳細な説明

アクセス割り当ては、プレビューサービス、特殊なツール、または制限されたリソースへのアクセスを提供する場合があります。説明フィールドには、パートナーが付与されたアクセスをアクティブ化して使用する方法について説明します。

新しい利点アプリケーションへの割り当ての適用

メリット配分の ApplicableBenefitIds フィールドは、この配分を活用できるその他のメリットを特定します。これにより、パートナーが既存の割り当てを使用して追加の利点を申請できる利点の連鎖が可能になります。

例えば、パートナーには次のようなものがあります。

1. 事前承認された MAP 資金配分を受け取る
2. 個々の顧客移行プロジェクトの特典アプリケーションを作成する
3. これらのアプリケーションを事前承認された割り当てに関連付ける
4. プロジェクトが承認されると、配分から資金を引き出す

このアプローチにより、事前に承認された資金を持つパートナーのメリット申請プロセスが合理化され、個々のプロジェクトリクエストのレビューサイクルが短縮されます。

AWS Partner Central Channel API の使用

AWS Partner Central Channel APIsを使用すると、承認されたAWS再販ビジネスをプログラムで管理できます。これらの APIsにより、AWSソリューションプロバイダー、ディストリビューター、およびAWSディストリAWSビューション販売者は以下を実行できます。

- 再販プログラムに使用されるAWS管理アカウントを Partner Central に報告する
- パートナー関連付けを受け入れるための招待をAWSアカウントに送信する
- 再販されたエンドユーザーAWSアカウントのオンボーディングに必要な関係を作成し、パートナー割引を適用する

AWS Partner Central Channel APIs を使用すると、以下のカスタムオートメーションを構築できます。

- 新しいパートナー所有AWSアカウントを APN 再販プログラムにオンボードする
- 新しいエンドユーザーアカウントをオンボーディングしてプログラムを再販売する
- 顧客のオンボーディングと割引資格の促進

プログラム管理アカウント (PMAs) の使用

プログラム管理アカウント (PMA) はAWS、管理アカウントをチャネルプログラム登録に関連付けます。プログラム管理アカウントは、チャネルハンドシェイクを使用してセルフサービスがアクティブ化されます。PMA を作成してアクティブ化すると、チャネルプログラムの特典と割引が、関連するAWS管理アカウントに配信されるすべての請求書に適用されます。これらの APIs を使用して、再販とチャネル割引を処理するAWSアカウントをレポートおよび管理します。

- `CreateProgramManagementAccount`: 顧客請求管理用の新しいアカウントを追加する
- `UpdateProgramManagementAccount`: 既存のアカウントを変更する
- `DeleteProgramManagementAccount`: 再販プログラムからアカウントを削除する
- `ListProgramManagementAccounts`: 既存のアカウントとそのステータスを表示する

チャネルハンドシェイクの使用

チャネルハンドシェイクは、重要なチャネル管理オペレーションに対する安全で相互の同意を可能にします。AWS Partner Central は、チャネルハンドシェイクを使用して、プログラム管理アカウント (PMAs) のアクティブ化、サービス期間の確立、サービス期間の早期終了の 3 つの異なる目的で、AWS管理アカウントからの同意を検証します。次の 3 つのタイプがあります。

- `PROGRAM_MANAGEMENT_ACCOUNT`: プログラム管理アカウントのハンドシェイクは、PMAs をアクティブ化してチャネルプログラムの利点を適用するときに同意を検証します
- `START_SERVICE_PERIOD`: サービス期間作成ハンドシェイクは、最小通知期間または固定期間で請求転送コミットメントの相互契約を確立します。
- `REVOKE_SERVICE_PERIOD`: サービス期間終了ハンドシェイクは、両当事者が同意したときに、アクティブなサービス期間の早期終了を可能にします。すべてのハンドシェイクタイプを有効にするには、ターゲットAWS管理アカウントからの承諾が必要です。

これらの APIs を使用して、PMA アクティベーションとサービス期間管理のチャネルハンドシェイクを管理します。

- `CreateChannelHandshake`: PMA の招待、サービス期間の設定、またはサービス期間の終了のハンドシェイクを作成する
- `AcceptChannelHandshake`: ターゲットAWSアカウントからのハンドシェイクを受け入れる (ハンドシェイクタイプに応じてパートナーまたは顧客が使用できます)
- `RejectChannelHandshake`: ターゲットAWSアカウントからのハンドシェイクを拒否する
- `CancelChannelHandshake`: ハンドシェイクが承諾、拒否、または期限切れになる前に、保留中のハンドシェイクをキャンセルする
- `ListChannelHandshakes`: タイプとステータスでフィルタリングしてハンドシェイクを追跡および表示する

チャネル関係の使用

関係により、エンドユーザー、内部組織、またはダウンストリーム販売者を管理できます。各関係には以下が含まれます。

- 関連付けられたAWS組織のAWS管理アカウント
- チャネル割引資格に必要なAWSパートナーネットワークメタデータ

リレーションシップを作成すると、関連するAWS組織からのすべての使用には、関連するチャネルプログラムの利点があります。これらの APIs を使用して、販売者とエンドユーザーを報告します。

- `CreateRelationship`: 新しいエンドユーザー、ディストリビューション販売者、または内部AWSアカウントをオンボードする
- `GetRelationship`: 特定の関係の詳細を表示する
- `ListRelationships`: すべての関係を表示する
- `UpdateRelationship`: 既存の関係を変更する
- `DeleteRelationship`: 関係を削除する

収益測定 API の使用

トピック

- Marketplace 収益共有の使用

収益属性 IDs の使用

収益属性 ID は、AWSマーケットプレイス製品の収益を特定のAWSマーケットプレイスオファーや AWS Partner Central Opportunities にマッピングするディールレベルの識別子です。これは、製品レベルの Partner Revenue Measurement (PRM) 実装に基づいて構築されます。PRM は Marketplace 製品の属性収益の合計をキャプチャし、Revenue Attribution ID は、指定した毎月のコスト配分の割合に従ってその収益を特定の取引にマッピングします。

Revenue Attribution Service は、Revenue Attribution IDs を作成、取得、一覧表示、更新し、毎月のコスト配分エントリを非同期的に管理するための APIs を公開します。

収益属性 ID とは

収益属性 ID には 2 つのレイヤーがあります。

- `ra-` プレフィックス ID (例: `ra-aabbccddee001`) と ARN によって識別される、パートナーの名前付き、パートナーの説明付きリソースの属性。各属性は、 `Catalog` (AWS本番稼働用、テストSandbox用) とパートナーのアカウントに限定されます。
- 各エントリの 1 つ以上の毎月のコスト配分エントリは、単一の (オファー ID またはオポチュニティ ID、請求月) の組み合わせをコスト配分の割合にマッピングします。エントリは、割り当てタスクパターンを通じて非同期的にバッチで管理されます。

収益属性 ID は、次の 2 つの方法で使用できます。

- 既存の製品レベルの PRM 実装のディールレベルのオーバーレイとして。収益属性 ID を作成し、該当する Marketplace オファーや ACE オポチュニティを関連付け、すでに測定済みの製品収益をそれらの特定の取引にAWSマッピングします。既存のリソースタグやユーザーエージェント文字列に変更を加える必要はありません。
- リソースタグ値 (`aws-apn-id = <RA ID>`) またはユーザーエージェント文字列 (`APN_1.1/pc_<RA ID>$`) として直接使用されるスタンドアロン識別子として、消費を最初から特定のディールに属性AWS付けします。

収益属性 IDs の使用

パートナーは、Revenue Attribution Service を通じて、Revenue Attribution IDs とその毎月のコスト配分エントリを管理できます。ライフサイクルは、属性レコードフロー (作成、更新、取得、一覧表

示) と割り当てフロー (バッチタスクの開始、結果のポーリング、個々のエントリの取得、属性のエントリのリスト) の 2 つの独立したフローを通じて進行します。

収益属性 ID の作成

最初のステップは、 CreateRevenueAttribution API アクションを使用して収益属性 ID を作成することです。返された Idと は、リソースタグ値としてすぐに使用したり、ユーザーエージェント文字列で使用してAWS消費を属性Arn化したりできます。

収益属性 ID を作成する場合、パートナーは以下を指定する必要があります。

- **Catalog** — 本番AWS稼働用またはテストSandbox用。
- **Name** — アトリビューションの人間が読み取れる名前。Catalog とパートナーのアカウント内で一意である必要があります。最大 128 文字

パートナーはオプションで以下を提供できます。

- **Description** — アトリビューションのフリーテキストの説明。最大1024文字
- **MarketplaceProduct** —AWSこの属性に関連付ける Marketplace 製品。(25 文字の Marketplace 製品 ID) と TenancyModel (MULTI_TENANT または) ProductIdentifierを指定しますSINGLE_TENANT。作成時に省略すると、顧客が購入した Marketplace 製品に関係なく、Revenue Attribution ID で測定されるすべての消費に属性が適用されます。これは、単一のデプロイが複数の Marketplace リストにまたがる場合に便利です。
- **Tags** — リソース組織に対して最大 200 個のキーと値のペア。タグキーは、リクエスト内で一意である必要があります。

ベストプラクティス: 取引が特定の Marketplace 出品に固定されるMarketplaceProduct.ProductIdentifierたびに を指定します。これによりAWS、 は、製品が呼び出し元のパートナーアカウントまたは Partner Account Connection (PAC) を介して接続された子会社アカウントによって所有されていることを検証し、解決された ProductCode と ProductType (、SaaSAMI、 などML) をレスポンスに表示します。製品が認可されたアカウントによって所有されていない場合、API は理由 ValidationExceptionで を返しますPRODUCT_NOT_FOUND_OR_NOT_OWNED。

レスポンスは、新しい Id (形式 ra-[a-z0-9]{13})Arn、、解決されたMarketplaceProduct属性、および初期Version数 (1 から始まり、それ以降の更新ごとに増分) を返します。

毎月のコスト配分エントリの追加

収益属性 ID が作成されると、パートナーは StartRevenueAttributionAllocationsTask API アクションを通じて毎月のコスト配分エントリのバッチを送信することで、AWSマーケットプレイスオファーや ACE オポチュニティを関連付けます。これは、タスクごとに最大 250 個の割り当て変更 (CREATE および/または UPDATE) を受け入れる非同期オペレーションです。

各割り当てエントリは以下を指定します。

- オファー ID またはオポチュニティ ID — 関連付けられている取引の一意の識別子。Marketplace オファーが既に ACE オポチュニティにリンクされている場合、パートナーはオファー ID を提供するだけで済みます。リンクされたオポチュニティに収益AWSを自動的に属性付けます。
- 請求月 — コスト配分が適用される暦月 (例: 2026-04)。
- コスト配分 % — この請求月のこの取引に起因する製品の合計AWS消費量の一部。お客様がインフラストラクチャを共有するハイブリッドデプロイ用のパートナーホストコンポーネントなど、マルチテナント SaaS 製品に必要です。
- 顧客AWSアカウント ID — 製品を消費する顧客のAWSアカウント。マルチテナント SaaS 製品に必要です。

特定の収益属性 ID と同じ請求月のすべての配分エントリに対するコスト配分率の合計は、100% を超えることはできません。合計が 100% を超えると、レコードごとのエラーコードを使用して非同期ビジネス検証中に問題のあるエントリが拒否されます。

同期検証: は、基本的なシェイプ検証を同期的にStartRevenueAttributionAllocationsTask実行します (フィールドタイプ、パターン、バッチサイズ) 。基本検証に合格すると、API はステータス TaskIdの を返しますIN_PROGRESS。ビジネス検証 (上限チェック、イミュータビリティルール、 Marketplace と ACE に対する依存関係検索) は非同期的に実行され、レコードごとの結果は を通じて検出されますGetRevenueAttributionAllocationsTask。

送信されたタスクをモニタリングするために、パートナーは収益属性リソース ID または ARN GetRevenueAttributionAllocationsTaskを使用してポーリングします。レスポンスは から COMPLETED (個々のレコードが成功したか失敗したかに関係なく) IN_PROGRESSに進行し、以下を返します。

- **TaskStatus** — IN_PROGRESS、COMPLETED、または FAILED (最上位タスクの失敗) 。

- **RecordResults** — 各入力レコード: 割り当てられた AllocationId (成功した場合) 、レコードごとのステータス (SUCCEEDED または FAILED)、およびレコードが失敗した場合は構造化エラーコードとメッセージ。

パートナーはタスク結果をすぐに取得する必要があります。タスクが完了すると、レコードごとの結果を照合し、失敗したエントリを修正して新しいタスクで再送信できます。

毎月のコスト配分エントリの編集

コスト配分エントリは、請求月ごとに以下のルールで管理されます。

- パートナーは、現在または将来の請求月の新しい月次エントリをいつでも追加できます。
- パートナーは、現在または将来の請求月の毎月のエントリをいつでも更新できます。
- パートナーは、先月のエントリを当月の 7 日まで更新できます。 このウィンドウにより、パートナーは前月の割り当てを確定する前に、AWS Cost Explorer で実際の使用状況を確認できます。

当月 7 日以降は、前の請求月の属性を変更できなくなります。当月または将来の月のエントリの更新は、次の毎月の請求サイクルから適用されます。過去 1 か月の履歴属性は遡及的に再計算されません。

割り当てエントリの更新は、影響を受けるエントリUPDATEごとに を Operation に設定して、同じ StartRevenueAttributionAllocationsTask API アクションを介して送信されます。非同期タスクパターンは、更新と作成を均一に処理します。

収益属性 ID の更新

パートナーは UpdateRevenueAttribution API アクションを使用して、既存の収益属性 ID Descriptionの を更新できます。属性レコード自体は変更不可です。Name、MarketplaceProduct、および tag-on-create 値は作成後に変更できません。リソースに再度タグを付けるには、属性の ARN で標準のAWSタグ付けオペレーションを使用します。

収益属性 ID を更新する場合、パートナーは以下を提供する必要があります。

- **Catalog** — AWSまたは Sandbox。
- **Identifier** — 更新する属性の ra- ID。
- **Version** — オプティミスティックロックの属性の最新バージョン。

パートナーはオプションで以下を提供できます。

- **Description** — 更新されたフリーテキストの説明。最大1024文字
- **ClientToken** — 更新のべき等性トークン。

オプティミスティックロック: Versionフィールドは、属性が最後に取得されてから変更されていない場合にのみ更新が適用されるようにします。送信された Versionが現在のリソースバージョンと一致しない場合、更新は ConflictExceptionと、送信された と現在のバージョン番号を含むメッセージで拒否されます。ベストプラクティスは、各更新GetRevenueAttributionの前に で最新バージョンを取得することです。

収益属性 ID の詳細の表示

パートナーは GetRevenueAttribution API アクションを使用して、単一の収益属性 ID の完全な情報を取得できます。これにより、次が返されます。

リソースメタデータ:

- 一意の識別子 (Id) と Amazon リソースネーム ()Arn。
- 属性Catalogが存在する 。
- 「Name」および「Description」。
- 取得された Versionと LatestVersion (後続の更新には最新の を使用します) 。

Marketplace 製品属性 (作成時に MarketplaceProductが設定されている場合):

- ProductId — パートナーから提供された Marketplace 製品 ID。
- ProductCode — から解決されたAWS Marketplace 製品コードProductId。
- ProductType — SaaS、AMI、Container、ML、Data、または Professional Services。
- TenancyModel — MULTI_TENANTまたは SINGLE_TENANT。

監査情報:

- 「CreatedDate」および「LastModifiedDate」。

パートナーはオプションVersionで、属性の履歴バージョンを取得するリクエストに特定の を指定できます。を省略Versionして最新の を返します。

特定の月額コスト配分エントリを取得するには、パートナーはエントリの で GetRevenueAttributionAllocation API アクションを使用しますAllocationId。これにより、オファー ID またはオポチュニティ ID、請求月、コスト配分 %、顧客AWSアカウント ID、エントリのステータスと監査情報が返されます。

収益属性 IDs の一覧表示

パートナーは、 ListRevenueAttributions API アクションを使用して、アカウント内のすべての収益属性 IDs を表示できます。これにより、フィルタリングとソート機能を備えた属性の概要のページ分割されたリストが返されます。

パートナーは、次の方法で結果をフィルタリングできます。

- **Catalog** — AWSまたは Sandbox (必須) 。
- **Identifiers** — 取得する最大 100 個の特定の ra- ID のリスト。 IDs
- **CreatedAfter** / **CreatedBefore** — 作成タイムスタンプ範囲 (含む) でフィルタリングします。

パートナーは、 Sortパラメータを使用して結果のソートを設定できます。

- SortBy — CreatedDateまたは LastModifiedDate。
- SortOrder — ASCENDINGまたは DESCENDING。

レスポンスには、各属性の概要が含まれます: Arn、Id、Catalog、Name、解決されたMarketplaceProduct属性、CreatedDate、LastModifiedDate最新の Version、および TotalRevenueAttributionAssociationCount (属性にリンクされたアクティブな毎月のコスト配分エントリの数) 。

(MaxResultsデフォルトは 25、最大は 100) と NextToken を使用して、大きな結果セットをページ分割します。追加ページが利用可能なNextToken場合、レスポンスには が含まれます。

毎月のコスト配分エントリの一覧表示

パートナーは、 ListRevenueAttributionAllocations API アクションを使用して、特定の収益属性 ID の毎月のコスト配分エントリを一覧表示できます。これにより、フィルタリング機能を備えた割り当ての概要のページ分割されたリストが返されます。

パートナーは、次の方法で結果をフィルタリングできます。

- **MarketplaceOfferId** — 特定の Marketplace オファーに関連付けられたエントリのみを一覧表示します。
- **AwsPartnerCentralOpportunityId** — 特定の Partner Central オポチュニティに関連付けられたエントリのみを一覧表示します。
- **BillingMonth** — 特定の暦月のエントリのみを一覧表示します。

レスポンスには、各エントリの概要情報としてAllocationId、、関連付けられたオファー ID またはオポチュニティ ID、顧客AWSアカウント ID、請求月、コスト配分 %、エントリの名前とステータス、監査タイムスタンプが含まれます。

ダッシュボードの構築: ListRevenueAttributionsと の組み合わせListRevenueAttributionAllocationsは、パートナーダッシュボードの作成をサポートするように設計されています。パートナーは、次のようなビューを構築できます。

- 過去 30 日間に作成されたすべての収益属性 IDs、作成日でソートされます。
- 複数の収益属性 IDs にわたる特定の Marketplace オファーのすべての月額コスト配分エントリ。
- 現在の請求月のすべてのコスト配分エントリ。合計が属性ごとに 100% を超えないことを検証します。
- 少なくとも 1 つのアクティブな割り当てエントリ () を持つすべての収益属性 IDsTotalRevenueAttributionAssociationCount > 0。

Marketplace 収益共有の使用

Marketplace 収益共有は、AWS Marketplace を通じて収集された製品の総収益の一部を宣言する AWS Marketplace 製品の入力です。 はこの入力AWSを使用して、パートナーの Marketplace で収集された収益を他の収益シグナルと関連付けます。

Marketplace Revenue Share Service は、 Marketplace Revenue Share リソースを作成、取得、一覧表示し、その割り当て (パートナーが宣言した収益共有の割合と発効日ウィンドウ) を管理する APIs を公開します。標準のAWSタグ付けオペレーションは、 Marketplace Revenue Share リソースでサポートされています。

Marketplace 収益配分とは

Marketplace 収益共有には 2 つのレイヤーがあります。

- 共有 — Marketplace によって識別される単一の Marketplace 製品の入力AWSProductId。製品ごとに Marketplace 収益配分が 1 つだけあります。

- 1 つ以上の割り当て — 各割り当ては、 RevenueSharePercentと有効日ウィンドウ (EffectiveDate、オプションで) を宣言しますExpirationDate。共有は時間の経過とともに多くの割り当てを持つことができますが、いつでも適用できるACTIVE割り当ては 1 つだけです。

割り当てには、次のキープロパティがあります。

- ステータス — ACTIVEまたは INACTIVE。作成時、ステータスはデフォルトで になりますACTIVE。ACTIVE 配分のみが収益計算と重複チェックに参加します。
- 重複不変 — 1 つの共有内では、2 つのACTIVE割り当てが重複[EffectiveDate, ExpirationDate]範囲を持つことはできません。共有ごとにいつでも最大 1 つのオープンエンドACTIVE割り当て (なしExpirationDate) が許可されます。

Marketplace 収益共有の使用

パートナーは、 Marketplace Revenue Share Service を通じて Marketplace Revenue Shares とその割り当てを管理します。ライフサイクルは、共有フロー (作成、取得、リスト、タグ) と割り当てフロー (作成、更新、ソフト削除、取得、リスト) の 2 つのフローを通じて進行します。

Marketplace 収益共有の作成

最初のステップでは、 CreateMarketplaceRevenueShare API アクションを使用して Marketplace 製品のAWS Marketplace 収益共有を作成します。共有は Marketplace によって識別ProductIdされ、後続のすべての割り当てのコンテナとして機能します。

Marketplace 収益共有を作成する場合、パートナーは以下を提供する必要があります。

- **Catalog** — 本番AWS稼働用またはテストSandbox用。
- **ProductId** — 25 文字の Marketplace 製品 ID。パターン: [a-z0-9]{25}。

パートナーはオプションで以下を提供できます。

- **Tags** — 作成時にリソース組織に対して最大 50 個のキーと値のペア。を使用してTagResource、作成後にタグを追加します。
- **ClientToken** — 再試行時の重複作成を防ぐためのべき等性トークン。最大 64 文字。SDKsトークンを自動生成します。

製品検証: 作成時に、サービスは、製品が発信者のアカウント、または Partner Account Connection (PAC) を介して発信者に接続された子会社アカウントによって所有されていることをAWS Marketplace で検証します。製品が存在しないか、所有アカウントが発信者に対して承認されていない場合、API は理由 ValidationExceptionで を返しますPRODUCT_NOT_FOUND_OR_NOT_OWNED。

製品ごとに 1 つの共有: 同じ ProductId の 2 回目のCreateMarketplaceRevenueShare呼び出しは Catalogを返しますConflictException。

レスポンスは、共有 Arn (形式 arn:{partition}:partnercentral:{region}:{account-id}:catalog/{catalog}/marketplace-revenue-share/{productId})、Catalog、、ProductId初期Version数 (各割り当てミューテーションで 1 から始まり、増分) 、および作成時に適用されるタグを返します。

割り当ての追加

Marketplace 収益共有が作成されると、パートナーは CreateMarketplaceRevenueShareAllocation API アクションを使用して配分を作成して収益共有の割合と有効日ウィンドウを宣言します。

割り当てを作成する場合、パートナーは以下を提供する必要があります。

- **Catalog** — AWSまたは Sandbox。
- **MarketplaceRevenueShareIdentifier** — 親共有の識別子。
- **RevenueSharePercent** — 10 進数で提供される Marketplace を通じて収集された製品の収益の割合 (例: 0.45 45%)。
- **EffectiveDate** — この割り当てが有効になるカレンダーの日付。形式: YYYY-MM-DD。

レスポンスは、割り当ての Id (形式 mrsa-[A-Za-z0-9]{13})、割り当てステータス (ACTIVE作成時) 、および親共有のバンプ Version (割り当てミューテーションごとに 1 つのバージョン整数がバンプされ、その後の割り当て更新で楽観的ロックに使用されます) を返します。

割り当ての更新

パートナーは API UpdateMarketplaceRevenueShareAllocationアクションを使用して割り当てを更新できます。更新可能なフィールドは、割り当てが将来の日付であるか、現在有効であるか、過去の日付であるかによって異なります。

割り当てを更新する場合、パートナーは以下を提供する必要があります。

- **Catalog** — AWSまたは Sandbox。
- **MarketplaceRevenueShareIdentifier** — 親共有の識別子。
- **AllocationId** — 更新する割り当ての mrsa- ID。
- **MarketplaceRevenueShareVersion** — 割り当てを楽観的に口ックするための親共有の最新バージョン。バンプ競合は を返しますConflictException。

パートナーはオプションで以下を提供できます。

- **RevenueSharePercent** — 更新された収益共有の割合。将来の日付の割り当てでのみ編集できます。
- **EffectiveDate** — 更新された発効日。将来の日付の割り当てでのみ編集できます。
- **ExpirationDate** — 更新された有効期限。将来の日付の割り当て (任意の値 > EffectiveDate) と、現在有効またはオープンエンドの割り当て (任意の値) で編集できます≥ today。
- **Status** — ACTIVEまたは INACTIVE。を に設定すると、割り当てがINACTIVEソフトに削除されます。将来の日付の割り当てでのみ編集できます。
- **ClientToken** — べき等性トークン。

Marketplace 収益共有の詳細の表示

パートナーは GetMarketplaceRevenueShare API アクションを使用して、単一の Marketplace 収益共有の完全な情報を取得できます。これにより、共有の Arn、CatalogProductId、、Version (親共有の最新バージョン)CreatedDate、LastModifiedDate、、および共有に適用されるタグが返されます。

パートナーは、割り当ての で GetMarketplaceRevenueShareAllocation API アクションを使用して 1 つの割り当てを取得できますId。これにより、次が返されます。

- 割り当ての Idと親共有の MarketplaceRevenueShareArn。
- RevenueSharePercent, EffectiveDate, ExpirationDate.
- Status (ACTIVE または INACTIVE)。
- 後続の更新で楽観的に口ックMarketplaceRevenueShareVersionするための親共有の 。
- 「CreatedDate」および「LastModifiedDate」。

Marketplace 収益共有の一覧表示

パートナーは、 ListMarketplaceRevenueShares API アクションを使用して、自分のアカウントが所有するすべての Marketplace 収益共有を表示できます。これにより、共有の概要のページ分割されたリストが返されます。

パートナーは、次の方法で結果をフィルタリングできます。

- **Catalog** — AWSまたは Sandbox (必須) 。
- **ProductIds** — 取得する特定の Marketplace 製品 IDsのリスト。

レスポンスには、各共有の概要として、Arn、Catalog、ProductId、最新の Version、CreatedDate、および が含まれますLastModifiedDate。

MaxResults と を使用してNextToken、大きな結果セットをページ分割します。

割り当ての一覧表示

パートナーは、 ListMarketplaceRevenueShareAllocations API アクションを使用して、特定の Marketplace 収益共有の割り当てを一覧表示できます。これにより、フィルタリング機能を備えた割り当ての概要のページ分割されたリストが返されます。

パートナーは、次の方法で結果をフィルタリングできます。

- **Status** — ACTIVEまたは INACTIVE。
- 有効日付範囲 — EffectiveDateOnOrAfter / EffectiveDateOnOrBefore 有効日が特定のウィンドウ内にある割り当てを取得します。

レスポンスには、各割り当ての概要情報、Id、、RevenueSharePercentEffectiveDate、ExpirationDateStatusMarketplaceRevenue および監査タイムスタンプが含まれます。

Marketplace 収益共有のタグ付け

Marketplace Revenue Share リソースでは、標準のAWSタグ付けオペレーションがサポートされています。

- **TagResource** — Marketplace 収益共有にタグを追加または上書きします。共有あたり最大 50 タグ。

- **`UntagResource`** — キーで特定のタグを削除します。
- **`ListTagsForResource`** — Marketplace 収益共有に現在適用されているすべてのタグを取得します。

割り当てはタグをサポートしていません。親共有にタグを付けて、関連する割り当てを整理します。

各タグ付けオペレーションは、共有の `Arn` (形式 `arn:{partition}:partnercentral:{region}:{account-id}:catalog/{catalog}/marketplace-revenue-share/{productId}`) をリソース識別子として受け入れます。Standard AWS tag-on-create は、 の `Tags`フィールドでもサポートされていますCreateMarketplaceRevenueShare。

サポートされている AWS リージョン

以下のセクションでは、サポートされている AWS リージョンについて説明します。

トピック

- AWS Partner Central Account API でサポートされている AWS リージョン
- AWS Partner Central Selling API でサポートされている AWS リージョン
- AWS Partner Central Benefits API でサポートされている AWS リージョン
- AWS Partner Central Channel API でサポートされている AWS リージョン
- AWS Partner Central Revenue Measurement API でサポートされている AWS リージョン

AWS Partner Central Account API でサポートされている AWS リージョン

AWS Partner Central アカウントサービスには、次のエンドポイントを使用して、任意の AWS 米国東部 (バージニア北部) リージョンからアクセスできます。

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

AWS Partner Central Selling API でサポートされている AWS リージョン

AWS Partner Central 販売サービスには、次のエンドポイントを使用して、任意の AWS 米国東部 (バージニア北部) リージョンからアクセスできます。

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

AWS Partner Central Benefits API でサポートされている AWS リージョン

AWS Partner Central の特典サービスは、次のエンドポイントを使用して、任意の AWS 米国東部 (バージニア北部) リージョンからアクセスできます。

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

AWS Partner Central Channel API でサポートされている AWS リージョン

AWS Partner Central チャネル管理サービスには、次のエンドポイントを使用して、任意の AWS 商用リージョンからアクセスできます。

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

SigV4a を使用したリクエストの署名

AWS Partner Central Channel API はグローバルエンドポイントを使用するため、リクエストは署名バージョン 4a (SigV4a) で署名されます。これは、非対称暗号化による署名をサポートする SigV4 の拡張機能であり、複数の AWS リージョンで検証可能な署名を有効にします。

AWS SDK または CLI の使用

AWS SDK または CLI AWS を使用する場合、グローバルエンドポイントを呼び出すと SigV4a 署名が自動的に処理されます。追加の設定は必要ありません。

リクエストを手動で署名する

SDK を使用せずに手動でリクエストに署名する場合は、標準の SigV4 と以下の違いに注意してください。

- SigV4a は、HMAC ベースのキー取得を使用するのではなく、既存の AWS シークレットアクセスキーから楕円曲線デジタル署名アルゴリズム (ECDSA) キーペアを取得します。
- 認証情報スコープは、署名がリージョン間で有効であるため、特定のリージョン名ではなくリージョンコンポーネント*に を使用します。

SigV4a の仕組みの詳細については、「API リクエストの AWS 署名バージョン 4」を参照してください。SigV4a 署名のコード例については、GitHub の sigv4a-signing-examples プロジェクトを参照してください。

AWS Partner Central Revenue Measurement API でサポートされている AWS リージョン

AWS Partner Central Revenue Measurement サービスには、次のエンドポイントを使用して、任意の AWS 商用リージョンからアクセスできます。

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

SigV4a を使用したリクエストの署名

収益測定サービスはグローバルエンドポイントを使用するため、リクエストは署名バージョン 4a (SigV4a) で署名されます。これは、非対称暗号化による署名をサポートする SigV4 の拡張機能であり、複数の AWS リージョンで検証可能な署名を有効にします。

AWS SDK または CLI の使用

AWS SDK または CLI AWS を使用する場合、グローバルエンドポイントを呼び出すと SigV4a 署名が自動的に処理されます。追加の設定は必要ありません。

リクエストを手動で署名する

SDK を使用せずに手動でリクエストに署名する場合は、標準の SigV4 と以下の違いに注意してください。

- SigV4a は、HMAC ベースのキー取得を使用するのではなく、既存の AWS シークレットアクセスキーから楕円曲線デジタル署名アルゴリズム (ECDSA) キーペアを取得します。
- 認証情報スコープは、署名がリージョン間で有効であるため、特定のリージョン名ではなくリージョンコンポーネント*に を使用します。

SigV4a の仕組みの詳細については、「API リクエストの AWS 署名バージョン 4」を参照してください。SigV4a 署名のコード例については、GitHub の sigv4a-signing-examples プロジェクトを参照してください。

アクセス許可

以下のセクションでは、 アクセス許可について説明します。

トピック

アカウント API へのアクセス

アクセスコントロールとアクセス許可はAWS Identity and Access Management(IAM) によって管理されます。このセクションでは、 アカウント API とやり取りするために必要なアクセス許可を設定するためのガイダンスを提供します。

前提条件

アクセス許可を設定する前に、AWS アカウントが にリンクされ、必要な IAM ロールとユーザーが作成されていることを確認します。詳細については、「セットアップと認証」を参照してください。

AWS管理ポリシーの使用

AWSには、 アカウント API を操作するために必要なアクセス許可を付与する マネージドポリシーが用意されています。アカウントリソースを管理するために必要なアクセスを提供するには、`AWSPartnerCentralFullAccess`ポリシーを IAM ID にアタッチします。詳細については、AWS「 ユーザーの 管理ポリシー」を参照してください。

IAM ロールとユーザーにポリシーを割り当てる

IAM ロールとユーザーにポリシーを割り当てるには、次の手順に従います。

1. AWS Management Consoleにサインインします。
2. IAM サービスに移動します。
3. ロールまたはユーザーを選択し、ポリシーをアタッチする IAM ロールまたはユーザーを選択します。
4. 選択した IAM ロールまたはユーザーに`AWSPartnerCentralFullAccess`ポリシーをアタッチします。

詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

条件キーを使用したアクセス許可の管理

IAM ポリシーの条件キーは、ステートメントポリシーを適用するタイミングに関するリソースレベルのアクセス許可を提供します。条件キーを使用して、特定のアクセス許可がいつ許可または拒否されるかを決定する条件を指定できます。

詳細については、「IAM JSON ポリシー要素: 条件演算子」を参照してください。

条件キーの概要

条件キー	説明	適用可能なアクション	有効値
partnercentral:カタログ	は、関連付けられたカタログエンティティのタイプでアクセスをフィルタリングします	すべてのアクション	AWS、サンドボックス

必要なアクセス許可の概要

必要なアクセス許可の概要

[アクション]	説明
partnercentral:AcceptConnectionInvitation	は接続の招待の受け入れを許可します
partnercentral:AssociateAwsTrainingCertificationEmailDomain	トレーニングAWS証明書の E メールドメインの関連付けを許可
partnercentral:CancelConnection	は接続のキャンセルを許可します
partnercentral:CancelConnectionInvitation	では、接続の招待をキャンセルできます。
partnercentral:CancelProfileUpdateTask	プロファイル更新タスクのキャンセルを許可
partnercentral:CreateConnectionInvitation	接続の招待の作成を許可
partnercentral:CreatePartner	でパートナーの作成を許可

[アクション]	説明
partnercentral:DisassociateAwsTrainingCertificationEmailDomain	AWSトレーニング証明書 E メールドメインの関連付け解除を許可
partnercentral:GetAllianceLeadContact	では、提携リードの連絡先の詳細を取得できます
partnercentral:GetConnection	接続の詳細の取得を許可
partnercentral:GetConnectionInvitation	接続招待の詳細の取得を許可
partnercentral:GetConnectionPreferences	接続設定の取得を許可
partnercentral:GetPartner	パートナーの詳細の取得を許可
partnercentral:GetProfileUpdateTask	プロファイル更新タスクの詳細の取得を許可
partnercentral:GetProfileVisibility	プロファイルの可視性設定の取得を許可
partnercentral:GetVerification	検証の詳細の取得を許可
partnercentral:ListConnectionInvitations	接続の招待の一覧表示を許可
partnercentral:ListConnections	接続の一覧表示を許可
partnercentral:ListPartners	が出品パートナーを許可
partnercentral:PutAllianceLeadContact	が提携リードの連絡先の詳細の更新を許可
partnercentral:PutProfileVisibility	プロファイルの可視性設定の更新を許可
partnercentral:RejectConnectionInvitation	接続の招待の拒否を許可する
partnercentral:SendEmailVerificationCode	は E メール検証コードの送信を許可します
partnercentral:StartProfileUpdateTask	プロファイル更新タスクの開始を許可
partnercentral:StartVerification	は検証プロセスの開始を許可します
partnercentral:UpdateConnectionPreferences	接続設定の更新を許可

販売 API へのアクセス

アクセスコントロールとアクセス許可はAWS Identity and Access Management(IAM) によって管理されます。このセクションでは、AWS Marketplaceエンティティを一覧表示するために必要なアクセス許可など、API とやり取りするために必要なアクセス許可を設定するためのガイダンスを提供します。

前提条件

アクセス許可を設定する前に、AWS アカウントが Partner Central にリンクされ、必要な IAM ロールとユーザーが作成されていることを確認します。詳細については、「セットアップと認証」を参照してください。

AWS管理ポリシーの使用

AWSは、 API とやり取りするために必要なアクセス許可を付与する マネージドポリシーを提供します。機会を管理するために必要なアクセスを提供するには、`AWSPartnerCentralOpportunityManagement`ポリシーを IAM ID にアタッチします。詳細については、AWSAWS「 Partner Central ユーザーの マネージドポリシー」を参照してください。

AWSPartnerCentralOpportunityManagement ポリシー

このポリシーは、Partner Central オポチュニティ管理アクションへのフルアクセスを許可します。

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/*"
            ]
        }
    ]
```

```
}
```

カスタムポリシー

マネージドポリシーがニーズに合わない場合は、ユースケースに必要なアクセス許可を付与するカスタム IAM ポリシーを作成します。次の例は、AWS Marketplaceエンティティを一覧表示するアクセス許可を付与するカスタムポリシーです。

Exampleカスタムポリシーの例

JSON

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

カスタム許可ポリシー

このポリシーは、ポリシーの更新を必要とせずに将来追加される可能性のある機能など、Partner Central の販売アクションへの広範なアクセスを提供します。ワイルドカードアクション を使用することで`partnercentral:*`、このポリシーは新しい Partner Central 販売機能が利用可能になると自動的にアクセスを許可し、手動更新の必要性を減らします。このポリシーには、AWS Marketplace工ンティティとやり取りするためのアクセス許可も含まれています。これにより、販売アクションと Marketplace アクションの両方でアクセスが維持されます。

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"
                }
            }
        }
    ]
}
```

IAM ロールとユーザーにポリシーを割り当てる

IAM ロールとユーザーにポリシーを割り当てるには、次の手順に従います。

1. AWS Management Consoleにサインインします。
2. IAM サービスに移動します。
3. ロールまたはユーザーを選択し、ポリシーをアタッチする IAM ロールまたはユーザーを選択します。
4. 選択した IAM ロールまたはユーザーに`AWSPartnerCentralOpportunityManagement`ポリシーまたはカスタムポリシーをアタッチします。

詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

条件キーを使用したアクセス許可の管理

IAM ポリシーの条件キーは、ステートメントポリシーを適用するタイミングに関するリソースレベルのアクセス許可を提供します。条件キーを使用して、特定のアクセス許可がいつ許可または拒否されるかを決定する条件を指定できます。

詳細については、「IAM JSON ポリシー要素: 条件演算子」を参照してください。

条件キーの概要

条件キー	説明	適用可能なアクション	有効値
partnercentral:Catalog	は、関連付けられたカタログエンティティのタイプによってアクセスをフィルタリングします	すべてのアクション	AWS、サンドボックス
aws-marketplace:PartyType	は、パーティのタイプ (提案者など) に基づいてアクセスをフィルタリングします	SearchAgreements、DescribeAgreement	提案者

必要なアクセス許可の概要

必要なアクセス許可の概要

[アクション]	説明
partnercentral: CreateOpportunity	はオポチュニティの作成を許可します
partnercentral:UpdateOpportunity	がオポチュニティの更新を許可
partnercentral:ListOpportunities	が出品の機会を許可
partnercentral:GetOpportunity	では、オポチュニティの詳細を取得できます。
partnercentral:ListSolutions	がソリューションの一覧表示を許可
partnercentral:AssociateOpportunity	は、機会を他のエンティティに関連付けることができます
partnercentral: DisassociateOpportunity	では、他のエンティティから機会の関連付けを解除できます。
partnercentral:AcceptEngagementInvitation	はエンゲージメントの招待の承諾を許可します
partnercentral:RejectEngagementInvitation	はエンゲージメントの招待を拒否できます
partnercentral:GetEngagementInvitation	では、エンゲージメントの招待の詳細を取得できます。
partnercentral:ListEngagementInvitations	では、エンゲージメントの招待を一覧表示できます。
partnercentral:SubmitOpportunity	がオポチュニティの送信を許可
partnercentral:GetAwsOpportunitySummary	では、AWSオポチュニティの概要を取得できます
partnercentral:StartProspectingFromEngagementTask	エンゲージメントからプロスペクティングタスクを作成します

[アクション]	説明
partnercentral:GetProspectingFromEngagementTask	プロスペクティングタスクの詳細を返します
partnercentral:ListProspectingFromEngagementTasks	プロスペクティングタスクのリストを返します
aws-marketplace:ListEntities	AWS Marketplaceエンティティの一覧表示を許可
aws-marketplace:DescribeEntity	AWS Marketplaceエンティティの記述を許可
aws-marketplace:SearchAgreements	では、 で契約を検索できます。AWS Marketplace
aws-marketplace:DescribeAgreement	では、 での契約の説明を許可します。AWS Marketplace

Benefits API へのアクセス

アクセスコントロールとアクセス許可はAWS Identity and Access Management(IAM) によって管理されます。このセクションでは、 Benefits API を操作するために必要なアクセス許可を設定するためのガイダンスを提供します。

前提条件

アクセス許可を設定する前に、AWS アカウントが にリンクされ、必要な IAM ロールとユーザーが作成されていることを確認します。詳細については、「セットアップと認証」を参照してください。

AWS管理ポリシーの使用

AWSには、 Benefits API を操作するために必要なアクセス許可を付与する マネージドポリシーが用意されています。特典リソースを管理するために必要なアクセスを提供するには、`AWSPartnerCentralFullAccess`ポリシーを IAM ID にアタッチします。詳細については、AWS「 ユーザーの 管理ポリシー」を参照してください。

IAM ロールとユーザーにポリシーを割り当てる

IAM ロールとユーザーにポリシーを割り当てるには、次の手順に従います。

1. AWS Management Consoleにサインインします。
2. IAM サービスに移動します。
3. ロールまたはユーザーを選択し、ポリシーをアタッチする IAM ロールまたはユーザーを選択します。
4. 選択した IAM ロールまたはユーザーにAWSPartnerCentralFullAccessポリシーをアタッチします。

詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

条件キーを使用したアクセス許可の管理

IAM ポリシーの条件キーは、ステートメントポリシーを適用するタイミングに関するリソースレベルのアクセス許可を提供します。条件キーを使用して、特定のアクセス許可がいつ許可または拒否されるかを決定する条件を指定できます。

詳細については、「IAM JSON ポリシー要素: 条件演算子」を参照してください。

条件キーの概要

条件キー	説明	適用可能なアクション	有効値
partnercentral:カタログ	は、関連付けられたカタログエンティティのタイプによってアクセスをフィルタリングします	すべてのアクション	AWS、サンドボックス

必要なアクセス許可の概要

必要なアクセス許可の概要

[アクション]	説明
partnercentral:AmendBenefitApplication	では、メリットアプリケーションを修正できます。

[アクション]	説明
partnercentral:AssociateBenefitApplicationResource	では、リソースをメリットアプリケーションに関連付けることができます
partnercentral:CancelBenefitApplication	で特典アプリケーションのキャンセルを許可する
partnercentral:CreateBenefitApplication	でメリットアプリケーションを作成可能に
partnercentral:DisassociateBenefitApplicationResource	では、メリットアプリケーションからリソースの関連付けを解除できます
partnercentral:GetBenefit	では、メリットの詳細を取得できます。
partnercentral:GetBenefitAllocation	では、メリット配分の詳細を取得できます。
partnercentral:GetBenefitApplication	では、メリットアプリケーションの詳細を取得できます
partnercentral:ListBenefitAllocations	でメリット配分の一覧表示を許可
partnercentral:ListBenefitApplications	でメリットアプリケーションの出品を許可
partnercentral:ListBenefits	では、利点を一覧表示できます。
partnercentral:RecallBenefitApplication	では、メリットアプリケーションを再現できます。
partnercentral:SubmitBenefitApplication	はメリットアプリケーションの送信を許可します
partnercentral:UpdateBenefitApplication	メリットアプリケーションの更新を許可

チャネル API へのアクセス

アクセスコントロールとアクセス許可はAWS Identity and Access Management(IAM) によって管理されます。このセクションでは、AWS Partner Central Channel API とやり取りするために必要なアクセス許可を設定するためのガイダンスを提供します。

チャネル管理アカウントの設定

AWS Partner Central のチャネル管理機能では、IAM ロールを次の 2 種類のAWSアカウントにデプロイする必要があります。

1. AWS Partner Central リンクAWS アカウント:

   これは、AWS Partner Network および Partner Central アカウントAWS アカウントに関連付けられている です。パートナーAWS アカウントごとにリンクされたAWS Partner Central は 1 つだけです。ワンタイムセットアップアクティビティとして、Partner Central チャネル管理ポリシーAWS アカウントを使用して、この IAM ロールを作成します。
2. プログラム管理アカウントAWS アカウント:

   これは、エンドユーザーの消費に対する請求と支払いを管理するための請求書転送アカウントとしてAWS アカウント使用される です。これは Partner AWS Central チャネル管理のプログラム管理アカウントとして報告AWS アカウントされます。複数のプログラム管理アカウントがあり、プログラム管理アカウントとしてAWS アカウント報告する各 に IAM ロールを作成する必要があります。

AWS アカウントタイプごとに、異なる IAM ロールを特定のアクセス許可と信頼ポリシーに関連付ける必要があります。

リンクされたAWS Partner Central へのアクセスAWS アカウント

AWS Partner Central リンク内でAWS アカウント、チャネルリソースを管理する IAM ロールを作成します。これは Partner Central ユーザーに付与されます。この IAM ロールにより、ユーザーはAWS Partner Central でチャネル管理リソースを表示および管理できます。ロール名は で始まる必要がありPartnerCentralRoleFor、推奨名は ですPartnerCentralRoleForChannel。ロールを設定するには、次の手順を実行します。

- 次のカスタム信頼ポリシーを追加して、AWS Partner Central クラウド管理者と提携リードが IAM ロールを Partner Central ユーザーにマッピングできるようにします。

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

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- AWSPartnerCentralChannelManagement 管理ポリシーをPartnerCentralRoleForChannelロールに関連付けます。

プログラム管理アカウントAWSアカウントのアクセス (複数可)

AWS Partner Central のプログラム管理アカウントとしてAWS アカウント報告された各 内で、以下の IAM ロールを作成し、必要な名前を付けます。

- PartnerCentralChannelHandshakeApprovalManagement: このロールを使用して、 Partner Central とプログラム管理の間のハンドシェイクを管理しますAWS アカウント。この IAM ロールは、AWS Partner Central でプログラム管理アカウントをアクティブ化するために応答するために使用できます。このロールを作成するときは、AWSPartnerCentralChannelHandshakeApprovalManagement管理ポリシーを関連付けます。
- PartnerCentralChannelBillingTransferReadOnly: このクロスアカウントロールを使用して、AWS Partner Central からの請求転送ステータスを可視化します。このロールは、 からAWS Partner Central AWS アカウントへの請求転送ステータスを共有し、請求転送ステータスを一元的に表示します。このロールが作成されると、AWS Partner Central のユーザーはAWS Partner Central で請求転送ステータスを表示できます。ただし、このロールは、AWS請求情報とコスト管理コンソールで請求転送にアクセスするアクセス許可をAWS Partner Central ユーザーに付与しません。
  - 信頼ポリシー:
    - 以下の JSON 内で、ロールの作成時に {PartnerCentralLinkedAccountID}を実際の AWS Partner Central linked AWS アカウント 12 桁の ID に置き換えます。

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

```
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- アクセス許可ポリシー:

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

- `PartnerCentralChannelBillingTransferManagement` (オプション): このクロスアカウントロールを使用して、AWS Partner Central からの請求転送の作成と管理を有効にします。このロールにより、AWS Partner Central ユーザーは `AWSPartnerCentralChannelManagement`マネージドポリシーを含む IAM ロールにアクセスして、プログラム管理アカウントの請求転送を管理できるようになりますAWS アカウント。このクロスアカウントロールを使用すると、AWS Partner Central ユーザーは Partner AWS Central チャネル管理の「請求の管理」ボタンを使用してAWS、請求情報とコスト管理コンソールでAWS請求転送にアクセスして管理できます。
  - 信頼ポリシー:
    - 以下の JSON 内で、ロールの作成時に `{PartnerCentralLinkedAccountID}`を実際の AWS Partner Central linked AWS アカウント 12 桁の ID に置き換えます。

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

```
        "AWS": "arn:aws:iam::{PartnerCentralLinkedAccountId}:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- アクセス許可ポリシー:

```
{
  "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": "*"
    }
  ]
}
```

AWS管理ポリシーの使用

AWSには、チャネル API とやり取りするために必要なアクセス許可を付与する マネージドポリシーが用意されています。すべてのチャネルリソースを管理するために必要なアクセスを提供するには、AWSPartnerCentralChannelManagementポリシーを IAM ID にアタッチします。チャネルハンドシェイクリクエストを表示して応答するために必要なアクセスを提供するに

は、AWSPartnerCentralChannelHandshakeApprovalManagementポリシーを IAM ID にアタッチします。詳細については、AWSAWS「 Partner Central ユーザーの マネージドポリシー」を参照してください。

AWSPartnerCentralChannelManagement ポリシー

このポリシーは、Partner Central チャネル管理アクションへのフルアクセスを許可します。

```
{
  "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 ポリシー

このポリシーは、チャネルハンドシェイクリクエストを表示して応答するためのアクセスを許可します。このポリシーは、チャネルハンドシェイクリクエストを受信するAWSアカウントのロールに適用する必要があります。

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

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

IAM ロールとユーザーにポリシーを割り当てる

IAM ロールとユーザーにポリシーを割り当てるには、次の手順に従います。

1. AWS Management Consoleにサインインします。
2. IAM サービスに移動します。
3. ロールまたはユーザーを選択し、ポリシーをアタッチする IAM ロールまたはユーザーを選択します。
4. 選択した IAM ロールまたはユーザーに`AWSPartnerCentralChannelManagement`ポリシーまたはカスタムポリシーをアタッチします。

詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

条件キーを使用したアクセス許可の管理

IAM ポリシーの条件キーは、ステートメントポリシーを適用するタイミングに関するリソースレベルのアクセス許可を提供します。条件キーを使用して、特定のアクセス許可がいつ許可または拒否されるかを決定する条件を指定できます。

詳細については、「IAM JSON ポリシー要素: 条件演算子」を参照してください。

条件キーの概要

条件キー	説明	適用可能なアクション	有効値
partnercentral:カタログ	関連付けられたカタログエンティティのタイプでアクセスをフィルタリングします	すべてのチャネル管理アクション	AWS、サンドボックス
partnercentral:ChannelHandshakeType	チャネルハンドシェイクのタイプに基づいてアクセスをフィルタリングします	CreateChannelHandshake、AcceptChannelHandshake、RejectChannelHandshake、CancelChannelHandshake、ListChannelHandshakes	PROGRAM_MANAGEMENT_ACCOUNT、START_SERVICE_PERIOD、REVOKE_SERVICE_PERIOD

必要なアクセス許可の概要

必要なアクセス許可の概要

[アクション]	説明
partnercentral:CreateProgramManagementAccount	はプログラム管理アカウントの作成を許可します
partnercentral:UpdateProgramManagementAccount	はプログラム管理アカウントの更新を許可します
partnercentral:DeleteProgramManagementAccount	プログラム管理アカウントの削除を許可する
partnercentral:ListProgramManagementAccounts	プログラム管理アカウントの一覧表示を許可

[アクション]	説明
partnercentral:GetProgramManagementAccount	では、プログラム管理アカウントの詳細を取得できます。
partnercentral:CreateRelationship	関係の作成を許可
partnercentral:UpdateRelationship	関係の更新を許可
partnercentral:DeleteRelationship	はリレーションシップの削除を許可します
partnercentral:GetRelationship	では、関係の詳細を取得できます。
partnercentral:ListRelationships	リレーションシップの一覧表示を許可
partnercentral:CreateChannelHandshake	チャネルハンドシェイクの作成を許可
partnercentral:AcceptChannelHandshake	はチャネルハンドシェイクの受け入れを許可します
partnercentral:RejectChannelHandshake	はチャネルハンドシェイクを拒否できます
partnercentral:CancelChannelHandshake	はチャネルハンドシェイクのキャンセルを許可します
partnercentral:ListChannelHandshakes	チャネルハンドシェイクの一覧表示を許可

収益測定 API へのアクセス

アクセスコントロールとアクセス許可はAWS Identity and Access Management(IAM) によって管理されます。このセクションでは、AWS Marketplaceエンティティを一覧表示するために必要なアクセス許可など、収益測定 API とやり取りするために必要なアクセス許可を設定するためのガイダンスを提供します。

前提条件

アクセス許可を設定する前に、AWS アカウントが Partner Central にリンクされ、必要な IAM ロールとユーザーが作成されていることを確認します。詳細については、「セットアップと認証」を参照してください。

AWS管理ポリシーの使用

AWSには、収益測定 API を操作するために必要なアクセス許可を付与する マネージドポリシーが用意されています。ペルソナに応じて 2 つの管理ポリシーを使用できます。

- パートナー — AWSPartnerCentralRevenueAttributionManagementポリシーを IAM ID にアタッチします。
- 顧客 — AWSRevenueAttributionManagementポリシーを IAM ID にアタッチします。

詳細については、AWSAWS「 Partner Central ユーザーの マネージドポリシー」を参照してください。

AWSPartnerCentralRevenueAttributionManagement ポリシー

このポリシーは、AWS Partner Central に登録されたAWSアカウントの収益帰属管理アクティビティに必要なアクセスを提供します。

```
{
    "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 ポリシー

このポリシーは、AWS Partner Central に登録されていないAWSアカウントの収益属性管理アクティビティに必要なアクセスを提供します。

```
{
    "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"
                    ]
                }
            }
```

```
        }
    ]
}
```

IAM ロールとユーザーにポリシーを割り当てる

IAM ロールとユーザーにポリシーを割り当てるには、次の手順に従います。

1. AWS Management Consoleにサインインします。
2. IAM サービスに移動します。
3. ロールまたはユーザーを選択し、ポリシーをアタッチする IAM ロールまたはユーザーを選択します。
4. 選択した IAM ロールまたはユーザーに`AWSPartnerCentralRevenueAttributionManagement`ポリシー (パートナー向け) または`AWSRevenueAttributionManagement`ポリシー (顧客向け) をアタッチします。

詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

条件キーを使用したアクセス許可の管理

IAM ポリシーの条件キーは、ステートメントポリシーを適用するタイミングに関するリソースレベルのアクセス許可を提供します。条件キーを使用して、特定のアクセス許可がいつ許可または拒否されるかを決定する条件を指定できます。

詳細については、「IAM JSON ポリシー要素: 条件演算子」を参照してください。

条件キーの概要

条件キー	説明	適用可能なアクション	有効値
partnercentral:カタログ	関連付けられたカタログエンティティのタイプでアクセスをフィルタリングします	すべてのアクション	AWS、サンドボックス

必要なアクセス許可の概要

必要なアクセス許可の概要

[アクション]	説明
partnercentral:CreateRevenueAttribution	新しい収益属性レコードの作成を許可する
partnercentral:UpdateRevenueAttribution	既存の収益属性レコードの更新を許可する
partnercentral:GetRevenueAttribution	特定の収益属性の詳細の取得を許可する
partnercentral:ListRevenueAttributions	オプションのフィルターを使用した収益属性の一覧表示を許可する
partnercentral:GetRevenueAttributionAllocation	ID による単一の割り当ての取得を許可する
partnercentral:ListRevenueAttributionAllocations	フィルタリングサポートによるコミットされた割り当ての一覧表示を許可する
partnercentral:StartRevenueAttributionAllocationsTask	非同期処理のために最大 250 個の割り当て変更のバッチの送信を許可します
partnercentral:GetRevenueAttributionAllocationsTask	以前に送信された割り当てタスクのステータスの取得を許可します。
partnercentral:CreateMarketplaceRevenueShare	新しいマーケットプレイス収益共有リソースの作成を許可する
partnercentral:GetMarketplaceRevenueShare	特定のマーケットプレイス収益配分の詳細の取得を許可する
partnercentral:ListMarketplaceRevenueShares	オプションのフィルターを使用したマーケットプレイス収益共有の一覧表示を許可する
partnercentral:CreateMarketplaceRevenueShareAllocation	新しいマーケットプレイス収益配分の作成を許可する
partnercentral:GetMarketplaceRevenueShareAllocation	特定のマーケットプレイス収益配分の詳細の取得を許可する

[アクション]	説明
partnercentral:UpdateMarketplaceRevenueShareAllocation	既存のマーケットプレイス収益配分の更新を許可する
partnercentral:ListMarketplaceRevenueShareAllocations	マーケットプレイス収益配分での配分の一覧表示を許可する
partnercentral:TagResource	リソースのタグの追加または上書きを許可する
partnercentral:UntagResource	リソースからのタグの削除を許可する
partnercentral:ListTagsForResource	リソースに関連付けられたタグの一覧表示を許可する

AWS Partner Central API のクォータ

以下のセクションでは、サービスクォータについて説明します。

トピック

- AWS Partner Central Account API のクォータ
- AWS Partner Central Selling API のクォータ
- AWS Partner Central Benefits API のクォータ
- AWS Partner Central Channel API のクォータ
- AWS Partner Central Revenue Measurement API のクォータ

AWS Partner Central Account API のクォータ

AWS Partner Central Account API には次のクォータがあります。

追加のクォータ

追加のクォータ

Display name (表示名)	カタログ	説明	デフォルトの値
アカウントあたりのオープン接続の招待	AWS	AWSカタログ内のパートナーアカウントで維持できるオープン接続の招待の最大数	1,000
アカウントあたりのアクティブな接続数	AWS	AWSカタログ内のパートナーアカウントで維持できるアクティブな接続の最大数	1,000
パートナーあたりの E メールドメイン	AWS	AWSカタログのAWSトレーニング認定のためにパートナーアカウントに関連付けることができる E メールドメインの最大数	50
アカウントあたりの接続招待のレート	AWS	AWSカタログで送信できる 1 日あたりの接続招待の最大数	50
アカウントあたりのプロファイル更新タスクのレート	AWS	AWSカタログで開始できる 1 日あたりのプロファイル更新タスクの最大数	10
アカウントあたりのプロファイル可視性更新のレート	AWS	AWSカタログで開始できる 1 日あたりの	5

Display name (表示名)	カタログ	説明	デフォルトの値
		プロファイル可視性更新の最大数	
E メールアドレスあたりの E メール検証コードのレート	AWS	AWSカタログでリクエストできる 1 日あたりの E メール検証コードの最大数	5
アカウントあたりのオープン接続の招待	サンドボックス	サンドボックスカタログ内のパートナーアカウントで維持できるオープン接続招待の最大数	1,000
アカウントあたりのアクティブな接続数	サンドボックス	サンドボックスカタログ内のパートナーアカウントで維持できるアクティブな接続の最大数	1,000
パートナーあたりの E メールドメイン	サンドボックス	サンドボックスカタログのAWSトレーニング認定のためにパートナーアカウントに関連付けることができる E メールドメインの最大数	50
アカウントあたりの接続招待のレート	サンドボックス	サンドボックスカタログで送信できる 1 日あたりの接続招待の最大数	50

Display name (表示名)	カタログ	説明	デフォルトの値
アカウントあたりのプロファイル更新タスクのレート	サンドボックス	サンドボックスカタログで開始できる 1 日あたりのプロファイル更新タスクの最大数	10
アカウントあたりのプロファイル可視性更新のレート	サンドボックス	サンドボックスカタログで開始できる 1 日あたりのプロファイル可視性更新の最大数	5
E メールアドレスあたりの E メール検証コードのレート	サンドボックス	サンドボックスカタログでリクエストできる 1 日あたりの E メール検証コードの最大数。サンドボックスカタログから E メールは送信されません。	該当なし

クォータの理解と管理

レート制限

API レート制限に達すると、サービスは ThrottlingException で応答します。レート制限をより適切に処理するために、AWS ではエクスポネンシャルバックオフと再試行戦略をアプリケーションに実装することをお勧めします。

クォータ引き上げのリクエスト

デフォルトのクォータが要件を満たしていない場合は、Service Quotas ページからクォータの引き上げをリクエストできます。Service Quotas コンソールは、サービスクォータの表示と管理に使用できるブラウザベースのインターフェイスです。Service Quotas には、上部のナビゲーションバー

で選択するか、 で Service Quotas を検索することで、任意のAWS Management ConsoleページからアクセスできますAWS Management Console。

AWS Partner Central Selling API のクォータ

AWS Partner Central 販売 API は、公平な使用を確保し、サービスを誤用から保護するためにクォータを適用します。以下は、オポチュニティごとのさまざまな API オペレーションと関連付けの詳細なクォータです。

API オペレーションクォータ

タイプ	API オペレーション:	クォータ (パートナーアカウントごと)
読み取りアクション	GetOpportunity	10/秒、100,000/24 時間
	GetAwsOpportunitySummary	
	ListOpportunities	
	ListSolutions	
	GetEngagementInvitation	
	ListEngagementInvitations	
読み取りアクションのプロスペクティング	GetProspectingFromEngagementTask	100/秒
読み取りアクションのプロスペクティング	ListProspectingFromEngagementTasks	1 秒あたり 5
書き込みアクション	CreateOpportunity	1 秒あたり 1、24 時間あたり 10,000
	UpdateOpportunity	
	AssociateOpportunity	
	DisassociateOpportunity	
	RejectEngagementInvitation	

タイプ	API オペレーション:	クォータ (パートナーアカウントごと)
	AssignOpportunity	
	StartEngagementFromOpportunityTask	
	StartEngagementByAcceptingInvitationTask	
書き込みアクションのプロスペクティング	StartProspectingFromEngagementTask	2/秒
エンゲージメント書き込みアクション	CreateEngagement	15/秒

オポチュニティあたりの関連付けクォータ

関連エンティティ	クォータ
AWS製品	オポチュニティあたり 20
パートナーソリューション	オポチュニティあたり 10
AWS Marketplaceソリューション	オポチュニティあたり 10
AWS Marketplace製品	オポチュニティあたり 10
AWS Marketplaceプライベートオファー	オポチュニティごとに 1

クォータの理解と管理

レート制限

API レート制限に達すると、サービスは ThrottlingException で応答します。レート制限をより適切に処理するために、AWS ではエクスポネンシャルバックオフおよび再試行戦略をアプリケーションに実装することをお勧めします。

クォータの時間枠

日次クォータはローリング 24 時間にリセットされます。過去 24 時間以内に 10,000 件の書き込みアクションを実行し、10,001 回目のリクエストを実行しようとしている場合など、リクエストはスロットリングされます。意図しないスロットリングを防ぐために、アプリケーションの使用パターンでこれを考慮に入れてください。

クォータ引き上げのリクエスト

デフォルトのクォータが要件を満たしていない場合は、Service Quotas ページからクォータの引き上げをリクエストできます。Service Quotas コンソールは、サービスクォータの表示と管理に使用できるブラウザベースのインターフェイスです。Service Quotas には、上部のナビゲーションバーで選択するか、 で Service Quotas を検索することで、任意のAWS Management ConsoleページからアクセスできますAWS Management Console。

AWS Partner Central Benefits API のクォータ

AWS Partner Central Benefits API には次のクォータがあります。

クォータのリクエスト

クォータのリクエスト

API オペレーション	クォータ (AWSアカウントあたり)
ListBenefits	20/秒
GetBenefit	20/秒
AmendBenefitApplication	1 秒あたり 10
CancelBenefitApplication	1 秒あたり 10
CreateBenefitApplication	1 秒あたり 10
GetBenefitApplication	20/秒
ListBenefitApplications	30/秒
RecallBenefitApplication	1 秒あたり 10
SubmitBenefitApplication	1 秒あたり 10

API オペレーション	クォータ (AWSアカウントあたり)
UpdateBenefitApplication	1 秒あたり 10
GetBenefitAllocation	20/秒
ListBenefitAllocations	20/秒
AssociateBenefitApplicationResource	1 秒あたり 10
DisassociateBenefitApplicationResource	1 秒あたり 10

追加のクォータ

追加のクォータ

Display name (表示名)	カタログ	説明	デフォルトの値
アプリケーションのメリット	AWS	AWSカタログに作成できる利点アプリケーションの最大数	10,000
アプリケーションのメリット	サンドボックス	サンドボックスカタログで作成できる利点アプリケーションの最大数	10,000

クォータの理解と管理

レート制限

API レート制限に達すると、サービスは ThrottlingException で応答します。レート制限をより適切に処理するために、AWS ではエクスポネンシャルバックオフおよび再試行戦略をアプリケーションに実装することをお勧めします。

クォータ引き上げのリクエスト

デフォルトのクォータが要件を満たしていない場合は、Service Quotas ページからクォータの引き上げをリクエストできます。Service Quotas コンソールは、サービスクォータの表示と管理に使用できるブラウザベースのインターフェイスです。Service Quotas には、上部のナビゲーションバーで選択するか、 で Service Quotas を検索することで、任意のAWS Management ConsoleページからアクセスできますAWS Management Console。

AWS Partner Central Channel API のクォータ

AWS Partner Central Channel API には次のクォータがあります。

クォータのリクエスト

クォータのリクエスト

API オペレーション	クォータ (AWSアカウントあたり)
CreateProgramManagementAccount	3/秒
UpdateProgramManagementAccount	3/秒
ListProgramManagementAccounts	5/秒
DeleteProgramManagementAccount	3/秒
CreateChannelHandshake	3/秒
ListChannelHandshakes	5/秒
AcceptChannelHandshake	3/秒
RejectChannelHandshake	3/秒
CancelChannelHandshake	3/秒
CreateRelationship	3/秒
GetRelationship	5/秒
ListRelationships	5/秒

API オペレーション	クォータ (AWSアカウントあたり)
UpdateRelationship	3/秒
DeleteRelationship	3/秒

追加のクォータ

追加のクォータ

Display name (表示名)	カタログ	説明	デフォルトの値
プログラム管理アカウント	AWS	AWSカタログに作成できるプログラム管理アカウントの最大数	50
プログラム管理アカウントあたりの関係	AWS	AWSカタログ内のプログラム管理アカウントごとに確立できるリレーションシップの最大数	1,000
アカウントあたりのプログラム管理アカウントのハンドシェイク	AWS	AWSカタログ内のアクティブなプログラム管理アカウントのハンドシェイクの最大数	100
アカウントごとにサービス期間ハンドシェイクを開始する	AWS	AWSカタログでサービス期間を確立するためのアクティブなハンドシェイクの最大数	200
アカウントあたりのサービス期間のハン	AWS	AWSカタログのサービス期間を取り消す	200

Display name (表示名)	カタログ	説明	デフォルトの値
ドシェイクを取り消す		ためのアクティブなハンドシェイクの最大数	
1 日あたりのプログラム管理アカウントのハンドシェイクのレート	AWS	AWSカタログで同じアカウントに送信できる 1 日あたりのプログラム管理AWSアカウントのハンドシェイクの最大数	5
1 日あたりのサービス期間ハンドシェイクの開始レート	AWS	AWSカタログで同じAWSアカウントに送信できる 1 日あたりの開始サービス期間のハンドシェイクの最大数	5
1 日あたりのサービス期間ハンドシェイクの取り消しレート	AWS	AWSカタログで同じAWSアカウントに送信できる 1 日あたりのサービス期間のハンドシェイクの取り消しの最大数	5
プログラム管理アカウント	サンドボックス	サンドボックスカタログで作成できるプログラム管理アカウントの最大数	50
プログラム管理アカウントあたりの関係	サンドボックス	サンドボックスカタログのプログラム管理アカウントごとに確立できるリレーションシップの最大数	1,000

Display name (表示名)	カタログ	説明	デフォルトの値
アカウントあたりのプログラム管理アカウントのハンドシェイク	サンドボックス	サンドボックスカタログ内のアクティブなプログラム管理アカウントのハンドシェイクの最大数	100
アカウントごとにサービス期間ハンドシェイクを開始する	サンドボックス	サンドボックスカタログでサービス期間を確立するためのアクティブなハンドシェイクの最大数	200
アカウントあたりのサービス期間のハンドシェイクを取り消す	サンドボックス	サンドボックスカタログのサービス期間を取り消すためのアクティブなハンドシェイクの最大数	200
1 日あたりのプログラム管理アカウントのハンドシェイクのレート	サンドボックス	サンドボックスカタログで同じアカウントに送信できる 1 日あたりのプログラム管理AWSアカウントのハンドシェイクの最大数	5
1 日あたりのサービス期間ハンドシェイクの開始レート	サンドボックス	サンドボックスカタログで同じAWSアカウントに送信できる 1 日あたりのサービス開始期間のハンドシェイクの最大数	5

Display name (表示名)	カタログ	説明	デフォルトの値
1 日あたりのサービス期間ハンドシェイクの取り消しレート	サンドボックス	サンドボックスカタログで 1 日あたり同じAWSアカウントに送信できる取り消しサービス期間のハンドシェイクの最大数	5

クォータの理解と管理

レート制限

API レート制限に達すると、サービスは ThrottlingException で応答します。レート制限をより適切に処理するために、AWS ではエクスポネンシャルバックオフと再試行戦略をアプリケーションに実装することをお勧めします。

クォータ引き上げのリクエスト

デフォルトのクォータが要件を満たしていない場合は、Service Quotas ページからクォータの引き上げをリクエストできます。Service Quotas コンソールは、サービスクォータの表示と管理に使用できるブラウザベースのインターフェイスです。Service Quotas には、上部のナビゲーションバーで選択するか、 で Service Quotas を検索することで、任意のAWS Management ConsoleページからアクセスできますAWS Management Console。

AWS Partner Central Revenue Measurement API のクォータ

AWS Partner Central Revenue Measurement API は、クォータを適用して公平な使用を確保し、サービスを誤用から保護します。以下は、さまざまな API オペレーションの詳細なクォータです。

API オペレーションクォータ

タイプ	API オペレーション:	クォータ (パートナーアカウントごと)
読み取りアクション	GetRevenueAttribution	1 秒あたり 5

タイプ	API オペレーション:	クォータ (パートナーアカウントごと)
	GetRevenueAttributionAllocation	
	GetRevenueAttributionAllocationsTask	
	GetMarketplaceRevenueShare	
	GetMarketplaceRevenueShareAllocation	
アクションを一覧表示する	ListRevenueAttributions	1 秒あたり 10
	ListRevenueAttributionAllocations	
	ListMarketplaceRevenueShares	
	ListMarketplaceRevenueShareAllocations	
	ListTagsForResource	
書き込みアクション	CreateRevenueAttribution	2/秒
	UpdateRevenueAttribution	
	CreateMarketplaceRevenueShare	
	CreateMarketplaceRevenueShareAllocation	
	UpdateMarketplaceRevenueShareAllocation	

タイプ	API オペレーション:	クォータ (パートナーアカウントごと)
	TagResource	
	UntagResource	
非同期書き込みアクション	StartRevenueAttributionAllocationsTask	1/秒

クォータの理解と管理

レート制限

API レート制限に達すると、サービスは ThrottlingException で応答します。レート制限をより適切に処理するために、 はエクスポネンシャルバックオフおよび再試行戦略をアプリケーションに実装AWSすることをお勧めします。

クォータ引き上げのリクエスト

デフォルトのクォータが要件を満たしていない場合は、Service Quotas ページからクォータの引き上げをリクエストできます。Service Quotas コンソールは、サービスクォータの表示と管理に使用できるブラウザベースのインターフェイスです。Service Quotas には、上部のナビゲーションバーで選択するか、 で Service Quotas を検索することで、任意のAWS Management ConsoleページからアクセスできますAWS Management Console。

AWS Partner Central API のログ記録

以下のセクションでは、ログ記録について説明します。

トピック

- AWS Partner Central Account API のログ記録
- AWS Partner Central Selling API のログ記録
- AWS Partner Central Benefits API のログ記録
- AWS Partner Central Channel API のログ記録
- AWS Partner Central Revenue Attribution API のログ記録

AWS Partner Central Account API のログ記録

AWS CloudTrail は、 のガバナンス、コンプライアンス、運用監査、リスク監査を可能にするサービスですAWS アカウント。を使用するとAWS CloudTrail、AWSインフラストラクチャ全体のアクションに関連するアカウントアクティビティをログに記録し、継続的にモニタリングし、保持できます。AWSパートナーセントラルアカウント API アクティビティは、イベントとして CloudTrail に記録されます。証跡を作成できます。証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。

概要

AWS Partner Central Account API はAWS CloudTrail、AWS Partner Central のユーザー、ロール、または のサービスによって実行されたアクションを記録するAWSサービスである と統合されています。CloudTrail は、AWS Partner Central Account API のすべての API コールをイベントとしてキャプチャします。キャプチャされた呼び出しには、AWS Partner Central からの呼び出しとAWS、Partner Central Account API オペレーションへのコード呼び出しが含まれます。

証跡を作成する場合は、AWS Partner Central Account API のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。

CloudTrail で収集された情報を使用して、AWS Partner Central Account API に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

AWS Partner Central Account API ログファイルエントリについて

証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。証跡がAWS Partner Central Account API イベントを追跡すると、CloudTrail はすべてのリージョンでイベントをログファイルとして処理します。各ログファイルには、1 つ以上のイベントを含めることができます。

次の例は、AWS Partner Central Account API での `CreatePartner`アクションを示す CloudTrail ログエントリを示しています。

```
{
    "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"
    }
}
```


この例では、 CreatePartnerアクションは CloudTrailTestUser という名前の IAM ユーザーによって呼び出されました。リクエストは 2025 年 11 月 7 日 19:45:58 UTC に行われました。リクエストは、AWS カタログEXAMPLE_PARTNER_IDの ID がExampleLegalName」である新しいパートナーを作成しました。

AWS Partner Central Account API ログファイルエントリのフィールド

CloudTrail ログファイルの各エントリには、リクエストを行ったユーザー、リクエストで処理されたリソース、およびAWS Partner Central Account API によって返されるレスポンス要素に関する情報が含まれています。、、 などのログエントリのフィールドのリストはeventVersionuserIdentityeventTime、アクションに関する詳細情報を提供します。たとえば、 sourceIPAddressフィールドには、リクエストが行われた IP アドレスが表示されます。

AWS Partner Central Selling API のログ記録

AWS CloudTrail は、 のガバナンス、コンプライアンス、運用監査、リスク監査を可能にするサービスですAWS アカウント。を使用するとAWS CloudTrail、AWSインフラストラクチャ全体のアクションに関連するアカウントアクティビティを記録、継続的にモニタリング、保持できます。AWS Partner Central API アクティビティはCloudTrail のイベントとして記録されます。証跡を作成できます。証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。

概要

AWS Partner Central API はAWS CloudTrail、AWS Partner Central のユーザー、ロール、または のサービスによって実行されたアクションを記録するAWSサービスである と統合されています。CloudTrail は、AWS Partner Central のすべての API コールをイベントとしてキャプチャします。キャプチャされた呼び出しには、AWS Partner Central からの呼び出しとAWS、Partner Central API オペレーションへのコード呼び出しが含まれます。

証跡を作成する場合は、AWS Partner Central のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。

CloudTrail によって収集された情報を使用して、AWS Partner Central に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

AWS Partner Central Selling API ログファイルエントリについて

証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。証跡がAWS Partner Central イベントを追跡すると、CloudTrail はすべてのリージョンでイベントをログファイルとして処理します。各ログファイルには、1 つ以上のイベントを含めることができます。

次の例は、AWS Partner Central での ListOpportunitiesアクションを示す CloudTrail ログエントリを示しています。

```
{
    "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"
}
```


この例では、 `ListOpportunities`アクションは CloudTrailTestUser という名前の IAM ユーザーによって呼び出されました。アクションは us-east-1 で呼び出AWS リージョンされ、リクエストは 2023 年 10 月 17 日 21:49:23 UTC に行われました。

AWS Partner Central Selling API ログファイルエントリのフィールド

CloudTrail ログファイルの各エントリには、リクエストを行ったユーザー、リクエストで処理されたリソース、およびAWS Partner Central によって返されるレスポンス要素に関する情報が含まれます。、、 などのログエントリのフィールドのリストはeventVersionuserIdentityeventTime、アクションに関する詳細情報を提供します。たとえば、 `sourceIPAddress`フィールドには、リクエストが行われた IP アドレスが表示されます。

AWS Partner Central Benefits API のログ記録

AWS CloudTrail は、 のガバナンス、コンプライアンス、運用監査、リスク監査を可能にするサービスですAWS アカウント。を使用するとAWS CloudTrail、AWSインフラストラクチャ全体のアクションに関連するアカウントアクティビティを記録、継続的にモニタリング、保持できます。AWS Partner Central Benefits API アクティビティは、CloudTrail にイベントとして記録されます。証跡を作成できます。証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。

概要

AWS Partner Central Benefits API はAWS CloudTrail、AWS Partner Central のユーザー、ロール、または のサービスによって実行されたアクションを記録するAWSサービスである と統合されています。CloudTrail は、AWS Partner Central Benefits API のすべての API コールをイベントとしてキャプチャします。キャプチャされた呼び出しには、AWS Partner Central からの呼び出しとAWS、Partner Central Benefits API オペレーションへのコード呼び出しが含まれます。

証跡を作成する場合は、AWS Partner Central Benefits API のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。

CloudTrail で収集された情報を使用して、AWS Partner Central Benefits API に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

AWS Partner Central Benefits API ログファイルエントリについて

証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。証跡がAWS Partner Central Benefits API イベントを追跡すると、CloudTrail はすべてのリージョンでイベントをログファイルとして処理します。各ログファイルには、1 つ以上のイベントを含めることができます。

次の例は、AWS Partner Central Benefits API での `ListBenefitApplications`アクションを示すCloudTrail ログエントリを示しています。

```
{
    "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"
}
```


この例では、ListBenefitApplicationsアクションは Alice という名前の IAM ユーザーによって呼び出されました。リクエストは 2025 年 10 月 21 日 03:10:59 UTC に行われました。リクエストには、AWS カタログの特典アプリケーションが記載されています。

AWS Partner Central Benefits API ログファイルエントリのフィールド

CloudTrail ログファイルの各エントリには、リクエストを行ったユーザー、リクエストで処理されたリソース、およびAWS Partner Central Benefits API によって返されるレスポンス要素に関する情報が含まれています。、、 などのログエントリのフィールドのリスト

はeventVersionuserIdentityeventTime、アクションに関する詳細情報を提供します。たとえば、 sourceIPAddressフィールドには、リクエストが行われた IP アドレスが表示されます。

AWS Partner Central Channel API のログ記録

AWS CloudTrail は、 のガバナンス、コンプライアンス、運用監査、リスク監査を可能にするサービスですAWS アカウント。を使用するとAWS CloudTrail、AWSインフラストラクチャ全体のアクションに関連するアカウントアクティビティをログに記録し、継続的にモニタリングし、保持できます。AWSパートナーセントラルチャネル API アクティビティはCloudTrail のイベントとして記録されます。証跡を作成できます。証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。

概要

AWS Partner Central Channel API はAWS CloudTrail、AWS Partner Central のユーザー、ロール、または のサービスによって実行されたアクションを記録するAWSサービスである と統合されています。CloudTrail は、AWS Partner Central Channel API のすべての API コールをイベントとしてキャプチャします。キャプチャされた呼び出しには、AWS Partner Central からの呼び出しとAWS、Partner Central Channel API オペレーションへのコード呼び出しが含まれます。

証跡を作成する場合は、AWS Partner Central Channel API のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。

CloudTrail で収集された情報を使用して、AWS Partner Central Channel API に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

AWS Partner Central Channel API ログファイルエントリについて

証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。証跡がAWS Partner Central Channel API イベントを追跡すると、CloudTrail はすべてのリージョンでイベントをログファイルとして処理します。各ログファイルには、1 つ以上のイベントを含めることができます。

次の例は、AWS Partner Central Channel API での CreateProgramManagementAccountアクションを示す CloudTrail ログエントリを示しています。

```
{
    "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"
    }
}
```


この例では、引き受けたロールセッションを通じて ExampleRole という名前の IAM ロールによってCreateProgramManagementAccountアクションが呼び出されました。リクエストは、2025 年 10 月 21 日 17:07:26 UTC に行われました。リクエストは、ソリューションプロバイダープログラムの ID を持つ新しいプログラム管理アカウントを作成pma-example123456789しました。

AWS Partner Central Channel API ログファイルエントリのフィールド

CloudTrail ログファイルの各エントリには、リクエストを行ったユーザー、リクエストで処理されたリソース、およびAWS Partner Central Channel API によって返されるレスポンス要素に関する情報が含まれています。、、 などのログエントリのフィールドのリストはeventVersionuserIdentityeventTime、アクションに関する詳細情報を提供します。たとえば、 sourceIPAddressフィールドには、リクエストが行われた IP アドレスが表示されます。

AWS Partner Central Revenue Attribution API のログ記録

AWS CloudTrail は、 のガバナンス、コンプライアンス、運用監査、リスク監査を可能にするサービスですAWS アカウント。を使用するとAWS CloudTrail、AWSインフラストラクチャ全体のアクションに関連するアカウントアクティビティをログに記録し、継続的にモニタリングし、保持できます。AWS Partner Central Revenue Measurement API アクティビティは、CloudTrail のイベントとして記録されます。証跡を作成できます。証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。

概要

AWS Partner Central 収益測定 API はAWS CloudTrail、AWS Partner Central のユーザー、ロール、または サービスによって実行されたアクションを記録するAWSサービスである と統合されていま

す。CloudTrail は、AWS Partner Central Revenue Attribution のすべての API コールをイベントとしてキャプチャします。キャプチャされた呼び出しには、AWS Partner Central コンソールからの呼び出しと、AWS Partner Central 収益測定 API オペレーションへのコード呼び出しが含まれます。

証跡を作成する場合は、AWS Partner Central Revenue Measurement のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。

CloudTrail で収集された情報を使用して、AWS Partner Central Revenue Measurement に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

次のAWS Partner Central Revenue Measurement API アクションが CloudTrail に記録されます。

- `CreateRevenueAttribution`
- `UpdateRevenueAttribution`
- `GetRevenueAttribution`
- `ListRevenueAttributions`
- `TagResource`
- `UntagResource`
- `ListTagsForResource`
- `CreateMarketplaceRevenueShare`
- `GetMarketplaceRevenueShare`
- `ListMarketplaceRevenueShares`
- `CreateMarketplaceRevenueShareAllocation`
- `GetMarketplaceRevenueShareAllocation`
- `UpdateMarketplaceRevenueShareAllocation`
- `ListMarketplaceRevenueShareAllocations`
- `StartRevenueAttributionAllocationsTask`
- `GetRevenueAttributionAllocationsTask`
- `ListRevenueAttributionAllocations`
- `GetRevenueAttributionAllocation`

AWS Partner Central 収益測定 API ログファイルエントリについて

証跡は、イベントをログファイルとして Amazon S3 バケットに配信できるようにする設定です。証跡がAWS Partner Central Revenue Measurement イベントを追跡すると、CloudTrail はすべてのリージョンでイベントをログファイルとして処理します。各ログファイルには、1 つ以上のイベントを含めることができます。

次の例は、AWS Partner Central Revenue Attribution に対する `CreateRevenueAttribution`アクションを示す CloudTrail ログエントリを示しています。


```
{
    "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"
}
```

この例では、 CreateRevenueAttributionアクションは CloudTrailTestUser という名前の IAM ユーザーによって呼び出されました。アクションは us-east-1AWS リージョンで呼び出され、リクエストは 2026 年 6 月 2 日 11:53:54 UTC に行われました。新しい収益属性リソースが ID で作成されましたra-0123456789abcdef0。

AWS Partner Central Revenue Measurement API ログファイルエントリのフィールド

CloudTrail ログファイルの各エントリには、リクエストを行ったユーザー、リクエストで処理されたリソース、およびAWS Partner Central Revenue Measurement によって返されるレスポンス要素に関する情報が含まれます。、、 などのログエントリのフィールドのリストはeventVersionuserIdentityeventTime、アクションに関する詳細情報を提供します。たとえば、 sourceIPAddressフィールドには、リクエストが行われた IP アドレスが表示されます。

AWS Partner Central API の通知

以下のセクションでは、通知について説明します。

トピック

- AWS Partner Central Account API の通知
- AWS Partner Central Selling API の通知
- AWS Partner Central Benefits API の通知
- AWS Partner Central Channel API の通知

AWS Partner Central Account API の通知

パートナーアカウント接続通知を使用すると、パートナーはビジネス関係と運用ワークフローに直接影響する接続ライフサイクルイベントについて最新情報を得ることができます。これらのイベントは、パートナーが以下を行う上で重要です。

- 新しいコラボレーションに迅速に対応
- 接続ポートフォリオのステータスの認識を維持する
- 接続が確立または終了したときに適切なアクションを実行する

- 関係が変わるタイミングを把握してビジネス継続性を確保する

トピック

- イベントをモニタリングするための前提条件を満たす
- イベントをモニタリングするように Amazon EventBridge を設定する
- アカウント接続 API イベントの詳細

イベントをモニタリングするための前提条件を満たす

ユーザーには、アカウント接続 API によって発行されたイベントにアクセスして管理するための適切な IAM アクセス許可が必要です。EventBridge で使用可能なアクション、リソース、および条件キーの詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge での IAM ポリシー条件EventBridge」を参照してください。条件キーの 1 つは です。これはevents:detail-type、特定のイベントタイプにアクセス許可の範囲を設定するために使用できます。

次のポリシー例は、提案されたイベントのアクセス許可をカスタマイズして範囲を設定する方法を示しています。AllowPutRuleForPartnercentralAccountEvents ステートメントは、aws.partnercentral-accountソースからのイベントに対してのみ、ルールの作成を許可します。

IAM ポリシーの詳細な例については、EventBridge アクセス許可に関する AWS ドキュメントを参照してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralAccountEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-account.connection"
        }
      }
    }
  ]
```

```
}
```

イベントをモニタリングするように Amazon EventBridge を設定する

アカウント接続 API イベントをモニタリングするには、キャプチャするイベントに一致する EventBridge ルールを作成します。AWS Management ConsoleまたはAWS SDKs を使用して、ルールを作成および管理できます。以下のセクションでは、両方の方法を使用してルールを作成する方法について説明します。使用する方法にかかわらず、米国東部 (バージニア北部) `us-east-1`リージョンでルールを作成する必要があります。

AWS Management Consoleセットアップ

を使用して EventBridge ルールを設定するにはAWS Management Console、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントに対応するルールの作成EventBridge」の手順に従います。ルールを作成するときは、イベントバスをデフォルトに設定し、米国東部 (バージニア北部) `us-east-1`リージョンでルールを作成する必要があります。

イベントルールの例を次に示します。

```
{
    "source": ["aws.partnercentral-account"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS SDK のセットアップ

AWS SDKs を使用して、プログラムで EventBridge ルールを作成および管理できます。詳細については、「Amazon EventBridge API リファレンス」の「PutRule」を参照してください。

次の例では、AWS SDK for Python (Boto3) を使用しています。

```
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'])
```

アカウント接続 API イベントの詳細

以下のセクションでは、アカウント接続 API イベントタイプ、それらをトリガーするシナリオ、およびイベント例について説明します。

イベントタイプ

以下は、イベントタイプとそのトリガーです。

- パートナー接続招待の受信: 別のパートナーが接続を確立することを受信者に通知します。
- パートナー接続招待の承諾: 招待が承諾され、接続がアクティブになったことを送信者に通知します
- パートナー接続の招待が拒否されました: 招待が拒否されたことを送信者に通知します
- パートナー接続のキャンセル: アクティブな接続が終了したときに、接続された他の参加者に通知します。
- パートナー接続招待のキャンセル: アクションが実行される前に送信者が招待を取り消したことを受信者に通知します。
- パートナー接続招待の有効期限切れ: 招待の有効期限が切れたことを送信者と受信者の両方に通知します。

イベントの例

以下のセクションでは、前のセクションで前述したイベントの例を示します。

トピック

- パートナー接続の招待を受信しました
- パートナー接続の招待が承諾されました
- パートナー接続の招待が拒否されました
- パートナー接続がキャンセルされました
- パートナー接続の招待がキャンセルされました

- パートナー接続の招待の有効期限が切れました

パートナー接続の招待を受信しました

トリガーコンテキスト

このイベントは、接続招待が正常に作成され、CreateConnectionInvitation API を介して PAC のデータストアに保持されると生成されます。イベントは、招待の作成が正常に完了した直後に送信され、API コールが行われたときだけでなく、PAC のデータストアに保持されます。

新しい接続招待が作成されると、イベントが自動的に送信されます。同じ招待に対して重複するイベントなしで作成された招待ごとに 1 つのイベント。

受取人

接続招待の受信者のAWSアカウント。

予想される処理

一般的なお客様のアクション:

- 即時通知: 新しいパートナーシップの機会についてビジネスチームにアラートをトリガーする
- ワークフロー自動化: 保留中の招待でパートナー管理システムを自動的に更新する
- 決定サポート: 接続タイプに基づいて適切な意思決定者に招待をルーティングする
- 監査ログ記録: コンプライアンスとパートナーシップの追跡のための招待受信を記録する
- レスポンスの自動化: 信頼できるパートナーの場合、特定の招待タイプを自動承認する可能性があります

一般的な統合パターン:

- Lambda 関数 → パートナーポータルダッシュボードを更新する
- SQS キュー → レビューのためのバッチ処理の招待
- SNS トピック → ビジネスチームへの E メール/Slack 通知
- API 送信先 → 外部 CRM/パートナー管理システムへのウェブフック

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>"
    }
  }
}
```

パートナー接続の招待が承諾されました

トリガーコンテキスト

このイベントは、接続の招待が正常に受け入れられ、接続が作成され、AcceptConnectionInvitation API を介して PAC のデータストアに保持されると生成されます。イベントは、招待の承諾が正常に完了し、接続が確立されるとすぐに送信されます。

接続の招待が承諾されると、イベントが自動的に送信されます。招待ごとに 1 つのイベントが受け入れられ、同じ承認に対して重複するイベントはありません。

受取人

接続の招待に関係する両方のAWSアカウント: 招待を最初に送信した送信者アカウントと、招待を承諾した受信者アカウント。

予想される処理

一般的なお客様のアクション:

- パートナーシップのアクティベーション: ワークフローをトリガーしてレイヤー 2 ビジネスアクティビティを有効にする
- ステータスの更新: アクティブな接続ステータスでパートナー管理システムを更新する
- 通知: パートナーシップの確立が成功したことをビジネスチームに警告する
- アクセスプロビジョニング: コラボレーションに適切なアクセス許可を自動的に付与する
- 分析: パートナーシップの変換率と成功メトリクスを追跡する

一般的な統合パターン:

- Lambda 関数 → データ共有アクセス許可を有効にし、パートナーポータルを更新する
- SQS キュー → ビジネスセットアップ用のバッチプロセス接続のアクティベーション
- SNS トピック → ビジネスチームと技術チームへの E メール/Slack 通知
- API 送信先 → CRM システムへのウェブフックでパートナーシップのステータスを更新する

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"
     }
```

```
  }
}
```

パートナー接続の招待が拒否されました

トリガーコンテキスト

このイベントは、接続の招待が RejectConnectionInvitation API を介して正常に拒否されたときに生成されます。イベントは、招待拒否が処理され、PAC のデータストアに保持された直後に送信されます。

接続の招待が拒否されると、イベントが自動的に送信されます。招待ごとに 1 つのイベントが拒否され、同じ拒否に対して重複するイベントはありません。

受取人

接続の招待を最初に送信したAWSアカウント (送信者アカウント) 。

予想される処理

一般的なお客様のアクション:

- ステータスの更新: パートナー管理システムを拒否ステータスで更新する
- フォローアップアクション: 代替パートナーシップアプローチのワークフローをトリガーする
- 通知: パートナーシップの決定についてビジネスチームに警告する
- クリーンアップ: 内部システムから保留中の招待参照を削除する

一般的な統合パターン:

- Lambda 関数 → パートナーポータルを更新し、フォローアップワークフローをトリガーする
- SQS キュー → ビジネス分析のバッチプロセス拒否
- SNS トピック → ビジネス開発チームへの E メール/Slack 通知
- API 送信先 → CRM システムへのウェブフックでオポチュニティステータスを更新する

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-****"
    }
  }
}
```

パートナー接続がキャンセルされました

トリガーコンテキスト

このイベントは、アクティブな接続が正常にキャンセルされ、アクティブなステータスが CancelConnection API を介して PAC のデータストアで終了するように更新されると生成されます。イベントは、接続のキャンセルが処理され、接続ステータスが更新された直後に送信されます。

接続が終了すると、イベントが自動的に送信されます。接続ごとに 1 つのイベントがキャンセルされ、同じキャンセルに対して重複するイベントはありません。

受取人

接続内の他の参加者のAWSアカウント (キャンセルを開始した参加者ではありません) 。

予想される処理

一般的なお客様のアクション:

- アクセス失効: アクセス許可を直ちに取り消し、データ共有を無効にする
- ステータスの更新: パートナー管理システムを終了した接続ステータスで更新する
- 通知: パートナーシップの終了についてビジネスチームと技術チームに警告する

- クリーンアップ: 接続リファレンスを削除し、共有リソースをクリーンアップする
- 分析: 接続時間と終了パターンを追跡する

一般的な統合パターン:

- Lambda 関数 → アクセス許可を取り消し、パートナーポータルのステータスを更新する
- SQS キュー → クリーンアップワークフローのバッチプロセス接続の終了
- SNS トピック → ビジネスチームと技術チームへの E メール/Slack 通知
- API 送信先 → CRM システムへのウェブフックでパートナーシップのステータスを更新する

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>>"
    }
  }
}
```

パートナー接続の招待がキャンセルされました

トリガーコンテキスト

このイベントは、保留中の接続招待が CancelConnectionInvitation API を介して正常にキャンセルされたときに生成されます。イベントは、招待のキャンセルが処理され、招待ステータスが CANCELED に更新された直後に送信されます。

接続の招待がキャンセルされると、イベントが自動的に送信されます。接続の招待ごとに 1 つのイベントがキャンセルされ、同じキャンセルに対して重複するイベントはありません。

受取人

接続の招待を受け取るはずのAWSアカウント (受信者アカウント) 。

予想される処理

一般的なお客様のアクション:

- ステータスの更新: パートナー管理システムを更新して保留中の招待リファレンスを削除する
- 通知: 潜在的なパートナーシップの機会が取り消されたことをビジネスチームに警告する
- クリーンアップ: 内部追跡システムから招待リファレンスを削除する
- 分析: パートナーシップインサイトの招待キャンセルパターンを追跡する

一般的な統合パターン:

- Lambda 関数 → パートナーポータルを更新し、保留中の招待通知を削除する
- SQS キュー → ビジネス分析のためのバッチプロセスのキャンセル
- SNS トピック → ビジネス開発チームへの E メール/Slack 通知
- API 送信先 → CRM システムへのウェブフックでオポチュニティステータスを更新する

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-****"
    }
  }
}
```

パートナー接続の招待の有効期限が切れました

トリガーコンテキスト

このイベントは、受信者によって承諾または拒否されずに ExpiresAt タイムスタンプに達した後、保留中の接続招待が自動的に期限切れになると生成されます。招待の有効期限に達すると、イベントは自動的にシステムによってトリガーされます。

接続の招待の有効期限が切れると、イベントが自動的に送信されます。招待ごとに 1 つのイベントが期限切れになり、同じ有効期限の重複イベントはありません。

受取人

接続の招待に関係する両方のAWSアカウント: 招待を最初に送信した送信者アカウントと、招待を受信した受信者アカウント。

予想される処理

一般的なお客様のアクション:

- ステータスの更新: パートナー管理システムを更新して招待を期限切れとしてマークする
- フォローアップアクション: 再エンゲージメントまたは代替パートナーシップアプローチのワークフローをトリガーする
- 通知: 欠落したパートナーシップの機会についてビジネスチームに警告する
- クリーンアップ: 内部追跡システムから期限切れの招待参照を削除する

一般的な統合パターン:

- Lambda 関数 → パートナーポータルを更新し、フォローアップワークフローをトリガーする
- SQS キュー → ビジネス分析のバッチプロセスの有効期限
- SNS トピック → ビジネス開発チームへの E メール/Slack 通知
- API 送信先 → CRM システムへのウェブフックでオポチュニティステータスを更新する

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 の通知

販売 API のイベントは、ステータスの変更や機会の詳細に関するリアルタイムの通知を提供します。これらのイベントは、システムをAWS Partner Central と同期させ、タイムリーな応答と更新を確保するのに役立ちます。

トピック

- イベントをモニタリングするための前提条件を満たす
- イベントをモニタリングするように Amazon EventBridge を設定する
- API イベントの販売の詳細

イベントをモニタリングするための前提条件を満たす

ユーザーには、 Partner Central 販売 API によって発行されたイベントにアクセスして管理するための適切な IAM アクセス許可が必要です。EventBridge で使用可能なアクション、リソー

ス、および条件キーの詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge での IAM ポリシー条件EventBridge」を参照してください。条件キーの 1 つは です。これはevents:detail-type、特定のイベントタイプにアクセス許可の範囲を設定するために使用できます。

次のポリシー例は、提案されたイベントのアクセス許可をカスタマイズして範囲を設定する方法を示しています。AllowPutRuleForPartnercentralSellingEvents ステートメントは、aws.partnercentral-sellingソースからのイベントに対してのみ、ルールの作成を許可します。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AllowPutRuleForPartnercentralSellingEvents",
    "Effect": "Allow",
    "Action": "events:PutRule",
    "Resource": "*",
    "Condition": {
      "ForAllValues:StringEquals": {
        "events:source": "aws.partnercentral-selling"
      }
    }
  }
}
```

イベントをモニタリングするように Amazon EventBridge を設定する

販売 API イベントをモニタリングするには、キャプチャするイベントに一致する EventBridge ルールを作成します。AWS Management ConsoleまたはAWS SDKs を使用して、ルールを作成および管理できます。以下のセクションでは、両方の方法を使用してルールを作成する方法について説明します。使用する方法にかかわらず、米国東部 (バージニア北部) us-east-1リージョンでルールを作成する必要があります。

AWS Management Consoleセットアップ

を使用して EventBridge ルールを設定するにはAWS Management Console、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントに対応するルールの作成EventBridge」の手

順に従います。ルールを作成するときは、イベントバスをデフォルトに設定し、米国東部 (バージニア北部) us-east-1リージョンでルールを作成する必要があります。

イベントルールの例を次に示します。

```
{
    "source": ["aws.partnercentral-selling"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS SDK のセットアップ

AWS SDKs を使用して、プログラムで EventBridge ルールを作成および管理できます。詳細については、「Amazon EventBridge API リファレンス」の「PutRule」を参照してください。

次の例では、AWS SDK for Python (Boto3) を使用しています。

```
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'])
```

API イベントの販売の詳細

以下のセクションでは、販売 API イベントタイプ、それらをトリガーするシナリオ、およびイベント例について説明します。

イベントタイプ

以下は、イベントタイプとそのトリガーです。

- 作成されたオポチュニティ: 新しいオポチュニティが作成されるとトリガーされます。
- オポチュニティの更新: オポチュニティ (オポチュニティまたは対応するオポチュニティ概要)AWS が更新されるとトリガーされます。
- エンゲージメント招待の作成:AWS紹介招待の作成時にトリガーされます。
- Engagement Invitation Accepted: パートナーがAWS Engagement Invitation を受け入れ、機会に対する とのコラボレーションへの関心を確認するAWSとトリガーされます。
- エンゲージメントの招待が拒否されました: パートナーがAWSエンゲージメントの招待を拒否するとトリガーされます。
- エンゲージメント招待の有効期限切れ: このイベントは、パートナーが EngagementInvitation を拒否したときにトリガーされます。招待の有効期限が切れたことを送受信パートナーに通知します。
- エンゲージメントメンバーの追加: このイベントは、招待を承諾した後に新しいメンバーがエンゲージメントに参加するとトリガーされます。これは、エンゲージメントに追加された新しいメンバーについて、エンゲージメントのすべての現在のメンバーに通知します。
- エンゲージメントリソーススナップショットの作成: このイベントは、エンゲージメントに関連付けられたリソース (機会など) のスナップショットの新しいリビジョンが作成されたときにトリガーされます。関連するリソーススナップショットの変更について、エンゲージメントの現在のメンバー全員に通知します。
- 作成されたエンゲージメント: 新しいエンゲージメントが作成されるとトリガーされます。
- エンゲージメントの更新: エンゲージメントが更新されるとトリガーされます。

イベントシナリオ

次の表は、イベントがさまざまなシナリオでどのように動作するかを示しています。これらのシナリオでは、次の前提が適用されます。

- AWSは、これらのシナリオのパートナーでもあります。
- `ResourceSnapshotJob` はパートナーによって正しく設定されています。
- オポチュニティで更新されたフィールドは、リソーススナップショットのオポチュニティテンプレートにも記載されています。

パートナーとしてのユーザーによるアクション	受信したイベント	参加者タイプ
オポチュニティを作成する	作成されたオポチュニティ	

パートナーとしてのユーザーによるアクション	受信したイベント	参加者タイプ
`StartEngagementFromOpportunityTask` を使用してオポチュニティを送信する	機会の更新、エンゲージメントの作成、エンゲージメントリソーススナップショットの作成、エンゲージメントメンバーの追加、エンゲージメント招待の作成	
AWSは送信されたオポチュニティを承認します	エンゲージメント招待の承諾、エンゲージメントメンバーの追加、オポチュニティの更新、エンゲージメントリソーススナップショットの作成	
AWSは送信されたオポチュニティを拒否します	エンゲージメント招待の拒否、機会の更新、エンゲージメントリソーススナップショットの作成	
AWS Marketplaceオファーをオポチュニティに関連付ける	オポチュニティの更新	
ソリューションをオポチュニティに関連付ける	機会が更新され、エンゲージメントリソーススナップショットが作成されました	
オポチュニティを更新する	機会の更新、エンゲージメントリソーススナップショットの作成 (`ResourceSnapshotJob` がセットアップされ、フィールドの更新がテンプレートにある場合はオプション)	

パートナーとしてのユーザーによるアクション	受信したイベント	参加者タイプ
AWSがオポチュニティを更新する	機会が更新され、エンゲージメントリソーススナップショットが作成されました	
エンゲージメントに別のパートナーを招待する	エンゲージメントの招待が作成されました	送信者
エンゲージメントへの参加への招待がパートナーによって承諾される	エンゲージメント招待の承諾、エンゲージメントメンバーの追加	送信者
エンゲージメントへの招待がパートナーによって拒否されました	エンゲージメントの招待が拒否されました	送信者
エンゲージメントへの参加の招待がパートナーによって 15 日間処理されない	エンゲージメントの招待の有効期限が切れました	送信者
別のパートナーからエンゲージメントへの参加の招待を受け取る	エンゲージメントの招待が作成されました	レシーバー
別のパートナーからの招待を受け入れて、 経由でエンゲージメントに参加する `StartEngagementByAcceptingInvitationTask`	エンゲージメント招待の承諾、エンゲージメントメンバーの追加、オポチュニティの作成、オポチュニティの更新、エンゲージメントリソーススナップショットの作成	レシーバー
別のパートナーからのエンゲージメントへの参加の招待を拒否した場合	エンゲージメントの招待が拒否されました	レシーバー

パートナーとしてのユーザーによるアクション	受信したイベント	参加者タイプ
15 日間、別のパートナーからのエンゲージメントへの参加を招待しない	エンゲージメントの招待の有効期限が切れました	レシーバー
エンゲージメントを作成する	作成されたエンゲージメント	送信者
エンゲージメントを更新する	エンゲージメントの更新	送信者、受信者

イベントの例

以下のセクションでは、前のセクションで示したイベントの例を示します。

トピック

- 作成されたオポチュニティ
- オポチュニティの更新
- エンゲージメントの招待が作成されました
- エンゲージメントの招待が承諾されました
- エンゲージメントの招待が拒否されました
- エンゲージメントの招待の有効期限が切れました
- エンゲージメントメンバーの追加
- エンゲージメントリソーススナップショットの作成
- 作成されたエンゲージメント
- エンゲージメントの更新

作成されたオポチュニティ

パートナーは、次の目的でこのイベントを使用します。

- 新しいオポチュニティが作成されたことをユーザーに通知します。
- CRM システムの更新や、新しいオポチュニティの作成に関する営業チームへの通知などの自動ワークフローをトリガーします。

次の例は、一般的なオポチュニティ作成イベントを示しています。

```
{
    "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"
        }
    }
}
```

オポチュニティの更新

パートナーは、次の目的でこのイベントを使用します。

- 既存のオポチュニティが更新されたことをユーザーに通知します。
- CRM システムの更新や営業チームへの通知などの自動ワークフローをトリガーします。

次の例は、一般的なオポチュニティ更新イベントを示しています。

```
{
    "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"
        }
```

```
    }
}
```

エンゲージメントの招待が作成されました

パートナーは、次の目的でこのイベントを使用します。

- 受信した新しい EngagementInvitation をユーザーに通知します。
- CRM システムの更新や営業チームへの通知など、招待を承諾または拒否する自動ワークフローをトリガーします。

次の例は、一般的なエンゲージメント招待作成イベントを示しています。

```
{
    "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
      }
    }
}
```

エンゲージメントの招待が承諾されました

パートナーは、このイベントを使用して以下を行います。

- EngagementInvitation が受け入れられたことをユーザーに通知します。
- エンゲージメントの新しいパートナーを反映するように内部レコードを更新します。
- 自動ワークフローをトリガーしてデータを同期するか、新しいエンゲージメントメンバーについて他のチームメンバーに通知します。

次の例は、一般的なエンゲージメント招待の承諾イベントを示しています。

```
{
    "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
        }
    }
}
```

エンゲージメントの招待が拒否されました

パートナーは、このイベントを使用して以下を行います。

- EngagementInvitation が拒否されたことをユーザーに通知します。

- 拒否された招待を反映するように内部レコードを更新します。
- 拒否についてセールスチームに通知するなど、必要なワークフローをトリガーします。

次の例は、一般的なエンゲージメント招待の拒否イベントを示しています。

```
{
    "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
        }
    }
}
```

エンゲージメントの招待の有効期限が切れました

パートナーは、このイベントを使用して以下を行います。

- EngagementInvitation の有効期限が切れており、対応できなくなったことをユーザーに通知します。
- 期限切れの招待を反映するように内部レコードを更新します。
- 期限切れの招待をセールスチームに通知するなど、必要なワークフローをトリガーします。

次の例は、一般的なエンゲージメント招待の有効期限が切れたイベントを示しています。

```
{
    "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
      }
    }
}
```

エンゲージメントメンバーの追加

パートナーは、このイベントを使用して以下を行います。

- エンゲージメントメンバーシップの変更をユーザーに通知します。
- 新しいエンゲージメントメンバーを反映するように内部レコードを更新します。
- チームメンバーに変更を通知するなど、必要なワークフローをトリガーします。

次の例は、一般的なエンゲージメントメンバー追加イベントを示しています。

```
{
    "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"
      }
    }
}
```

エンゲージメントリソーススナップショットの作成

パートナーはこのイベントを使用して、エンゲージメントに関連するリソースの変更を追跡し、CRM システムの更新や営業チームへの通知などの自動ワークフローをトリガーします。スナップショットテンプレートは、更新のタイプを決定します。

`OpportunitySummaryView`

- エンゲージメントが更新されたことをエンゲージメントのすべてのメンバーに通知します。

`AwsOpportunitySummaryFullView`

- AWS がエンゲージメント内の機会に対して特に行った更新を追跡します。
- この通知とスナップショットは、オポチュニティ所有者にのみ表示されます。
- `OpportunitySummaryView` テンプレートでは利用できない、ディールサイジングの更新が含まれています。

次の例は、一般的なエンゲージメントリソーススナップショット作成イベントを示しています。

```
{
    "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"
      }
    }
}
```

作成されたエンゲージメント

パートナーは、このイベントを使用して以下を行います。

- 新しいエンゲージメントが作成されたことをユーザーに通知します。
- CRM システムの更新や、新しいエンゲージメントの作成に関するセールスチームへの通知などの自動ワークフローをトリガーします。

次の例は、一般的なエンゲージメント作成イベントを示しています。

```
{
    "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"]

        }
    }
}
```

エンゲージメントの更新

パートナーは、このイベントを使用して以下を行います。

- 既存のエンゲージメントが更新されたことをユーザーに通知します。
- CRM システムの更新や営業チームへの通知などの自動ワークフローをトリガーします。

次の例は、一般的なエンゲージメント更新イベントを示しています。

```
{
    "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 の通知

Benefits Service は、 BenefitApplications と BenefitAllocations の両方の EventBridge イベントを提供します。これらのイベントにより、お客様は、最初のアプリケーションから割り当て、最終的なフルフィルメントまで、リソースのライフサイクル全体で変更に対応する包括的なイベント駆動型アーキテクチャを構築できます。

トピック

- イベントをモニタリングするための前提条件を満たす
- イベントをモニタリングするように Amazon EventBridge を設定する
- メリット API イベントの詳細

イベントをモニタリングするための前提条件を満たす

ユーザーは、メリット API によって発行されたイベントにアクセスして管理するための適切な IAM アクセス許可が必要です。EventBridge で使用できるアクション、リソース、および条件キーの詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge での IAM ポリシー条件EventBridge」を参照してください。条件キーの 1 つは です。これは`events:detail-type`、特定のイベントタイプにアクセス許可の範囲を設定するために使用できます。

次のポリシー例は、提案されたイベントのアクセス許可をカスタマイズして範囲を設定する方法を示しています。`AllowPutRuleForPartnercentralBenefitsEvents` ステートメントは、`aws.partnercentral-benefits`ソースからのイベントに対してのみ、ルールの作成を許可します。

IAM ポリシーの詳細な例については、EventBridge アクセス許可に関する AWS ドキュメントを参照してください。

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

```
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-benefits"
        }
      }
    }
  ]
}
```

イベントをモニタリングするように Amazon EventBridge を設定する

メリット API イベントをモニタリングするには、キャプチャするイベントに一致する EventBridge ルールを作成します。AWS Management ConsoleまたはAWS SDKs を使用して、ルールを作成および管理できます。以下のセクションでは、両方の方法を使用してルールを作成する方法について説明します。使用する方法にかかわらず、米国東部 (バージニア北部) `us-east-1`リージョンでルールを作成する必要があります。

AWS Management Consoleセットアップ

を使用して EventBridge ルールを設定するにはAWS Management Console、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントに対応するルールの作成EventBridge」の手順に従います。ルールを作成するときは、イベントバスをデフォルトに設定し、米国東部 (バージニア北部) `us-east-1`リージョンでルールを作成する必要があります。

イベントルールの例を次に示します。

```
{
    "source": ["aws.partnercentral-benefits"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS SDK のセットアップ

AWS SDKs を使用して、プログラムで EventBridge ルールを作成および管理できます。詳細については、「Amazon EventBridge API リファレンス」の「PutRule」を参照してください。

次の例では、AWS SDK for Python (Boto3) を使用しています。

```
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'])
```

メリット API イベントの詳細

以下のセクションでは、API イベントタイプ、イベントをトリガーするシナリオ、およびイベント例について説明します。

イベントタイプ

以下は、イベントタイプとそのトリガーです。

メリットアプリケーション

- メリットアプリケーションの作成: メリットアプリケーションの作成時
- レビュー中のメリットアプリケーション: メリットアプリケーションが送信されるか、アプリケーションが「必要なアクション」から「レビュー中」に移行すると、このイベントがトリガーされます。
- メリットアプリケーションアクションが必要: このイベントは、内部レビューワーがアプリケーションステータスを更新して、レビューを続行する前にパートナーが追加情報、明確化、または変更を提供する必要があることを示す場合に発生します。
- メリットアプリケーションの拒否: このイベントは、内部レビューワーがアプリケーションのステータスを更新して、アプリケーションがメリットの基準または要件を満たしておらず、拒否されたことを示す場合に発生します。
- メリットアプリケーション承認済み: このイベントは、内部レビューワーがアプリケーションのステータスを更新して、アプリケーションがメリット配分のために承認されたことを示す場合に発生します。

- メリットアプリケーションステージの変更: このイベントは、ビジネスレビュー、AWSレビュー、財務レビューなどのレビュープロセス中に内部レビューワーがアプリケーションのステージを更新したときに発生します。

メリット配分

- メリット配分のアクティブ化: メリット配分が作成または内部サービスによってアクティブ化された に更新された場合。
- メリット配分が無効: このイベントは、内部レビューワーが配分ステータスを更新するか、メリット配分が自動的に期限切れになったときに発生します。
- メリット配分達成: このイベントは、内部レビューワーが配分ステータスを更新して、配分が完全に使用されたことを示す場合に発生します。

イベントの例

以下のセクションでは、前のセクションで示したイベントの例を示します。

トピック

- 作成されたメリットアプリケーション
- レビュー中の利点アプリケーション
- メリットアプリケーションアクションが必要
- メリットアプリケーションが拒否されました
- 特典アプリケーションが承認されました
- メリットアプリケーションステージが変更されました
- メリット配分がアクティブ化されました
- メリット配分が非アクティブ化されました
- メリット配分を達成

作成されたメリットアプリケーション

トリガーコンテキスト

Benefit Application Created イベントは、Benefit Service を通じて新しい Benefit アプリケーションが正常に作成されたときに Amazon EventBridge に発行されます。このイベント

は、CreateBenefitApplication API コールが正常に完了し、パートナーが特定の利点のリクエストを開始したことを示すときに発生します。

このイベントは、利点アプリケーションのライフサイクルの開始を表し、新しいアプリケーションがアカウントに送信されるとすぐに通知されます。

受取人

所有アカウントは、メリット配分のイベントを受け取ります。

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"]
    }
  }
}
```

レビュー中の利点アプリケーション

トリガーコンテキスト

Benefit Application In Review イベントは、Benefit Service を通じて Benefit Application が IN_REVIEW ステータスに移行したときに Amazon EventBridge に発行されます。このイベントは、次の場合に発生します。

- パートナーが SubmitBenefitApplication API コールを介してアプリケーションを送信する
- パートナーがフィードバックに対処した後、アプリケーションが ACTION_REQUIRED から IN_REVIEW に戻る
- アプリケーションがリコールされ、再送信される

このイベントは、アプリケーションが正式なレビュープロセスに入り、AWS内部チームによって評価されていることを示します。

受取人

所有アカウントは、メリット配分のイベントを受け取ります。

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"]
    }
  }
}
```

メリットアプリケーションアクションが必要

トリガーコンテキスト

Benefit Application Action Required イベントは、Benefit Service を通じて Benefit Application が ACTION_REQUIRED ステータスに移行したときに Amazon EventBridge に発行されます。このイベントは、内部レビューワーが UpdateBenefitApplicationInternal API を介してアプリケーションステータスを更新し、レビューを続行する前にパートナーが追加情報、明確化、または変更を提供する必要があることを示す場合に発生します。

このイベントは、レビュープロセスでアプリケーションを前進させるためにパートナーの介入が必要な、アプリケーションライフサイクルの重要なポイントを表します。

受取人

所有アカウントは、メリット配分のイベントを受け取ります

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"]
    }
  }
}
```

メリットアプリケーションが拒否されました

トリガーコンテキスト

Benefit Application Rejected イベントは、Benefit Service を通じて Benefit Application が REJECTED ステータスに移行したときに Amazon EventBridge に発行されます。このイベントは、内部レビューワーが UpdateBenefitApplicationInternal API を介してアプリケーションのステータスを更新して、アプリケーションが利点の基準または要件を満たしておらず、拒否されたことを示す場合に発生します。

このイベントは、アプリケーションが正式に拒否され、メリット配分に進まないアプリケーションライフサイクルのターミナル状態を表します。

受取人

所有アカウントは、メリット配分のイベントを受け取ります。

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"]
    }
  }
}
```

特典アプリケーションが承認されました

トリガーコンテキスト

Benefit Application Approved イベントは、Benefit Service を通じて Benefit Application が APPROVED ステータスに移行したときに Amazon EventBridge に発行されます。このイベントは、内部レビューワーが UpdateBenefitApplicationInternal API を介してアプリケーションのステータスを更新して、アプリケーションがすべてのメリット基準と要件を満たし、メリット配分が承認されていることを示すときに発生します。

このイベントは、アプリケーションレビュープロセスの正常な完了を表し、通常、対応するメリット配分の作成をトリガーします。

受取人

所有アカウントは、メリット配分のイベントを受け取ります。

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"]
    }
  }
}
```

メリットアプリケーションステージが変更されました

トリガーコンテキスト

Benefit Application Stage Changed イベントは、Benefit Service を通じて Benefit Application のレビューステージが更新されると、Amazon EventBridge に発行されます。このイベントは、内部レビューワーがアプリケーションのステージを可能な列挙値 (FINANCE_REVIEW、BUSINESS_REVIEW、TECH_REVIEW、AWS_REVIEW) のいずれかに更新したときに発生します。

ステータスの変更とは異なり、ステージの変更は読み取り専用の情報更新であり、パートナーアクションを必要とせずに内部レビューワークフローの進行状況を可視化できます。

受取人

所有アカウントは、メリット配分のイベントを受け取ります。

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"]
    }
  }
}
```

メリット配分がアクティブ化されました

トリガーコンテキスト

メリット配分がアクティブ化されたイベントは、メリット配分が Benefits Service を通じて ACTIVE ステータスに移行したときに Amazon EventBridge に発行されます。このイベントは、内部レビューワーが UpdateBenefitAllocationInternal API を介して割り当てステータスを更新して、以前に非アクティブな割り当てを再アクティブ化した場合に発生します。

このイベントは通常、一時的に非アクティブ化された割り当て (有効期限、ポリシーの変更、または管理上の理由による) がアクティブステータスに復元され、パートナーの利用に再びメリットをもたらす場合に発生します。

受取人

パートナーに属するAWSアカウントがメッセージを受信します。つまり、リソースを所有するAWSアカウントです。

サンプルイベント


```
{
  "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"]
    }
  }
}
```


メリット配分が非アクティブ化されました

トリガーコンテキスト

メリット配分の非アクティブ化イベントは、メリット配分が Benefits Service を通じて INACTIVE ステータスに移行したときに Amazon EventBridge に発行されます。このイベントは、内部レビューワーが UpdateBenefitAllocationInternal API を介して割り当てステータスを更新するか、利点割り当てが自動的に期限切れになるときに発生します。

このイベントは通常、割り当てが一時的に停止され (有効期限、ポリシーの変更、コンプライアンスの問題、または管理上の理由により)、再アクティブ化されるまでパートナーの利用にメリットが利用できなくなった場合に発生します。

受取人

パートナーに属するAWSアカウントがメッセージを受信します。つまり、リソースを所有するAWSアカウントです。

サンプルイベント

```
{
  "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"]
    }
  }
}
```

メリット配分を達成

トリガーコンテキスト

Benefit Allocation Fulfilled イベントは、Benefit Service を通じて Benefit Allocation が FULFILLED ステータスに移行したときに Amazon EventBridge に発行されます。このイベントは、内部レビュー

ワーが UpdateBenefitAllocationInternal API を介して割り当てステータスを更新して、割り当てが完全に使用されたこと、または利点条件が満たされたことを示す場合に発生します。

このイベントは、メリットライフサイクルの完了を表し、パートナーが割り当てられたメリットを正常に活用したか、割り当てが自然な結論に達したことを示します。

受取人

パートナーに属するAWSアカウントがメッセージを受信します。つまり、リソースを所有するAWSアカウントです。

サンプルイベント

```
{
  "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 の通知

チャネル API のイベントは、ステータスの変更やチャネル関連のアクティビティの詳細に関するリアルタイムの通知を提供します。これらのイベントは、システムをAWS Partner Central と同期させ、タイムリーな応答と更新を確保するのに役立ちます。

前提条件

AWS Partner Central Channel API からイベントにアクセスして管理するには、適切な IAM アクセス許可が必要です。EventBridge で使用可能なアクション、リソース、および条件キーの詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge での IAM ポリシー条件EventBridge」を参照してください。条件キーの 1 つは です。これを使用して`events:detail-type`、特定のイベントタイプにアクセス許可の範囲を設定できます。

次のポリシー例は、イベントのアクセス許可をカスタマイズして範囲を設定する方法を示しています。`AllowPutRuleForPartnercentralChannelEvents` ステートメントは、`aws.partnercentral-channel`ソースからのイベントに対してのみ、ルールの作成を許可します。

IAM ポリシーの詳細な例については、EventBridge アクセス許可に関する AWS ドキュメントを参照してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralChannelEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-channel"
        }
      }
    }
  ]
}
```

イベントをモニタリングするように Amazon EventBridge を設定する

チャネル API イベントをモニタリングするには、キャプチャするイベントに一致する EventBridge ルールを作成します。AWS Management ConsoleまたはAWS SDKs を使用して、ルールを作成および管理できます。以下のセクションでは、両方の方法を使用してルールを作成する方法について説明します。使用する方法にかかわらず、米国東部 (バージニア北部) `us-east-1`リージョンでルールを作成する必要があります。

AWS Management Consoleセットアップ

を使用して EventBridge ルールを設定するにはAWS Management Console、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントに対応するルールの作成EventBridge」の手順に従います。ルールを作成するときは、イベントバスをデフォルトに設定し、米国東部 (バージニア北部) `us-east-1`リージョンでルールを作成する必要があります。

イベントルールの例を次に示します。

```
{
    "source": ["aws.partnercentral-channel"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS SDK のセットアップ

AWS SDKs を使用して、プログラムで EventBridge ルールを作成および管理できます。詳細については、「Amazon EventBridge API リファレンス」の「PutRule」を参照してください。

次の例では、AWS SDK for Python (Boto3) を使用しています。

```
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'])
```

チャネル API イベントの詳細

以下のセクションでは、チャネル API イベントタイプ、それらをトリガーするシナリオ、およびイベント例について説明します。

イベントタイプ

以下は、イベントタイプとそのトリガーです。

- チャネルハンドシェイクの作成: 新しいチャネルハンドシェイクの作成時にトリガーされます。
- Channel Handshake Accepted: 新しいチャネルハンドシェイクが受け入れられるとトリガーされます。
- チャネルハンドシェイクが拒否されました: 新しいチャネルハンドシェイクが拒否されるとトリガーされます。

作成されたチャネルハンドシェイク

次の例は、さまざまなハンドシェイクタイプの一般的なチャネルハンドシェイク作成イベントを示しています。

サービス期間ハンドシェイクの開始:

```
{
    "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"
            }
        }
    }
}
```

サービス期間のハンドシェイクを取り消す:

```
{
    "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"
            }
        }
    }
}
```

プログラム管理アカウントのハンドシェイク:

```
{
    "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"
            }
        }
```

```
    }
}
```

チャネルハンドシェイクが受け入れられました

次の例は、さまざまなハンドシェイクタイプの一般的な Channel Handshake Accepted イベントを示しています。

サービス期間ハンドシェイクの開始:

```
{
    "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"
            }
        }
    }
}
```

サービス期間のハンドシェイクを取り消す:

```
{
    "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"
            }
        }
    }
}
```

プログラム管理アカウントのハンドシェイク:

```
{
    "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"
            }
        }
    }
}
```

チャネルハンドシェイクが拒否されました

次の例は、さまざまなハンドシェイクタイプの一般的なチャネルハンドシェイク拒否イベントを示しています。

サービス期間ハンドシェイクの開始:

```
{
    "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"
            }
        }
    }
}
```

サービス期間のハンドシェイクを取り消す:

```
{
    "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"
            }
        }
    }
}
```

プログラム管理アカウントハンドシェイク:

```
{
    "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"
```

```
            }
        }
    }
}
```

サンドボックスでのテスト

パートナーは、サンドボックスを使用して、安全で隔離された環境でAWS Partner Central API インタラクションをテストおよび検証できるため、ソリューションを本番環境に昇格する前にスムーズなオペレーションを実現できます。 は、本番環境と同様のレスポンスを返す動的サンドボックスをAWS Partner Central API ユーザーAWSに提供します。AWSは、サンドボックス環境にユーザーインターフェイスを提供しません。したがって、パートナーはプログラムによる応答に基づいてソリューションをテストする必要があります。

サンドボックス環境へのアクセス

パートナーは、 をAWS アカウント Partner Central アカウントにリンクするとすぐにテスト環境にアクセスできます。詳細については、AWS「アカウントをAWS Partner Central アカウントにリンクする」を参照してください。各リクエストには、データ環境を決定する `catalog` パラメータが含まれています。`catalog` が に設定されている場合AWS、本番データを参照し、 に設定されている場合Sandbox、サンドボックスデータを参照します。

サンドボックス環境に関する重要な詳細

1. データの更新: は 1 年に 1 回、サンドボックス環境 (通常は年初) のデータAWSを更新します。この更新後、テスト環境のデータの一部が失われる可能性があります。
2. テスト範囲: サンドボックス環境は通常、スケーラビリティやパフォーマンスのテストではなく、機能テストに使用されます。

トピック

- AWS Partner Central Account API のサンドボックスでのテスト
- AWS Partner Central Selling API のサンドボックスでのテスト
- AWS Partner Central Benefits API のサンドボックスでのテスト
- AWS Partner Central Channel API のサンドボックスでのテスト
- AWS Partner Central Revenue Measurement API のサンドボックスでのテスト

AWS Partner Central Account API のサンドボックスでのテスト

サンドボックスの使用方法

サンドボックスを使用するには、次の手順を実行します。

1. IAM ロールを作成します。

   AWS Partner Central アカウントにリンクAWS アカウントされた に IAM ロールを作成します。
2. ポリシーの割り当て:

   IAM ロールに次のポリシーを割り当てます。詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

```
{
  "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. IAM ロールの認証情報を使用します。

   ソリューションでこの IAM ロールの認証情報 (シークレットキーとアクセスキー) を使用して API アクションを実行します。

AWSアクションのテスト

テストフェーズでは、多くの場合、AWSアクションをシミュレートする必要があります。このシミュレーションにより、パートナーはAWSサービスとの統合の完全なend-to-endフローを徹底的にテストできます。各リクエストには、データ環境を決定するカタログパラメータが含まれています。

パートナーの作成のシミュレーション

パートナーの作成をシミュレートするには、CreatePartnerアクションのペイロードに をペイロード"catalog": "Sandbox"に含めます。

例えば、次のようになります。

```
{
  "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: サンドボックス内の E メール検証は無効になっています。テストには任意の 6 桁の EmailVerificationCode を使用できます。また、リクエストに「タグ」を使用しないでください。

その他のテストノート

1. 他のアクションをテストするには、ペイロード"catalog": "Sandbox"で を設定します。
2. サンドボックスでパートナーアカウント接続をテストするには、サンドボックスカタログでパートナーを作成した後にStartProfileUpdateTaskアクションを実行する必要があります。
3. 本番稼働に移行するには、カタログ値を Sandbox から に変更しますAWS。

サンドボックス環境でのイベントのテスト

パートナーはサンドボックス環境のイベントを使用して、イベントベースの実装をテストできます。イベントの詳細で を指定してサンドボックスイベントをリッスンするルールAWS アカウントを使用

して、同じ catalog: Sandbox で EventBridge を設定します。詳細については、「AWS Partner Central Account API の通知」を参照してください。

イベントルールの例:

```
{
  "source": ["aws.partnercentral-account"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

catalog として を指定するイベントルールSandboxは、サンドボックス環境で実行するアクションによって生成されたサンドボックスからのイベントのみに一致します。

AWS Partner Central Selling API のサンドボックスでのテスト

サンドボックスの使用方法

1. IAM ロールを作成します。

   AWS Partner Central アカウントにリンクAWS アカウントされた に IAM ロールを作成します。

2. ポリシーの割り当て:

   IAM ロールに次のポリシーを割り当てます。詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

   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. IAM ロールの認証情報を使用します。

   ソリューションでこの IAM ロールの認証情報 (シークレットキーとアクセスキー) を使用して API アクションを実行します。

AWSアクションのテスト

テストフェーズでは、多くの場合、AWSアクションをシミュレートする必要があります。このシミュレーションにより、パートナーはAWSサービスとの統合の完全なend-to-endフローを徹底的にテストできます。

AWS開始されたオポチュニティの作成をシミュレートする

AWS開始されたオポチュニティの作成をシミュレートするには、CreateOpportunityアクションのペイロードに "Catalog": "Sandbox"と を含めます"Origin": "AWS Referral"。

例えば、次のようになります。

```
{
    "Catalog": "Sandbox",
    "Origin": "AWS Referral",
    "OpportunityIdentifier": "O123456",
    "Title": "Test Opportunity",
    ...
}
```

パートナー起点の機会の更新をシミュレートする

パートナーが起案したオポチュニティの更新やその他のアクションをシミュレートするには、ペイロード“Action Required”で "Catalog": "Sandbox"および Lifecycle.ReviewStatus: “Approved”または で UpdateOpportunity アクションを使用します。

更新を実行するGetOpportunityアクションからペイロードを取得する場合は、ID を に変更しIdentifier、次のフィールドを削除してください。

- CreatedDate
- OpportunityTeam
- RelatedEntityIdentifiers

例えば、次のようになります。

```
{
    "Catalog": "Sandbox",
    "Identifier": "O123456",
    "Title": "Updated Test Opportunity",
    "Lifecycle": {
        "ReviewStatus": "Approved"
    }
    ...
}
```

サンドボックス環境でのイベントのテスト

パートナーはサンドボックス環境のオポチュニティイベントを使用して、イベントベースの実装をテストできます。イベントの詳細で を指定してサンドボックスイベントをリッスンするルールAWS アカウントを使用して、同じ catalog: sandbox で EventBridge を設定します。詳細については、「AWS Partner Central Selling API の通知」を参照してください。

イベントルールの例:

```
{
    "source": ["aws.partnercentral-selling"],
    "detail": {
        "catalog": ["Sandbox"]
    }
}
```

catalog として を指定するイベントルールSandboxは、サンドボックス環境で実行するアクションによって生成されたサンドボックスからのイベントのみと一致します。

その他のテストノート

1. テストAssociateOpportunityアクション:
   a. AssociateOpportunity アクションをテスト"S-1234567"するには、デフォルトの ソリューションを使用します。
   b. Marketplace オファーをテストするには、 アカウントの実際のオファー ID を使用します。
2. 本番稼働に移行するには、catalog値を から Sandbox に変更しますAWS。

ヘルプの利用

CRM を と統合する際に課題が発生した場合AWS、またはここで説明されていない特定のシナリオをテストする必要がある場合は、次の手順でケースを提起して サポートにお問い合わせください。

1. Partner Network 認証情報を使用してAWS Partner AWS Central にサインインします。
2. Partner Central のサポートセンターで、新しいケースのログOpen New Case記録を選択します。フィールドに以下のように入力します。
   a. サポートケースのタイプ: AWS Partner Central。
   b. に関する質問: Partner Central Tools or ACE leads and opportunities。
   c. 詳細: 最適な CRM 統合ケースタイプを選択します。
   d. 件名: リクエストの簡単な説明を含めます。
   e. 説明: 問題、質問、エラー、トラブルシューティング手順の詳細な説明を入力します。
   f. 添付ファイル: 必要に応じて、ログとスクリーンショットを含めます。

AWS Partner Central Benefits API のサンドボックスでのテスト

サンドボックスの使用方法

サンドボックスを使用するには、次の手順を実行します。

1. IAM ロールを作成します。

   AWS Partner Central アカウントにリンクAWS アカウントされた に IAM ロールを作成します。
2. ポリシーの割り当て:

   IAM ロールに次のポリシーを割り当てます。詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

```
{
  "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. IAM ロールの認証情報を使用します。

   ソリューションでこの IAM ロールの認証情報 (シークレットキーとアクセスキー) を使用して API アクションを実行します。

AWSアクションのテスト

テストフェーズでは、多くの場合、AWSアクションをシミュレートする必要があります。このシミュレーションにより、パートナーはAWSサービスとの統合の完全なend-to-endフローを徹底的にテストできます。各リクエストには、データ環境を決定するカタログパラメータが含まれています。

メリットアプリケーションの作成のシミュレーション

メリットアプリケーションの作成をシミュレートするには、`CreateBenefitApplication`アクションのペイロードで、ペイロード`"catalog": "Sandbox"`に を含めます。

例えば、次のようになります。

```
{
  "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"
    }
  ]
}
```

その他のテストノート

1. 他のアクションをテストするには、ペイロード"catalog": "Sandbox"で を設定します。
2. サンドボックスカタログと AWS カタログの特典 IDs は異なります。で GetBenefit API コールを行い"catalog": "Sandbox"、サンドボックスでメリット識別子を取得します。
3. サンドボックスカタログと AWS カタログの利点の割り当ては異なります。
4. 本番稼働に移行するには、カタログ値を から Sandbox に変更しますAWS。

サンドボックス環境でのイベントのテスト

パートナーはサンドボックス環境のイベントを使用して、イベントベースの実装をテストできます。イベントの詳細で を指定してサンドボックスイベントをリッスンするルールAWS アカウントを使用して、同じ catalog: Sandbox で EventBridge を設定します。詳細については、「AWS Partner Central Benefits API の通知」を参照してください。

イベントルールの例:

```
{
  "source": ["aws.partnercentral-benefits"],
  "detail": {
    "catalog": ["Sandbox"]
```

```
  }
}
```

catalog として を指定するイベントルールSandboxは、サンドボックス環境で実行するアクションによって生成されたサンドボックスからのイベントのみと一致します。

AWS Partner Central Channel API のサンドボックスでのテスト

サンドボックス内のアカウントはチャネル特典の対象ではなく、すべてのプログラム要件に対して検証されるわけではありません。

サンドボックスの使用方法

サンドボックスを使用するには、次の手順を実行します。

1. IAM ロールを作成します。

   AWS Partner Central アカウントにリンクAWS アカウントされた に IAM ロールを作成します。

2. ポリシーの割り当て:

   IAM ロールに次のポリシーを割り当てます。詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

```
{
  "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. IAM ロールの認証情報を使用します。

   ソリューションでこの IAM ロールの認証情報 (シークレットキーとアクセスキー) を使用して API アクションを実行します。

AWSアクションのテスト

テストフェーズでは、多くの場合、AWSアクションをシミュレートする必要があります。このシミュレーションにより、パートナーはAWSサービスとの統合の完全なend-to-endフローを徹底的にテストできます。各リクエストには、データ環境を決定するカタログパラメータが含まれています。

プログラム管理アカウントの作成のシミュレーション

プログラム管理アカウントの作成をシミュレートするには、`CreateProgramManagementAccount`アクションのペイロードで、ペイロード`"catalog": "Sandbox"`に を含めます。

例えば、次のようになります。

```
{
  "catalog": "Sandbox",
  "accountId": "123456789012",
  "displayName": "Test PMA Name",
  "clientToken": "123abc456def789ghi",
  "program": "SOLUTION_PROVIDER",
  "tags": [
    {
      "key": "testkey",
      "value": "testvalue"
    }
  ]
}
```

関係の更新をシミュレートする

関係の更新をシミュレートするには、ペイロード`"catalog": "Sandbox"`で で`UpdateRelationship`アクションを使用します。

例えば、次のようになります。

```
{
  "catalog": "Sandbox",
  "displayName": "Updated Test PMA",
  "identifier": "rs-abc123def456g",
  "programManagementAccountIdentifier": "pma-123abc456def7"
}
```

その他のテストノート

1. 他のアクションをテストするには、ペイロード"catalog": "Sandbox"で を設定します。
2. 本番稼働に移行するには、カタログ値を Sandbox から に変更しますAWS。

サンドボックス環境でのイベントのテスト

パートナーはサンドボックス環境のイベントを使用して、イベントベースの実装をテストできます。イベントの詳細で を指定してサンドボックスイベントをリッスンするルールAWS アカウントを使用して、同じ catalog: Sandbox で EventBridge を設定します。詳細については、「AWS Partner Central Channel API の通知」を参照してください。

イベントルールの例:

```
{
  "source": ["aws.partnercentral-channel"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

catalog として を指定するイベントルールSandboxは、サンドボックス環境で実行するアクションによって生成されたサンドボックスからのイベントのみと一致します。

ヘルプの利用

テストで課題が発生した場合、またはここで説明されていない特定のシナリオをテストする必要がある場合は、次の手順でケースを提起して サポートにお問い合わせください。

1. Partner Network 認証情報を使用してAWS Partner AWS Central にサインインします。
2. Partner Central のサポートセンターで、新しいケースのログOpen New Case記録を選択します。フィールドに以下のように入力します。
   a. サポートケースのタイプ: Channel Program
   b. 以下に関する質問: General Program Inquiry
   c. 詳細: 最も適切なチャネルケースタイプを選択します。
   d. 件名: リクエストの簡単な説明を含めます。
   e. 説明: 問題、質問、エラー、トラブルシューティング手順の詳細な説明を入力します。
   f. 添付ファイル: 必要に応じて、ログとスクリーンショットを含めます。

AWS Partner Central Revenue Measurement API のサンドボックスでのテスト

サンドボックスの使用方法

サンドボックスを使用するには、次の手順を実行します。

1. IAM ロールを作成します。

   AWS Partner Central アカウントにリンクAWS アカウントされた に IAM ロールを作成します。
2. ポリシーの割り当て:

   IAM ロールに次のポリシーを割り当てます。詳細については、「Adding and removing IAM identity permissions」(IAM アクセス許可の追加と削除) を参照してください。

```
{
    "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. IAM ロールの認証情報を使用します。

   ソリューションでこの IAM ロールの認証情報 (シークレットキーとアクセスキー) を使用して API アクションを実行します。

AWSアクションのテスト

テストフェーズでは、多くの場合、AWSアクションをシミュレートする必要があります。このシミュレーションにより、パートナーはAWSサービスとの統合の完全なend-to-endフローを徹底的にテストできます。各リクエストには、データ環境を決定するカタログパラメータが含まれています。

収益属性 ID の作成のシミュレーション

収益属性 ID の作成をシミュレートするには、 `CreateRevenueAttribution`アクションのペイロードで、ペイロード`"Catalog": "Sandbox"`に を含めます。

例えば、次のようになります。

```
{
  "Catalog": "Sandbox",
  "ClientToken": "String",
  "Name": "String",
  "Description": "String",
```

```
  "TenancyModel": "MULTI_TENANT | SINGLE_TENANT",
  "ProductIdentifier": "String",
  "Tags": [
    {
      "Key": "String",
      "Value": "String"
    }
  ]
}
```

その他のテストノート

1. 他のアクションをテストするには、ペイロード"Catalog": "Sandbox"で を設定します。
2. 本番稼働に移行するには、カタログ値を から Sandbox に変更しますAWS。

AWS SDKsコード例

次のコード例は、AWS Software Development Kit (SDK) で Partner Central を使用する方法を示しています。

アクションはより大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。アクションは個々のサービス機能を呼び出す方法を示していますが、コンテキスト内のアクションは、関連するシナリオで確認できます。

シナリオは、1 つのサービス内から、または他のAWS のサービスと組み合わせて複数の関数を呼び出し、特定のタスクを実行する方法を示すコード例です。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

コードの例

- AWS SDKs基本的な例
  - AWS SDKsアクション
    - AWS SDK AssignOpportunityで を使用する
    - AWS SDK AssociateOpportunityで を使用する
    - AWS SDK CreateOpportunityで を使用する
    - AWS SDK DisassociateOpportunityで を使用する
    - AWS SDK GetAwsOpportunitySummaryで を使用する
    - AWS SDK GetEngagementInvitationで を使用する
    - AWS SDK GetOpportunityで を使用する
    - AWS SDK ListEngagementInvitationsで を使用する
    - AWS SDK ListOpportunitiesで を使用する
    - AWS SDK ListSolutionsで を使用する
    - AWS SDK RejectEngagementInvitationで を使用する
    - AWS SDK StartEngagementByAcceptingInvitationTaskで を使用する
    - AWS SDK StartEngagementFromOpportunityTaskで を使用する
    - AWS SDK UpdateOpportunityで を使用する
- AWS SDKsシナリオ

- オポチュニティの関連エンティティを更新する

AWS SDKs基本的な例

次のコード例は、 SDKs でAWSAWS Partner Central の基本を使用する方法を示しています。

例

- AWS SDKsアクション
  - AWS SDK AssignOpportunityで を使用する
  - AWS SDK AssociateOpportunityで を使用する
  - AWS SDK CreateOpportunityで を使用する
  - AWS SDK DisassociateOpportunityで を使用する
  - AWS SDK GetAwsOpportunitySummaryで を使用する
  - AWS SDK GetEngagementInvitationで を使用する
  - AWS SDK GetOpportunityで を使用する
  - AWS SDK ListEngagementInvitationsで を使用する
  - AWS SDK ListOpportunitiesで を使用する
  - AWS SDK ListSolutionsで を使用する
  - AWS SDK RejectEngagementInvitationで を使用する
  - AWS SDK StartEngagementByAcceptingInvitationTaskで を使用する
  - AWS SDK StartEngagementFromOpportunityTaskで を使用する
  - AWS SDK UpdateOpportunityで を使用する

AWS SDKsアクション

次のコード例は、AWS SDKs を使用して個々の Partner Central アクションを実行する方法を示しています。それぞれの例には、GitHub へのリンクがあり、そこにはコードの設定と実行に関する説明が記載されています。

これらは Partner Central API を呼び出すもので、コンテキスト内で実行する必要がある大規模なプログラムからのコード抜粋です。アクションは AWS SDKsシナリオ のコンテキスト内で確認できます。

以下の例には、最も一般的に使用されるアクションのみ含まれています。完全なリストについては、「AWSパートナーセントラル API リファレンス」を参照してください。

例

- AWS SDK AssignOpportunityで を使用する
- AWS SDK AssociateOpportunityで を使用する
- AWS SDK CreateOpportunityで を使用する
- AWS SDK DisassociateOpportunityで を使用する
- AWS SDK GetAwsOpportunitySummaryで を使用する
- AWS SDK GetEngagementInvitationで を使用する
- AWS SDK GetOpportunityで を使用する
- AWS SDK ListEngagementInvitationsで を使用する
- AWS SDK ListOpportunitiesで を使用する
- AWS SDK ListSolutionsで を使用する
- AWS SDK RejectEngagementInvitationで を使用する
- AWS SDK StartEngagementByAcceptingInvitationTaskで を使用する
- AWS SDK StartEngagementFromOpportunityTaskで を使用する
- AWS SDK UpdateOpportunityで を使用する

AWS SDK **AssignOpportunity**で を使用する

次のサンプルコードは、AssignOpportunity を使用する方法を説明しています。

Java

SDK for Java 2.x

既存のオポチュニティを他のユーザーに再割り当てします。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「AssignOpportunity」を参照してください。

Python

SDK for Python (Boto3)

既存のオポチュニティを他のユーザーに再割り当てします。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「AssignOpportunity」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **AssociateOpportunity**で を使用する

次のサンプルコードは、AssociateOpportunity を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- オポチュニティの関連エンティティを更新する

Java

SDK for Java 2.x

オポチュニティとさまざまな関連エンティティとの正式な関連付けを作成します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x APIリファレンス」の「AssociateOpportunity」を参照してください。

Python

SDK for Python (Boto3)

オポチュニティとさまざまな関連エンティティとの正式な関連付けを作成します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「AssociateOpportunity」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **CreateOpportunity**で を使用する

次のサンプルコードは、CreateOpportunity を使用する方法を説明しています。

.NET

SDK for .NET

オポチュニティを作成します。

```
// 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.");
            }
        }
    }
}
```

- API の詳細については、「AWS SDK for .NET API リファレンス」の「CreateOpportunity」を参照してください。

Java

SDK for Java 2.x

オポチュニティを作成します。

```
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;
    }

}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「CreateOpportunity」を参照してください。

Python

SDK for Python (Boto3)

オポチュニティを作成します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「CreateOpportunity」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **DisassociateOpportunity**で を使用する

次のサンプルコードは、DisassociateOpportunity を使用する方法を説明しています。

アクション例は、より大きなプログラムからのコードの抜粋であり、コンテキスト内で実行する必要があります。次のコード例で、このアクションのコンテキストを確認できます。

- オポチュニティの関連エンティティを更新する

Java

SDK for Java 2.x

オポチュニティと関連エンティティ間の既存の関連付けを削除します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「DisassociateOpportunity」を参照してください。

Python

SDK for Python (Boto3)

オポチュニティと関連エンティティ間の既存の関連付けを削除します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「DisassociateOpportunity」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **GetAwsOpportunitySummary**で を使用する

次のサンプルコードは、GetAwsOpportunitySummary を使用する方法を説明しています。

Java

SDK for Java 2.x

AWSオポチュニティの概要を取得します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「GetAwsOpportunitySummary」を参照してください。

Python

SDK for Python (Boto3)

AWSオポチュニティの概要を取得します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「GetAwsOpportunitySummary」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **GetEngagementInvitation**で を使用する

次のサンプルコードは、GetEngagementInvitation を使用する方法を説明しています。

Java

SDK for Java 2.x

がパートナーAWSと共有しているエンゲージメントの招待の詳細を取得します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「GetEngagementInvitation」を参照してください。

Python

SDK for Python (Boto3)

がパートナーAWSと共有しているエンゲージメントの招待の詳細を取得します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンス の「GetEngagementInvitation」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **GetOpportunity**で を使用する

次のサンプルコードは、GetOpportunity を使用する方法を説明しています。

.NET

SDK for .NET

オポチュニティを取得します。

```
// 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.");
            }
        }
    }
}
```

- API の詳細については、「AWS SDK for .NET API リファレンス」の「GetOpportunity」を参照してください。

Go

SDK for Go V2

オポチュニティを取得します。

```
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))
}
```

- API の詳細については、「AWS SDK for Go API リファレンス」の「GetOpportunity」を参照してください。

Java

SDK for Java 2.x

オポチュニティを取得します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「GetOpportunity」を参照してください。

Python

SDK for Python (Boto3)

オポチュニティを取得します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「GetOpportunity」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **ListEngagementInvitations**で を使用する

次のサンプルコードは、ListEngagementInvitations を使用する方法を説明しています。

Java

SDK for Java 2.x

パートナーに送信されたエンゲージメントの招待の一覧を取得します。

```
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;
  }
 }
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「ListEngagementInvitations」を参照してください。

Python

SDK for Python (Boto3)

パートナーに送信されたエンゲージメントの招待の一覧を取得します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「ListEngagementInvitations」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **ListOpportunities**で を使用する

次のサンプルコードは、ListOpportunities を使用する方法を説明しています。

.NET

SDK for .NET

オポチュニティを一覧表示します。

```
// 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.");
            }
        }
    }
}
```

- API の詳細については、「AWS SDK for .NET API リファレンス」の「ListOpportunities」を参照してください。

Go

SDK for Go V2

オポチュニティを一覧表示します。

```
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))
}
```

- API の詳細については、「AWS SDK for Go API リファレンス」の「ListOpportunities」を参照してください。

Java

SDK for Java 2.x

オポチュニティを一覧表示します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「ListOpportunities」を参照してください。

Python

SDK for Python (Boto3)

オポチュニティを一覧表示します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「ListOpportunities」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **ListSolutions**で を使用する

次のサンプルコードは、ListSolutions を使用する方法を説明しています。

Java

SDK for Java 2.x

パートナーがパートナーセントラルに登録したパートナーソリューションの一覧を取得します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「ListSolutions」を参照してください。

Python

SDK for Python (Boto3)

パートナーがパートナーセントラルに登録したパートナーソリューションの一覧を取得します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「ListSolutions」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **RejectEngagementInvitation**で を使用する

次のサンプルコードは、RejectEngagementInvitation を使用する方法を説明しています。

Java

SDK for Java 2.x

がAWS共有した EngagementInvitation を拒否します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」のRejectEngagementInvitation」を参照してください。

Python

SDK for Python (Boto3)

がAWS共有した EngagementInvitation を拒否します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「RejectEngagementInvitation」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **StartEngagementByAcceptingInvitationTask**で を使用する

次のサンプルコードは、StartEngagementByAcceptingInvitationTask を使用する方法を説明しています。

Java

SDK for Java 2.x

EngagementInvitation を受け入れることでエンゲージメントを開始します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「StartEngagementByAcceptingInvitationTask」を参照してください。

Python

SDK for Python (Boto3)

EngagementInvitation を受け入れることでエンゲージメントを開始します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「StartEngagementByAcceptingInvitationTask」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **StartEngagementFromOpportunityTask**で を使用する

次のサンプルコードは、StartEngagementFromOpportunityTask を使用する方法を説明しています。

Java

SDK for Java 2.x

エンゲージメントの招待を受け入れ、パートナーのシステムで対応するオポチュニティを作成することで、既存のオポチュニティからエンゲージメントプロセスを開始します。

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「StartEngagementFromOpportunityTask」を参照してください。

Python

SDK for Python (Boto3)

エンゲージメントの招待を受け入れ、パートナーのシステムで対応するオポチュニティを作成することで、既存のオポチュニティからエンゲージメントプロセスを開始します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「StartEngagementFromOpportunityTask」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDK **UpdateOpportunity**で を使用する

次のサンプルコードは、UpdateOpportunity を使用する方法を説明しています。

Java

SDK for Java 2.x

オポチュニティを更新します。

```
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;
  }
    }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の「UpdateOpportunity」を参照してください。

Python

SDK for Python (Boto3)

オポチュニティを更新します。

```
#!/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()
```

- API の詳細については、AWS SDK for Python (Boto3) API リファレンスの「UpdateOpportunity」を参照してください。

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

AWS SDKsシナリオ

次のコード例は、AWS SDKs を使用して Partner Central で一般的なシナリオを実装する方法を示しています。これらのシナリオは、パートナーセントラル内で複数の機能を呼び出すか、他のAWS のサービスと組み合わせて、特定のタスクを実行する方法を示します。各シナリオには、完全なソースコードへのリンクが含まれており、そこからコードの設定方法と実行方法に関する手順を確認できます。

シナリオは、サービスアクションをコンテキストで理解するのに役立つ中級レベルの経験を対象としています。

例

- オポチュニティの関連エンティティを更新する

オポチュニティの関連エンティティを更新する

次のコード例は、以下を実行する方法を示しています。

- 古いエンティティの関連付けを解除します。
- 新しいエンティティを関連付けます。

Java

SDK for Java 2.x

> **Note**
>
> GitHub には、その他のリソースもあります。シナリオリポジトリで完全な例を見つけて、設定と実行の方法を確認してください。

オポチュニティの関連エンティティを更新する

```
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;
 }
}
```

- API の詳細については、「AWS SDK for Java 2.x API リファレンス」の以下のトピックを参照してください。
  - AssociateOpportunity
  - DisassociateOpportunity

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub には、その他のリソースもあります。用例一覧を検索し、AWSコード例リポジトリでの設定と実行の方法を確認してください。

オポチュニティの関連エンティティを更新する

```
#!/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()
```

- API の詳細については、『AWS SDK for Python (Boto3) API リファレンス』の以下のトピックを参照してください。
  - AssociateOpportunity
  - DisassociateOpportunity

AWS SDK 開発者ガイドとコード例の完全なリストについては、「」を参照してくださいAWS SDK でのAWS Partner Central API の使用。このトピックには、使用開始方法に関する情報と、以前の SDK バージョンの詳細も含まれています。

リリースノート

このページには、最新の更新から始まるAWS Partner Central API のドキュメントに対する重要な変更の概要が含まれています。

改訂履歴

変更	説明	日付
RC 6.0	• AWS Partner Central に新しい Partner Revenue Measurement APIs。	2026-06-30
RC 5.3	• ACE Resonate の更新: `GetAwsOpportunitySummary` レスポンスが更新されました。	2026-06-16
RC 5.2	• EUR 通貨サポート: `CreateOpportunity` 、`UpdateOpportunity` 、 が MRR 通貨コードUSDの EUR に加えて を受け入れる`CreateEngagementInvitation` ようになりました。AWS パーティションが `aws-eusc` (AWS 欧州主権クラウド) の場合、通貨コードは である必要がありますEUR。 • 読み取りオペレーションの EUR: `GetOpportunity` 、`ListOpportunities` 、およびが、`aws-eusc`パーティションEUR内のオポチュニティの `CurrencyC`	2026-03-30

変更	説明	日付
	ode フィールドに返るGetAwsOpportunitySummary ようになりました。	
RC 5.1	• EUR 通貨サポート: CreateOpportunity 、UpdateOpportunity 、 が MRR 通貨コードUSDの EUR に加えて を受け入れるCreateEngagementInvitation ようになりました。AWS パーティションが aws-eusc (AWS 欧州主権クラウド) の場合、通貨コードは である必要がありますEUR。 • 読み取りオペレーションの EUR: GetOpportunity 、ListOpportunities 、および が、aws-euscパーティションEUR内のオポチュニティの CurrencyCode フィールドに返るGetAwsOpportunitySummary ようになりました。	2026-03-30

変更	説明	日付
RC 5	• Account API の導入:AWS Partner Central に新しい Account API を追加しました。これにより、パートナーはアカウント情報、登録、プロファイル、ドメイン管理、パートナー接続を管理できます。 • Account API リファレンスドキュメント: パートナー登録、プロファイル管理、ドメイン検証、アカウント接続ワークフローなど、Account API の包括的な API リファレンスドキュメントを追加しました。 • アカウント API ログ記録: API の使用状況の監査と追跡に役立つ、アカウント API の口グ記録のサポートとドキュメントを追加しました。 • アカウント API 通知: Account API のイベント通知サポートを追加しました。これにより、パートナーは、パートナーアカウント接続を含むアカウント関連のアクティビティに関するリアルタイムの更新を受け取ることができます。 • Account API クォータ: Account API のサービス	2025-11-30

変更	説明	日付
	クォータドキュメントを追加しました。 • Account API のサンドボックステスト: Account API のサンドボックステスト機能を追加しました。これにより、パートナーは非本番環境でアカウント管理ワークフローをテストできます。 • Benefits API を導入:AWS Partner Central に新しい Benefits API を追加しました。これにより、パートナーは一元化されたインターフェイスを通じてAWS、パートナープログラム全体でメリットを検出、申請、管理できます。 • Benefits API リファレンスドキュメント: Benefits API の包括的な API リファレンスドキュメントを追加し、特典アプリケーション、割り当て、フルフィルメントプロセスについて説明しました。 • Benefits API ログ記録: API の使用状況の監査と追跡に役立つ Benefits API の口グ記録サポートとドキュメントを追加しました。 • Benefits API 通知: Benefits API のイベント通知サポートを追加し、パートナーが	

変更	説明	日付
	特典アプリケーションと割り当てに関するリアルタイムの更新を受信できるようになりました。 • Benefits API クォータ: Benefits API のサービスクォータドキュメントを追加しました。 • Benefits API のサンドボックステスト: Benefits API のサンドボックステスト機能を追加しました。これにより、パートナーは非本番環境でメリット管理ワークフローをテストできます。 • 販売 API の更新: リード管理ワークフローをサポートするように販売 API エンゲージメントアクションを強化しました。 • 販売 API ドキュメントの更新: 構造が改善され、リード管理のワークフローが更新され、販売 API ドキュメントが強化されました。 • API 通知の更新の販売: エンゲージメントイベントとエンゲージメント招待イベントのイベント通知ドキュメントを強化しました。 • API 構造の拡張: 4 つの異なる APIs (アカウント、メリット、販売、チャネル) をサポートするようにドキュメ	

変更	説明	日付
	ント構造を拡張し、各 API のアクセス許可、ログ記録、通知、クォータ、サンドボックステスト専用のセクションを追加しました。 • 拡張 API 使用ドキュメント: 詳細なワークフローと統合パターンを含む Account and Benefits APIs包括的な使用ガイドを追加しました。 • サポートされているリージョンのドキュメント: Account and Benefits API でサポートされているAWS リージョンのドキュメントを追加しました。 APIs • の新しいフィルター `ListOpportunities` : YYYY-MM-DD 形式の `AfterTargetCloseDate` および `BeforeTargetCloseDate` フィールドを使用して、予想される終了日でフィルタリングの機会を有効にする `TargetCloseDate` フィルターを追加しました。	

変更	説明	日付
RC 4	• チャネル API の導入:AWS Partner Central に新しいチャネル API を追加し、パートナーがチャネル関係とpartner-to-partnerコラボレーションを管理できるようになりました。 • チャネル API リファレンスドキュメント: 使用可能なすべてのアクションとデータモデルを含む、チャネル API の包括的な API リファレンスドキュメントを追加しました。 • チャネル API アクセス許可: チャネル API に固有の IAM アクセス許可ポリシーとアクセスコントロールドキュメントを追加しました。 • チャネル API ログ記録: API の使用状況の監査と追跡に役立つ、チャネル API の口グ記録のサポートとドキュメントを追加しました。 • チャネル API 通知: チャネル API のイベント通知サポートが追加され、パートナーがチャネル関連のアクティビティに関するリアルタイムの更新を受信できるようになりました。 • Channel API クォータ: Channel API のサービス	2025-11-19

変更	説明	日付
	クォータドキュメントを追加しました。 • チャネル API のサンドボックステスト: チャネル API のサンドボックステスト機能を追加しました。これにより、パートナーは非本番環境でチャネルワークフローをテストできます。 • API 構造の再編成: Selling API と Channel API を区別するためにドキュメントを再編成しました。各 API のアクセス許可、ログ記録、通知、クォータ専用のセクションがあります。 • サポートされているリージョンのドキュメント: 販売 API とチャネル API の両方でサポートされているAWSリージョンのドキュメントを追加しました。 APIs	

変更	説明	日付
RC 3	• API アクセス許可ポリシーの更新: このリリースで導入された新しいアクションを反映するようにポリシーが更新されました。IAM ロールとアクセス許可がそれに応じて調整されていることを確認します。 • カスタムヘッダー使用状況の詳細: API 使用状況の監査と測定X-Amzn-User-Agentに役立つカスタムヘッダーの使用方法の詳細を追加しました。 • アクションを非同期アクションに置き換えました: を非同期アクション AcceptOpportunityEngagementInvitation に置き換えStartEngagementByAcceptingInvitationTask 、ペイロード形式が RC2 に関して変更されました。 • エンティティとアクションの名前変更: OpportunityEngagementInvitation を に変更しましたEngagementInvitation 。これらのアクションの属性も、新しいエンティティに合わせて変更されました。次のアクションが置き換えられました。	2024-10-17

変更	説明	日付
	• GetOpportunityEngagementInvitation が になりましたGetEngagementInvitation 。招待の複数の属性が変更、追加、または削除されました。 • ListOpportunityEngagementInvitations が になりましたListEngagementInvitations 。 • RejectOpportunityEngagementInvitation が になりましたRejectEngagementInvitation 。 • AssignOpportunity パラメータの更新: が Assigneeの代わりに を使用するAssignOpportunity ようになりましたAssigneeEmail 。それに応じて実装を更新します。 • SubmitOpportunity 機能の変更: SubmitOpportunity は非同期アクション によって実行されるようになりましたStartEngagementFro	

変更	説明	日付
	mOpportunityTask 。は InvolvementType とを受け入れるSubmitOpportunity ようになりましたVisibility 。 • 属性の名前変更と拡張: をEstimatedAwsMonthlyRevenue に変更しましたExpectedCustomerSpend 。には、 Frequency や などの新しい属性が含まれるExpectedCustomerSpend ようになりましたTargetCompany 。 • AWSオポチュニティ概要の新しいアクション: オポチュニティの可視性が、 という別のアクションを通じて提供されるようになりましたGetAwsOpportunitySummary 。 • オポチュニティを使用するためのワークフローの更新: オポチュニティを使用するためのワークフローが、このリリースで導入された新しいアクションを使用するように更新されました。 • オポチュニティを使用するためのワークフローの更新: オポチュニティを使用するためのワークフローが更新されEngagemen	

変更	説明	日付
	tInvitations 、AWS が提供する処理オポチュニティが改善されました。 • 更新を追跡するためのワークフローの更新: 更新を追跡するワークフローは、オポチュニティの更新を処理し、オポチュニティステータスのモニタリングの明確性と効率を高めるようになりました。 • アクションを一覧表示するための新しいフィルター: 新しいフィルターは ListSolutions および ListOpportunities アクションで使用できます。 • の場合ListSolutions 、新しいフィルターは FilterCategory 、FilterIdentifier 、および ですFilterStatus 。 • の場合ListOpportunities 、新しいフィルターは FilterCustomerCompanyName 、、FilterIdentifier 、FilterLastModifiedDate 、FilterLifeCycleApprovalStatus FilterLif	

変更	説明	日付
	eCycleStage 、および ですFilterOrigin 。 • AccessControl サブオブジェクトの削除: オポチュニティモデルのAccessControl サブオブジェクトが削除されました。NationalSecurity は最上位属性になりました。 • 属性が に再配置されましたGetAwsOpportunitySummary : AWSOpportunityTeam 、インサイト (EngagementScore 、NextBestAction)、オリジン、などの属性InvolvementType が、 GetAwsOpportunitySummary の代わりに から利用可能になりましたGetOpportunity 。 • InvolvementType フィールドの導入: コセルと可視性のみの機会を区別する InvolvementType フィールドを導入しました。 • イベント名の更新: 変更を反映するためにイベントがで更新されました。たとえば、作成されたオポチュニティエンゲージメントの招	

変更	説明	日付
	待が、エンゲージメントの招待が作成されました。 • レスポンスペイロードの更新: すべてのレスポンスペイロードに ID や ARN が含まれるようになりました。 • リクエストペイロードの更新: すべてのリクエストペイロードが ID または ARN を受け入れる入力として識別子を使用するようになりました。 • エンゲージメントエンティティの更新: エンゲージメントエンティティの title 属性の名前が に変更されましたEngagementTitle 。 • フィルタープレフィックスの削除: すべてのフィルターにフィルタープレフィックスが含まれなくなり、フィルタリングメカニズムが簡素化されました。 • StartEngagementFromOpportunity アクションの更新: 属性が に変更AwsInvolvement されましたAwsSubmission 。 • 招待プロジェクトの詳細の更新: の名前Invitation.ProjectDetails.TargetCompletionDate が に変更されまし	

変更	説明	日付
	たInvitation.ProjectDetails.EstimatedCompletionDate 。 • 収益見積もりの更新: Invitation.PartnerRevenueEstimate がになりましたExpectedCustomerSpend 。データ型が 1 つのオブジェクトから 1 つのオブジェクトを含む配列に変更されました。 • 受信者アカウントの更新: フィールドの名前ReceiverAccount が に変更されましたAccountReceiver 。 • エンゲージメント招待の新しい属性: SenderContact 属性がエンゲージメント招待に導入されました。 • Customer.Contact はオブジェクトではなく配列になりました。 PartnerAccountManager は OpportunityOwner としてサポートされる値ですBusinessTitle 。 • PartnerOpportunityTeam は OpportunityTeam という名前になりました。	

変更	説明	日付
	• APNPrograms は ApnPrograms になりました • エンゲージメントの招待ステータスが更新されました。 • ListEngagementInvitations が ParticipantType として を必要とするようになりましたRECEIVER。	

変更	説明	日付
RC 2	• OpportunityEngagementInvitation という新しいエンティティを導入しました。 • 新しいアクション「ListOpportunityEngagementInvitations」が導入されました。 • 新しいアクション「GetOpportunityEngagementInvitation」が導入されました。 • 新しいイベント「機会エンゲージメントの招待が作成されました」が導入されました。 • AcceptOpportunity を AcceptOpportunityEngagementInvitation に置き換えました。 • RejectOpportunity を RejectOpportunityEngagementInvitation に置き換えました。 • 「オポチュニティ承諾」を「オポチュニティエンゲージメント招待承諾」に置き換えました。 • 「機会の拒否」を「機会エンゲージメントの招待の拒否」に置き換えました。	2024-08-07

変更	説明	日付
	• PrimaryNeedsFromAWS PrimaryNeedsFromAws に変更されました。 • AWSOpportunityTeam が に変更されましたAwsOpportunityTeam 。 • AWSSalesLifeCycle を に変更しましたAwsSalesLifeCycle 。 • PrimaryNeedsFromAWSChangeReason が に変更されましたPrimaryNeedsFromAwsChangeReason 。 • AWSAccountId を に変更しましたAwsAccountId 。 • ExpectedMonthlyAWSRevenue を に変更しましたExpectedMonthlyAwsRevenue 。 • AWSFundingUsed が に変更されましたAwsFundingUsed 。 • AWSMarketplaceOffers を に変更しましたAwsMarketplaceOffers 。 • Country が に変更されましたCountryCode 。 • PartnerOpportunityContact を に変更しまし	

変更	説明	日付
	たPartnerOpportunity Team 。 • フィールドを Contact.Title に更新しましたContact.BusinessTitle 。 • フィールドを ContactInfo に更新しましたOwnerInfo 。 • AWSチームをマップからリストに変更しました。 • 受け入れられた Rejection Reasons のリストに関する詳細を追加しました。 • クライアントトークンの使用方法に関する情報が提供されました。 • UpdateOpportunity の使用方法の詳細を追加しました。 • AWS製品とパートナーソリューションの使用に関するガイダンスを追加しました。 • OpportunityEngagementInvitation エンティティに関する詳細を提供しました。 • StateOrRegion を更新し、承認された値のリストを改訂しました。 • パフォーマンスとエラーメッセージを改善するため	

変更	説明	日付
	に、複数のバグ修正を実装しました。	

変更	説明	日付
RC 1	• [ブレーク中] ApprovalStatus を ReviewStatus に変更: ApprovalStatus フィールドの名前を ReviewStatus に変更し、目的をより適切に反映しました。さらに、新しいステータス「送信保留中」を導入し、オポチュニティのドラフトを示します。ReviewStatus は、送信保留中、送信済み、レビュー中、承認済み、拒否済み、および必要なアクションの値を受け入れるようになりました。以前のドラフトステータスは、保留中の送信に置き換えられます。 • [ブレーク] SubmitOpportunity アクションの概要: 新しいアクションである SubmitOpportunity が導入されました。CreateOpportunity がオポチュニティも送信する現在の動作とは異なり、ソリューションまたはAWS製品をすぐにリンクすることなく、保留中の送信状態でオポチュニティを作成できるようになりました。送信を完了するには、AssociateOpportunity アクションを使用してソリューションまたは製品を	2024-06-21

変更	説明	日付
	リンクし、次に SubmitOpportunity を使用します。この分離により、API レスポンスのパフォーマンスが向上し、送信前の柔軟な準備フェーズが可能になります。 • [ブレーク] カタログパラメータの要件とサンドボックスエンドポイントの廃止: 7/30/2024 以降、「ガンマ」エンドポイント (partnercentral-selling-gamma.us-east-1.amazonaws.com) は使用できなくなります。すべての API アクションで、オペレーションがサンドボックスまたはAWSカタログで実行されるかどうかを指定する Catalog パラメータが必要になりました。通知ルールが environmentName ではなくカタログでフィルタリングされるようになりました。 • [ブレーク] OpportunityIdentifier を Identifier に名前変更: Actions AssignOpportunity、AcceptOpportunity、および RejectOpportunity が、UpdateOpportunity との整合性を保つために OpportunityIdentifier	

変更	説明	日付
	の代わりに Identifier を使用するようになりました。 • Enum 型から String for APNProgram CustomerUseCase、および UseCase への変更: Project.APNPrograms、Marketing.ActivityUseCases、および Project.CustomerUseCase を列挙型から文字列型に変換して、値の変更に伴うリスクを軽減します。これらのフィールドは、事前定義されたリストの値のみを受け入れますが、SDKs でネイティブには使用できません。 • エラー処理の強化: APIs内のエラー処理が大幅に改善され、デバッグが容易になり、問題を迅速に解決できるようになりました。新しいエラー処理は、エラーを 1 の 2 つの段階で構造化します。リクエストの検証に失敗しました: データ型と形式、または 2 をチェックします。ビジネス検証に失敗しました: ペイロード内のすべての問題が 1 つのレスポンスに一覧表示され、エラーのあるフィールドが指定されます。この統合されたフィードバックにより、	

変更	説明	日付
	エラーのトラブルシューティングと修正をより効率的に行うことができます。エラーコードと形式が標準化されました。 • クライアントソース識別子: パートナー向けの機能では、リクエストに X-Amzn-User-Agent ヘッダーを含めることで、トラフィックのソースを識別できます。[ソリューションプロバイダーを含むソリューション名] | [バージョン] 形式を使用する例:AWSパートナー CRM コネクタ | v2.0.1 • すでにデプロイされており、SDK に変更がないバグ修正] オポチュニティの作成とAWSアカウント ID: パートナーは、エラーが発生することなく、AWSアカウント ID を null から有効な値に変更しながらオポチュニティを起動できるようになりました。以前は、バグにより、パートナーはこれらのアクションを個別に実行できませんでした。 • すでにデプロイされているバグ修正、SDK への変更なし] ソリューションと新しいオポチュニティの関連付け: パートナーはソリュ	

変更	説明	日付
	ーションを新しく作成されたオポチュニティに関連付けることができます。以前は、パートナーが新しいオポチュニティを作成し、ソリューションを同時に関連付けたとき、ソリューション情報は失われていました。この問題は解決されました。 • すでにデプロイされているバグ修正、SDK への変更なし] オポチュニティ所有者の詳細表示: 予想される「Email」、LastName」、LastName」フィールドではなくFirstName 、「LastName」フィールドの組み合わせに所有者の詳細が表示されていた問題を修正しました。 • すでにデプロイされており、SDK に変更がないバグ修正] Customer.Account.WebsiteUrl フィールドオプション: AccessControls.NationalSecurity フィールドが「はい」の場合、Customer.Account.WebsiteUrl フィールドはオプションとなり、API の動作が既存の UI の動作と一致します。AccessControls.NationalSecurity フィールドが「いいえ」または null の	

変更	説明	日付
	場合、ビジネス検証としてWebsiteUrl フィールドが必要です。 • すでにデプロイされているバグ修正、SDK への変更なし〕コンサルティングパートナーのAWSアカウント ID 要件: コンサルティングパートナーのAWSアカウント ID フィールドがオプションとして表示された UI の不整合が解決されました。ドキュメントに従って、条件付き必須フィールドになりました。 • バグ修正、6/19 にデプロイ、SDK への変更なし] 共同販売機会のステージからクローズドロストへ: パートナーは、クローズドロスト理由が指定されていれば、共同販売機会のステージをクローズドロストに正常に更新できるようになりました。これにより、「Missing Required Field: Closed Lost Reason is required when closing the opportunity」という前のエラーが解決されます。 • ドキュメント] ドキュメントの改善: 変更を反映するためにドキュメントにいくつかの改善を実施しました。	

変更	説明	日付
	• 修正保留中の既知の問題と今後の機能のリスト: • ソリューションまたは製品で AssociateOpportunity アクションを実行している間、オファーからのエラーメッセージが正しく表示されません。 • サンドボックスで機会をリモートで作成し、AWS 検証プロセスをシミュレートする機能 • 郵便番号は、米国の XXXXX-XXXX 形式の場合に null を返します。 • 次のステップと次のステップ履歴は、「サポートは不要」とマークされたオポチュニティでは使用できませんAWS。 • Customer Business Problem の説明が 255 文字を超えると、過去の機会を取得できません。 • 送信が保留中のオポチュニティを削除する機能。 • LeadSource はデータモデルでは使用できません。	

変更	説明	日付
ベータ 5	• LifeCycle.ApprovalStatus を LifeCycle.ReviewStatus に変更 • Insights.ReviewComments を LifeCycle.ReviewComments に変更 • LifeCycle.ReviewStatusReason (新しい属性) を追加 • PartnerAcceptance.Status に新しい値「保留中」を追加 • dotnet、java、javascript、python、ruby SDKs をリリースしました。 • 上記の変更を反映するように API リファレンスガイドを更新しました。	2024-04-19

変更	説明	日付
ベータ 4	• サンドボックステスト: サンドボックスで機会イベント通知をテストする機能を追加しました。 • ドキュメントの追加: オポチュニティイベント通知のサンプルルールを導入しました。 • SDK リリースを追加: Java (V2)、Javascript (V2)、DotNet (V3)、Python (V3) 用の SDKs をリリースしました。 • 更新された日付形式: ISO 標準に準拠するように更新されました。	2024-03-04

変更	説明	日付
ベータ 3	• 変更ログファイルを追加: 更新と変更をより適切に追跡するための変更ログファイルを導入しました。 • API リファレンスガイドの更新 (PDF): API リファレンスガイドの新しいバージョンを PDF 形式でリリースしました。 • API リファレンスガイドの更新 (ウェブバージョン): API リファレンスガイドのウェブバージョンを更新しました。 • DotNet SDK を追加: DotNet SDK をリリースし、さまざまなプログラミング環境のサポートを拡張しました。 • Python Boto 3 SDK (V2): Python Boto 3 SDK のバージョン 2 をリリースし、新機能と改善点を導入しました。	2024-01-24
ベータ 2	• API リファレンスガイド (ウェブバージョン): API リファレンスガイド (ウェブバージョン) の新しいバージョンをリリースしました。注: このバージョンでは、適切な機能のために、抽出後にローカルウェブサーバーを実行する必要があります。	2024-01-07

変更	説明	日付
ベータ 1	• Boto3 (初期リリース): サービス用の Python SDK である Boto3 の初期リリースを開始しました。 • ベータプログラムガイドを追加: ベータテストプログラムの参加者向けのガイドを導入しました。 • API リファレンスガイド (PDF): API リファレンスガイドの最初のバージョンを PDF 形式でアップロードし、API の包括的なドキュメントを提供しました。	2023-12-15
ベータ 0	この初回リリースでは、AWS Partner Central のパートナーエンゲージメントと機会を管理および最適化するための一連の機能が提供されます。	2023-11-15

ドキュメント履歴

この API リファレンスのドキュメントリリースのリストを次に示します。

変更	説明	日付
Partner Revenue Measurement APIsのドキュメントを追加	AWSパートナーセントラル APIs を追加しました。	2026 年 6 月 30 日
ACE Resonate のドキュメントを更新しました	AWS Cosell Motion をサポートする GetAwsOpportunitySummary アクションのパートナー中央更新 API レスポンス。	2026 年 6 月 16 日
Partner Central MCP Server のドキュメントを追加	AWSパートナー中央エージェント MCP サーバーは、モデルコンテキストプロトコル (MCP) を通じてパートナー中央ツールを提供します。	2026 年 3 月 16 日
RC 5 リリース	Partner AWS Central API の候補 5 バージョンがリリースされました。AWS Partner Central に新しい Account API を導入し、パートナーがアカウント情報、登録、プロファイル、ドメイン管理、パートナー接続を管理できるようになりました。新しい Benefits API をAWS Partner Central に導入しました。これにより、パートナーは一元化されたインターフェイスを通じてAWS パートナープログラム全体でメリットを検出、申請、管理できます。リード管理ワー	2025 年 11 月 30 日

	クフローをサポートするように API エンゲージメントアクションの販売を強化しました。詳細なワークフローと統合パターンを含む Account、APIs の包括的な使用ガイドと API ドキュメントを追加しました。リードコンテキストを使用したエンゲージメント管理のための販売 API ドキュメントを強化しました。	
RC 4 リリース	Partner AWS Central API の候補 4 バージョンがリリースされました。チャネル関係とpartner-to-partnerコラボレーションを管理するチャネル API を導入しました。これにより、パートナーは Billing Transfer を使用してエンドユーザーと取引する際にプログラムのメリットを受けることができます。ドキュメントは、複数の AWS Partner Central APIs をサポートするために再編成されました。	2025 年 11 月 19 日
マルチパートナーの機会に関するドキュメントを追加	マルチパートナーオポチュニティ (プレビュー) は、AWS Partner Central の統合エクスペリエンスです。これにより、AWSパートナーは他のパートナーを見つけて接続し、顧客の取引で共同販売できます。	2024/12/04

RC 3 リリース	AWS Partner Central API の候補 3 バージョンがリリースされました	2024 年 10 月 17 日
RC 2 リリース	Partner AWS Central API の候補 2 バージョンがリリースされました	2024 年 8 月 7 日
RC 1 リリース	Partner Central API の候補 1 AWSバージョンがリリースされました	2024 年 6 月 21 日
ベータ 5 リリース	AWS Partner Central API のベータ 5 がリリースされました	2024 年 4 月 19 日
ベータ 4 リリース	AWS Partner Central API のベータ 4 がリリースされました	2024 年 3 月 4 日
ベータ 3 リリース	AWS Partner Central API のベータ 3 がリリースされました	2024 年 1 月 24 日
ベータ 2 リリース	AWS Partner Central API のベータ 2 がリリースされました	2024 年 1 月 7 日
ベータ 1 リリース	AWS Partner Central API のベータ 1 がリリースされました	2023 年 12 月 15 日
ベータ 0 リリース	AWS Partner Central API の初期リリース	2023 年 11 月 15 日