aws

开发人员指南

AWS 合作伙伴中心

AWS 合作伙伴中心: 开发人员指南

Table of Contents

AWS Partner Central API 参考已重新构建。有关支持的 API 操作的更多信息，请参阅 AWS Partner Central API 参考。

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。

欢迎来到AWS合作伙伴中心开发者指南

本开发人员指南介绍了AWS合作伙伴如何使用服务 API 来集成和管理构建、营销、销售和发展活动 AWS。

主题

使用AWS合作伙伴中心 API，其中包含AWS SDK

AWS软件开发套件 (SDK) 可用于许多流行的编程语言。每个软件开发工具包都提供 API、代码示例和文档，使开发人员能够更轻松地以其首选语言构建应用程序。

SDK 文档	代码示例
适用于 C++ 的 AWS SDK	适用于 C++ 的 AWS SDK代码示例
AWS CLI	AWS CLI代码示例
适用于 Go 的 AWS SDK	适用于 Go 的 AWS SDK代码示例
适用于 Java 的 AWS SDK	适用于 Java 的 AWS SDK代码示例
适用于 JavaScript 的 AWS SDK	适用于 JavaScript 的 AWS SDK代码示例
适用于 Kotlin 的 AWS SDK	适用于 Kotlin 的 AWS SDK代码示例
适用于 .NET 的 AWS SDK	适用于 .NET 的 AWS SDK代码示例
适用于 PHP 的 AWS SDK	适用于 PHP 的 AWS SDK代码示例
AWS Tools for PowerShell	AWS Tools for PowerShell代码示例
适用于 Python (Boto3) 的 AWS SDK	适用于 Python (Boto3) 的 AWS SDK代码示例
适用于 Ruby 的 AWS SDK	适用于 Ruby 的 AWS SDK代码示例
适用于 Rust 的 AWS SDK	适用于 Rust 的 AWS SDK代码示例

SDK 文档	代码示例
适用于 SAP ABAP 的 AWS SDK	适用于 SAP ABAP 的 AWS SDK代码示例
适用于 Swift 的 AWS SDK	适用于 Swift 的 AWS SDK代码示例

> **示例可用性**
>
> 找不到所需的内容？ 通过使用此页面底部的提供反馈链接请求代码示例。

设置和身份验证

使用AWS合作伙伴中心 API 进行设置和身份验证包括三个步骤。以下是该过程的概述：

1. 将您的AWS Marketplace卖家账户关联至合作伙伴中心。
2. 使用 IAM 设置权限。
3. 使用签名版本 4 (Sigv4) 对您的 API 调用进行身份验证。

链接你的AWS 账户前往合作伙伴中心

将您链接AWS 账户到合作伙伴中心是使用 API 的先决条件。有关更多信息，请参阅将AWS合作伙伴平台账户与AWS Marketplace卖家账户关联起来。您必须使用具有联盟领导权限或云管理员权限的账户登录合作伙伴中心，导航至该**`Account Linking`**部分，然后按照提示进行操作。

设置 IAM

要使用AWS合作伙伴中心 API，您需要一个AWS Identity and Access Management(IAM) 角色或 IAM 用户才能开始拨打电话。有关更多信息，请参阅何时使用 IAM？ 。请按照AWS 账户指南中创建 IAM 角色和创建 IAM 用户的步骤进行操作。您必须在您的合作伙伴 Central-linked AWS Marketplace卖家账户 Role/User 中创建此 IAM。 role/user 创建 IAM 不会产生任何费用。

1. 创建 IAM Role/User

   登录AWS 管理控制台，导航到 IAM 服务，然后按照步骤创建 IAM 角色或 IAM 用户。

2. 分配策略：

根据需要附加托管策略或创建自定义策略。要修改或扩展权限，请对 IAM 角色应用其他策略，而不是复制来自的内容并将其AWSPartnerCentralOpportunityManagement与其他权限合并。避免复制托管策略，因为这样做会使您无法在新功能发布时自动获得访问权限，并且将来您必须手动更新策略。有关访问策略的更多详细信息，请参阅访问控制文档。

管理AWS Marketplace优惠

要管理AWS Marketplace优惠并将其与机会关联起来，合作伙伴必须授予 IAM 角色访问目录 API 的权限。确保 role/user 具有权限，例如aws-marketplace:ListEntities和aws-marketplace:SearchAgreements。

对 API 调用进行身份验证

AWS合作伙伴中心 API 使用签名版本 4 (Sigv4) 进行身份验证。以下是实现它的方法：

使用AWS SDK

AWS SDK 会自动处理请求签名。提供您的AWS凭证，剩下的就交给 SDK。

1. 对于 Java，请参阅为适用于 Java 的AWS SDK 提供临时证书。
2. 对于 Python (Boto3)，请参阅凭证。
3. 对于 JavaScript (Node.js)，请参阅中的设置凭据 Node.js。
4. 有关.NET 的信息，请参阅凭据和配置文件解析。
5. 有关其他编程语言和更多示例，请参阅构建工具AWS。

不使用进行身份验证AWS SDK

如果您选择的编程语言没有可用的AWS SDK，则身份验证包括手动创建规范请求、对请求进行签名和处理会话令牌。AWS为使用 SigV4 签名提供了全面的指导。但是，请注意，建议使用AWS软件开发工具包，因为手动请求签名会增加复杂性，并且需要仔细管理安全令牌。

每个对 awsjson1_0 协议的请求都必须使用以下标头使用 HTTP “POST” 方法发送到根 URL (/)。

必需的标头

每个 AWSJson1_0 协议的 API 请求都必须使用以下标头使用 HTTP “POST” 方法发送到根网址 (/)。

标题	必需	说明
Content-Type	true	此标头的静态值为`application/x-amz-json-1.0` 。
Content-Length	true	由 RFC 9110 #section -8.6 定义的标准 Content-Length 标头。
X-Amz-Target	对于请求则为真	例如，服务操作`CreatePartner` 的值`PartnerCentralAccount` 为`PartnerCentralAccount.CreatePartner` 。

使用自定义用户代理对通话进行签名

在向 P AWS artner Central 提出 API 请求时，我们建议添加`X-Amzn-User-Agent`标题以帮助AWS识别客户端应用程序的来源、监控使用情况和审计绩效。AWS使用此标头来区分发出调用的客户端应用程序的类型，并收集有关不同客户端实现成功率的见解。

自定义用户代理标头

标题名称：`X-Amzn-User-Agent`

目的：区分发出 API 请求的客户端类型，对交互来源进行分类。

Format: `{Integrator}|{Product}`

示例值：`AWS|AWS Partner CRM Connector`

在每个请求中都包含此标头可以AWS分析请求模式、跟踪集成并改善不同客户端系统的 API 体验。

在 SDK 中使用自定义标头

要在 SDK 调用中包含`X-Amzn-User-Agent`标头，您可以在调用 API 之前修改客户端请求行为。以下是使用适用于 Python 的AWS SDK (Boto3) 的销售 API 示例：

```
import boto3

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

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

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

# Function to add the custom User-Agent header

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

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

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

此示例演示如何在 Boto3 SDK 中注册事件，以自动将`X-Amzn-User-Agent`标头附加到每个 API 请求。通过修改其各自的请求拦截机制，可以将同样的方法应用于其他AWS SDK。

> **Note**
>
> 标`X-Amzn-User-Agent`题格式已得到简化。我们建议使用新格式。为了向后兼容，仍支持以前的格式 (`CompanyName|ProductName|CRMName|ProductVersion`)。现有的集成无需修改即可继续运行。

合作伙伴中心代理 MCP 服务器

Partner Central 代理 MCP 服务器通过模型上下文协议 (MCP) 提供合作伙伴中心工具，使您的 AI 代理和工具能够通过自然语言发现机会管理、客户见解和融资计划并与之交互。

服务器负责身份验证、会话管理和写入操作的人机在环审批，在简化工作流程的同时保持控制。

概述

Partner Central 代理 MCP 服务器是一项完全托管的AWS托管服务，可将 MCP-compatible AI 客户端连接到合作伙伴中心代理。它使用 HTTPS 上的 JSON-RPC 2.0 和 Sigv4 身份验证，并支持 Server-Sent 事件 (SSE) 进行实时流媒体响应。

该服务器支持多轮对话、用于文档分析的文件附件，以及内置的批准工作流程，该工作流程需要在执行任何写入操作之前获得您的明确同意。

代理能力

- 渠道见解 — 获取有关销售渠道的对话情报，包括风险机会、阶段进展和已完成的分析
- 创造机会 — 通过自然语言对话、上传会议记录、提案或通话记录，或者通过克隆现有机会来创造新的机会
- 机会摘要 — 生成任何交易的简洁、一目了然的摘要，涵盖阶段、支出、截止日期等
- 销售策略生成 — 结合机会详情、行业背景和AWS解决方案建议，制定定制的销售策略
- 创建客户档案 — 使用涵盖行业、商业模式、地理位置和最新发展的公开信息生成公司简介
- 解决方案建议 — 根据机会要求 Cross-reference 注册的解决方案，以找到最匹配的解决方案
- 资金建议 — 根据可用AWS资金计划评估机会，估算金额并提出资金申请
- 下一步建议 — 以AWS共同销售标准和阶段进展指导为基础，制定按优先顺序排列的行动计划
- 机会进展 — 上传支持文档，提取相关数据，并通过渠道阶段推进机会
- AI-assisted 产品清单 — 利用您现有的数字资产生成高质量AWS Marketplace的产品清单，根据AWS Marketplace标准对列表强度进行评分，并获得现场级推荐以提高可发现性。有关更多信息，请参阅《AWS Marketplace卖家指南》中的AI-assisted 产品清单。

主要优势

- 通过对话方式访问合作伙伴中心 — 使用自然语言提问，无需浏览复杂的控制台工作流程

- 更多销售时间 — 通过简短的对话而不是填写多步骤表单来创造机会，这样可以减少数据输入，让合作伙伴销售团队将更多的时间花在销售上，代理商会建议改进措施，以便合作伙伴提交更高质量的机会并改善渠道卫生
- Human-in-the-loop 安全 — 所有写入操作在执行前都需要您的明确批准
- Multi-turn 对话 — 在会话中完善您的查询，而无需重复上下文
- 文件分析 — 附上文档（PDF、DOCX、XLSX、CSV、图片），供客服人员与您的问题一起分析
- 集中访问管理 — 通过具有细粒度权限的 IAM 策略控制访问权限
- 沙盒测试 — 在接触生产数据之前，在隔离的沙盒环境中测试工作流程
- 直播回复 — 在代理处理您的请求时通过 SSE 获取反馈

用法示例

所有 AI-generated 见解都包含用于可追溯性的会话 ID。数据与登录合作伙伴自己的机会隔离开来。所有内容都带有明确的披露标签，并受AWS负责任的人工智能政策的约束。

创造机会

代理根据自然语言描述或上传的文档（PDF、DOCX、XLSX、TXT）创建新的机会。它提取客户名称、项目范围、预计结束日期和其他必填字段，从公开的网络数据中丰富客户详细信息，根据 AWS 就绪要求验证草稿，并在您批准后通过合作伙伴中心销售 API 创造机会。

- “为 Acme Corp 创造机会 — 他们希望将数据仓库迁移到 Redshift，目标截止日期为第三季度末，预计每月 AWS 支出为 4 万美元”
- “这是我昨天与之会面的电话记录 GlobalTech ——用这份记录创造机会”
- “使用此计划书 PDF 为客户的 SAP 迁移创造机会”

机会克隆

代理在现有机会的基础上创建新的机会，合并您提供的新客户或项目详细信息，并在提交之前验证至少一个区分字段已更改，以便 AWS 审查不会拒绝重复项。

- “为新客户克隆机会 O1234567890 — Globex，工作负载相同，目标截止日期为季度末”
- “创造像 O9876543210 这样的机会，但要针对客户的生产环境而不是沙箱”

管道见解

获取有关您的销售渠道的对话情报。代理人分析阶段进展、截止日期、停滞不前的交易和已完成的亏损模式，以揭示最重要的事情。

- “本周有哪些机会需要我关注？”
- “下个月有多少机会要关闭？”
- “在过去的6个月中，我们失去机会的主要原因是什么？”

机会摘要

该代理将公司名称、行业、阶段、预计每月AWS支出、目标截止日期和其他关键细节综合成一个简洁的摘要，无需扫描单个表单字段。

- “给我一个机会摘要 O1234567890”
- “Acme Corp交易的现状和关键细节如何？”

销售游戏生成

代理商通过将机会的详细信息、客户的行业背景和相关的AWS解决方案建议整合到一个随时可用的销售计划中，来制定定制的销售策略。

- “为机会生成销售策略 O1234567890”
- “向该金融服务客户推销云迁移的最佳方法是什么？”
- “为我制定 GlobalTech 数据分析交易的销售策略”

创建客户档案

代理商使用公开可用的信息生成公司简介，包括行业分类、商业模式 (B2B/B2C/hybrid)、地理分布、公司规模、市场重点和最近的业务发展。个人资料标有 “使用公开数据和AWS人工智能见解生成”。

- “为 Acme Corp 创建客户档案”
- “我们对这个客户的行业和商业模式了解多少？”
- “为我即将与之会面整理一份公司概述 GlobalTech”

解决方案推荐

代理会根据机会要求交叉引用您注册的解决方案，显示解决方案名称、描述以及该解决方案是否已附加到机会中。

- “我们的哪个解决方案最适合机会 O1234567890？”
- “我们的数据分析解决方案是否已经附属于这笔交易？”
- “为想要迁移 SAP 工作负载的客户推荐解决方案”

资金建议

代理人根据机会详情和计划资格标准，根据可用的AWS资助计划对每个机会进行评估。找到匹配项后，它会显示程序名称、描述和详细理由。然后，您可以估算资金金额、创建自动填充的资金申请草稿或通过对话了解有关计划的更多信息。SCA（战略合作协议）预算可用性会在相关时浮出水面，并且所有操作都尊重 IAM 权限。

- “我是否有资格获得机会 O6789012345 的任何资助计划？”
- “估算该客户的 POC 的资金金额”
- “为这个机会创建 MAP 福利申请”

下一步建议

代理人根据阶段进展指导评估机会，将当前数据与合格机会的标准进行比较，并准确确定您还需要收集哪些信息。AWS结果是制定了基于AWS共同销售标准的优先行动计划。

- “为了提前获得机会 O1234567890，接下来我需要做什么？”
- “这个机会准备好提交了吗？ 缺少哪些字段？”
- “将这笔交易从 “潜在客户” 转为 “合格” 有哪些要求？”

机会进展

代理接受支持文件（会议记录、电话记录、电子邮件摘要），提取相关信息，将其映射到机会字段，评估阶段要求并更新机会。如果仍存在差距，则会返回已满足需求与未满足需求的明细表。

- “这是我的电话记录 — 使用相关细节更新机会 O1234567890”
- “我附上了会议记录。根据我们讨论的内容，推进这个机会。”

- “查看此电子邮件摘要，告诉我它满足了哪些机会字段”

合作伙伴入职的代理经验

代理可以自动将合作伙伴加入合作伙伴中心，并通过自然语言为Marketplace卖家设置和PRM合规提供指导性帮助。

代理能力：

- 合作伙伴资料自动化 — 扫描您的网站，自动填充您的合作伙伴资料，并管理知名度、联盟负责人联系人、培训域名和账户连接
- 卖家设置 Step-by-step 指南 — Marketplace 卖家注册指南，包括纳税申报表、银行、合规 (KYC/BAV/SU)、ESC 目录和服务相关角色
- PRM 合规指南 — 评估 PRM 准备情况，检索用于收入标签的产品代码，并验证子公司账户关联以实现合并收入归因

合作伙伴资料自动化

代理会扫描您的公司网站以提取并自动填充您的合作伙伴档案，然后引导您填补任何剩余的空白。它可以更新个人资料字段、更改知名度、管理您的联盟负责人联系人、关联培训认证域以及处理账户连接邀请。

- “你能看看我的网站并填写我的个人资料吗？”
- “公开我们的个人资料，这样 AWS 客户就可以找到我们”
- “将我们的联盟牵头联系人更新为 [姓名]，地址为 [email]”
- “Connect 我们的欧洲、中东和非洲子公司账户”

卖家设置指南

代理人端到端地完成 Marketplace 卖家注册，根据您的国家和实体类型确定正确的纳税申报表，解释按地区划分的银行和支付设置，并确定适用于您的合规要求（KYC、银行账户验证、二级用户验证）。

- “我想开始在 Marketplace 上销售商品，我该从哪里开始？”
- “我需要什么税表？ 我是德国的一家公司”
- “我需要KYC吗？ 我要向法国的买家销售商品”
- “如果我在美国境外，如何设置付款？”

PRM 合规指南

代理会检查您的账户是否已连接 PRM-ready ——验证您的账户是否已连接、是否存在列表以及您的产品代码是否可供标记。对于拥有子公司账户的合作伙伴，它会验证所有卖家账户是否已正确关联以进行合并收入归属。

- “我是否符合 PRM 标准？”
- “我在哪里可以找到用于标记的产品代码？”
- “告诉我为了符合 PRM 要求我需要执行的所有步骤”

开始使用

准备好设置合作伙伴中心代理 MCP 服务器了吗？ 有关分步设置说明，请前往入门指南。

合作伙伴中心代理 MCP 服务器入门

本指南将引导您使用自定义 MCP 客户端设置对合作伙伴中心代理 MCP 服务器的编程访问权限。服务器使用带有 Sigv4 身份验证的直接 HTTPS，无需代理或 IDE 插件。

先决条件

开始之前，请确保您已具备以下条件：

- 有效的合作伙伴中心账户（已迁移到AWS控制台）
- 拥有合作伙伴中心的 IAM 权限的AWS账户
- AWS CLI 已安装并配置了凭据
- 访问 us-east-1（弗吉尼亚北部）区域
- 通过 HTTPS 连接到 `partnercentral-agents-mcp.us-east-1.api.aws`
- 你的 HTTP 客户端支持 TLS 1.2+
- 支持 JSON-RPC 2.0 和 Sigv4 请求签名的 MCP-compatible 客户端

步骤 1：设置 IAM 权限

Partner Central Agent MCP 服务器需要两个级别的 IAM 权限：协议访问权限（用于与 MCP 端点通信）和数据访问权限（用于执行合作伙伴中心操作）。

附加 IAM 策略

要使用AWS管理控制台将策略附加到您的 IAM 身份，请执行以下操作：

1. 打开 IAM 控制台。
2. 在左侧导航窗格中，根据要将策略附加到的身份选择用户、用户组或角色，然后选择特定用户、组或角色的名称。
3. 选择权限选项卡。
4. 选择附加策略（如果是第一次，则选择添加权限）。
5. 在策略列表中，搜索并选择要附加的托管策略（例如，您根据以下 JSON 块创建的自定义策略）。
6. 选择 “附加策略”（或选择 “下一步”，然后选择 “添加权限”）进行确认。

权限立即生效。您可以将多个策略附加到同一个身份。

推荐：使用托管策略

授予 MCP 协议访问权限的最简单方法是将`AWSMcpServiceActionsFullAccess`托管策略附加到您的 IAM 身份。此策略包括与 MCP 服务器交互所需的所有权限。

为了实现精细控制，您可以使用 IAM 策略中的`aws:IsMcpServiceAction`条件密钥来确定权限范围，专门针对 MCP 服务操作。

MCP 协议访问的最低权限

您的 IAM 身份至少需要此操作才能与 MCP 服务器进行交互：

处理建议	说明
`partnercentral:UseSession`	创建、更新和检索对话会话所必需的

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

```
            "Resource": "*",
   "Condition": {
                "Bool": {
                    "aws:IsMcpServiceAction": "true"
                }
            }
        }
    ]


}
```

数据访问权限

要通过代理实际执行 Partner Central 操作，您需要根据自己的用例获得额外的权限。

机会管理：

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

资助计划：

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

市场访问权限：

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

合作伙伴中心入职培训：

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

在 Marketplace 卖家设置中，还要添加：

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

对于与服务相关的角色管理（转售 Authorizations/CPPO）：

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

完全访问策略

对于开发和测试，您可以将所有权限合并到一个策略中：

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

Read-only 政策

对于生产环境或只读用例，请限制读取操作的权限：

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

第 2 步：连接您的 MCP 客户端

合作伙伴中心代理 MCP 服务器使用带有 Sigv4 请求签名的直接 HTTPS。没有代理层 — 您的 MCP 客户端直接向端点发送 JSON-RPC 2.0 请求。

端点

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

身份验证

所有请求都必须使用签AWS名版本 4 进行签名，使用以下命令进行签名：

- 服务名称：`partnercentral-agents-mcp`
- 区域：`us-east-1`

初始化 MCP 连接

发送建立协议的`initialize`请求：

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

预期的回应：

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

列出可用工具

```
{
    "jsonrpc": "2.0",
    "id": 2,
```

```
    "method": "tools/list",
    "params": {}
}
```

使用 MCP 标头对通话进行签名

在向 Partner Central 代理 MCP 提出请求时，我们建议使用以下方法添加自定义 MCP 标头，以帮助AWS识别客户端应用程序的来源、监控使用情况和审计性能。AWS使用此标头来区分发出调用的客户端应用程序的类型，并收集有关不同客户端实现成功率的见解。

方法 1：_meta 字段 (programmatic/stateless MCP)

对于直接构造 MCP `tools/call` 请求的代码，请在请求上提供`_meta`字段。

```
{
    "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（基于会话的自定义代理）

对于建立会话的自定义 MCP 客户端，请在字段内提供 MCP 标头信息：`clientInfo`

```
{
    "method": "initialize",
```

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

中的字段clientInfo：

- integrator— 公司名称或合作伙伴的 “直接”

  示例：AWS
- sourceProduct— Product/agent 名字

  示例：AWS CRM Connector

方法 3：网址参数（仅限托管 MCP）

仅适用于集成商无法修改协议字段的托管 MCP 客户端。使用 URL 参数：

服务器网址：https://mcp.partnercentral.aws?appId=<Integrator's Company Name / Direct>

步骤 3：验证设置

发送一条简单的消息以确认一切正常。使用Sandbox目录进行测试：

```
{
    "jsonrpc": "2.0",
    "id": 3,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "content": [
                {
                    "type": "text",
                    "text": "Hello, what can you help me with?"
                }
```

```
            ],
            "catalog": "Sandbox"
        }
    }
}
```

如果您收到代理的回复"status": "complete"和短信回复，则说明您的设置工作正常。回复中还将包括sessionId可用于发送后续消息的。

步骤 4：运行第首批任务

查询您的机会

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

查看资金资格

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

```
        "arguments": {
            "sessionId": "session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
            "content": [
                {
                    "type": "text",
                    "text": "Am I eligible for MAP funding for opportunity
 O1234567890?"
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

检索会话历史记录

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

合作伙伴入职

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

其他可以尝试的入门任务：

- “指导我完成税务调查流程”
- “你能看看我的网站并填写我的合作伙伴资料吗？”
- “我还需要做些什么才能做好在 Marketplace 上销售商品的准备？”

安全注意事项

- 不要通过 MCP 工具参数传递AWS凭证。身份验证由传输层的 SigV4 请求签名处理。
- 使用沙盒目录进行测试和开发。该"Sandbox"目录提供了一个不影响生产合作伙伴数据的隔离环境。
- 在生产环境中应用最低权限 IAM 策略。使用只读策略来监控和报告用例。仅在用户需要更新机会或提交资金申请时才授予写入权限。
- 仔细检查写入操作。服务器对所有写入操作都使用人工在环审批。当提出写入操作时，请在批准之前查看参数。
- 会话数据是暂时的。会话将在创建 48 小时后过期。不要依赖会话进行长期数据存储。
- 文件上传到临时的 S3 存储桶。上传的文件是临时存储的，不会永久保留。请勿上传包含凭证、机密或其他敏感信息的文件。

后续步骤

- 配置参考 — 终端节点、IAM 操作、会话管理和错误代码的完整参考
- 工具参考 — `sendMessage` 和`getSession`工具的详细文档

配置引用

本页提供了连接和配置 Partner Central Agent MCP 服务器的完整参考。

端点

属性	值
URL	`https://partnercentral-agents-mcp.us-east-1.api.aws/mcp`
Region	`us-east-1` 仅限（弗吉尼亚北部）
协议	JSON-RPC 2.0 通过 HTTPS
流式传输	Server-Sent 活动 (SSE)
身份验证	AWS签名版本 4
sigv4 服务名称	`partnercentral-agents-mcp`
TLS 要求	TLS 1.2 或更高版本

IAM 权限

除非另有说明，否则所有 IAM 操作都使用`partnercentral:`前缀。

MCP 协议访问

授予 MCP 协议访问权限的最简单方法是将`AWSMcpServiceActionsFullAccess`托管策略附加到您的 IAM 身份。此策略包括与 MCP 服务器交互所需的所有权限。为了实现精细控制，您可以使用 IAM 策略中的`aws:IsMcpServiceAction`条件密钥来确定权限范围，专门针对 MCP 服务操作。

处理建议	说明
`partnercentral:UseSession`	创建、更新和检索对话会话

机会管理

处理建议	说明
partnercentral:List*	列出机会、解决方案和其他合作伙伴中心资源
partnercentral:Get*	检索个人机会和其他资源的详细信息
partnercentral:CreateOpportunity	创造机会
partnercentral:UpdateOpportunity	修改机会字段（阶段、截止日期、收入等）
partnercentral:SubmitOpportunity	提交机会进行AWS审核
partnercentral:AssignOpportunity	向合作伙伴代表分配机会
partnercentral:AssociateOpportunity	将机会链接到其他资源（解决方案等）
partnercentral:DisassociateOpportunity	移除机会与其他资源之间的链接
partnercentral:StartEngagementFromOpportunityTask	提交机会以从机会开始互动

资助计划

处理建议	说明
partnercentral:ListBenefitAllocations	列出您账户的可用福利分配
partnercentral:ListBenefitApplications	列出已提交的福利申请
partnercentral:CreateBenefitApplication	创建新的福利申请（MAP、POC、WMP）

处理建议	说明
partnercentral:GetBenefitApplication	检索特定福利申请的详细信息
partnercentral:UpdateBenefitApplication	更新待处理的福利申请
partnercentral:AssociateBenefitApplicationResource	将资源链接到福利申请
partnercentral:DisassociateBenefitApplicationResource	从福利申请中移除资源链接

市场访问权限

处理建议	说明
aws-marketplace:DescribeEntity	检索商城实体的详细信息
aws-marketplace:SearchAgreements	搜索市场协议
aws-marketplace:ListEntities	列出市场实体

入职代理

合作伙伴账户管理 (**partnercentral**)：

处理建议	说明
partnercentral:ListPartners	列出该账户的合作伙伴注册情况
partnercentral:GetPartner	检索合作伙伴注册和个人资料详情
partnercentral:GetProfileVisibility	检索当前的个人资料可见性 (PUBLIC/PRIVATE)

处理建议	说明
partnercentral:GetAllianceLeadContact	检索联盟负责人联系信息
partnercentral:GetProfileUpdateTask	检查待处理的异步配置文件更新的状态
partnercentral:StartProfileUpdateTask	提交合作伙伴资料更新（异步）
partnercentral:CancelProfileUpdateTask	取消待处理的个人资料更新任务
partnercentral:PutProfileVisibility	将个人资料可见性更改为公开或私人
partnercentral:PutAllianceLeadContact	更新联盟牵头联系人（姓名、职务、电子邮件）
partnercentral:SendEmailVerificationCode	为基于电子邮件的更改发送 domain/contact 验证码
partnercentral:AssociateAwsTrainingCertificationEmailDomain	链接电子邮件域名以进行培训认证跟踪
partnercentral:DisassociateAwsTrainingCertificationEmailDomain	移除链接的培训认证电子邮件域名
partnercentral:GetAccountConnections	列出活跃的账户连接（例如子公司链接）
partnercentral:GetConnectionInvitations	列出待发送和已接收的连接邀请
partnercentral:CreateConnectionInvitation	向其他AWS账号发送连接邀请

处理建议	说明
partnercentral:CancelConnectionInvitation	取消待处理的传出连接邀请
partnercentral:RespondConnectionInvitation	接受或拒绝传入的连接邀请
partnercentral:CancelConnection	取消活跃账户连接
partnercentral:ManageConnectionPreferences	获取或更新关联账户之间的共享偏好

Marketplace 卖家设置 (**aws-marketplace**)：

处理建议	说明
aws-marketplace:ListEntities	列出 Marketplace 实体（例如卖家实体）
aws-marketplace:DescribeEntity	检索 Marketplace 实体的完整详情
aws-marketplace:DescribeChangeSet	检查已提交的更改集的状态
aws-marketplace:StartChangeSet	提交 Marketplace 变更集（例如，卖家资料更新）

IAM — 服务相关角色 (**iam**)：

处理建议	说明
iam:GetRole	检查 Marketplace 转售授权服务相关角色是否存在
iam:CreateServiceLinkedRole	创建 AWSServiceRoleForMarketplaceResaleAuthorization 角色

目录环境

目录	说明
"AWS"	生产环境。所有操作都会影响实时合作伙伴数据。
"Sandbox"	隔离的测试环境。对生产数据没有影响。用于开发和验证。

会话管理

属性	值
会话 ID 格式	带session-前缀的 UUID v4（例如）session-550e8400-e29b-41d4-a716-446655440000
会话创建	首次sendMessage 通话时自动通话，不用sessionId
会话到期	会话创建后 48 小时（绝对到期，不基于非活动状态）

文件上传

属性	值
每封邮件的最大文件数	3
图像大小限制	3.75 MB
文档大小限制	4.5 MB
允许的扩展	doc, docx, pdf, png, jpeg, xlsx, csv, txt

属性	值
S3 存储桶（生产）	aws-partner-central marketplace-phemeral-writeonly
上传路径	s3://{bucket}/{aws-account-id}/
S3 网址要求	必须包含`versionId` 查询参数

上传工作流程

1. 将您的文件上传到AWS账户 ID 前缀下的相应的 S3 存储桶。
2. 记下包括上传`versionId`返回的 S3 URI。
3. 将该文件作为`document`内容块包含在`sendMessage`通话中。

S3 URI 格式示例：

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

错误代码

代码	Name	说明
-32001	AUTHENTICATION_FAILURE	Sigv4 签名无效或证书已过期
-31004	TOOL_PERMISSION_DENIED	IAM 身份缺少此`partnercentral:` 操作所需的操作
-32002	ACCESS_DENIED	常规访问被拒绝（账号未注册、区域不匹配等）
-32004	LIMIT_EXCEEDED	已超过速率限制或配额。使用指数退避重试。

代码	Name	说明
-30001	RESOURCE_NOT_FOUND	请求的资源（会话、机会等）不存在
-32600	INVALID_REQUEST	JSON-RPC 请求格式错误或参数无效
-32603	INTERNAL_ERROR	Server-side 错误。重试请求。

速率限制

服务器强制执行每个账户的速率限制：

操作	持续汇率	突增
sendMessage	每分钟 2 个请求	10
所有其他操作	每分钟 10 个请求	20

如果超过这些限制，服务器将返回错误代码 -32004 (LIMIT_UXCHERED)。在客户端中使用抖动实现指数级退避，以优雅地处理速率限制。

直播 (SSE) 事件类型

当stream在sendMessage调用true中设置为时，服务器会返回具有以下 Server-Sent 事件类型的事件：

事件类型	说明
stream_start	直播连接已建立
assistant-response.start	代理已开始生成响应
assistant-response.delta	代理响应的增量文本块
assistant-response.completed	代理已完成响应的生成

事件类型	说明
server-tool-use	代理正在调用内部工具（读取操作）
server-tool-response	内部工具调用的结果
tool_approval_request	代理正在请求人工批准写入操作
stream_end	直播连接关闭
done	最后一个事件，表示 SSE 直播已完成

后续步骤

- 工具参考 — sendMessage 和getSession工具的详细文档

工具参考

Partner Central Agent MCP 服务器公开了两个 MCP 工具：sendMessage用于所有代理交互和getSession用于检索会话状态。Partner Central 的所有操作（机会查询、资金申请、文件分析）均通过自然语言处理sendMessage。

工具概述

工具	说明	类别
sendMessage	向合作伙伴中心 AI 代理发送消息。支持文本、文件附件和人工审批响应。	读/写
getSession	检索会话状态，包括对话历史记录、事件和元数据。	Read-only

####

所有 Partner Central AI 代理互动的主要工具。使用此工具可以提问、请求操作、附加文档以供分析，以及回应写入操作的批准请求。

代理在会话中维护对话上下文，因此您可以提问后续问题，而无需重复之前的上下文。

参数

- content（必填）-内容块数组。每个区块必须包含一个用于确定区块结构的type字段。您可以在一封邮件中包含多个方块（例如，文本 + 文档附件）。

  内容区块类型：

Type	字段	说明
text	type（必填）、text（必填）	发送给代理的用户消息文本
document	type（必填）、filename（必填）、s3Uri（必填）	文件附件供代理分析。s3Uri必须包含versionId 参数。
tool_approval_response	type（必填）、toolUseId （必填）、decision（必填）、message（可选）	回应人性化批准请求

- catalog（必填）-操作的目标环境。

  有效值："AWS"（生产）、"Sandbox"（测试）

- sessionId（可选）— UUID v4 用于标识要继续的现有会话。省略创建新会话。格式：session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx。

  默认：自动创建新会话。

- stream（可选）— 启用 Server-Sent 事件 (SSE) 流式传输以实现实时响应交付。

  有效值：true、false

  默认值：false

响应

响应包括以下内容：

字段	说明
sessionId	后续消息的会话标识符
status	响应状态："complete" "requires_approval" 、或 "error"
content	来自代理的响应内容块数组

示例

基本短信（新会话）

请求:

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

响应:

```
{
    "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"
    }
}
```

Follow-up 消息（现有会话）

请求:

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

文件附件

先将文档上传到 S3，然后在消息中引用该文档：

```
{
```

```
    "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"
        }
    }
}
```

文件上传限制：

- 每封邮件最多 3 个文件
- 图像大小限制：3.75 MB
- 文件大小限制：4.5 MB
- 允许的扩展名：docdocx、pdf、png、jpeg、xlsx、csv、txt
- 必须将文件上传到 s3://{bucket}/{your-aws-account-id}/
- S3 URI 必须包含versionId查询参数

Human-in-the-loop 批准工作流程

当代理需要执行写入操作（例如，更新机会、提交资金申请）时，它会返回包含建议操作详细信息的"requires_approval"状态。您必须使用tool_approval_response内容块进行响应。

步骤 1-代理请求批准：

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

步骤 2-批准操作：

```
{
    "jsonrpc": "2.0",
    "id": 5,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "tool_approval_response",
                    "toolUseId": "tool-use-98765",
                    "decision": "approve"
```

```
                }
            ],
            "catalog": "AWS"
        }
    }
}
```

步骤 2（备选）-拒绝操作：

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

第 2 步（备选）-使用自定义响应进行覆盖：

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

批准决策值：

决策	行为
"approve"	使用建议的参数执行工具
"reject"	请勿执行该工具。message（可选）解释了原因。
"override"	通过以下方式提供自定义响应或修改后的指令 message

通过 SSE 进行直播

启用流媒体功能，以便在代理处理您的请求时接收增量响应块：

请求:

```
{
    "jsonrpc": "2.0",
    "id": 6,
    "method": "tools/call",
    "params": {
        "name": "sendMessage",
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "content": [
                {
                    "type": "text",
                    "text": "Analyze my pipeline and identify opportunities at risk"
```

```
                }
            ],
            "catalog": "AWS",
            "stream": true
        }
    }
}
```

服务器以 SSE 事件流进行响应：

```
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: {}
```

####

检索对话会话的当前状态，包括完整的对话历史记录、事件和元数据。使用它来检查会话状态、查看过去的互动或恢复对话。

参数

- `sessionId`（必填）— 要检索的会话的 UUID。格式：`session-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`。
- `catalog`（必填）-会话所属的环境。

  有效值："AWS"、"Sandbox"

响应

字段	Type	说明
`sessionId`	字符串	会话标识符
`createdAt`	字符串	ISO 8601 会话创建的时间戳
`lastActivity`	字符串	ISO 8601 上次活动的时间戳
`sequenceNumber`	integer	当前事件序列号
`stateType`	字符串	当前会话状态
`events`	array	完整的对话历史记录（用户消息、代理回复、工具使用）
`variables`	object	会话变量和元数据
`eventCount`	integer	会话中的事件总数

示例

请求:

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

```
        "arguments": {
            "sessionId": "session-550e8400-e29b-41d4-a716-446655440000",
            "catalog": "AWS"
        }
    }
}
```

响应:

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

错误处理

所有错误都遵循 JSON-RPC 2.0错误格式：

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

the section called “错误代码”有关错误代码及其含义的完整列表，请参阅。

推荐的重试策略

- 对于 -32004 (LIMIT_UXCHERED)：从 1 秒开始使用指数退避重试
- 对于 -32603 (INTERNAL_ERROR)：使用指数退避最多重试 3 次
- 对于-32001（AUTHENTICATION_FAILURE）：刷新凭据并重
- 对于所有其他错误：不要自动重试 — 请检查错误消息并更正请求

关于AWS合作伙伴中心 API

以下各节提供有关 API 的一般信息。

主题

使用AWS合作伙伴中心 API

以下部分提供有关使用AWS合作伙伴中心 API 的信息。

主题

使用AWS合作伙伴中心账户 API

管理合作伙伴账户

合作伙伴账户已注册并与现有AWS账户关联，以提供完整的 IAM-based 体验。这种方法无需用户级证书，确保所有注册和操作均通过账户内的 IAM 实体进行管理。AWSAWS Partner Central 支持与AWS服务的无缝集成，此模式提供合作伙伴实体管理，支持多个目录，并建立AWS账户级授权系统。

合作伙伴可以利用可用的合作伙伴账户 API 来注册、管理和更新其账户：

合作伙伴注册 API

合作伙伴可以使用其账户注册其AWS账户进行合作伙伴互动。使用 Create API，合作伙伴将提供所需的联盟牵头人信息，接受 APN 条款并提供唯一的法定名称。

合作伙伴档案 API

合作伙伴通过 public/private 可见性控制、异步验证和状态跟踪来管理包含公司详细信息（名称、描述、网站、徽标）的个人资料，允许一次更新一次。

域名管理 API

合作伙伴可以通过电子邮件验证注册和验证业务域，以建立组织身份并关联员工的培训和认证。

合作伙伴连接 API

合作伙伴出于各种目的将其AWS账户与其他AWS账户关联起来，例如与其他合作伙伴合作进行共同销售、在账户之间共享AWS Marketplace Offers 数据、授权 distributors/channel 合作伙伴 distribute/re销售其产品等。

合作伙伴验证 API

合作伙伴验证是注册合作伙伴账户的必备条件。合作伙伴必须先完成业务验证和身份验证流程，然后才能创建合作伙伴账户。这些验证可验证企业是否合法注册，并确认注册AWS账户的人的身份。

主题

处理合作伙伴注册

合作伙伴注册

合作伙伴可以使用其账户注册其AWS账户进行合作伙伴互动。使用 Create API，合作伙伴将提供所需的联盟牵头人信息，接受 APN 条款并提供唯一的法定名称。

注册期间

1. 同步创建合作伙伴实体 — 客户通过调用 Create API 启动合作伙伴注册流程，该API执行输入验证并在同一请求中创建合作伙伴实体。
2. 联盟负责人信息要求 — 在合作伙伴注册期间，客户必须为AWS合作伙伴网络 (APN) 相关的沟通提供联盟负责人联系信息。
3. 条款和条件执行 — 客户在注册时必须接受 APN 条款和条件。必须接受才能完成注册并访问任何功能。这些条款适用于所有 APN 计划的优势和功能。
4. Tag-On-Create Su@@ pp ort — 作为AWS一致授权体验 (CAE) 和 Tag-Based 授权的一部分，客户可以在创建合作伙伴实体时添加标签。
5. 法律 Name-Based 验证 — 注册将根据合作伙伴的法定名称强制实现唯一性，确保同一实体下不会重复注册。

注册后

1. Tag Management Supp ort — 注册后，合作伙伴可以为现有合作伙伴实体标记、取消标签和列出标签。
2. 获取合作伙伴实体-创建合作伙伴实体后，客户可以使用列表 API 检索合作伙伴 ID，然后使用 Get API 获取特定合作伙伴实体的详细信息。

API 摘要

1. CreatePartner API：客户通过调用 API 启动合作伙伴注册流程， CreatePartner API 会执行输入验证并在同一请求中创建合作伙伴实体。
2. ListPartners API：注册后，客户可以使用列表 API 来检索合作伙伴实体列表。
3. GetPartner API：注册后，客户可以使用 Get API 来检索特定合作伙伴实体的详细信息。
4. PutAllianceLeadContact API：更新主要联盟负责人联系信息。此操作允许合作伙伴在保持组织连续性的同时，转移主要联盟牵头人的称号或修改联系方式。
5. GetAllianceLeadContact API：更新主要联盟负责人联系信息。此操作允许合作伙伴在保持组织连续性的同时，转移主要联盟牵头人的称号或修改联系方式。

使用合作伙伴档案

配置文件管理

合作伙伴通过 public/private 可见性控制、异步验证和状态跟踪来管理包含公司详细信息（名称、描述、网站、徽标）的个人资料，允许一次更新一次。

1. 个人资料信息 — 合作伙伴资料必须包含基本的公司详细信息，例如显示名称、描述、网站 URL、徽标 IndustrySegments 和 PrimarySolutionType。
2. 多语言 Support — 合作伙伴资料支持多种语言。配置文件的每个本地化版本都包含一个语言环境属性。
3. 个人资料可见性控制 — 合作伙伴可以配置其个人资料可见性（公开或私密）。
4. 个人资料存储-个人资料信息将存储在合作伙伴账户对象中。
5. 个人资料状态管理-配置文件更新在发布前会进行异步验证。合作伙伴可以查看最新的个人资料更新状态（正在进行中、成功、失败和已取消）。只有经过批准的个人资料才会公开显示。
6. 请求并发控制-一次只允许一个配置文件更新请求。如果个人资料更新已在进行中，则客户必须先通过 CancelPartnerProfileUpdateTask API 明确取消正在进行的更新，然后才能启动新的更新。
7. 取消更新任务-合作伙伴可以通过调用 CancelPartnerProfileUpdateTask API 来取消正在进行的个人资料更新。只有在更新处于 IN_PROGRESS 状态时才允许取消，并且必须在开始新的更新之前完成。

API 摘要

1. StartPutProfileTask API：此 API 接受合作伙伴资料更新并执行基本的同步输入验证。
2. GetProfileUpdateTask API：此 API 允许合作伙伴获取当前的个人资料更新状态。它旨在在客户通过 StartPutPartnerProfile API 开始更新个人资料后使用，使合作伙伴能够检查其请求是处于 “进行中”、“失败”、“已成功” 还是 “已取消”。
3. CancelProfileTask API：允许合作伙伴明确取消当前处于 IN_PROGRESS 状态的个人资料更新任务。这包括自动审查和手动审查阶段。取消申请后，合作伙伴无需等待当前审核流程完成即可启动新的个人资料更新。
4. PutProfileVisibility API：通过调用此 API，合作伙伴可以控制个人资料的可见性，不受区域限制。这允许合作伙伴在不修改内容的情况下将个人资料设为公开或私有。
5. GetProfileVisibility API：通过调用 UpdatePartnerProfileVisibility API，合作伙伴可以控制个人资料的可见性，不受区域限制。这允许合作伙伴在不修改内容的情况下将个人资料设为公开或私有。

使用域名管理

域管理

合作伙伴可以通过电子邮件验证注册和验证业务域，以建立组织身份并关联员工的培训和认证。

API 摘要

1. SendEmailVerificationCode API：通过向指定的电子邮件地址发送验证码来启动电子邮件验证流程。用于创建新的联系人和更新电子邮件地址。
2. AssociateAwsTrainingCertificationEmailDomain API：将电子邮件域与合作伙伴账户的 AWS 培训和认证相关联，从而自动验证员工认证。
3. DisassociateAwsTrainingCertificationEmailDomain API：移除电子邮件域与合作伙伴账户的 AWS 培训和认证之间的关联。

使用合作伙伴账户连接

合作伙伴可以使用 Partner Connection API 管理自己的AWS账户之间的连接或与其他AWS合作伙伴账户的关联。该过程包括两个阶段：

1. 连接邀请阶段：发起和响应连接请求
2. 活动连接阶段：管理已建立的连接及其相关的业务活动

创建和发送连接邀请

步骤 1：创建连接邀请

建立合作伙伴连接的第一步是使用`CreateConnectionInvitation`操作创建和发送连接邀请。创建邀请时，必须指定：

- 目录：用于管理连接的AWS目录（AWS或沙盒）
- ConnectionType: 您要建立的关系类型。选择下列选项之一：
  - **`OPPORTUNITY_COLLABORATION`**: 当您想向其他合作伙伴的账户发送连接请求时，请使用此类型，以便您可以向他们发送机会参与邀请，以进行机会协作
  - **`SUBSIDIARY`**：如果您想将您拥有的多个AWS Marketplace 卖家账户关联到您与 APN 关联的 “主要” 账户或您注册为合作伙伴的账户，请使用此类型
- ReceiverIdentifier：收款人的公共合作伙伴档案 ID（适用于机会协作连接类型）或AWS账户 ID（适用于子公司连接类型）

- 电子邮件：用于通信的联系电子邮件地址
- 姓名：您作为发件人的姓名
- 消息：解释连接请求目的的描述

创建后，邀请将进入待处理状态，等待接收者的操作。

> **Important**
>
> 发送连接邀请即表示您同意向接受邀请的接收者透露您的AWS账户 ID。这种披露对于建立账户级别的联系是必要的。

第 2 步：监控您已发送的邀请

您可以使用`ParticipantType`参数设置为的`ListConnectionInvitations`操作来跟踪已发送的邀请的状态`SENDER`。从而让您能够实现以下目的：

- 查看所有已发出的邀请
- 按状态筛选 (`PENDING`、`ACCEPTED`、`REJECTED`、`EXPIRED`)
- 按连接类型筛选
- 查看向其他合作伙伴发出的具体邀请

步骤 3：取消邀请（可选）

如果您需要在待处理的邀请被接受之前将其撤回，则可以使用该`CancelConnectionInvitation`操作。请注意：

- 您只能取消处于 “待处理” 状态的邀请
- 邀请被接受并建立连接后，就无法取消
- 要终止已建立的连接，必须改为使用`CancelConnection`操作

接收和回复连接邀请

步骤 1：列出收到的邀请

要查看您收到的连接邀请，请使用 ParticipantType 参数设置为的`ListConnectionInvitations`操作`RECEIVER`。您可以按以下条件进行筛选：

- 连接类型
- 状态 (PENDING,ACCEPTED,REJECTED)
- 发件人的标识符

第 2 步：查看邀请详情

使用GetConnectionInvitation操作可查看特定邀请的完整详细信息，包括：

- 发件人姓名和联系人邮箱
- 解释目的的邀请消息
- 正在请求的连接类型
- 到期日期
- 发件人的公开个人资料信息

步骤 3：接受或拒绝邀请

查看邀请详情后，您有两个选择：

选项 A：接受邀请

使用AcceptConnectionInvitation操作建立连接。当你接受时：

- 在活动状态下创建了一个新连接
- 邀请的状态更改为 “已接受”
- 您的AWS账户 ID 会被泄露给发件人
- 您可以开始执行由连接类型启用的业务活动

> **Important**
>
> 重要同意：接受连接邀请即表示您同意向发件人透露您的AWS账户 ID。这种披露对于建立账户级别的联系是必要的。

选项 B：拒绝邀请

如果您不想建立连接，请使用该RejectConnectionInvitation操作。您可以选择提供拒绝的理由。一旦被拒绝：

- 邀请的状态更改为 “已拒绝”
- 未建立连接
- 该邀请将不再出现在您的待处理邀请列表中

自动过期

如果在到期时间内未采取任何操作，系统会自动将邀请转换为 “已过期” 状态。无法接受或拒绝已过期的邀请。

管理活动连接

查看您的连接

建立连接后，双方均可使用以下操作查看和管理连接：

`ListConnections`: 使用筛选选项检索您的连接列表：

- 按连接类型和状态筛选（例如`OPPORTUNITY_COLLABORATION:ACTIVE`）
- 按另一方的标识符筛选
- 查看每个连接的摘要信息

`GetConnection`: 检索特定连接的完整详细信息，包括：

- 与此合作伙伴关系相关的所有连接类型
- 每种连接类型的状态（“活动” 或 “已终止”）
- 原始邀请函中的联系信息
- AWS双方的账户 ID 和公开个人资料 ID
- 每种连接类型的创建和终止时间戳

向现有连接添加连接类型

如果您已经与合作伙伴建立了活跃的联系并想要建立一种新型的关系，则可以使用不同的连接类型发送新的连接邀请。接受后：

- 新的连接类型已添加到现有连接中
- 两种连接类型均可独立管理
- 关系保持相同的连接 ID

> **Note**
>
> 在任何时候，给定目录中的两个账户之间只能有一个任一给定类型的活动连接。

终止连接类型

任何一方都可以使用CancelConnection操作终止特定的连接类型。终止时：

- 指定要终止的连接 ID 和连接类型
- 提供解雇理由
- 连接类型的状态更改为 CANCELED
- 同一连接中的其他连接类型不受影响

完全终止连接

当连接中的所有连接类型都终止时，连接本身将变为“已终止”状态。完全终止的连接：

- 无法重新激活，但您可以发送新的连接请求以继续使用该账户开展业务
- 留在系统中用于保存历史记录
- 可以使用 GetConnection API 查看
- 需要新的邀请才能重新建立关系

监视连接事件

合作伙伴应使用 Amazon 监控与连接相关的事件 EventBridge ，以便随时了解以下信息：

- 新的传入连接邀请
- 已发送邀请的状态变更 (ACCEPTED、REJECTED、EXPIRED)
- 由另一方发起的连接终止

收到事件后，使用相应的 Get API 操作来获取最新的详细信息：

- GetConnectionInvitation获取邀请更新
- GetConnection用于连接更新

可能发生的事件

- 已创建连接邀请：将收到的连接邀请通知收件人帐户。
- 连接邀请已取消：当发件人取消传入的连接邀请时，通知收件人账户。
- 连接邀请已过期：当连接邀请过期但未执行任何操作时，会同时通知发件人和收件人帐户。
- 连接邀请被拒绝：当发件人的连接邀请被收件人拒绝时，通知发件人账户。
- 已接受连接邀请：当发件人账户的连接邀请被接受并建立连接时，通知其账户。
- 连接已取消：当所有连接完全终止时，通知连接的两方。

了解连接状态

连接邀请状态

州	说明	可以过渡到
PENDING	创建时的初始状态；正在等待接收器操作	ACCEPTED, REJECTED, EXPIRED, CANCELED
ACCEPTED	接收器已接受；连接已创建	无（终端状态）
REJECTED	收件人已拒绝；包含可选的拒绝原因	无（终端状态）
EXPIRED	System-expired 由于在到期期内没有采取任何行动	无（终端状态）
CANCELED	发件人在接受邀请之前撤回了邀请	无（终端状态）

连接类型状态

州	说明
ACCEPTED	连接类型为可操作；已启用关联业务活动
CANCELED	连接类型由任何一方终止

最佳实践

1. 清晰的沟通：在连接邀请中提供详细、清晰的消息，说明目的和预期的协作。
2. 及时响应：及时监控和回复连接邀请，以避免自动过期。
3. 定期审查：定期检查您的活跃连接，确保它们与您当前的业务需求保持相关性。
4. 正确终止关系：在终止业务关系时，使用具有明确理由的 CancelConnection 行动来保持专业沟通。
5. 事件监控：设置 EventBridge 规则以自动接收连接状态更改通知，以帮助随时了解重要更新。
6. 连接类型管理：在建立连接之前，请考虑需要哪些连接类型，因为每种连接类型支持不同的业务功能。
7. 安全意识：请记住，建立连接会泄露各方之间的AWS账户 ID。仅与值得信赖的合作伙伴建立联系。

连接类型及其用途

连接类型	用途	使用场景
OPPORTUNITY_COLLABORATION	允许与其他合作伙伴就共同销售机会进行协作	如果您希望其他合作伙伴访问您的机会，反之亦然，请使用此连接类型。此连接允许您向关联的合作伙伴发送机会参与邀请。
SUBSIDIARY	在您的多个AWS Marketplace卖家账户和您的主要合作伙伴账户之间建立连接	当您有多个AWS Marketplace卖家账户想要关联到注册为合作伙伴和卖家的主账户时使用。

使用合作伙伴验证

管理合伙人验证

合作伙伴验证是合作伙伴账户注册的强制性先决条件。合作伙伴必须先完成业务验证和身份验证流程，然后才能创建合作伙伴账户。这些验证可验证企业是否合法注册，并确认注册AWS账户的人的身份。

合作伙伴可以利用可用的验证 API 来启动和监控验证流程：

在验证期间

1. 启动企业验证 — 合作伙伴通过调用 StartVerification API 来启动业务验证，并提供企业注册详细信息，包括法定名称、注册 ID、国家/地区代码和公司司法管辖区。
2. 身份验证启动 — 合作伙伴通过调用包含注册人信息的 StartVerification API 来启动身份验证。API 会返回一个安全的完成 URL，其中注册人使用政府颁发的身份证件完成身份验证工作流程。
3. 验证状态监控 — 合作伙伴使用 GetVerification API 检索验证流程的当前状态和详细信息。API 会返回验证类型、状态、时间戳和特定于验证的详细信息。
4. 验证完成 — 业务验证和身份验证都必须达到SUCCEEDED状态，合作伙伴才能继续创建合作伙伴账户。在注册账户之前，必须先解决验证失败或已过期的问题。

后期验证

1. 创建合作伙伴账户 — 成功验证后，合作伙伴将继续使用合作伙伴账户文档中描述CreatePartner的 API 创建其合作伙伴账户。
2. 验证记录检索 — 合作伙伴可以随时使用 GetVerification API 检索验证详情，并在启动期间返回验证 ID。

API 摘要

1. **StartVerification**API：合作伙伴通过调用 StartVerification API 启动验证流程。为了进行企业验证，API 会验证企业注册详细信息。对于身份验证，API 会生成一个有时间限制的完成 URL，供注册人完成身份验证。
2. **GetVerification**API：启动验证后，合作伙伴使用该 GetVerification API 检索验证状态和详细信息。API 返回当前状态、时间戳和特定于验证的信息。

使用AWS Partner Central Selling API

本AWS合作伙伴中心 API 参考旨在帮助AWS合作伙伴将客户关系管理 (CRM) 系统与合作伙伴中心集成。API 可自动与合作伙伴中心进行互动，这有助于确保有效参与联合业务活动。

该 API 提供标准AWS的 API 功能。您可以使用 API 操作或针对您的编程语言或平台量身定制的AWS SDK 来访问它。有关更多信息，请参阅入门AWS和构建工具AWS。

提供的功能AWS合作伙伴中心 API

1. 机会管理：AWS使用CreateOpportunity、、、UpdateOpportunity和AssignOpportunity API 操作促进共同销售机会的管理。ListOpportunities GetOpportunity
2. AWS推荐管理：AWS使用诸如ListEngagementInvitations、GetEngagementInvitation、和RejectEngagementInvitation之类的操作便于接收共享的推荐。StartEngagementByAcceptingInvitation
3. 实体关联：使用操作AssociateOpportunity和将AWS产品、合作伙伴解决方案、AWS Marketplace解决方案、AWS Marketplace产品和AWS Marketplace私人优惠等相关实体与机会关联起来DisassociateOpportunity。
4. 查看AWS机会详情：使用GetAWSOpportunitySummary操作可检索与您的AWS机会相关的机会的实时摘要。
5. 列出解决方案：为列出合作伙伴提供的解决方案提供列表 API ListSolutions。
6. 活动订阅：合作伙伴可以通过收听诸如 “机会创建”、“机会已更新”、“已接受参与邀请”、“拒绝参与邀请” 和 “使用亚马逊创建的参与邀请” 等活动来订阅机会的实时更新 EventBridge。

支持的区域和端点

AWS合作伙伴中心 API 已在美国东部（弗吉尼亚北部）区域推出。

Region	端点
us-east-1	partnercentral-selling.us-east-1.api.

合作伙伴可以在安全的沙盒环境中测试和验证 API 集成。这使您能够在不影响实时数据的情况下测试 API 操作。有关更多信息，请参阅 在沙箱中测试AWS Partner Central Selling API。

出售 API 实体

AWS Partner Central 实体代表合作伙伴与之间的共同销售活动中使用的关键业务组件。AWS这些实体封装了与机会、解决方案、产品和优惠相关的信息，从而实现了对联合销售活动的顺畅协作和管理。以下各节描述了合作伙伴平台销售 API 中的核心实体。

Opportunity

机会是企业识别并积极追求的潜在销售或交易。这是具有特定需求的合格潜在客户或潜在客户，公司的产品或服务可以满足这些需求。机会通常在销售渠道或CRM系统中进行跟踪，并构成未来收入的基础。有效的机会管理包括通过销售流程（从最初的资格认证到成交）培育潜在客户。有关更多信息，请参阅处理您的机会和数据类型

合作伙伴解决方案

代表合作伙伴解决方案（在 Partner Central 上AWS称为产品），它是由AWS合作伙伴创建和交付的软件产品或咨询业务。合作伙伴解决方案使用AWS服务帮助客户应对特定的业务挑战或实现特定目标。有关更多信息，请参阅什么是解决方案？

AWS产品

代表特定的AWS服务或产品。AWS提供各种产品和服务，旨在提供可扩展、可靠且经济实惠的基础架构解决方案。合作伙伴可以从 Partner Cent ral 的AWS批量导入页面（开始导入 >AWS产品和解决方案）获取最新的产品列表。要了解更多信息，您可以了解AWS产品或查看所有AWS产品的列表。

AWS Marketplace专属优惠

AWS Marketplace私人报价是一项允许AWS Marketplace卖家向个人AWS买家提供特定定价和条款的功能。通过私人报价，卖家可以协商定制价格、付款时间表和最终用户许可条款。AWS客户可以获得满足其特定要求的软件解决方案，同时还可能受益于与标准产品相比更优惠的条款或价格。私下报价流程涉及卖方创建具有量身定制条款的独特报价，然后与指定AWS客户私下共享该报价，以供他们审查和接受。有关更多信息，请参阅中的私人优惠AWS Marketplace。

订婚邀请

参与邀请是指合作伙伴就AWS特定推荐进行合作的正式请求。这使AWS合作伙伴能够共同努力，推动机会向前发展。合作伙伴可以接受或拒绝邀请。

主题

- 利用你的机会
- 利用来自的机遇AWS
- 处理机会更新
- 利用多合作伙伴机会
- 关联、解除关联和分配机会

- 与潜在客户合作
- 处理交易规模见解
- 最佳实践

利用你的机会

什么是机会？

在销售过程的初始阶段，销售代表会评估感兴趣的个人（称为潜在客户）是否有可能成为客户。此评估和验证阶段被称为资格认证。一旦销售线索被视为合格并且被认为更有可能转化为客户，则将其归类为机会。

利用你的机会

合作伙伴可以管理在其 CRM 系统中创建的机会，并使用AWS合作伙伴中心销售 API 将其与AWS合作伙伴中心同步。这使合作伙伴能够跟踪和管理从启动到结束的整个机会。

创造机会

管理机会的第一步是使用`CreateOpportunity`行动创造机会。这为`Lifecycle.ReviewStatus`设定创造了机会`Pending Submission`。但是，该机会尚未提交AWS给验证。

商机创建后，您必须使用`AssociateOpportunity`操作将至少一个合作伙伴AWS Marketplace解决方案、解决方案或AWS Marketplace产品与机会相关联。此操作清楚地定义了机会试图出售的内容。您可以使用 `ListSolutions` API 查看账户中可用解决方案的完整列表，也可以在一到十个解决方案之间进行关联。

或者，您也可以使用`AssociateOpportunity`操作将相关AWS产品与机会关联起来。此步骤可帮助AWS销售团队了解预计将与该机会一起销售哪些AWS产品。

关联所需的解决方案或产品，并且（可选）关联AWS产品后，您可以使用`StartEngagementFromOpportunityTask`操作开始与商机互动。这是您提交开始互动的机会的时候。

审核流程

使用`StartEngagementFromOpportunityTask`操作开始对机会进行互动后，机会进入AWS验证阶段，其设置`Lifecycle.ReviewStatus`为`Submitted`。在审核过程完成之前，不能对机会进行任何更改。

在此验证阶段，AWS确保机会详情准确完整。在验证过程中，设置Lifecycle.ReviewStatus为In-Review。

如果合作伙伴需要更改或提供其他详细信息，请将其AWS设置为Action Required，Lifecycle.ReviewStatus并通过该Lifecycle.ReviewComments字段传达任何必需的更新。

机会通过验证后，该机会就会Lifecycle.ReviewStatus变为Approved，从而为共同销售活动做好准备。

合作伙伴应使用 Amazon 监控Opportunity Updated活动 EventBridge。这将通知他们任何状态更改或来自的反馈AWS。收到活动后，合作伙伴可以使用 GetOpportunity API 操作来获取最新的机会详细信息并验证该Lifecycle.ReviewStatus字段。

解决验证问题

如果设置Lifecycle.ReviewStatus为Action Required，则合作伙伴需要解决突出显示的问题AWS。要解决这些问题，合作伙伴可以使用 UpdateOpportunity API 操作更新机会。

在此Action Required状态下，只有某些字段是可编辑的。这些字段包括：

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

在进行必要的更改后，机会重新进入验证阶段，该过程会重复进行，直到机会设置Lifecycle.ReviewStatus为Approved或。Disqualified

Post-Approval 更新

一旦机会出现Approved，合作伙伴就可以继续根据需要使用UpdateOpportunity操作更新机会，从而促进无缝的共同销售活动。

合作伙伴应继续通过 Amazon 监控Opportunity Updated事件 EventBridge ，以便随时了解任何更改。有关跟踪更AWS新的更多信息，请参阅 “处理机会更新” 部分。合作伙伴还可以根据业务验证规则更新选定字段。

利用来自的机遇AWS

1. 收到AWS Opportunity

当AWS销售主管将合作伙伴关联到 CRM 系统中的AWS机会时，机会就会与合作伙伴共享。这些机会被称为AWS机会，不同于在合作伙伴自己的账户中创建的机会。

当 Opp AWS ortunity 附上合作伙伴时，AWS会创建一份包含AWS机会数据子集的参与邀请。合作伙伴将收到 “创建参与邀请” 活动。

2. 查看参与邀请

参与邀请函包含一些基本信息Project.Title，例如Project.CustomerUseCase、Lifecycle.Stage、Project.CustomerBusinessProblem、和一些其他字段。合作伙伴可以使用这些数据来决定是否抓住机会。

但是，AWS机会中的以下字段未包含在参与邀请中：

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

3. 处理订婚邀请

在收到 “创建参与邀请” 活动后，合作伙伴可以使用该GetEngagementInvitation操作来检索AWS机会的详细信息。

如果合作伙伴决定不抓住机会，他们可以使用操作以及所需的RejectEngagementInvitation操作来拒绝邀请RejectionReason。一旦被拒绝，机会的访问权限就会丧失。

要接受邀请并继续，合作伙伴应使用StartEngagementByAcceptingInvitationTask操作。这是一个异步操作，按顺序执行以下任务：

1. 接受订婚邀请。
2. 使用机会中的数据在合作伙伴的账户中创建新AWS商机。
3. 包括识别合作伙伴账户中的客户所需的其他详细信息。

完成后，将使用相应的商机 ID 触发 “创建机会” 事件。然后，合作伙伴可以在GetOpportunity操作中使用此 ID 来检索机会的完整详细信息。

如果合作伙伴未使用事件，他们可以用相同的有效载荷StartEngagementByAcceptingInvitationTask再次致电以查看最新状态。

4. 管理AWS机会 Post-Acceptance

一旦机会进入合作伙伴的账户，就可以像系统中的任何其他机会一样对其进行管理和更新。合作伙伴可以使用诸如此类的操作UpdateOpportunity来进行任何必要的更改。

有关如何跟踪AWS更新和管理机会更新的更多详细信息，请参阅处理机会更新部分。

处理机会更新

更新机会

合作伙伴可以使用该UpdateOpportunity操作来更新机会，并使用具体规则来管理哪些字段的更新以及更新时间：

1. 如果是Submitted或，则Lifecycle.ReviewStatus无法进行更新In-Review。
2. 在提交之前，合作伙伴可以进行更新，但除非调用该StartEngagementFromOpportunityTask操作，否则AWS不会对其进行处理。
3. 当机会处于Submitted或In-review状态时，所有更新都将被阻止。
4. 如果商机处于Action Required状态，则AWS打开选择字段进行更新。这些字段包括：
   - Customer.Account.Address.City
   - Customer.Account.Address.Country
   - Customer.Account.Address.PostalCode
   - Customer.Account.Address.StateOrRegion
   - Customer.Account.Address.StreetAddress

- Customer.Account.WebsiteUrl
- LifeCycle.TargetCloseDate
- Project.ExpectedCustomerSpend.Amount
- Project.ExpectedCustomerSpend.CurrencyCode
- Project.ExpectedCustomerSpend.EstimationURL
- Project.ExpectedCustomerSpend.Frequency
- Project.ExpectedCustomerSpend.TargetCompany
- Project.CustomerBusinessProblem
- PartnerOpportunityIdentifier

5. 审阅过程结束后（即何时设置Lifecycle.ReviewStatus为Approved），将无法更新以下字段：
   - Customer.Account.Address.Country
   - Customer.Account.Address.PostalCode
   - Customer.Account.Industry
   - Customer.Account.WebsiteUrl
   - Project.CustomerBusinessProblem
   - PartnerOpportunityIdentifier
   - Project.Title
6. 对于所有其他字段，可以使用UpdateOpportunity操作进行更新。但是，根据特定计划或机会类型的业务规则，可能会适用其他限制。有关更多详细信息，请参阅字段级验证。
7. 对于通过 UI 和 API 进行的所有更新，都会生成机会更新事件。

接收来自的更新AWS关于机会

AWS通常会更新AWS机会，并且每次进行更新时，都会生成机会更新事件。

要从中检索最新更新AWS，合作伙伴需要提起两项单独的行动

1. GetOpportunity以检索合作伙伴机会的详细信息。
2. GetAWSOpportunitySummary检索AWS机会数据的实时摘要。

来自的大多数常规更新都AWS将通过GetAWSOpportunitySummary回复获得。但是，偶尔AWS可能会直接更新合作伙伴商机中的属性。

要使用来自的更新AWS

1. 调用GetAWSOpportunitySummary操作以检索AWS机会的最新详细信息。
2. 如果更改需要反映在合作伙伴的机会中，请使用UpdateOpportunity操作将相关数据复制到合作伙伴的机会上。

合作伙伴可以选择将此流程自动化，将其作为直接更新机制，也可以实施手动审查流程来验证和更新数据。

利用多合作伙伴机会

AWS PartnerAWS作为一个积极的参与者，可以共同寻找机会。

主题

- 互动和快照
- 为创建自定义策略 ResourceSnapshotJobRole
- 邀请合作伙伴抓住机会
- 检索参与邀请详情
- 回复订婚邀请
- 查看和更新多合作伙伴机会
- 使用快照接收合作伙伴最新消息
- 查看参与成员
- 监控资源快照任务状态

互动和快照

交互是指多个AWS Partner客户之间以及AWS针对客户机会进行协作而AWS拥有的一种资源。互动促进了合作伙伴之间的安全信息共享和协作，同时保持了个人机会的所有权和控制权。参与包括来自双方AWS和合作伙伴的机会，并以积极参与者的AWS身份出现。合作伙伴可以使用邀请AWS或其他合作伙伴加入互动EngagementInvitations。当被邀请的合作伙伴接受时，他们将在同一个参与中获得自己的机会。

参与成员通过快照共享进度，快照是来自基础资源（例如机会）的特定字段的不可变副本。快照是在互动中创建的，并与成员共享。当底层资源发生变化时，所有者可以创建新的快照修订版以反映更新。AWS提供快照作业，即客户拥有的作业，可在资源更改时自动创建新的修订版。共享后，参与成员仍可永久访问快照。

为创建自定义策略 ResourceSnapshotJobRole

合作开发多合作伙伴机会需要与项目中的其他合作伙伴共享您的机会快照。要保持对最新机会详细信息的访问权限，您需要创建一个名为的自定义角色ResourceSnapshotJobRole。此角色允许系统代表您创建机会的快照，并从同一项目中的其他合作伙伴那里检索快照。如果没有这个角色，参与活动的其他合作伙伴将无法看到您对机会所做的任何更新，他们将继续看到您的机会数据的过时快照。

> **Note**
>
> 你可以使用除之外的名称ResourceSnapshotJobRole。如果您使用其他名称，请用您的策略名称替换以下策略ResourceSnapshotJobRole中的所有实例。

要创建此角色，请转到您的 IAM 控制台并使用以下自定义信任策略创建该ResourceSnapshotJobRole角色：

JSON

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

创建此角色后，您需要附加AWSPartnerCentralSellingResourceSnapshotJobExecutionRolePolicyAWS托管策略或附加以下权限策略：

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/*"
            ]
        }
    ]
}
```

将 {account} 替换为您的 Amazon Web Services account ID。

创建ResourceSnapshotJobRole角色并附加权限后，您需要为组织ResourceSnapshotJobRole进行设置。使用PutSellingSystemSettings操作将您创建的角色设置ResourceSnapshotJobRole为公司的角色（由您的AWS账户标识）：

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

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

{account}替换为您的AWS账户 ID ResourceSnapshotJobRole 和您创建的角色名称。资源快照任务将隐含地担任此角色，以访问您的机会并创建快照，以便与项目中的其他合作伙伴共享。

邀请合作伙伴抓住机会

合作伙伴可以就来自双方的机会进行协作，也可以AWS发起参与邀请。您最多可以邀请九位合作伙伴参与一个机会。

邀请合作伙伴抓住机会

1. 按照《合作伙伴中心销售指南》中的 “寻找合作伙伴并与之建立联系”AWS中的步骤进行操作。您必须先与合作伙伴建立联系，然后才能邀请他们参与机会。此连接允许相互访问彼此的账户 ID，从而确保邀请到达目标合作伙伴账户。
2. 使用StartEngagementFromOpportunityTask操作创建交互并将机会与之关联。此异步操作按顺序执行以下任务：
    1. 创建互动。
    2. 将机会与参与联系起来。
    3. 调用该动CreateEngagementInvitation作。AWS
    4. 将机会提交给AWS。
    5. 启动ResourceSnapshotJob以创建快照。
3. 邀请其他合作伙伴加入。
    a. 要获取与您的机会关联的互动 ID，请使用按上一步TaskIdentifier返回的ListEngagementFromOpportunityTasks操作进行筛选。
    b. 使用接收合作伙伴的参与 ID 和账户 ID 向发送邀请CreateEngagementInvitation。

成功的邀请会向接收的合作伙伴发送参与邀请创建的事件。

检索参与邀请详情

当您收到互动邀请创建的活动时，请使用GetEngagementInvitation操作来检索邀请的详细信息。回复包括与该项目相关的客户机会的基本信息。

查看邀请详情，特别是InvitationMessage、Project.TitleProject.CustomerUseCase、和Project.CustomerBusinessProblem字段。这些信息提供了有关客户机会和邀请合作伙伴对合作的期望的背景信息。

使用这些详细信息来评估您是否想通过接受或拒绝参与邀请来抓住机会。

回复订婚邀请

当合作伙伴向您发送参与邀请时，它会创建一个创建互动邀请的事件，通知您该邀请。您可以选择接受或拒绝邀请。此决定决定了您是否参与多合作伙伴机会。

要拒绝订婚邀请，请执行以下操作：

使用RejectEngagementInvitation操作拒绝邀请。您必须提供一个`RejectionReason`参数来解释您的决定。一旦被拒绝，您将无法访问邀请的详细信息，而参与邀请被拒绝的事件会通知发送合作伙伴您已拒绝他们的邀请。

要接受订婚邀请，请执行以下操作：

要接受邀请并继续使用多合作伙伴机会，请使用StartEngagementByAcceptingInvitationTask操作。此异步操作按顺序执行以下任务：

1. 接受订婚邀请。
2. 使用来自发送合作伙伴机会的数据，在您的合作伙伴账户中创建新商机。您将收到一个机会创建的活动。
3. 包括识别您账户中的客户所需的其他详细信息。
4. 发布已接受参与邀请的事件，通知合作伙伴您已接受邀请并已添加到互动中。

订婚邀请已过期

如果您未在十五天内回复参与邀请，则该邀请将过期，并且您无法参与多合作伙伴机会。参与邀请过期事件会通知发送合作伙伴邀请已过期。

> **Note**
>
> 如果您仍想协作，则发送方合作伙伴必须发起新的参与邀请。

查看和更新多合作伙伴机会

当合作伙伴就机会发起互动时，接收合作伙伴账户中新创建的机会的审核状态由发送合作伙伴的机会状态决定。

评论状态为已提交的机会：

如果发送伙伴的机会Lifecycle.ReviewStatus设置为Submitted，则会发生以下过程：

1. 在接收合作伙伴的账户中创建的新机会将Lifecycle.ReviewStatus设置为Submitted。
2. 该Submitted状态将保持不变，直到发送伙伴有机会为止Approved。
3. 一旦发送合作伙伴的机会得到验证并设置Lifecycle.ReviewStatus为Approved，其他合作伙伴的机会将自动继承该Approved状态。

此流程可确保多合作伙伴机会的商机详情保持一致，无需进行多次审查。

完成后，将使用相应的商机 ID 触发商机创建事件。合作伙伴可以在GetOpportunity操作中使用此 ID 来检索完整的机会详情。

审核状态为已批准的机会：

如果发送方合作伙伴针对Lifecycle.ReviewStatus设置为的商机发起互动Approved，则会发生以下过程：

1. 在接收合作伙伴的账户中创建的新机会将Lifecycle.ReviewStatus设置为Approved。
2. 使用相应的商机 ID 触发商机创建事件。
3. 合作伙伴可以使用带有商机 ID 的GetOpportunity操作来检索完整的机会详情。

但是，已批准的机会中可能没有一些未从发送合作伙伴的机会中复制的特定于合作伙伴的详细信息。当机会阶段更新到Qualified或更晚阶段时，接收合作伙伴应使用UpdateOpportunity操作来更新这些详细信息：

- Project.CustomerUseCase
- Project.DeliveryModels
- Solution

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

一旦在接收合作伙伴的账户中创建了机会，就可以像合作伙伴系统中的任何其他机会一样对其进行管理和更新。合作伙伴可以利用该UpdateOpportunity操作进行更改或提供有关其参与机会的更多信息。

使用快照接收合作伙伴最新消息

在项目中，合作伙伴独立维护和更新其机会。当有人修改项目资源（例如添加机会）时，系统会发布一个创建的互动资源快照事件。

要随时了解变化并从其他合作伙伴的机会中获取最新信息，您可以使用以下操作：

- ListEngagementResourceAssociations— 使用此操作检索与机会关联的参与 ID。
- ListResourceSnapshots— 使用此操作可检索与该项目相关的所有机会快照的完整列表，从而概述所有合作伙伴的可用快照。
- GetResourceSnapshot— 使用此操作可获取特定机会快照的实时摘要，从而无需直接访问其他合作伙伴的机会即可查看最新信息。

如果您发现其他合作伙伴机会的相关更改或更新应反映在您自己的机会中，请使用UpdateOpportunity操作。此操作有助于有选择地将相关数据整合到您的机会中，从而确保整个项目的一致性和一致性。

查看参与成员

参与多合作伙伴机会最多可以有十个合作伙伴。每当新的合作伙伴接受参与邀请时，就会发布参与成员添加的活动，通知所有当前成员有关变更的信息。

要查看参与中所有成员的协作，请按照以下步骤操作：

1. 使用ListEngagementResourceAssociations操作检索与机会关联的互动 ID。
2. 提供从步骤 1 到ListEngagementMembers操作的 ID，以获取参与成员的合作伙伴详细信息。

> **Note**
>
> 只有参与的成员才能调用该`ListEngagementMembers`操作。

监控资源快照任务状态

在处理多合作伙伴机会时，您必须创建一个角色，允许您为其他合作伙伴创建机会快照，并从参与中其他合作伙伴那里检索机会快照。如果没有这个角色，您对机会所做的任何更新都将无法提供给参与该项目的其他合作伙伴，他们将继续看到您的机会的过时快照。

要跟踪与项目中的多合作伙伴商机相关的资源快照任务的状态，请按照以下步骤操作：

1. 使用ListEngagementResourceAssociations操作检索与机会关联的互动 ID。
2. 提供从步骤 1 到ListResourceSnapshotJobs操作的 ID，以生成项目中来电者拥有的所有快照任务的列表。从响应中检索任务 ID。
3. 向GetResourceSnapshotJob操作提供作业 ID 以跟踪任务状态并查看其是否正在运行。

该ResourceSnapshotJob操作向 Amazon 发布其异步操作 CloudWatch 的指标。 CloudWatch 将数据处理成可读的近乎实时的指标，以帮助您监控服务的性能和运行状况。

可供使用的指标如下：

- 故障-计算作业执行期间的内部问题。使用此指标来监控服务稳定性并设置故障阈值警报。
- 错误-跟踪任务处理期间与客户相关的问题。该指标有助于识别客户输入或配置存在的问题。
- 调用-表示任务调用的总数，包括成功和失败的尝试。使用它来跟踪一段时间内的服务使用趋势。

> **Note**
>
> CloudWatch 仅当ResourceSnapshotJob服务中有活动时，才会向其报告指标。如果没有任务或没有特定指标的数据，则不会报告该指标。

为确保操作顺利进行：ResourceSnapshotJob

- 使用这些指标来验证服务是否按预期执行。
- 创建 CloudWatch 警报以监控指标，并在指标超过可接受范围时触发操作（例如发送通知）。
- 设置主动监控以识别和解决问题。

有关使用 CloudWatch 指标的更多信息，请参阅 Amazon CloudWatch 用户指南。

关联、解除关联和分配机会

在机会生命周期的任何阶段，可以将机会与合作伙伴AWS Marketplace解决方案、AWSAWS Marketplace产品、解决方案、产品、AWS MarketplaceAWS Marketplace优惠和优惠集关联或取消关联。

将机会与其他实体关联起来

关联的实体是从RelatedEntityIdentifiers对象中的GetOpportunity方法中检索的。RelatedEntityIdentifiers可以使用进行更新AssociateOpportunity。请注意，无法使用UpdateOpportunity或CreateOpportunity方法更新此字段。

Solutions

在开始使用StartEngagementFromOpportunityTask操作进行互动之前，必须关联至少一个、最多十个合作伙伴解决方案。AWSAWS推荐人可能包含也可能不包含合作伙伴解决方案。

要查看您现有的解决方案，请使用ListSolutions操作。

合作伙伴可以在AWS合作伙伴中心Build部分创建、更新和管理他们的解决方案。

> **Important**
>
> 要将机会阶段推进到 “已提交” 或 “已发布”，您必须将有效的解决方案或AWS Marketplace产品列表与机会相关联。

> **Note**
>
> 仍然支持使用Solutions实体类型的传统合作伙伴解决方案（S-XXXX标识符）。但是，我们建议将AwsMarketplaceSolutions实体类型与机会相关联，因为Solution实体类型将来会被弃用。

AWS产品

一个机会最多可以关联 20 个AWS产品。要查看可用AWS产品列表，请使用托管AWS的产品列表GitHub。与AWS产品的关联完全使用该AssociateOpportunity操作完成。要更换或移除产品，请使用DisassociateOpportunity操作。

Solutions

机会可以与您的卖家账户中的解决方案相关联。使用RelatedEntityType设置为的AssociateOpportunity操作AwsMarketplaceSolutions。

该实体必须处于 “公开” 或 “受限” 状态。该服务拒绝草稿或非活动实体。

要关联解决方案，需要一个 ARN。以下示例显示了解决方案的 ARN 格式：

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

您也可以通过 “合作伙伴AWS Marketplace账户连接” 关联关联与您的主账户关联的子公司账户的解决方案。输入完整的 ARN，包括子公司账户 ID。

该`ListSolutions`操作将在每个解决方案摘要中返回`AwsMarketplaceSolutionArn`，并支持按 ARN 筛选。

AWS Marketplace产品

机会可以与您的卖家账户中的AWS Marketplace产品相关联。使用`RelatedEntityType`设置为的`AssociateOpportunity`操作`AwsMarketplaceProducts`。

该实体必须处于 “公开” 或 “受限” 状态。该服务拒绝草稿或非活动实体。

要关联产品，必须提供 ARN。ARN 包括产品类型（例如、`AmiProductSaaSProduct`、`ContainerProduct`）。以下示例显示了 AMI 产品的 ARN 格式：

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

您也可以通过 “合作伙伴AWS Marketplace账户连接” 关联关联与您的主账户关联的子公司账户中的产品。输入完整的 ARN，包括子公司账户 ID。

AWS Marketplace优惠和优惠套装

AWS Marketplace优惠

机会可以与AWS Marketplace私人报价挂钩。要查看可用报价，请使用AWS Marketplace目录 API 中的。 ListEntities 目前，您只能关联与AWS合作伙伴平台关联的AWS Marketplace卖家账户中的报价。

要关联私募报价，必须提供 ARN。示例：

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

AWS Marketplace优惠套装

机会也可以与AWS Marketplace优惠套餐相关联。优惠集允许将多个相关的商城报价组合在一起，以实现全面的解决方案打包。要查看可用的优惠套装，请 ListEntities 使用AWS Marketplace目录 API 中的。

要关联优惠套餐，需要提供 ARN。示例：

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

重要提示：一个机会可以与一个优惠或一个优惠套餐相关联，但不能同时与两者关联。一次只能将其中一种实体类型与机会相关联。

解除机会与其他实体的关联

使用`DisassociateOpportunity`操作取消机会与以下实体类型的关联：

- Solutions
- AWS产品
- AWS Marketplace解决方案
- AWS Marketplace产品
- AWS Marketplace优惠
- AWS Marketplace优惠套装

根据机会的状态，当您取消关联这些相关对象时，将适用不同的验证规则。

分配机会

用于`AssignOpportunity`更改机会的所有者。您可以将任何合作伙伴中心用户设置为机会所有者。

与潜在客户合作

什么是线索？

在企业对企业 (B2B) 销售中，潜在客户代表对公司的产品或服务表示兴趣但尚未完全符合销售机会资格的潜在客户。潜在客户是销售渠道的初始阶段，需要对其进行培育和资格认证，然后才能将其转化为积极的共同销售机会。AWS

AWS与合作伙伴共享的潜在客户基于客户参与信号，例如网络研讨会出席人数、活动参与度或合作伙伴解决方案查询。这些潜在客户包括宝贵的背景信息，例如客户公司信息、业务问题和参与细节，以帮助合作伙伴确定潜在机会的优先顺序和确定潜在机会。

处理潜在客户邀请

AWS当潜在客户对合作伙伴解决方案或服务表示兴趣时，合作伙伴就会收到潜在客户邀请。潜在客户邀请流程允许合作伙伴在承诺参与之前对潜在客户进行评估，从而确保有效的资源分配和更高的转化率。

收到潜在客户邀请

AWS创建潜在客户邀请并通过销售 API 与合作伙伴共享。当有新的潜在客户邀请可用时，AWS会向符合条件的合作伙伴发送包含潜在客户上下文的参与邀请。

合作伙伴应使用 Amazon 监控`Engagement Invitation Created`活动 EventBridge。此事件通知包括设置为的`payloadType`字段`LeadInvitation`，允许合作伙伴区分潜在客户邀请和机会邀请。收到活动后，合作伙伴可以检索邀请详细信息以评估潜在客户。

列出潜在客户邀请

合作伙伴可以使用 `ListEngagementInvitations` API 操作查看其所有潜在客户邀请。此操作会检索主叫方是发送者或接收者的邀请。

要专门筛选潜在客户邀请，合作伙伴应使用带有值的`payloadType`筛选条件`LeadInvitation`。此筛选器可确保仅返回潜在客户邀请，结果中不包括机会邀请。

潜在客户邀请可能处于以下状态：

- 待定：等待合作伙伴接受或拒绝
- 已@@ 接受：合作伙伴已接受邀请并获得完整的潜在客户详细信息
- 已@@ 拒绝：合作伙伴已拒绝邀请
- 已过期：邀请已过期，但未采取任何行动

评估潜在客户邀请

在接受潜在客户邀请之前，合作伙伴应使用 `GetEngagementInvitation` API 操作检索详细的邀请信息。这为做出知情的接受或拒绝决定提供了基本背景。

潜在客户邀请有效载荷包括：

客户信息（预验收时可用）：

- 公司名称、行业和国家
- 网站 URL
- 细分市场（企业、大型、中型、小型、微型）
- AWS成熟度级别（评估、 Single-Account、 Multi-Account）

客户互动：

- 来源类型和广告活动信息
- 客户采取的行动（填写表格、参加网络研讨会等）
- 客户提供的业务问题描述
- 用例类别
- 联系人标题（提供 BANT 资格的权限背景）

> **Important**
>
> 客户联系信息（姓名、电子邮件、电话号码）在邀请阶段不可见。只有在接受邀请后才会提供联系方式，从而确保客户隐私并防止铅种植行为。

接受潜在客户邀请

当合作伙伴决定寻找潜在客户时，他们必须使用 `AcceptEngagementInvitation` API 操作接受邀请。此操作会将合作伙伴添加到项目中，并允许其访问完整的潜在客户详细信息，包括客户联系信息。

验收后：

- 合作伙伴被添加为项目成员
- 通过互动潜在客户上下文可以看到客户联系信息
- 在系统中，潜在客户被归类为合作伙伴合格潜在客户 (PQL)AWS
- 潜在客户出现在合作伙伴的活跃潜在客户列表中
- 合作伙伴可以开始培养潜在客户并与客户建立联系

合作伙伴可以通过调用AcceptEngagementInvitation每份邀请，以编程方式接受多个潜在客户邀请，从而高效地批量处理高质量的潜在客户。

拒绝潜在客户邀请

如果潜在客户与合作伙伴的能力、能力或业务重点不一致，则合作伙伴可以使用 RejectEngagementInvitation API 操作拒绝邀请。拒绝潜在客户邀请时，合作伙伴应提供拒绝理由，以帮助AWS改善潜在客户路由和匹配。常见的拒绝原因包括：

- 缺乏地理覆盖范围
- 该用例所需的技术专业知识不足
- 当前的容量限制
- 客户行业不匹配
- 预算错位

拒绝后：

- 潜在客户已从合作伙伴的待处理邀请队列中移除
- 客户联系信息绝不会与合作伙伴共享
- 在系统中，潜在客户被归类为合作伙伴拒绝的潜在客户 (PRL)AWS
- 该邀请在合作伙伴的已拒绝邀请列表中仍然可见，以供参考

只要满足客户同意要求，被拒绝的潜在客户可能有资格重新分配给其他合作伙伴。

管理已接受的潜在客户

接受潜在客户邀请后，合作伙伴可以访问完整的潜在客户信息，培养客户关系，并在资格认证阶段跟踪潜在客户。

查看潜在客户详情

合作伙伴使用 GetEngagement API 操作访问完整的潜在客户详细信息。该项目包含 Lead 上下文，其中包含有关客户及其与之互动的全面信息AWS。

潜在客户背景包括：

完整的客户资料：

- 原始邀请函中的所有公司信息
- 所有客户互动的完整联系方式（姓名、电子邮件、电话、公司职务）
- AWS成熟度水平和细分市场

互动记录：

- 将多个客户接触点整合到一个互动中
- 每次互动都包括来源信息、客户操作、业务问题和联系方式
- 互动按时间顺序列出，以提供完整的客户旅程视图

资格状态：

- 潜在客户资格的当前状态（不合格、合格、取消资格或自定义状态）
- 合作伙伴可以在资格认证过程中更新此状态

这种全面的视图使合作伙伴能够了解客户在多个AWS活动和接触点上的完整互动历史记录，而不必将每次互动视为单独的潜在客户。

列出活跃的潜在客户

合作伙伴可以使用 `ListEngagements` API 操作检索所有活跃的潜在客户，`contextType`筛选条件设置为`Lead`。这会返回合作伙伴是成员且参与包含潜在客户上下文的互动。

列表响应包括关键信息，例如：

- 参与 ID 和标题
- 客户公司名称
- 当前参与度分数
- 资格状态
- 互动次数
- 创建和修改时间戳

合作伙伴可以使用此列表来构建仪表板，跟踪潜在客户渠道的运行状况，并确定需要关注的潜在客户。

利用潜在客户洞察丰富潜在客户

合作伙伴可以利用AWS洞察来丰富已接受的潜在客户，以便在投资外联活动之前更好地确定优先顺序并对其进行资格审查。扩充是通过勘探任务异步进行的，这些任务通过AWS派生的信号来增强潜在客户的参与度，以支持资格决策。

开始探矿任务

合作伙伴使用 StartProspectingFromEngagementTask API 操作启动充实。此操作在单个请求中最多接受 100 个参与标识符，允许合作伙伴分批丰富潜在客户。该任务异步运行，并立即返回合作伙伴TaskArn用来跟踪进度的 a TaskId 和。首字母TaskStatus是、PENDINGIN_PROGRESSCOMPLETED、或之一FAILED。

合作伙伴可以选择提供 a TaskName 来标记任务，并提供 a ClientToken 来确保重试之间的等性。

轮询结果

由于发现潜在客户是异步的，因此合作伙伴使用 GetProspectingFromEngagementTask API 操作进行轮询以确定是否完成，并在开始时TaskId返回结果。该响应报告了提交的每个标识符的总体任务状态和每次互动的结果。

每项互动都是独立处理的，因此单个交互可以成功或失败，而不会影响同一任务中的其他参与者。对于每项互动，结果包括：

- 参与标识符：已处理的互动。
- 状态：PENDINGIN_PROGRESS、COMPLETED、或FAILED。
- 潜在客户情境标识符：成功处理交互时填充。此标识符引用了项目中添加的丰富探矿背景，可以在后续操作中使用。
- 原因代码和消息：仅在项目失败时填充，提供枚举的失败代码和易于理解的描述以及建议的恢复步骤。

监控多项任务

要跟踪许多潜在客户的丰富程度，合作伙伴可以使用 ListProspectingFromEngagementTasks API 操作。此操作会列出来电者账户发起的所有潜在客户任务，并支持按任务标识符、任务名称或开始时间范围进行可选筛选，并可配置排序。响应是分页的；使用每个响应的NextToken值来检索后续页面。

当扩充成功完成后，这些AWS见解将作为潜在客户背景添加到项目中。合作伙伴可以检索丰富的详细信息，GetEngagement并使用它们来设置潜在客户的资格状态，并决定哪些潜在客户可以实现转化。

更新潜在客户信息

在合作伙伴培育潜在客户并收集更多信息时，他们可以使用 UpdateEngagementContext API 操作更新潜在客户详细信息。此操作允许合作伙伴修改项目中的潜在客户上下文。

合作伙伴可以更新：

资格状态：合作伙伴应在完成内部资格认证流程时更新资格状态：

- 不合格：接受潜在客户后的默认状态；表示潜在客户需要评估
- 合格：潜在客户符合资格标准，并显示出转化为机会的巨大潜力
- 取消资格：潜在客户不符合标准或不适合合作伙伴的解决方案
- 自定义状态：合作伙伴可以定义自己的资格状态以匹配其内部流程

潜在客户元数据：合作伙伴可以添加或更新与其资格认证和培育流程相关的其他信息。

更新资格状态可以帮助合作伙伴跟踪潜在客户进展，生成准确的渠道报告，并确定准备转化为机会的潜在客户。

监控AWS更新

AWS可能会根据新的客户参与度信号或精确的参与度评分来更新潜在客户信息。AWS更新互动时，合作伙伴会通过 Amazon 收到一个Engagement Updated活动 EventBridge。

这些更新可能包括：

- 根据新客户活动细化参与度分数
- 来自新活动或接触点的更多客户互动
- 更新了客户信息

合作伙伴应监控这些事件并致电GetEngagement以检索最新的潜在客户详细信息。这样可以确保合作伙伴获得最新的信息，以确定潜在客户的优先顺序和培育潜在客户。

该Engagement Updated活动包括该contextTypes字段，允许合作伙伴专门筛选潜在客户（与潜在客户情境的互动）。

将潜在客户转化为机会

当潜在客户获得资格并表现出认真的购买意向时，合作伙伴可以将其转化为与AWS之进行正式共同销售合作的机会。这种转变标志着从潜在客户培育（主要是合作伙伴驱动）向机会管理（与合作）的过渡。AWS

推荐方法：便利 API

合作伙伴可以使用StartOpportunityFromEngagementTask便捷的 API 操作来简化机会创建和关联流程。此任务 API 会自动协调多个操作：

1. 使用来自潜在客户背景的初步信息创建草稿机会
2. 通过资源快照将机会与来源互动联系起来
3. 为互动添加 CustomerProject 上下文

这种便捷的方法减少了所需的 API 调用次数，并确保了正确的归因跟踪。使用这种方法的合作伙伴仍然需要：

- 使用填写机会详情 UpdateOpportunity
- 关联合作伙伴解决方案使用 AssociateOpportunity
- 准备就绪StartEngagementFromOpportunityTask后使用 “提交机会”

便捷性 API 对于那些希望最大限度地降低整合复杂性的自动化线索到机会工作流程的合作伙伴特别有用。

替代方法： Step-by-step API

创建选秀机会

转换潜在客户的第一步是使用 CreateOpportunity API 操作创建草稿机会。这为Lifecycle.ReviewStatus设定创造了机会Pending Submission。在现阶段，机会尚未提交给AWS进行验证。

合作伙伴应利用在潜在客户培育期间收集的信息填充机会，包括：

- 客户账户详情
- 项目信息和业务问题
- 客户预期支出

- 目标截止日期
- 资格认证期间收集的任何其他相关细节

通过草稿机会，合作伙伴可以在提交信息之前准备好完整的信息AWS，从而确保更高的批准率和更快的验证。

将机会与潜在客户互动联系起来

创建机会草稿后，合作伙伴必须使用 CreateResourceSnapshot API 操作将其与来源潜在客户互动关联起来。此步骤对以下方面至关重要：

- 归因跟踪：建立从潜在客户邀请到机会关闭的来源链，从而能够准确计算营销活动和潜在客户来源的投资回报率。
- 转化指标：允许AWS合作伙伴衡量潜在客户到机会的转化率并确定成功的潜在客户来源。
- 情境保存：维护完整的客户旅程历史记录，包括所有原始互动和参与信号。

在创建资源快照时，合作伙伴应指定：

- 互动 ID（来源潜在客户互动量）
- 机会标识符
- 资源类型（机会）

此操作会自动为交互添加 CustomerProject 背景信息，表示潜在客户已转化为活跃机会。现在，该项目既包含原始的潜在客户背景（保存历史记录），也包含新的 CustomerProject 上下文（表示活跃的机会）。

填写机会详情

机会创建并关联后，合作伙伴必须在提交之前完成其他机会要求。有关详细信息，请参阅 AssociateOpportunity API 操作。

更新项目详情：合作伙伴可以使用 UpdateOpportunity API 操作来完善项目信息、客户业务问题、预期支出以及在潜在客户资格认证期间收集的其他相关详细信息。

提交机会

完成所有必填信息后，合作伙伴将使用StartEngagementFromOpportunityTask便捷的 API 提交AWS验证机会。这会将机会从草稿状态过渡到AWS审核流程。

验证要求：在提交之前，项目必须有 CustomerProject 背景信息。当使用将机会与项目关联时`CreateResourceSnapshot`，会自动创建此上下文。如果缺少此上下文，则提交将失败。

提交后：

- 机会`Lifecycle.ReviewStatus`变为 `Submitted`
- 在系统中，潜在客户被归类为合作伙伴销售合格潜在客户 (PSQL)AWS
- AWS开始验证以确保机会详情准确完整
- 在审核过程完成之前，不能对机会进行任何更改

然后，机会将按照 “处理您的机会” 文档中描述的标准验证工作流程进行，按照 “需要操作”、“已批准” 或 “取消资格” 等 In-Review状态进行处理。

处理交易规模见解

什么是交易规模？

交易规模调整是 P AWS artner Central 中的 APN 客户互动 (ACE) 机会中的一项功能，它使用 AI 在合作伙伴创建或更新机会时自动提供交易规模估算和AWS服务建议。

AI 预测的 MRR 估计值

根据合作伙伴的历史AWS机会、用例和业务描述，交易规模洞察提供了预测的每月经常性收入 (MRR) 范围的中位数。当合作伙伴收集有关交易和整个销售周期进展的更多信息时，合作伙伴应审查和更新总的MRR。

AI-Recommended AWS产品

AI-recommended 产品是根据客户业务问题描述和机会详情来识别的。AI 推荐符合技术要求和用例的相关AWS产品。合作伙伴应查看这些建议，并根据客户的特定需求定制产品选择。

使用定价计算器网址增强见解

合作伙伴还可以导入AWS定价计算器网址以自动填充服务选择，并获得增强的见解，包括Migration Acceleration Program (MAP) 资格指标、优化建议和潜在的成本节省分析。

了解 Source-Separated 数据

交易规模提供来自两个不同来源的见解，让您全面了解机会价值：

- AWS来源：基于类似历史机会模式的 AI-recommended 产品

- 合作伙伴来源：从AWS定价计算器导入的您自己的估算值

处理源之间的差异

当合作伙伴提供的估算值与AWS建议存在很大差异时，合作伙伴应：

1. 查看机会详情，确保捕捉到所有相关信息
2. 验证定价计算器 URL 配置是否符合实际需求
3. 将AWS建议视为以类似历史机会为依据的数据点
4. 根据特定的客户环境和要求做出明智的决策

访问交易规模见解

合作伙伴通过 API 操作访问交易规模数据，`GetAwsOpportunitySummary`API 操作会返回一个包含交易规模预测、产品推荐和优化见解的扩展`AwsOpportunitySummary`对象。

当交易规模数据可用时

在创建了包含足够客户和项目详细信息的机会后，即可获得交易规模见解。系统分析机会属性并生成：

1. AI 预测 MRR：根据机会特征自动计算
2. AWS产品推荐：与客户的业务问题和技术要求相关的服务
3. 增强见解（提供定价计算器网址时）：MAP 资格、优化建议和成本节约分析

> **Note**
>
> 并非所有机会都可获得交易规模见解。资格取决于合作伙伴历史记录和所提供机会详情的完整性等因素。

提供交易规模输入

合作伙伴通过`CreateOpportunity`和 `UpdateOpportunity` API 操作为交易规模提供输入。系统使用机会属性来生成推荐。

提供合同总价值 (TCV)

以合同总价值而不是每月经常性收入进行估算的合作伙伴可以直接提供其交易价值。AWS 使用转换模型自动将 TCV AI-powered 转换为 AWS MRR。

对于 TCV 中的交易规模，请提供 a ExpectedContractDuration sMonths（有效值范围为 1 到 144 个月）和 a TargetCompany s Self 和 a Frequency s None。

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

AWS 得出 AWS MRR 估算值，并返回以下两个条目：GetOpportunity

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

在上面的示例中，ExpectedCustomerSpend第一个数组包含 AWS MRR 估计值 (TargetCompany: "AWS",Frequency: "Monthly")，第二个数组是合作伙伴的合同总价值 (TargetCompany: "Self",Frequency: "None")。

TCV 验证规则

所有 TCV 验证错误都会返回INVALID_VALUE带有的错误代码。Reason: "BUSINESS_VALIDATION_FAILED"区别因素是FieldName和Message：

条件	字段	Message
无需自行进入 ExpectedContractDuration	project.expectedContractDuration	ExpectedContractDuration 当有自助参赛者时是必填的
ExpectedContractDuration 不带自行输入（仅限创建）	project.expectedCustomerSpend	在场时 ExpectedContractDuration 需要自行进入
多个 Self 条目	project.expectedCustomerSpend	只允许一次自行进入
存在持续时间 TargetCompany 时无效	project.expectedCustomerSpend.targetCompany	TargetCompany 必须是 “AWS” 或 “Self”
AWS 和 Self 之间的货币不匹配	project.expectedCustomerSpend.currencyCode	CurrencyCode 条目之间必须匹配
自助进入 + EstimationUrl 一起进入	project.expectedCustomerSpend.estimationUrl	自行进入且 EstimationUrl 不能共存

Note

开启UpdateOpportunity，ExpectedContractDuration如果没有自我条目，但存储中存在合作伙伴交易值，则系统会自动重构自我条目——不会引发任何错误。

在 TCV 和手动 MRR 之间切换

要从 TCV 模式切换回手动 MRR，请明确设置`ExpectedContractDuration`为`null`并仅提供一个`AWS`条目：

```
{
  "Project": {
    "ExpectedContractDuration": null,
    "ExpectedCustomerSpend": [
      {
        "Amount": "8000.00",
        "CurrencyCode": "USD",
        "Frequency": "Monthly",
        "TargetCompany": "AWS"
      }
    ]
  }
}
```

> **Note**
>
> `ExpectedContractDuration`从有效载荷中省略（不发送字段）意味着 “没有变化” ——保留现有的 TCV 数据。明确发送`null`意味着 “清除 TCV 模式”。

TCV 和定价计算器网址

如果合作伙伴同时提供了`Self`条目（TCV 模式）和`EstimationUrl`，则 API 会返回验证错误。自行输入， EstimationUrl 不能在同一个请求中共存。

提供手动的 MRR 估算

合作伙伴可以使用以下`Project.ExpectedCustomerSpend[].Amount`字段手动输入 MRR 估算值：

> **Note**
>
> 该`CurrencyCode`字段接受`USD`和`EUR`。如果 AWS 分区为`aws-eusc`（AWS 欧洲主权云），则货币代码必须为`EUR`。

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

此字段可供所有合作伙伴使用，可用于通过设置相同值来接受 AI 预测，也可以使用 Partner-provided 值进行覆盖。

提供定价计算器网址

合作伙伴可以选择通过该`Project.ExpectedCustomerSpend[].EstimationUrl`字段提供AWS定价计算器网址，以自动导入服务配置并获得增强的见解，包括 MAP 资格指标、优化建议和成本节约分析。

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

提供有效的定价计算器 URL 后，系统会自动根据计算器配置计算 MRR 金额，并在合作伙伴来源中填充 “金额” 字段和详细的产品级支出数据。`AwsProductsSpendInsightsBySource`合作伙伴在提供金额时应将金额设置为空 EstimationUrl ——系统将自动计算金额。

格式无效的 URL 和无法解析的 URL 将被拒绝`ValidationException`。

金额和 EstimationUrl 运作方式如何

您可以灵活地为机会提供每月经常性收入 (MRR) 估算值。API 可以智能地处理手动金额和AWS定价计算器网址的各种组合，以确保您的机会始终可以成功创建。

仅使用手动金额

当您仅提供金额字段时，API 会保存您的手动估算值并将其设置 EstimationUrl 为空。当您通过客户对话或自己的计算得出具体的 MRR 估算值时，这种方法非常有用。

请求示例：

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

仅使用一个AWS定价计算器网址

当你只提供 EstimationUrl 字段时：

- 有效网址-API 根据您的计算器配置计算 MRR 金额，并保存网址和计算出的金额。
- 网址无效 — API 会返回一个 ValidationException ，其中包含有关网址无效原因的详细信息。

请求示例：

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

```
}
```

错误响应示例（网址无效）：

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

同时提供金额和 EstimationUrl

当您同时提供这两个字段时，API 会优先确保您的机会成功创建：

1. 如果两个值均未更改，则不会对这些字段执行任何更新
2. 网址无效 — API 会保存您提供的金额并设置 EstimationUrl 为空，这样您就可以继续创建机会。
3. URL 有效，计算出的金额与您提供的金额相符 — 两个值均已保存。
4. 网址有效，但计算出的金额与您提供的金额不符 — API 会保存您提供的金额并设置 EstimationUrl 为 null 而不会返回错误。

> **Note**
>
> 当两个值都与现有值不匹配时，API 会首先尝试根据网址计算金额。当出现不匹配或验证问题时，您手动提供的金额始终优先。

主要优点：无效 URL 永远不会阻止机会创建

这种方法可确保无效或无法访问的AWS定价计算器网址永远不会阻止您创建或更新机会。当出现冲突或验证问题时，您手动提供的金额始终优先，这样您就可以完全控制机会数据。

了解扩展 AwsOpportunitySummary 数据模型

本节介绍了 `GetAwsOpportunitySummary` API 操作返回的响应结构。

> **Note**
>
> AI-forecasted MRR 返回于 `Project.ExpectedCustomerSpend[].Amount`

`GetAwsOpportunitySummary`API 操作通过新的`Insights.AwsProductsSpendInsightsBySource`结构返回交易规模见解，该结构提供带有来源归因的全面支出分析。

源分离：AWS vs 合作伙伴数据

交易规模按来源区分见解，以提供透明度并防止重复计算：

- AWS来源：来自 AI 产品推荐AWS
- 合作伙伴来源：从合作伙伴提供的定价计算器网址导入的支出数据和产品详情

这种分离使合作伙伴能够：

1. 将他们的估计值与AWS建议进行比较
2. 验证他们的尺寸假设
3. 避免不小心合并两个来源，从而夸大总数
4. 保持对数据来源的可见性

结构概述

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

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

字段参考

AwsProductsSpendInsightsBySource 地图

Source-separated 花费为AWS推荐和合作伙伴估算提供独立分析的见解。

该`Insights.AwsProductsSpendInsightsBySource`字段是一个包含最多两个键的地图：

Key	Type	说明
AWS	AwsProductsSpendInsights	AI-generated 见解，包括来自的推荐产品AWS
Partner	AwsProductsSpendInsights	Partner-sourced 从定价计算器网址中获得的见解，包括详细的服务成本和优化

> **Note**
>
> 只有具有可用数据的来源才会包含在响应中。新的机会通常只从填充AWS来源开始。

AwsProductInsights 对象

针对单一来源（AWS或合作伙伴）的全面支出分析，包括总金额、优化节省、计划类别明细以及详细的产品层面见解。

每个源对象都包含以下字段：

字段	Type	说明
CurrencyCode	字符串	ISO 4217 货币代码。支持的值为 “美元” 和 “欧元”。
频率	字符串	指明客户预计花费预计金额的频率。“频率” 字段只允许使用 “每月” 值，表示每月经常性支出。
TotalAmount	字符串	优化前此来源的预计支出总额
TotalOptimizedAmount	字符串	应用建议的优化后的预计支出总额
TotalPotentialSavingsAmount	字符串	通过实施优化可以实现量化的节约
TotalAmountByCategory	object	与AWS计划和现代化路径对应的支出金额
AwsProducts	array	Product-level 包括成本和优化建议在内的详细信息

当前状态限制：对于AWS来源， TotalAmount TotalOptimizedAmount、和，如果没有按产品划分的支出建议，则 TotalPotentialSavingsAmount 可以省略。在这些情况下，合作伙伴应参考`Project.ExpectedCustomerSpend[].Amount`总的AWS MRR估算值。

TotalAmountByCategory 对象

地图支出相当于AWS计划和战略计划：

类别密钥	说明
MAP 符合条件	在有资格获得 Migration Acceleration Program 资助
符合CAT资格	支持成本分配和跟踪的服务

类别密钥	说明
途径-迁移到云原生	与云原生现代化保持一致的服务
途径——转向 Managed Services	支持采用托管服务的服务
途径-转向开源	开源服务替代方案
途径-转移到容器	Container-based 服务选项
途径-迁移到无服务器	无服务器计算服务
途径-移至数据库	数据库现代化服务
途径-转向分析	分析和数据处理服务

> **Note**
>
> TotalAmount 由于单个商品可能属于多个类别，因此类别总金额可能会超过。

AwsProducts 阵列

包含计划资格指标（MAP、现代化路径）、成本估算和优化建议的AWS服务清单。

每个产品对象都包含：

字段	类型	必需	描述
ProductCode	字符串	是	AWS用于机会关联的合作伙伴中心产品标识符
ServiceCode	字符串	否	定价计算器服务代码（指向原始计算器网址的链接）
类别	array	是	本产品有资格进入的计划和途径类别清单

字段	类型	必需	描述
使用额	字符串	否	优化前的基准服务成本（对于AWS源自的建议，可能为空）
OptimizedAmount	字符串	否	应用优化后的服务成本（对于AWS源自的建议，可能为空）
PotentialSavingsAmount	字符串	否	Service-specific 通过优化降低成本（对于AWS源自的建议，可能为空）
优化	array	否	此产品的具体优化建议清单

Null 处理：对于AWS来源的产品，如果没有按产品分类的支出建议 OptimizedAmount，则 “金额” 和 “ PotentialSavingsAmount 字段” 可能为空。`Project.ExpectedCustomerSpend[0].Amount`相反，系统通过提供总的MRR。

优化对象

每项优化建议都包含：

字段	Type	说明
Description	字符串	Human-readable 优化策略的解释
SavingsAmount	字符串	通过实施这种优化可以实现量化的成本节约

示例

示例 1：AWS仅限建议（当前状态）

此示例显示了当只有 AI 推荐可用且尚无法推荐每款产品的支出金额时的典型响应。

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

要点：AWS来源显示了推荐的产品，其中包含类别，但不包含每件商品的金额。总的MRR估算值可通过以下方式Project.ExpectedCustomerSpend[].Amount获得.

示例 2：合作伙伴定价计算器导入

此示例显示了当合作伙伴提供有效的定价计算器网址时，可以增强洞察力。

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

要点：合作伙伴来源包含完整的支出明细以及每件产品的金额。优化建议提供了切实可行的成本降低策略。类别细分显示了 MAP 资格（总计 32,500 美元中的 22,500 美元）。产品包括 ServiceCode 链接回定价计算器配置。

示例 3：两个来源都存在（比较方案）

此示例演示了AWS推荐和合作伙伴估算值如何合并在一起，从而便于进行比较。

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

要点：两个来源都有：AWS推荐（总计 2.8 万美元）和合作伙伴计算器（总计 32.5 万美元）。AWS推荐显示 4 种产品；合作伙伴计算器显示 4 种产品（3 种重叠）。合作伙伴可以将AWS建议与其详细估算值进行比较。AWS可通过以下方式获得的总数`Project.ExpectedCustomerSpend[].Amount`。

最佳实践

处理空值

1. EstimationUrl 字段：对于大多数合作伙伴，预计 EstimationUrl 为空。这是标准行为，不应被视为错误。将手动 MRR 输入选项显示为主路径。
2. 缺少类别：如果 TotalAmountByCategory AWS来源中没有，则表示不提供按产品分类的推荐。如果提供了定价计算器网址，则可以对 Partner-sourced 数据进行类别细分。

优化建议

1. 显示切实可行的见解：突出显示优化说明和节省金额，以帮助合作伙伴了解降低成本的机会。
2. 总节省额：对 PotentialSavingsAmount 所有产品进行汇总以显示总体优化潜力。
3. 按影响排序：对优化进行排序 SavingsAmount ，以帮助合作伙伴专注于影响力最高的建议。

监控更新

1. 定期轮询：在商机创建工作流程中`GetAwsOpportunitySummary`定期进行轮询（建议：每 5 分钟一次）。
2. 处理异步生成：在创造机会后，可能无法立即获得交易规模见解。实现适当的加载状态和重试逻辑。

API 集成模式

1. 利用现有集成：已经致电的合作伙伴`GetAwsOpportunitySummary`只需要在响应中使用其他字段，无需调用新的 API。
2. 优雅降级：设计用户界面以处理交易规模数据尚不可用或不完整的场景。

数据质量和准确性

1. 验证定价计算器网址：在提交之前确保 URL 有效，以避免 EstimationUrl 回复为空。
2. 提供丰富的机会详情：更完整的机会信息（客户详细信息、项目描述、技术要求）可以提供更准确的人工智能建议。
3. 审查差异：当合作伙伴的估计值与AWS建议存在显著差异时，请查看机会详细信息以确保捕捉到所有相关的背景。

最佳实践

对事件做出反应

处理来自 Partn AWS er Central API 的事件时，请确保您的处理逻辑是等的，可以处理重复的事件。与其立即GetOpportunity调用每个事件，不如考虑根据应用程序的需求批量或有选择地获取详细信息。要实现不间断的操作，请注意配额。

实现乐观锁定

乐观锁定可防止并发更新期间意外的数据覆盖。以下是一个典型场景：

1. 合作伙伴从其 CRM 系统中检索机会。
2. 用户在AWS合作伙伴中心A更新机会。
3. 用户通过 CRM 集成同时B更新相同的机会。
4. 如果数据发生变化，CRM 系统会尝试上传数据，但会返回`ConflictException`。
5. 用户查看错误并手动解决有冲突的数据。

为避免出现这种情况，所有UpdateOpportunity请求都必须包含`LastModifiedDate`参数，您可以从之前的CreateOpportunityUpdateOpportunity、和GetOpportunity操作中获取该参数。只有当我们的系统`LastModifiedDate`匹配时，更新才会成功。如果没有，则必须`LastModifiedDate`使用获取最新版本GetOpportunity并重新尝试更新。

在 CRM 和之间同步数据AWS合作伙伴中心

必须使您的系统与合作伙伴中心的最新数据保持同步。以下是确保您的系统反映最新数据的两种策略：

使用事件（推荐）

1. 使用加载数据ListOpportunities。
2. 订阅机会活动。
3. 应对新的机会或变化。
   - 当您收到`Opportunity Created`、`Opportunity Updated`或`Engagement Invitation Accepted`事件时`GetOpportunity`，使用获取最新数据。
   - 当您收到`Engagement Invitation Rejected`活动时，请删除相应的机会。

使用 ListOpportunities 投票

1. 使用加载数据ListOpportunities。
2. 选择轮询频率，确保轮询频率不太高，以免耗尽每日阅读配额。
3. `LastModifiedDate`从存储的数据中识别最新数据，确保其源自AWS。
4. 调ListOpportunities用时使用`AfterLastModifiedDate`过滤器中的时间戳。

```
{
   "FilterList": [
```

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

5. AWS将返回在时间戳上显示的值之后创建或更新的机会。
6. 使用遍历所有返回的页面`NextToken`，并使用更新系统的数据。GetOpportunity

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

使用AWS合作伙伴中心权益 API

P AWS artner Central Benefits API 简化了合作伙伴在所有AWS合作伙伴计划中发现、申请和管理福利的方式。通过将福利管理集中到一个直观的界面中，该服务创建了一种透明的精英统治，在简化运营的同时奖励合作伙伴。

该 API 提供标准AWS的 API 功能。您可以使用 API 操作或针对您的编程语言或平台量身定制的AWS SDK 来访问它。有关更多信息，请参阅入门AWS和构建工具AWS。

什么是福利？

在AWS合作伙伴网络 (APN) 中，权益代表合作伙伴可以从中获得的优势或能力，AWS以支持其业务增长和客户成功。福利旨在根据合作伙伴的资格和贡献来奖励他们，同时提供有形的价值，帮助合作伙伴扩大AWS业务规模。

福利可以采取多种形式，包括：

- 经济激励 ——积分、现金支出和资助计划，例如Migration Acceleration Program (MAP) 或营销发展基金 (MDF)

- 访问权益-预览对新服务、工具或专业资源的访问权限
- 认可-数字徽章、认证、能力称号和合作伙伴等级状态
- 技术资源-解决方案架构协助、技术支持和 ProServe 参与
- 营销支持- Co-marketing 机会、精选合作伙伴身份和成功案例出版物

探索可用的福利

合作伙伴的福利之旅始于发现他们可以获得哪些福利并了解资格要求。福利 API 通过 Benefits Discovery API 提供全面的发现功能。

列出可用的福利

合作伙伴可以使用 `ListBenefits` API 操作查看所有可用的权益。此操作会检索具有可选筛选功能的福利目录，以帮助合作伙伴快速找到相关的机会。

为了有效地发现收益，合作伙伴可以按以下条件进行筛选：

- 计划-按特定AWS合作伙伴计划进行筛选，例如 MAP（Migration Acceleration Program）、MDF（营销发展基金）、ISV 工作负载迁移、创新沙箱、概念验证或合作伙伴计划资金
- 配送类型-按福利交付方式筛选：`CREDITSCASH`、`DISCOUNT`、`ACCESS`、`RECOGNITION`、或 `RESOURCE`
- 状态-按福利状态筛选：`ACTIVE`或 `DEPRECATED`

`ListBenefits`API 会返回福利摘要，其中包含基本信息，例如福利 ID、名称、描述、关联计划和配送类型。通过此摘要视图，合作伙伴可以快速浏览可用的权益，并确定与其业务目标最相关的权益。

筛选方法示例：

专注于财务激励的合作伙伴可能会按配送类型进行筛选`CREDITS`，`CASH`以查看融资机会。寻求技术支持的合作伙伴可以按配送类型进行筛选，`RESOURCE`或者`ACCESS`查找技术支持权益和工具访问权限。

查看详细的福利信息

一旦合作伙伴确定了感兴趣的权益，他们就可以使用 `GetBenefit` API 操作检索完整的详细信息。这提供了全面的信息，包括：

- 资格标准-合作伙伴必须满足的详细要求才有资格享受福利
- 福利申请架构-合作伙伴在申请时需要提供的具体信息和文件

- 拨款详情-合作伙伴在批准后将获得什么
- 关联计划-福利支持哪些AWS合作伙伴计划

福利申请架构尤其重要，因为它定义了福利申请的结构和必填字段。合作伙伴在创建福利申请之前应仔细审查此架构，以确保他们提前收集所有必要的信息。

> **Important**
>
> 每项福利的资格确定由客户端或用户界面层处理。对于与资金相关的福利，资格通常取决于合作伙伴的记分卡绩效和计划参与情况。

主题

使用福利申请

福利申请可以模拟合作伙伴对特定福利的请求。它收集了根据特定福利条件评估和处理申请所需的所有信息。从草稿创建到最终批准或拒绝，福利申请生命周期经历多个状态。

创建福利申请

合作伙伴通过使用 `CreateBenefitApplication` API 操作创建福利申请来启动福利申请流程。应用程序在创建时进入`PENDING_SUBMISSION`状态，允许合作伙伴在提交审核之前准备完整的信息。

在创建福利申请时，合作伙伴必须提供：

- 福利标识符-所申请福利的 ID 或 ARN
- 配送类型-合作伙伴期望如何获得收益（`CREDITSCASH`、`DISCOUNT`、`ACCESS`、`RECOGNITION`、或`RESOURCE`）
- 福利申请详情-包含福利申请架构定义的福利特定信息的 JSON 文档
- 客户令牌-一种独特的等性令牌，可防止重复提交

合作伙伴可以选择提供：

- 名称和描述-应用程序的 Human-readable 标识符

- 合作伙伴联系人-管理此福利申请的人员的联系信息（最多 1 位联系人）
- 文件附件-支持文档，例如项目计划、SOW、成本证明或客户满意度调查（最多 10 个文件）
- 相关资源-指向相关机会或现有福利分配的链接（最多 10 个资源）
- 标签-用于资源组织和跟踪的 Key-value 配对（最多 200 个标签）

CreateBenefitApplicationAPI 执行软验证，仅检查基本字段类型和模式。在提交过程中会进行全面的业务逻辑验证，允许合作伙伴保存未完成的申请并稍后返回以完成申请。

最佳实践：在创建应用程序之前，合作伙伴应使用中的福利申请架构GetBenefit来准确了解需要哪些信息。这降低了由于数据丢失或无效而导致提交失败的可能性。

更新福利申请

合作伙伴可以使用 UpdateBenefitApplication API 操作修改福利申请草案。这允许合作伙伴在提交之前完善信息、添加文档或更正错误。

更新福利申请时，合作伙伴必须提供：

- 标识符-要更新的福利申请的 ID 或 ARN
- 修订版-乐观锁定的当前版本号
- 福利申请详情-完整的、更新的 JSON 文档

即使只有特定字段发生变化，合作伙伴也必须提交完整的福利申请对象。最佳做法是先使用检索最新的应用程序详细信息GetBenefitApplication，修改必要的字段，然后将完整更新的有效载荷提交给UpdateBenefitApplication。

O@@ ptimistic Locking：修订版字段可确保仅在应用程序自上次检索以来未发生更改时才应用更新。如果修订版与当前数据库值不匹配，则更新会因冲突错误而被拒绝。这样可以防止合作伙伴意外覆盖其他流程或系统所做的更改。

只有在应用程序处于PENDING_SUBMISSION状态时才能执行更新。提交后，申请将进入审核流程，无法再通过此 API 进行更新。

将资源与福利申请关联

合作伙伴可以使用 AssociateBenefitApplicationResource API 操作将福利申请关联到相关资源。这在收益与使用收益的业务环境之间建立了宝贵的联系。

支持的资源类型包括：

- 机会-将福利申请与中的特定客户机会相关联。这对于特定于机会的福利特别有用，例如与客户互动相关的MAP资金或POC积分。
- B@@ ENEFIT_AL LOCATION-链接到现有的福利分配，使合作伙伴能够连锁收益或演示如何利用以前的福利

资源关联只能在提交之前发生。一旦提交福利申请（状态更改为IN_REVIEW），就不允许再进行关联。合作伙伴可以将最多 10 个资源与每个福利申请相关联。

要取消资源关联，合作伙伴可以使用 DisassociateBenefitApplicationResource API 操作。与关联一样，解除关联只能在PENDING_SUBMISSION状态下发生。

> **Important**
>
> 合作伙伴必须拥有相应的权限才能访问福利申请和读取关联的特定资源。API 会在关联操作期间验证这些权限。

提交福利申请

当福利申请完成并可供AWS审核时，合作伙伴使用 SubmitBenefitApplication API 操作提交该申请。提交会触发状态从PENDING_SUBMISSION到的转换，IN_REVIEW并启动特定福利的批准工作流程。

提交后，API 将执行全面验证，包括：

- 必填字段验证-确保福利申请详细信息中的所有必填字段都存在
- 字段格式验证-验证字段值是否与预期的模式和数据类型相匹配
- 业务规则验证-应用特定于福利的业务逻辑和约束
- 资源验证-确认所有关联资源均有效且可访问
- 文件验证-验证所有文件附件是否已成功完成处理

如果验证失败，API 会返回 a，其中 ValidationException 包含详细的错误代码和消息，指示哪些字段需要更正。合作伙伴必须UpdateBenefitApplication先使用解决这些问题，然后才能再次尝试提交。

文件处理要求：如果任何附件仍处于PENDING状态，则无法提交福利申请。合作伙伴必须等待文件处理完成（状态更改为SUCCEEDED）后才能提交。如果文件处理失败（状态FAILED），合作伙伴应上传更正后的版本。

成功提交后：

- 应用程序状态更改为 IN_REVIEW
- 福利所有人的团队会收到有关新提交的通知
- 合作伙伴无法再通过标准更新 API 修改应用程序详细信息
- 申请进入定义的批准工作流程，其中可能包括业务审批、技术批准和财务审批阶段

合作伙伴可以通过检索申请并监视 “阶段” 字段来跟踪提交进度，该字段表示当前的批准阶段（例如，“业务批准”、“技术批准”、“财务批准”）。

管理已提交的申请

提交后，合作伙伴管理福利申请的选项有限，但是 API 提供了针对常见场景的特定操作。

召回福利申请

如果合作伙伴在提交后发现错误或需要进行更改，他们可以使用 RecallBenefitApplication API 操作撤回应用程序。Recall 会将应用程序恢复到PENDING_SUBMISSION状态，允许更新和重新提交。

在召回申请时，合作伙伴应提供：

- 标识符-要召回的福利申请的 ID 或 ARN
- 原因-召回的可选解释（最多 1000 个字符）

reason 字段可以提高可追溯性，有助于AWS了解导致召回的常见问题，为福利申请流程的未来改进提供信息。

召回后，合作伙伴可以使用UpdateBenefitApplication进行必要的更改，然后在准备就绪SubmitBenefitApplication后使用重新提交。

> **Important**
>
> 召回通常仅在早期审查阶段可用。已进入后期批准阶段或已获得批准的申请可能不符合召回资格。

修改福利申请

对于提交后的细微更正，合作伙伴可以使用 AmendBenefitApplication API 操作更新特定字段，而无需撤回整个应用程序。当AWS审阅者在审阅过程中要求作出澄清或更正时，这特别有用。

修改使用 JSON Patch-style 方法，其中合作伙伴指定：

- 路径-标识要更新的字段的 JSONPath 表达式（例如）$.CreditDisbursementDetails.AwsAccountIdForCredits
- 值-字段的新值
- 操作-要执行的操作（目前REPLACE仅支持）

合作伙伴在单次 API 调用中最多可以提交 10 项修改。每项修订都必须包括更新的版本号，以实现乐观锁定。

对于较小的更正，应使用修正案。对于重大更改，合作伙伴应使用RecallBenefitApplication将申请恢复为草稿状态以进行全面更新。

取消福利申请

如果合作伙伴不再需要福利或希望撤回申请，他们可以使用 CancelBenefitApplication API 操作取消申请。取消会将申请转换为CANCELED状态，从而永久结束申请流程。

取消申请时，合作伙伴应提供：

- 标识符-要取消的福利申请的 ID 或 ARN
- 原因-取消的可选解释（最多 1000 个字符）

一旦取消，就无法重新激活应用程序。如果合作伙伴后来决定需要该福利，则他们必须创建新的福利申请。

取消预订的常见原因包括：

- 客户机会丢失或延迟
- 合作伙伴容量限制已改变
- 业务优先事项发生了变化
- 不再需要福利来实现预期目的

查看福利申请详情

合作伙伴可以使用 GetBenefitApplication API 操作检索有关福利申请的完整信息。这提供了应用程序的全面视图，包括：

应用程序元数据：

- 唯一标识符 (ID) 和亚马逊资源名称 (ARN)
- 关联的福利标识符
- 应用程序名称和描述
- 当前状态（PENDING_SUBMISSION、IN_REVIEW、ACTION_REQUIRED、APPROVED、REJECTED、或CANCELED）
- 当前处理阶段

状态信息：

- 状态原因- Human-readable 对当前状态的解释
- 状态原因代码-表示特定问题或要求的结构化代码（例如，“清单不完整”、“缺少项目计划”、“缺少设计胜利”）

应用程序内容：

- 完整的福利申请详情 JSON 文档
- 合作伙伴联系信息
- 带有处理状态的文件附件
- 关联资源（机会或分配）
- 资源标签

审计信息：

- 创建时间戳
- 上次修改时间戳
- 当前修订号

当应用程序处于状态时，ACTION_REQUIRED状态原因代码特别有用。这些守则提供了具体、可操作的指导，说明合作伙伴需要解决哪些问题才能继续申请。

列出福利申请

合作伙伴可以使用 ListBenefitApplications API 操作查看其所有福利申请。这将返回具有强大筛选功能的分页应用程序摘要列表。

合作伙伴可以按以下方式筛选应用程序：

- 程序-查看特定程序（MAP、MDF、沙盒、POC 等）的应用程序
- 配送类型-按配送方式（CREDITS、CASHACCESS、等）筛选
- 福利标识符-查看特定福利的申请
- 状态-按应用程序状态 (PENDING_SUBMISSION、、、IN_REVIEWAPPROVEDREJECTED、CANCELED) 筛选
- 阶段-按批准阶段筛选（业务审批、技术审批、财务审批）
- 关联资源 ARN-查找与特定机会或分配相关的应用程序

列表回复包括基本的摘要信息，例如：

- 应用程序 ID、ARN 和名称
- 关联的福利 ID
- 计划和配送类型
- 当前状态和阶段
- 创建和修改时间戳
- 关联的资源
- 当前修订号
- 福利申请详细信息中的选定字段（由福利所有人定义）

合作伙伴可以使用 “排序” 参数配置结果排序，并提供按创建日期、状态、计划或福利标识符升序或降序排序的选项。

控制面板构建：ListBenefitApplicationsAPI 旨在支持合作伙伴控制面板的创建。通过对应用程序进行筛选和排序，合作伙伴可以生成如下视图：

- 需要操作的应用程序（ACTION_REQUIRED状态）

- 最近提交的申请（按创建日期、IN_REVIEW状态排序）
- 待分配的已批准申请（APPROVED状态）
- 按程序划分的应用程序（按特定程序筛选）

处理福利分配

福利分配是指已授予合伙人的福利。福利申请获得批准后，AWS会创建一个或多个福利分配，为合作伙伴提供实际的积分、资金、访问权限或其他福利履行情况。

了解福利分配

福利分配可以代表各种类型的兑现：

- 财务支出 (CASH)-通过支出跟踪功能直接向合作伙伴支付财务款项
- AWS积分（积分）-适用于具有使用情况跟踪功能的AWS账户的信用代码
- 消耗品分配-使用率跟踪功能的 Pre-approved 资金金额或钱包
- 访问授权（访问权限）-访问系统、工具或资源
- 认可（认可）-数字徽章、认证或身份标识
- 资源（资源）-技术支持、咨询服务或 ProServe 项目

每项福利分配都通过其 “状态” 字段跟踪其生命周期：

- 已@@ 激活-分配可供使用
- 不活跃-分配暂时不可用
- 已@@ 发货-配额已完全使用或已送达

福利分配还包括定义福利何时生效 (StartsAt) 和何时到期 (ExpiresAt) 的时间信息，使合作伙伴能够了解使用福利的时间框架。

列出福利分配

合作伙伴可以使用 `ListBenefitAllocations` API 操作查看其所有福利分配。这将返回具有全面筛选功能的分页分配摘要列表。

合作伙伴可以按以下方式筛选分配：

- 福利标识符-查看特定福利的分配

- 福利申请标识符-查看特定应用程序产生的分配
- 配送类型-按分配类型（CREDITS、CASHACCESS、等）筛选
- 状态-按分配状态 (ACTIVE、INACTIVE、FULFILLED) 筛选

列表响应包括分配摘要，其中包含：

- 分配 ID、ARN 和名称
- 关联的福利 ID 和福利申请 ID（如果适用）
- 配送类型
- 当前状态
- 创建时间戳
- 开始日期

此列表视图使合作伙伴能够构建分配跟踪仪表板并确定：

- 有效分配可供立即使用
- 即将到期的分配
- 已完成分配以进行历史跟踪
- 按计划或配送类型划分的总分配价值

查看详细的分配信息

合作伙伴可以使用 GetBenefitAllocation API 操作检索完整的分配详情。这提供了有关分配的全面信息，包括因分配类型而异的特定于配送的详细信息。

核心分配信息：

- 唯一标识符 (ID) 和亚马逊资源名称 (ARN)
- 分配名称和状态
- 状态原因（对当前状态的解释）
- 关联的福利 ID 和福利申请 ID
- 配送类型
- 创建、更新、开始和到期时间戳

配送详情 (Type-Specific)：

该 FulfillmentDetail 字段包含基于配送类型的不同信息结构：

现金支出详情（现金类型）

对于直接现金付款，配送详情包括：

- 支付金额-支付的资金总额，包括金额价值和货币代码
- 发放详情（如果适用）-有关采购订单或发放事件的信息：
  - 发放编号-付款的唯一标识符
  - 发行金额-以当地货币计算的金额
  - 签发时间-付款时间戳

这种结构既支持单笔付款，也支持预先批准的融资方案，在这种情景中，单笔拨款可能会发生多笔发行。

信用详情（积分类型）

对于AWS积分，配送详情包括：

- 已分配金额-分配的总积分金额
- 已发行金额-已发行的信用总额（可能少于分阶段发行的分配额）
- 信用代码-个人信用代码详细信息阵列：
  - AWS账户 ID-使用信用额度的账户
  - 价值-信用代码的货币价值
  - AWS信用代码-实际的信用代码字符串
  - 状态-信用代码状态 (ACTIVE、INACTIVE、FULFILLED)
  - 签发时间-信用代码的签发时间
  - 到期时间-信用码到期时

这种详细的信用跟踪使合作伙伴能够监控多个账户的信用使用情况，并了解哪些信用额度可用、部分使用或已全部使用。

消耗品分配详情 () Pre-Approved Amounts/Wallets

对于预先批准的资金池，配送详情包括：

- 分配金额-分配中的总金额
- 剩余金额-未使用的金额仍可用
- 已@@ 用金额-已消耗的金额
- 发放详情（如果适用）-分配的采购订单信息

消耗品分配使合作伙伴能够根据需要为符合条件的活动提取资金，并实时跟踪剩余余额。

访问详情（访问类型）

对于访问授权，配送详情包括：

- 描述-详细说明所授予的访问权限及其使用方法

访问权限分配可能会提供对预览服务、专用工具或受限资源的访问权限。描述字段说明了合作伙伴如何激活和使用授予的访问权限。

将分配应用于新的福利申请

福利分配中的 ApplicableBenefitIds 字段确定了可以利用此分配的其他福利。这可以实现福利链，合作伙伴可以使用现有分配来申请额外福利。

例如，合作伙伴可以：

1. 获得预先批准的 MAP 资金分配
2. 为个人客户迁移项目创建福利申请
3. 将这些申请与预先批准的分配相关联
4. 项目获得批准后，从拨款中提取资金

这种方法简化了拥有预先批准资金的合作伙伴的福利申请流程，缩短了个别项目申请的审查周期。

使用AWS合作伙伴中心渠道 API

AWS Partner Central 渠道 API 使您能够以编程方式管理您的授权AWS转售业务。这些 API 允许AWS解决方案提供商、AWS分销商和AWS分销商卖家：

- 向合作伙伴中心报告用于转售计划的AWS管理账户
- 向AWS账户发送邀请以接受合作伙伴协会

- 创建注册转售的最终客户AWS账户和应用合作伙伴折扣所需的关系

借助AWS合作伙伴中心渠道 API，您可以构建自定义自动化，以：

- 将合作伙伴拥有的新AWS账户加入 APN 转售计划
- 将新的最终客户账户加入转售计划
- 加快客户注册和 discount 资格认证

使用项目管理账户 (PMA)

计划管理账户 (PMA) 将您的AWS管理账户与您的渠道计划注册关联起来。使用频道握手自助激活计划管理帐户。当您创建并激活 PMA 时，渠道计划的优惠和折扣将应用于发送到关联AWS管理账户的所有发票。使用以下 API 来报告和管理处理转售和渠道折扣的AWS账户：

- `CreateProgramManagementAccount`: 为客户账单管理添加新账户
- `UpdateProgramManagementAccount`: 修改现有账户
- `DeleteProgramManagementAccount`: 从转售计划中删除账户
- `ListProgramManagementAccounts`: 查看现有账户及其状态

处理频道握手

渠道握手可以确保关键渠道管理操作的安全、相互同意。AWS Partner Central 使用渠道握手来验证AWS管理账户的同意，其目的有三个：激活计划管理账户 (PMA)、确定服务期和提前终止服务期。有三种类型：

- `PROGRAM_MANAGEMENT_ACCOUNT`: 激活 PMA 以应用渠道计划权益时，计划管理账户握手会验证同意
- `START_SERVICE_PERIOD`: 服务期创建握手会建立账单转账承诺的共同协议，并规定最短的通知期或固定的条款
- `REVOKE_SERVICE_PERIOD`: 服务期终止握手可以在双方同意的情况下提前终止现役期。所有握手类型都需要目标AWS管理账户的接受才能生效。

使用以下 API 来管理 PMA 激活和服务期限管理的频道握手：

- `CreateChannelHandshake`: 为 PMA 邀请、服务期的建立或服务期终止创建握手

- `AcceptChannelHandshake`: 接受目标AWS客户的握手（合作伙伴或客户可以使用，具体取决于握手类型）
- `RejectChannelHandshake`: 拒绝目标账户的握手AWS
- `CancelChannelHandshake`: 在待处理的握手被接受、拒绝或到期之前将其取消
- `ListChannelHandshakes`: 通过按类型和状态进行筛选来跟踪和查看握手

处理渠道关系

关系使您能够管理最终客户、内部组织或下游卖家。每种关系都包括：

- 关联AWS组织的AWS管理账户
- 渠道 discount 资格所需的AWS合作伙伴网络元数据

当您创建关系时，关联AWS组织的所有使用都将获得相关的渠道计划权益。使用以下 API 报告卖家和最终客户：

- `CreateRelationship`: 注册新的最终客户、分销商销售商或内部账户AWS
- `GetRelationship`：查看特定的关系详情
- `ListRelationships`: 查看所有关系
- `UpdateRelationship`：修改现有关系
- `DeleteRelationship`: 移除关系

使用收入衡量 API

主题

使用您的收入归因 ID

收入归因 ID 是一种交易级别标识符，可将您的AWS Marketplace 产品收入映射到特定的AWS Marketplace 优惠 and/or AWS合作伙伴中心机会。它建立在产品级合作伙伴收入测量 (PRM) 实施之上：PRM 捕获 Marketplace 产品的总归因收入，收入归因 ID 根据您指定的每月成本分配百分比将收入映射到特定交易。

收入归因服务提供了 API，用于创建、检索、列出和更新收入归因 ID，并异步管理其每月成本分配条目。

什么是收入归因 ID？

收入归因 ID 分为两层：

- 归因是由合作伙伴命名的、由合作伙伴描述的资源，由`ra-`前缀 ID（例如）`ra-aabbccddee001`和 ARN 标识。每个归因的范围都限AWS于`Catalog`（用于生产、`Sandbox`用于测试）和合作伙伴的账户。
- 每个@@ 条目都有一个或多个月度成本分配条目，将单个（优惠 ID 或机会 ID，账单月份）组合映射到成本分配百分比。通过分配任务模式对条目进行异步批量管理。

收入归因 ID 有两种使用方式：

- 作为对现有产品级 PRM 实施的交易级叠加。创建收入归因 ID，关联适用的 Marketpl and/or ace 优惠 ACE 机会，并将您已经衡量的产品收入AWS映射到这些特定交易——无需更改现有资源标签或用户代理字符串。
- 作为独立标识符，直接用作资源标签值 (`aws-apn-id=<RA ID>`) 或在用户代理字符串 (`APN_1.1/pc_<RA ID>$`) 中，从一开始就将AWS消耗归因于特定交易。

使用您的收入归因 ID

合作伙伴可以通过收入归因服务管理收入归因 ID 及其每月成本分配条目。生命周期通过两个独立的流程进行：归因记录流程（创建、更新、检索、列出）和分配流程（启动批量任务、轮询结果、检索单个条目、列出归因条目）。

创建收入归因 ID

第一步是使用 `CreateRevenueAttribution` API 操作创建收入归因 ID。返回的`Id`，`Arn`可以立即用作资源标签值，也可以在用户代理字符串中使用以归因消耗。AWS

在创建收入归因 ID 时，合作伙伴必须提供：

- **`Catalog`**— AWS 用于生产或`Sandbox`测试。
- **`Name`**— 一个人类可读的归因名称。在目录和合作伙伴账户中必须是唯一的。最多 128 个字符。

合作伙伴可以选择提供：

- **Description**— 对归因的自由文字描述。最多 1024 个字符。
- **MarketplaceProduct**— 要与此归因关联的AWS Marketplace 产品。提供ProductIdentifier（25 个字符的 Marketplace 商品编码）和TenancyModel（MULTI_TENANT或SINGLE_TENANT）。如果在创建时省略了归因，则无论客户购买了哪个 Marketplace 产品，归因都适用于以收入归因 ID 衡量的所有消费，这在单个部署跨越多个商城列表时非常有用。
- **Tags**— 最多 200 个用于资源组织的键值对。标签密钥在请求中必须是唯一的。

最佳实践：MarketplaceProduct.ProductIdentifier每当您的交易锚定到特定的 Marketplace 列表时，请提供。这样可以AWS验证产品是否归呼叫合作伙伴账户或通过合作伙伴账户连接 (PAC) 关联的子公司账户所有，并在响应中显示已解决的ProductCode和ProductType（例如AMI，SaaS、、ML）。如果产品不归授权账户所有，则 API 会返回ValidationException并说明原因PRODUCT_NOT_FOUND_OR_NOT_OWNED。

响应返回新的Id（格式ra-[a-z0-9]{13}）Arn、已解析的MarketplaceProduct属性和初始Version数字（从 1 开始，在每次后续更新时递增）。

添加每月成本分配条目

创建收入归因 ID 后，合作伙伴会通过 StartRevenueAttributionAllocationsTask API 操作提交一批每月成本分配条目，从而将AWS Marketplace Offers and/or ACE 机会与其关联起来。这是一个异步操作，每个任务最多可接受 250 个分配更改 (CREATE and/or UPDATE)。

每个分配条目都指定：

- 报价 ID 或机会 ID — 关联交易的唯一标识符。如果 Marketplace 优惠已与 ACE 机会相关联，则合作伙伴只需要提供优惠 ID；AWS自动将收入归因于关联的机会。
- 账单月份 — 成本分配适用的日历月（例如，2026-04）。
- 成本分配百分比 — 本账单月份产品总AWS消耗中归因于该交易的部分。适用于多租户 SaaS 产品，包括用于客户共享基础架构的混合部署的合作伙伴托管组件。
- 客户AWS账户 ID — 消费产品的客户的AWS账户。对于多租户 SaaS 产品来说是必需的。

给定收入归因 ID 和同一账单月份的所有分配条目的总成本分配百分比不得超过 100%。如果总数超过 100%，则在异步业务验证期间，违规条目将被拒绝，并显示每条记录的错误代码。

同步验证：同步StartRevenueAttributionAllocationsTask执行基本形状验证（字段类型、模式、批次大小）。如果基本验证通过，API 将返回TaskId带状态的IN_PROGRESS。业务验证（上限

检查、不可变性规则、针对 Marketplace 和 ACE 的依赖关系查询）异步运行，每条记录的结果是通过发现的。GetRevenueAttributionAllocationsTask

要监控已提交的任务，合作伙伴GetRevenueAttributionAllocationsTask可使用收入归因资源 ID 或 ARN 进行投票。响应从变IN_PROGRESS为COMPLETED（无论个别记录成功还是失败），并返回：

- **TaskStatus**— IN_PROGRESS COMPLETED、或FAILED（顶级任务失败）。
- **RecordResults**— 对于每条输入记录：分配的AllocationId（如果成功）、每条记录的状态（SUCCEEDED或FAILED），以及结构化错误代码和消息（如果记录失败）。

合作伙伴应及时检索任务结果。任务完成后，可以协调每条记录的结果，并且可以在新任务中更正任何失败的条目并重新提交。

编辑每月成本分配条目

按账单月份管理成本分配条目，规则如下：

- 合作伙伴可以随时为当前或任何未来的账单月度添加新的月度条目。
- 合作伙伴可以随时更新当前或任何未来账单月份的月度条目。
- 合作伙伴可以在当月 7 日之前更新上个月的条目。此窗口允许合作伙伴在最终确定上个月的分配之前，在AWS Cost Explorer 中查看实际使用情况。

当月 7 日之后，无法再修改前一个账单月份的归因。当前或未来月份条目的更新将从下一个月度账单周期开始生效。不对前几个月的历史归因进行追溯性重新计算。

分配条目的更新通过相同的 StartRevenueAttributionAllocationsTask API 操作提交，每个受影响的条目均Operation设置为。UPDATE异步任务模式统一处理更新和创建。

更新收入归因 ID

合作伙伴可以使用 UpdateRevenueAttribution API 操作更新现有收入归因 ID。Description否则，归因记录本身是不可变的—— Name MarketplaceProduct、，并且创建时标记的值在创建后无法更改。要重新标记资源，请对归因的 ARN 使用标准AWS标记操作。

更新收入归因 ID 时，合作伙伴必须提供：

- **Catalog**— AWS 或Sandbox。

- **Identifier**— 要更新的归因的 ra- ID。
- **Version**— 乐观锁定归因的当前版本。

合作伙伴可以选择提供：

- **Description**— 更新的自由文本描述。最多 1024 个字符。
- **ClientToken**— 更新的等性标记。

Optimistic Locking：该Version字段可确保只有在归因自上次检索以来未发生变化时才会应用更新。如果提交的内容与当前资源版本Version不匹配，则更新将被拒绝，ConflictException并显示一条包含已提交版本号和当前版本号的消息。最佳做法是在每次更新GetRevenueAttribution之前检索最新版本。

查看收入归因 ID 详情

合作伙伴可以使用 GetRevenueAttribution API 操作检索单个收入归因 ID 的完整信息。这将返回：

资源元数据：

- 唯一标识符 (Id) 和 Amazon 资源名称 (Arn)。
- Catalog存在归因的。
- Name 和 Description
- 检索到的Version和LatestVersion（使用最新版本进行后续更新）。

Marketplace 商品属性（如果MarketplaceProduct是在创建时设置的）：

- ProductId— 合作伙伴提供的 Marketplace 产品编号。
- ProductCode—AWS Marketplace 产品代码解析自ProductId。
- ProductType— SaaS、AMI、Container、ML、Data、或Professional Services。
- TenancyModel— MULTI_TENANT 或SINGLE_TENANT。

审计信息：

- CreatedDate 和 LastModifiedDate

合作伙伴可以选择在请求Version中提供具体信息，以检索归因的历史版本。省略Version返回最新版本。

要检索特定的每月成本分配条目，合作伙伴可使用带有该条目的GetRevenueAttributionAllocation API操作AllocationId。这将返回报价 ID 或机会 ID、账单月份、成本分配百分比、客户AWS账户 ID 以及条目的状态和审计信息。

列出收入归因 ID

合作伙伴可以使用 ListRevenueAttributions API 操作查看其账户中的所有收入归因 ID。这将返回具有筛选和排序功能的归因摘要分页列表。

合作伙伴可以按以下条件筛选结果：

- **Catalog**— AWS 或Sandbox（必填）。
- **Identifiers**— 最多可检索 100 个特定 ra- ID 的列表。
- **CreatedAfter**/**CreatedBefore**— 按创建时间戳范围（含）筛选。

合作伙伴可以使用以下Sort参数配置结果排序：

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

响应包括每个归因的摘要：Arn、Id、Catalog、Name、已解决的MarketplaceProduct属性CreatedDateLastModifiedDate、、、Version、最新和TotalRevenueAttributionAssociationCount（与归因关联的每月有效成本分配条目数量）。

使用MaxResults（默认 25，最大 100）和NextToken对大型结果集进行分页。NextToken如果有其他页面可用，则响应中包含一个。

列出每月成本分配条目

合作伙伴可以使用 ListRevenueAttributionAllocations API 操作列出特定收入归因 ID 的每月成本分配条目。这将返回具有筛选功能的分配摘要的分页列表。

合作伙伴可以按以下条件筛选结果：

- **MarketplaceOfferId**— 仅列出与特定 Marketplace 优惠相关的条目。

- **AwsPartnerCentralOpportunityId**— 仅列出与特定合作伙伴中心机会相关的条目。
- **BillingMonth**— 仅列出特定日历月的条目。

响应包括每个条目的摘要信息：AllocationId、关联的报价 ID 或机会 ID、客户AWS账户 ID、账单月份、成本分配百分比、条目的名称和状态以及审计时间戳。

控制面板构建：ListRevenueAttributions和的组合ListRevenueAttributionAllocations旨在支持合作伙伴控制面板的创建。合作伙伴可以建立视图，例如：

- 过去 30 天内创建的所有收入归因 ID，按创建日期排序。
- 跨多个收入归因 ID 的某个 Marketplace 优惠的所有月度成本分配条目。
- 当前账单月份的所有成本分配条目，以验证每次归因的总费用不超过 100%。
- 至少有一个有效分配条目的所有收入归因 ID (TotalRevenueAttributionAssociationCount > 0)。

处理你的 Marketplace 收入份额

Marketplace 收入份额是AWS Marketplace 产品的输入，它申报了通过 M AWS arketplace 收取的产品总收入的部分。AWS使用此输入将合作伙伴的 Marketplace-collected 收入与其他收入信号关联起来。

Marketplace 收入共享服务提供了 API，用于创建、检索和列出 Marketplace 收入分成资源并管理其分配（合作伙伴申报的收入分成百分比和生效日期窗口）。Marketplace 收入共享资源支持标准AWS标签操作。

什么是 Marketplace 收入分成？

Marketplace 收入份额有两个层次：

- 份额 — 单个AWS Marketplace 产品的输入，由 Marketplace 标识ProductId。每种产品只有一个 Marketplace 收入份额。
- 一个或多个分配 — 每个分配都声明一个RevenueSharePercent和一个有效日期窗口（EffectiveDate可选ExpirationDate）。随着时间的推移，一个股票可以有多个分配，但任何时候都只能适用一种ACTIVE分配。

分配具有以下关键属性：

- 状态 — ACTIVE 或INACTIVE。创建时，状态默认为ACTIVE。只有ACTIVE分配参与收入计算和重叠检查。
- 重叠不变 — 在单个份额内，任何两个ACTIVE分配的范围都不能重叠[EffectiveDate, ExpirationDate]。任何时候每股最多允许一次无限期ACTIVE配股（否ExpirationDate）。

处理你的 Marketplace 收入份额

合作伙伴通过 Marketplace 收入分成服务管理 Marketplace 收入份额及其分配。生命周期通过两个流程进行：共享流程（创建、检索、列出、标记）和分配流程（创建、更新、软删除、检索、列出）。

创建 Marketplace 收入分成

第一步是使用 CreateMarketplaceRevenueShare API 操作为 Marketplace 产品创建AWS Marketplace 收入分成。该份额由 Marketplace 标识，ProductId并作为所有后续分配的容器。

在创建 Marketplace 收入分成时，合作伙伴必须提供：

- **Catalog**— AWS 用于生产或Sandbox测试。
- **ProductId**— 25 个字符的 Marketplace 商品编码。图案：[a-z0-9]{25}。

合作伙伴可以选择提供：

- **Tags**— 创建时资源组织最多 50 个键值对。用于TagResource在创建后添加标签。
- **ClientToken**— 一种等性标记，用于防止在重试时重复创建。最多 64 个字符。SDK 会自动生成此令牌。

产品验证：创建后，该服务会向AWS Marketplace 验证产品是否归来电者的账户或通过合作伙伴账户连接 (PAC) 与来电者关联的子账户所有。如果产品不存在或调用者未授权拥有账户，则 API 会返回ValidationException并说明原因PRODUCT_NOT_FOUND_OR_NOT_OWNED。

每种产品一股：第二次CreateMarketplaceRevenueShare要求ProductId在相同的Catalog回报中获得相同的股份ConflictException。

响应返回共享的Arn（格式arn:{partition}:partnercentral:{region}:{account-id}:catalog/{catalog}/marketplace-revenue-share/{productId}）、、、初始Version数字（从 1 开始，每次分配突变时递增）以及创建时应用的所有标签。Catalog ProductId

添加分配

创建 Marketplace 收入份额后，合作伙伴会通过 API 操作创建分配，从而申报收入分成百分比和生效日期窗口。CreateMarketplaceRevenueShareAllocation

创建分配时，合作伙伴必须提供：

- **Catalog**— AWS 或Sandbox。
- **MarketplaceRevenueShareIdentifier**— 父共享的标识符。
- **RevenueSharePercent**— 通过Marketplace收集的产品收入的百分比，以十进制形式提供（0.45例如，45%）。
- **EffectiveDate**— 此分配生效的日历日期。格式：YYYY-MM-DD。

响应返回分配的Id（格式mrsa-[A-Za-z0-9]{13}）、分配状态（ACTIVE创建时）和父份额的增量Version（每次分配突变都会增加每股一个版本整数，用于对后续分配更新进行乐观锁定）。

更新分配

合作伙伴可以使用 UpdateMarketplaceRevenueShareAllocation API 操作更新分配。可更新的字段取决于分配是未来日期、当前有效还是过期。

更新分配时，合作伙伴必须提供：

- **Catalog**— AWS 或Sandbox。
- **MarketplaceRevenueShareIdentifier**— 父共享的标识符。
- **AllocationId**— 要更新的分配的 mrsa- ID。
- **MarketplaceRevenueShareVersion**— 当前版本的母股，用于乐观锁定分配。碰撞冲突又回ConflictException来了。

合作伙伴可以选择提供：

- **RevenueSharePercent**— 更新的收入分成百分比。仅在未来分配时可编辑。
- **EffectiveDate**— 更新的生效日期。仅在未来分配时可编辑。
- **ExpirationDate**— 更新的到期日期。在远期分配（任何值> EffectiveDate）和当前有效或开放式分配（任何值）上均可编辑。≥ today
- **Status**— ACTIVE 或INACTIVE。设置为INACTIVE软删除分配。仅在未来分配时可编辑。

- **ClientToken**— 一个等性代币。

查看 Marketplace 收入分享详情

合作伙伴可以使用 GetMarketplaceRevenueShare API 操作检索单个 Marketplace 收入份额的完整信息。这将返回共享的Arn、Catalog、ProductId、Version（父共享的当前版本）CreatedDate、LastModifiedDate、以及应用于共享的所有标签。

合作伙伴可以使用 GetMarketplaceRevenueShareAllocation API 操作和分配来检索单个分配Id。这将返回：

- 配额Id和母股MarketplaceRevenueShareArn。
- RevenueSharePercent, EffectiveDate, ExpirationDate.
- Status（ACTIVE或INACTIVE）。
- 父共享MarketplaceRevenueShareVersion用于乐观锁定后续更新。
- CreatedDate 和 LastModifiedDate

列出市场收入份额

合作伙伴可以使用 ListMarketplaceRevenueShares API 操作查看其账户拥有的所有 Marketplace 收入份额。这将返回分页的分享摘要列表。

合作伙伴可以按以下条件筛选结果：

- **Catalog**— AWS 或Sandbox（必填）。
- **ProductIds**— 要检索的特定 Marketplace 商品编号的列表。

响应包括每份共享的摘要：ArnCatalog、ProductId、、最新的VersionCreatedDate、和LastModifiedDate。

使用MaxResults和NextToken对大型结果集进行分页。

列出分配

合作伙伴可以使用 ListMarketplaceRevenueShareAllocations API 操作列出特定 Marketplace 收入份额的分配。这将返回具有筛选功能的分配摘要的分页列表。

合作伙伴可以按以下条件筛选结果：

- **Status**— ACTIVE 或INACTIVE。
- 有效日期范围 —EffectiveDateOnOrAfter/EffectiveDateOnOrBefore用于检索有效日期在特定窗口内的分配。

响应包括每个分配的摘要信息：Id、RevenueSharePercent、EffectiveDate、ExpirationDate、StatusMarketplaceReven和审计时间戳。

标记 Marketplace 收入份额

Marketplace 收入共享资源支持标准AWS标签操作：

- **TagResource**— 在 Marketplace 收入份额上添加或覆盖标签。每次共享最多 50 个标签。
- **UntagResource**— 按密钥删除特定标签。
- **ListTagsForResource**— 检索当前应用于 Marketplace 收入份额的所有标签。

分配不支持标签。为父共享添加标签以组织相关分配。

每个标记操作都接受共享的Arn（格式arn:{partition}:partnercentral:{region}:{account-id}:catalog/{catalog}/marketplace-revenue-share/{productId}）作为资源标识符。还通过AWS字段支持标准的创建时标记。Tags CreateMarketplaceRevenueShare

支持 AWS Regions

以下部分提供有关支持的 AWS 区域的信息。

主题

- 支持 AWS 的区域 AWS 合作伙伴中心账户 API
- 支持 AWS 的区域 AWS Partner Central Selling API
- 支持 AWS 的区域 AWS 合作伙伴中心权益 API
- 支持 AWS 的区域 AWS 合作伙伴中心渠道 API
- 支持 AWS 的区域 AWS 合作伙伴中心收入衡量 API

支持 AWS 的区域 AWS 合作伙伴中心账户 API

您可以通过以下终点从任何 AWS 美国东部（弗吉尼亚北部）地区访问 AWS 合作伙伴中心账户服务。

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

支持 AWS 的区域 AWS Partner Central Selling API

您可以通过以下终点从任何 AWS 美国东部（弗吉尼亚北部）地区访问 AWS 合作伙伴中心销售服务。

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

支持 AWS 的区域 AWS 合作伙伴中心权益 API

您可以通过以下终点从任何 AWS 美国东部（弗吉尼亚北部）地区访问 AWS 合作伙伴中心福利服务。

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

支持 AWS 的区域 AWS 合作伙伴中心渠道 API

您可以通过以下端点从任何 AWS 商业区域访问 AWS 合作伙伴中心渠道管理服务。

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

使用 sigv4a 对请求进行签名

由于 AWS 合作伙伴中心渠道 API 使用全球终端节点，因此请求使用签名版本 4a (Sigv4a) 进行签名，该版本是 Sigv 4 的扩展，支持使用非对称加密进行签名，从而实现跨多个区域可验证的签名。 AWS

使用 AWS SDK 或 CLI

如果您使用 S AWS DK 或 AWS CLI，则在您调用全局端点时会自动处理 Sigv4A 签名。无需其他配置。

手动签署请求

如果您在没有 SDK 的情况下手动签署请求，请注意与标准 Sigv4 的以下区别：

- Sigv4a 从您现有的 AWS 私有访问密钥中派生椭圆曲线数字签名算法 (ECDSA) 密钥对，而不是使用密钥派生。 HMAC-based
- 凭证范围*用于区域组件，而不是特定的区域名称，因为该签名在各个区域都有效。

有关 Sigv4a 工作原理的更多信息，请参阅适用于 API 请求的 AWS 签名版本 4。有关 sigv4a 签名的代码示例，请参阅上的 sigv4a-signing-examples 项目。 GitHub

支持 AWS 的区域 AWS 合作伙伴中心收入衡量 API

您可以通过以下终点从任何 AWS 商业区域访问 AWS 合作伙伴中心收入衡量服务。

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

使用 sigv4a 对请求进行签名

由于收入衡量服务使用全局终端节点，因此请求使用签名版本 4a (Sigv4a) 进行签名，该版本是 Sigv 4 的扩展，支持使用非对称加密进行签名，从而实现跨多个区域可验证的签名。 AWS

使用 AWS SDK 或 CLI

如果您使用 S AWS DK 或 AWS CLI，则在您调用全局端点时会自动处理 Sigv4A 签名。无需其他配置。

手动签署请求

如果您在没有 SDK 的情况下手动签署请求，请注意与标准 Sigv4 的以下区别：

- Sigv4a 从您现有的 AWS 私有访问密钥中派生椭圆曲线数字签名算法 (ECDSA) 密钥对，而不是使用密钥派生。 HMAC-based
- 凭证范围*用于区域组件，而不是特定的区域名称，因为该签名在各个区域都有效。

有关 Sigv4a 工作原理的更多信息，请参阅适用于 API 请求的 AWS 签名版本 4。有关 sigv4a 签名的代码示例，请参阅上的 sigv4a-signing-examples 项目。 GitHub

Permissions

以下各节提供有关权限的信息。

主题

- 访问福利 API
- 频道 API 的访问权限
- 访问收入衡量 API

账户 API 的访问权限

访问控制和权限由AWS Identity and Access Management(IAM) 管理。本节提供有关配置与账户 API 交互的必要权限的指导。

先决条件

在配置权限之前，请确保您的AWS 账户已关联并已创建必要的 IAM 角色和用户。有关更多信息，请参阅 设置和身份验证。

使用AWS托管策略

AWS提供托管策略，授予与账户 API 交互所需的权限。要提供管理账户资源的必要访问权限，请将`AWSPartnerCentralFullAccess`策略附加到您的 IAM 身份。有关更多信息，请参阅用户AWS托管策略。

向 IAM 角色和用户分配策略

按照以下步骤将策略分配给 IAM 角色和用户：

1. 登录到AWS 管理控制台。
2. 导航到 IAM 服务。
3. 选择角色或用户，然后选择要向其关联策略的 IAM 角色或用户。
4. 将`AWSPartnerCentralFullAccess`策略附加到选定的 IAM 角色或用户。

有关更多信息，请参阅添加和删除 IAM 身份权限。

使用条件键管理权限

IAM 策略中的条件密钥为何时强制执行声明策略提供资源级权限。您可以使用条件键来指定何时允许或拒绝某些权限的条件。

有关更多信息，请参阅 IAM JSON 策略元素：条件运算符。

条件键概述

条件键	说明	适用的行动	有效值
合作伙伴中心：目录	按关联目录实体的类型筛选访问权限	所有操作	AWS，沙箱

所需权限摘要

所需权限摘要

处理建议	说明
合作伙伴中心：AcceptConnectionInvitation	允许接受连接邀请
合作伙伴中心：AssociateAwsTrainingCertificationEmailDomain	允许关联AWS培训认证电子邮件域
合作伙伴中心：CancelConnection	允许取消连接
合作伙伴中心：CancelConnectionInvitation	允许取消连接邀请
合作伙伴中心：CancelProfileUpdateTask	允许取消个人资料更新任务
合作伙伴中心：CreateConnectionInvitation	允许创建连接邀请
合作伙伴中心：CreatePartner	允许创建合作伙伴
合作伙伴中心：DisassociateAwsTrainingCertificationEmailDomain	允许取消关联AWS培训认证电子邮件域
合作伙伴中心：GetAllianceLeadContact	允许检索联盟负责人的联系方式
合作伙伴中心：GetConnection	允许检索连接详细信息
合作伙伴中心：GetConnectionInvitation	允许检索连接邀请的详细信息
合作伙伴中心：GetConnectionPreferences	允许检索连接首选项
合作伙伴中心：GetPartner	允许检索合作伙伴的详细信息

处理建议	说明
合作伙伴中心：GetProfileUpdateTask	允许检索配置文件更新任务详情
合作伙伴中心：GetProfileVisibility	允许检索个人资料可见性设置
合作伙伴中心：GetVerification	允许检索验证详情
合作伙伴中心：ListConnectionInvitations	允许列出连接邀请
合作伙伴中心：ListConnections	允许列出连接
合作伙伴中心：ListPartners	允许列出合作伙伴
合作伙伴中心：PutAllianceLeadContact	允许更新联盟负责人的联系方式
合作伙伴中心：PutProfileVisibility	允许更新个人资料可见性设置
合作伙伴中心：RejectConnectionInvitation	允许拒绝连接邀请
合作伙伴中心：SendEmailVerificationCode	允许发送电子邮件验证码
合作伙伴中心：StartProfileUpdateTask	允许启动个人资料更新任务
合作伙伴中心：StartVerification	允许启动验证进程
合作伙伴中心：UpdateConnectionPreferences	允许更新连接首选项

访问销售 API

访问控制和权限由AWS Identity and Access Management(IAM) 管理。本节提供有关配置与 API 交互的必要权限的指导，包括列出AWS Marketplace实体所需的权限。

先决条件

在配置权限之前，请确保您的AWS 账户已链接到合作伙伴中心，并且您已创建必要的 IAM 角色和用户。有关更多信息，请参阅设置和身份验证。

使用AWS托管策略

AWS提供托管策略，授予与 API 交互所需的权限。要提供管理机会所需的访问权限，请将AWSPartnerCentralOpportunityManagement策略附加到您的 IAM 身份。有关更多信息，请参阅AWS合作伙伴中心用户的AWS托管政策。

AWSPartnerCentralOpportunityManagement 政策

此政策授予对合作伙伴中心机会管理操作的完全访问权限。

JSON

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

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

自定义策略

如果托管策略不能满足您的需求，请创建自定义 IAM 策略来授予您的用例所需的权限。以下示例是授予列出AWS Marketplace实体的权限的自定义策略：

Example自定义策略示例

JSON

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

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

自定义许可策略

本政策为合作伙伴平台的销售活动提供了广泛的访问权限，包括将来无需更新政策即可添加的功能。通过使用通配符操作`partnercentral:*`，此政策会在新的合作伙伴中心销售功能可用时自动授予访问权限，从而减少手动更新的需求。该政策还包括与AWS Marketplace实体互动的权限，这有助于确保销售和 Marketplace 操作均保持访问权限。

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"
                }
            }
        }
    ]
}
```

向 IAM 角色和用户分配策略

按照以下步骤向 IAM 角色和用户分配策略：

1. 登录到AWS 管理控制台。
2. 导航到 IAM 服务。
3. 选择角色或用户，然后选择要向其关联策略的 IAM 角色或用户。
4. 将`AWSPartnerCentralOpportunityManagement`策略或您的自定义策略附加到选定的 IAM 角色或用户。

有关更多信息，请参阅添加和删除 IAM 身份权限。

使用条件键管理权限

IAM 策略中的条件密钥为何时强制执行声明策略提供资源级权限。您可以使用条件键来指定何时允许或拒绝某些权限的条件。

有关更多信息，请参阅 IAM JSON 策略元素：条件运算符。

条件键概述

条件键	说明	适用的行动	有效值
合作伙伴中心：目录	按关联目录实体的类型筛选访问权限	所有操作	AWS，沙箱
aws 市场：PartyType	根据当事方的类型（例如提案人）筛选访问权限	SearchAgreements, DescribeAgreement	投标人

所需权限摘要

所需权限摘要

处理建议	说明
合作伙伴中心：CreateOpportunity	允许创造机会
合作伙伴中心：UpdateOpportunity	允许更新机会
合作伙伴中心：ListOpportunities	允许上市机会
合作伙伴中心：GetOpportunity	允许检索机会详情
合作伙伴中心：ListSolutions	允许列出解决方案
合作伙伴中心：AssociateOpportunity	允许将机会与其他实体关联起来
合作伙伴中心：DisassociateOpportunity	允许解除机会与其他实体的关联
合作伙伴中心：AcceptEngagementInvitation	允许接受订婚邀请
合作伙伴中心：RejectEngagementInvitation	允许拒绝参与邀请
合作伙伴中心：GetEngagementInvitation	允许检索参与邀请详情
合作伙伴中心：ListEngagementInvitations	允许发布订婚邀请
合作伙伴中心：SubmitOpportunity	允许提交机会

处理建议	说明
合作伙伴中心：GetAwsOpportunitySummary	允许检索AWS机会摘要
合作伙伴中心：StartProspectingFrom EngagementTask	根据互动创建潜在客户任务
合作伙伴中心：GetProspectingFromEn gagementTask	返回探矿任务详情
合作伙伴中心：ListProspectingFromE ngagementTasks	返回探矿任务列表
aws 市场：ListEntities	允许列出AWS Marketplace实体
aws 市场：DescribeEntity	允许描述AWS Marketplace实体
aws 市场：SearchAgreements	允许在中搜索协议AWS Marketplace
aws 市场：DescribeAgreement	允许在中描述协议AWS Marketplace

访问福利 API

访问控制和权限由AWS Identity and Access Management(IAM) 管理。本节提供有关配置与 Benefits API 交互的必要权限的指导。

先决条件

在配置权限之前，请确保您的AWS 账户已关联并已创建必要的 IAM 角色和用户。有关更多信息，请参阅 设置和身份验证。

使用AWS托管策略

AWS提供托管策略，这些策略授予与 Benefits API 交互所需的权限。要提供管理福利资源的必要访问权限，请将`AWSPartnerCentralFullAccess`策略附加到您的 IAM 身份。有关更多信息，请参阅用户AWS托管策略。

向 IAM 角色和用户分配策略

按照以下步骤将策略分配给 IAM 角色和用户：

1. 登录到AWS 管理控制台。
2. 导航到 IAM 服务。
3. 选择角色或用户，然后选择要向其关联策略的 IAM 角色或用户。
4. 将`AWSPartnerCentralFullAccess`策略附加到选定的 IAM 角色或用户。

有关更多信息，请参阅添加和删除 IAM 身份权限。

使用条件键管理权限

IAM 策略中的条件密钥为何时强制执行声明策略提供资源级权限。您可以使用条件键来指定何时允许或拒绝某些权限的条件。

有关更多信息，请参阅 IAM JSON 策略元素：条件运算符。

条件键概述

条件键	说明	适用的行动	有效值
合作伙伴中心：目录	按关联目录实体的类型筛选访问权限	所有操作	AWS，沙箱

所需权限摘要

所需权限摘要

处理建议	说明
合作伙伴中心：AmendBenefitApplication	允许修改福利申请
合作伙伴中心：AssociateBenefitApplicationResource	允许将资源与福利应用程序关联
合作伙伴中心：CancelBenefitApplication	允许取消福利申请
合作伙伴中心：CreateBenefitApplication	允许创建福利申请
合作伙伴中心：DisassociateBenefitApplicationResource	允许解除资源与福利应用程序的关联

处理建议	说明
合作伙伴中心：GetBenefit	允许检索福利详情
合作伙伴中心：GetBenefitAllocation	允许检索福利分配详情
合作伙伴中心：GetBenefitApplication	允许检索福利申请详情
合作伙伴中心：ListBenefitAllocations	允许列出福利分配
合作伙伴中心：ListBenefitApplications	允许列出福利申请
合作伙伴中心：ListBenefits	允许上市福利
合作伙伴中心：RecallBenefitApplication	允许召回福利申请
合作伙伴中心：SubmitBenefitApplication	允许提交福利申请
合作伙伴中心：UpdateBenefitApplication	允许更新福利申请

频道 API 的访问权限

访问控制和权限由AWS Identity and Access Management(IAM) 管理。本节提供有关配置与AWS合作伙伴中心渠道 API 进行交互的必要权限的指导。

频道管理账号设置

P AWS artner Central 上的渠道管理功能要求在两种类型的AWS账户中部署 IAM 角色：

1. AWS合作伙伴中心已链接AWS 账户：

   这是与您的AWS合作伙伴网络和合作伙伴中心账户AWS 账户关联的。AWS 账户每个AWS合作伙伴只能关联一个合作伙伴中心。作为一次性设置活动，您将AWS 账户使用 Partner Central 渠道托管策略在此活动中创建 IAM 角色。

2. 项目管理账户AWS 账户：

   此 Bill-Transfer 账户AWS 账户用作管理最终客户消费账单和付款的账户。AWS 账户这是在AWS合作伙伴中心渠道管理中报告为项目管理账户。您可能有多个项目管理账户，并且需要在AWS 账户您报告的每个账户中创建一个 IAM 角色作为项目管理账户。

对于每AWS 账户种类型，您需要将不同的 IAM 角色与特定的权限和信任策略相关联。

的访问权限AWS已链接合作伙伴中心AWS 账户

在链接的AWS合作伙伴中心中AWS 账户，创建一个 IAM 角色来管理渠道资源，该角色将授予合作伙伴中心用户。此 IAM 角色允许用户在 P AWS artner Central 上查看和管理渠道管理资源。角色名称必须以开头PartnerCentralRoleFor，推荐的名称必须是PartnerCentralRoleForChannel。要配置角色，请完成以下操作：

- 添加以下自定义信任策略，以允许 P AWS artner Central 云管理员和联盟潜在客户将 IAM 角色映射到合作伙伴中心用户：

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

- 将AWSPartnerCentralChannelManagement托管策略与PartnerCentralRoleForChannel角色关联。

项目管理账户的访问权限AWS账户

在 P AWS artner Central 中AWS 账户报告为计划管理账户的每个账户中，创建以下 IAM 角色，所需名称如下：

- PartnerCentralChannelHandshakeApprovalManagement: 利用此角色来管理合作伙伴中心与您的项目管理层之间的握手。AWS 账户此 IAM 角色可用于响应在AWS合作伙伴中心激活您的计划管理账户。创建此角色时，请关联AWSPartnerCentralChannelHandshakeApprovalManagement托管策略。
- PartnerCentralChannelBillingTransferReadOnly：利用此跨账户角色可以从AWS合作伙伴中心查看账单转账状态。此角色将共享从您AWS 账户到AWS合作伙伴中心的账单转账状态，以便集中查看账单转账状态。如果创建了此角色，则AWS合作伙伴平台中的用户可以在合作伙伴

中心中查看账单转账状态。AWS但是，此角色不向AWS合作伙伴中心用户授予在 Billing and Cost Management 控制台中访问AWS账单转账的权限。

- 信任策略：
  - 在以下 JSON 中，在{PartnerCentralLinkedAccountID}创建角色时，将替换为您实际的AWS合作伙伴中心关联的AWS 账户 12 位数字 ID。

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

- 权限策略：

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

- PartnerCentralChannelBillingTransferManagement（可选）：利用此跨账户角色可以从AWS合作伙伴中心创建和管理账单转账。此角色将允许访问包含AWSPartnerCentralChannelManagement托管策略的 IAM 角色的AWS合作伙伴中心用户管

理计划管理账户中的账单转账AWS 账户。使用此跨账户角色，Partner Central 用户将能够使用AWS合作伙伴中心渠道管理中的 “账单管理” 按钮，在 B AWS illing and Cost Management 控制台中访问和管理AWS账单转账。AWS

- 信任策略：
  - 在以下 JSON 中，在{PartnerCentralLinkedAccountID}创建角色时，将替换为您实际的AWS合作伙伴中心关联的AWS 账户 12 位数字 ID。

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

- 权限策略：

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

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

使用AWS托管策略

AWS提供托管策略，授予与频道 API 交互所需的权限。要提供管理所有渠道资源的必要访问权限，请将AWSPartnerCentralChannelManagement策略附加到您的 IAM 身份。要提供查看和响应频道握手请求所需的访问权限，请将AWSPartnerCentralChannelHandshakeApprovalManagement策略附加到您的 IAM 身份。有关更多信息，请参阅AWS合作伙伴中心用户的AWS托管政策。

AWSPartnerCentralChannelManagement 政策

此政策授予对合作伙伴中心渠道管理操作的完全访问权限。

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

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

AWSPartnerCentralChannelHandshakeApprovalManagement 政策

此政策授予查看和响应频道握手请求的权限。此政策应适用于接收频道握手请求的AWS账户中的角色。

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

向 IAM 角色和用户分配策略

按照以下步骤向 IAM 角色和用户分配策略：

1. 登录到AWS 管理控制台。
2. 导航到 IAM 服务。
3. 选择角色或用户，然后选择要向其关联策略的 IAM 角色或用户。
4. 将AWSPartnerCentralChannelManagement策略或您的自定义策略附加到选定的 IAM 角色或用户。

有关更多信息，请参阅添加和删除 IAM 身份权限。

使用条件键管理权限

IAM 策略中的条件密钥为何时强制执行声明策略提供资源级权限。您可以使用条件键来指定何时允许或拒绝某些权限的条件。

有关更多信息，请参阅 IAM JSON 策略元素：条件运算符。

条件键概述

条件键	说明	适用的行动	有效值
合作伙伴中心：目录	按关联目录实体的类型筛选访问权限	所有渠道管理操作	AWS，沙盒
合作伙伴中心：ChannelHandshakeType	根据频道握手的类型筛选访问权限	CreateChannelHandshake, AcceptChannelHandshake, RejectChannelHandshake, CancelChannelHandshake, ListChannelHandshakes	计划管理账户、开始服务周期、撤销服务周期

所需权限摘要

所需权限摘要

处理建议	说明
合作伙伴中心：CreateProgramManagementAccount	允许创建项目管理账户
合作伙伴中心：UpdateProgramManagementAccount	允许更新项目管理账户
合作伙伴中心：DeleteProgramManagementAccount	允许删除项目管理帐户
合作伙伴中心：ListProgramManagementAccounts	允许列出项目管理账户

处理建议	说明
合作伙伴中心：GetProgramManagement Account	允许检索项目管理账户详细信息
合作伙伴中心：CreateRelationship	允许创建关系
合作伙伴中心：UpdateRelationship	允许更新关系
合作伙伴中心：DeleteRelationship	允许删除关系
合作伙伴中心：GetRelationship	允许检索关系详情
合作伙伴中心：ListRelationships	允许列出关系
合作伙伴中心：CreateChannelHandshake	允许创建频道握手
合作伙伴中心：AcceptChannelHandshake	允许接受频道握手
合作伙伴中心：RejectChannelHandshake	允许拒绝频道握手
合作伙伴中心：CancelChannelHandshake	允许取消频道握手
合作伙伴中心：ListChannelHandshakes	允许列出频道握手

访问收入衡量 API

访问控制和权限由AWS Identity and Access Management(IAM) 管理。本节提供有关配置与收入衡量 API 交互的必要权限的指导，包括列出AWS Marketplace实体所需的权限。

先决条件

在配置权限之前，请确保您的AWS 账户已链接到合作伙伴中心，并且您已创建必要的 IAM 角色和用户。有关更多信息，请参阅设置和身份验证。

使用AWS托管策略

AWS提供托管策略，授予与收入衡量 API 交互所需的权限。根据您的角色，有两种托管策略可供选择：

- 合作伙伴 — 将AWSPartnerCentralRevenueAttributionManagement策略附加到您的 IAM 身份。
- 客户 — 将AWSRevenueAttributionManagement策略附加到您的 IAM 身份。

有关更多信息，请参阅AWS合作伙伴中心用户的AWS托管政策。

AWSPartnerCentralRevenueAttributionManagement 政策

该政策为在 P AWS artner Central 注册的AWS账户提供了必要的收入归因管理活动访问权限。

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

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

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

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

```
    ]
}
```

AWSRevenueAttributionManagement 政策

本政策为未在 P AWS artner Central 注册的AWS账户提供了必要的收入归因管理活动访问权限。

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

向 IAM 角色和用户分配策略

按照以下步骤将策略分配给 IAM 角色和用户：

1. 登录到AWS 管理控制台。

2. 导航到 IAM 服务。
3. 选择角色或用户，然后选择要向其关联策略的 IAM 角色或用户。
4. 将`AWSPartnerCentralRevenueAttributionManagement`策略（适用于合作伙伴）或`AWSRevenueAttributionManagement`策略（适用于客户）附加到选定的 IAM 角色或用户。

有关更多信息，请参阅添加和删除 IAM 身份权限。

使用条件键管理权限

IAM 策略中的条件密钥为何时强制执行声明策略提供资源级权限。您可以使用条件键来指定何时允许或拒绝某些权限的条件。

有关更多信息，请参阅 IAM JSON 策略元素：条件运算符。

条件键概述

条件键	说明	适用的行动	有效值
合作伙伴中心：目录	按关联目录实体的类型筛选访问权限	所有操作	AWS，沙盒

所需权限摘要

所需权限摘要

处理建议	说明
合作伙伴中心：CreateRevenueAttribution	允许创建新的收入归因记录
合作伙伴中心：UpdateRevenueAttribution	允许更新现有的收入归因记录
合作伙伴中心：GetRevenueAttribution	允许检索特定收入归因的详细信息
合作伙伴中心：ListRevenueAttributions	允许使用可选筛选条件列出收入归因
合作伙伴中心：GetRevenueAttributionAllocation	允许按其 ID 检索单个分配
合作伙伴中心：ListRevenueAttributionAllocations	允许列出支持筛选的已提交分配

处理建议	说明
合作伙伴中心：StartRevenueAttributionAllocationsTask	允许提交一批最多 250 个分配变更以进行异步处理
合作伙伴中心：GetRevenueAttributionAllocationsTask	允许检索先前提交的分配任务的状态
合作伙伴中心：CreateMarketplaceRevenueShare	允许创建新的市场收入分成资源
合作伙伴中心：GetMarketplaceRevenueShare	允许检索特定市场收入份额的详细信息
合作伙伴中心：ListMarketplaceRevenueShares	允许使用可选筛选器列出市场收入份额
合作伙伴中心：CreateMarketplaceRevenueShareAllocation	允许创建新的市场收入分成分配
合作伙伴中心：GetMarketplaceRevenueShareAllocation	允许检索特定市场收入分成分配的详细信息
合作伙伴中心：UpdateMarketplaceRevenueShareAllocation	允许更新现有的市场收入分成分配
合作伙伴中心：ListMarketplaceRevenueShareAllocations	允许在商城收入份额下进行上架分配
合作伙伴中心：TagResource	允许为资源添加或覆盖标签
合作伙伴中心：UntagResource	允许从资源中移除标签
合作伙伴中心：ListTagsForResource	允许列出与资源关联的标签

的配额AWS合作伙伴中心 API

以下各节提供有关服务配额的信息。

主题

- 的配额AWS合作伙伴中心账户 API

- 的配额AWS Partner Central Selling API
- 的配额AWS合作伙伴中心权益 API
- 的配额AWS合作伙伴中心渠道 API
- 的配额AWS合作伙伴中心收入衡量 API

的配额AWS合作伙伴中心账户 API

AWS合作伙伴中心账户 API 具有以下配额。

其他配额

其他配额

显示名称	目录	说明	默认 值
每个账户打开连接邀请	AWS	您可以通过AWS目录中的合作伙伴账户保留的最大打开连接邀请数量	1000
每个账户的活跃连接数	AWS	您可以与AWS目录中的合作伙伴账户保持的最大活跃连接数	1000
每个合作伙伴的电子邮件域名	AWS	AWS目录中可以与合作伙伴账户关联以获得AWS培训认证的最大电子邮件域数量	50
每个账户的连接邀请率	AWS	您每天可以在AWS目录中发送的最大连接邀请数量	50
每个账户的个人资料更新任务比率	AWS	您每天可以在AWS目录中启动的最大配置文件更新任务数	10

显示名称	目录	说明	默认 值
每个账户的个人资料可见性更新率	AWS	您每天可以在AWS目录中启动的最大个人资料可见性更新次数	5
每个电子邮件地址的电子邮件验证码比率	AWS	您每天可以在AWS目录中请求的最大电子邮件验证码数量	5
每个账户打开连接邀请	沙盒	您可以使用沙盒目录中的合作伙伴账户保留的最大打开连接邀请数量	1000
每个账户的活跃连接数	沙盒	您可以与沙盒目录中的合作伙伴账户保持的最大活跃连接数	1000
每个合作伙伴的电子邮件域名	沙盒	沙盒目录中可以与合作伙伴账户关联以获得AWS培训认证的最大电子邮件域数量	50
每个账户的连接邀请率	沙盒	您每天可以在沙盒目录中发送的最大连接邀请数量	50
每个账户的个人资料更新任务比率	沙盒	您每天可以在沙盒目录中启动的最大配置文件更新任务数	10
每个账户的个人资料可见性更新率	沙盒	您每天可以在沙盒目录中启动的最大个人资料可见性更新次数	5

显示名称	目录	说明	默认 值
每个电子邮件地址的电子邮件验证码比率	沙盒	您每天可以在 Sandbox 目录中请求的最大电子邮件验证码数量。沙盒目录中不会发送任何电子邮件。	N/A

了解和管理配额

速率限制

当达到 API 速率限制时，服务将返回一个 ThrottlingException。为了更好地处理速率限制，AWS 建议在您的应用程序中实施指数退避和重试策略。

请求提高配额

如果默认配额不符合您的要求，您可以通过 Service Quotas 页面申请增加配额。Service Quotas 控制台是一个基于浏览器的界面，可用于查看和管理您的服务配额。您可以从任何AWS 管理控制台页面访问服务配额，方法是在顶部导航栏上选择该页面，或者在中搜索服务配额AWS 管理控制台。

的配额AWS Partner Central Selling API

AWS Partner Central 销售 API 会强制执行配额，以确保公平使用并保护服务免遭滥用。以下是各种 API 操作和每个机会关联的详细配额。

API 操作配额

Type	API 操作	配额（每个合作伙伴账户）
阅读动作	GetOpportunity	每秒 10 个；每 24 小时 100,000 个
	GetAwsOpportunitySummary	
	ListOpportunities	
	ListSolutions	

Type	API 操作	配额（每个合作伙伴账户）
	GetEngagementInvitation	
	ListEngagementInvitations	
潜在客户阅读操作	GetProspectingFrom EngagementTask	每秒 100 个
潜在客户阅读操作	ListProspectingFro mEngagementTasks	每秒 50 个
写动作	CreateOpportunity	每秒 1 个；每 24 小时 10,000 个
	UpdateOpportunity	
	AssociateOpportunity	
	DisassociateOpportunity	
	RejectEngagementInvitation	
	AssignOpportunity	
	StartEngagementFro mOpportunityTask	
	StartEngagementByAcceptingI nvitationTask	
潜在客户写作行动	StartProspectingFr omEngagementTask	每秒 2 个
参与写作动作	CreateEngagement	每秒 15 个

每个机会的关联配额

关联实体	配额
AWS产品	每个机会 20
合作伙伴解决方案	每个机会 10 个
AWS Marketplace解决方案	每个机会 10 个
AWS Marketplace产品	每个机会 10 个
AWS Marketplace私人优惠	每个机会 1 个

了解和管理配额

速率限制

当达到 API 速率限制时，服务将返回一个 ThrottlingException。为了更好地处理速率限制，AWS 建议在您的应用程序中实施指数退避和重试策略。

配额的时间窗口

每日配额会以 24 小时的滚动周期重置。例如，如果您在过去 24 小时内执行了 10,000 次写入操作，并且正在尝试执行 10,001 个请求，则您的请求将被限制。确保您的应用程序的使用模式考虑到这一点，以防止意外限制。

请求提高配额

如果默认配额不符合您的要求，您可以通过 Service Quotas 页面申请增加配额。Service Quotas 控制台是一个基于浏览器的界面，可用于查看和管理您的服务配额。您可以从任何AWS 管理控制台页面访问服务配额，方法是在顶部导航栏上选择该页面，或者在中搜索服务配额AWS 管理控制台。

的配额AWS合作伙伴中心权益 API

AWS合作伙伴中心权益 API 具有以下配额。

请求配额

请求配额

API 操作	配额（每个AWS账户）
ListBenefits	每秒 20 个
GetBenefit	每秒 20 个
AmendBenefitApplication	每秒 10 个
CancelBenefitApplication	每秒 10 个
CreateBenefitApplication	每秒 10 个
GetBenefitApplication	每秒 20 个
ListBenefitApplications	每秒 30 个
RecallBenefitApplication	每秒 10 个
SubmitBenefitApplication	每秒 10 个
UpdateBenefitApplication	每秒 10 个
GetBenefitAllocation	每秒 20 个
ListBenefitAllocations	每秒 20 个
AssociateBenefitApplicationResource	每秒 10 个
DisassociateBenefitApplicationResource	每秒 10 个

其他配额

其他配额

显示名称	目录	说明	默认 值
福利申请	AWS	您可以在AWS目录中创建的最大福利申请数量	10000
福利申请	沙盒	您可以在沙盒目录中创建的最大福利应用程序数量	10000

了解和管理配额

速率限制

当达到 API 速率限制时，该服务将以 “” 作为响应 ThrottlingException。为了更好地处理速率限制，AWS 建议在您的应用程序中实施指数退避和重试策略。

请求提高配额

如果默认配额不符合您的要求，您可以通过 Service Quotas 页面申请增加配额。Service Quotas 控制台是一个基于浏览器的界面，可用于查看和管理您的服务配额。您可以从任何AWS 管理控制台页面访问服务配额，方法是在顶部导航栏上选择该页面，或者在中搜索服务配额AWS 管理控制台。

的配额AWS合作伙伴中心渠道 API

AWS合作伙伴中心渠道 API 具有以下配额。

请求配额

请求配额

API 操作	配额（每个AWS账户）
CreateProgramManagementAccount	每秒 3 个
UpdateProgramManagementAccount	每秒 3 个

API 操作	配额（每个AWS账户）
ListProgramManagementAccounts	每秒 5 个
DeleteProgramManagementAccount	每秒 3 个
CreateChannelHandshake	每秒 3 个
ListChannelHandshakes	每秒 5 个
AcceptChannelHandshake	每秒 3 个
RejectChannelHandshake	每秒 3 个
CancelChannelHandshake	每秒 3 个
CreateRelationship	每秒 3 个
GetRelationship	每秒 5 个
ListRelationships	每秒 5 个
UpdateRelationship	每秒 3 个
DeleteRelationship	每秒 3 个

其他配额

其他配额

显示名称	目录	说明	默认 值
项目管理账户	AWS	您可以在AWS目录中创建的最大项目管理帐户数	50
每个项目管理账户的关系	AWS	在AWS目录中每个项目管理账户可以建立的最大关系数	1000

显示名称	目录	说明	默认 值
每个账户的计划管理账户握手次数	AWS	目录中活动程序管理账户的最大握手次数AWS	100
开始服务期每个账户的握手次数	AWS	在目录中确定服务期限的最大主动握手次数AWS	200
撤消服务期限每个账户的握手次数	AWS	目录中撤消服务期的最大主动握手次数AWS	200
每天计划管理账户的握手率	AWS	在目录中，您每天可以向同一账户发送的项目管理AWS账户握手次数上限AWS	5
开始服务期每天握手的比率	AWS	在目录中，您每天可以向同一AWS账户发送的最大起始服务期握手次数AWS	5
撤销服务期每天握手次数的比率	AWS	在目录中，您每天可以向同一AWS账户发送的最大撤销服务期握手次数AWS	5
项目管理账户	沙盒	您可以在沙盒目录中创建的最大程序管理帐户数	50
每个项目管理账户的关系	沙盒	在沙盒目录中，每个项目管理账户可以建立的最大关系数	1000

显示名称	目录	说明	默认 值
每个账户的计划管理账户握手次数	沙盒	沙盒目录中活动程序管理账户的最大握手次数	100
开始服务期每个账户的握手次数	沙盒	在沙盒目录中确定服务期限的最大主动握手次数	200
撤消服务期限每个账户的握手次数	沙盒	沙盒目录中撤消服务期的最大主动握手次数	200
每天计划管理账户的握手率	沙盒	在沙盒目录中，您每天可以向同一账户发送的项目管理AWS账户握手次数上限	5
开始服务期每天握手的比率	沙盒	在沙盒目录中，您每天可以向同一个AWS账户发送的最大起始服务期握手次数	5
撤销服务期每天握手次数的比率	沙盒	在沙盒目录中，您每天可以向同一AWS账户发送的最大撤销服务期握手次数	5

了解和管理配额

速率限制

当达到 API 速率限制时，该服务将以 “” 作为响应 ThrottlingException。为了更好地处理速率限制，AWS 建议在您的应用程序中实施指数退避和重试策略。

请求提高配额

如果默认配额不符合您的要求，您可以通过 Service Quotas 页面申请增加配额。Service Quotas 控制台是一个基于浏览器的界面，可用于查看和管理您的服务配额。您可以从任何AWS 管理控制台页面访问服务配额，方法是在顶部导航栏上选择该页面，或者在中搜索服务配额AWS 管理控制台。

的配额AWS合作伙伴中心收入衡量 API

AWS Partner Central 收入衡量 API 强制实施配额，以确保公平使用并保护服务免遭滥用。以下是各种 API 操作的详细配额。

API 操作配额

Type	API 操作	配额（每个合作伙伴账户）
阅读动作	GetRevenueAttribution	每秒 50 个
	GetRevenueAttributionAllocation	
	GetRevenueAttributionAllocationsTask	
	GetMarketplaceRevenueShare	
	GetMarketplaceRevenueShareAllocation	
列出动作	ListRevenueAttributions	每秒 10 个
	ListRevenueAttributionAllocations	
	ListMarketplaceRevenueShares	
	ListMarketplaceRevenueShareAllocations	

Type	API 操作	配额（每个合作伙伴账户）
	ListTagsForResource	
写动作	CreateRevenueAttribution	每秒 2 个
	UpdateRevenueAttribution	
	CreateMarketplaceRevenueShare	
	CreateMarketplaceRevenueShareAllocation	
	UpdateMarketplaceRevenueShareAllocation	
	TagResource	
	UntagResource	
异步写入操作	StartRevenueAttributionAllocationsTask	每秒 1 个

了解和管理配额

速率限制

当达到 API 速率限制时，服务将返回一个 ThrottlingException。为了更好地处理速率限制，AWS建议在应用程序中实现指数退避和重试策略。

请求提高配额

如果默认配额不符合您的要求，您可以通过 Service Quotas 页面申请增加配额。Service Quotas 控制台是一个基于浏览器的界面，可用于查看和管理您的服务配额。您可以从任何AWS 管理控制台页面访问服务配额，方法是在顶部导航栏上选择该页面，或者在中搜索服务配额AWS 管理控制台。

正在登录AWS合作伙伴中心 API

以下各节提供有关日志记录的信息。

主题

正在记录AWS合作伙伴中心账户 API

AWS CloudTrail是一项支持对您的进行治理、合规、运营审计和风险审计的服务AWS 账户。借AWS CloudTrail助，您可以记录、持续监控和保留与整个AWS基础架构中的操作相关的账户活动。AWS合作伙伴中心账户 API 活动在中记录为事件 CloudTrail。您可以创建跟踪，该配置允许将事件作为日志文件传输到 Amazon S3 存储桶。

概述

P AWS artner Central Account API 与AWS CloudTrail一项服务集成，该服务提供用户、角色或AWS服务在AWS合作伙伴中心中采取的操作的记录。 CloudTrail 将AWS合作伙伴中心账户 API 的所有 API 调用捕获为事件。捕获的呼叫包括来自AWS合作伙伴中心的调用以及对AWS合作伙伴中心账户 API 操作的代码调用。

如果您创建跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括AWS合作伙伴中心账户 API 的事件。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的事件历史记录中查看最新的事件。

使用收集的信息 CloudTrail，您可以确定向AWS合作伙伴中心账户 API 发出的请求、发出请求的 IP 地址、谁发出了请求、何时发出请求以及其他详细信息。

了解AWS合作伙伴中心账户 API 日志文件条目

跟踪是一种允许将事件作为日志文件传输到 Amazon S3 存储桶的配置。当您的跟踪跟踪AWS合作伙伴中心账户 API 事件时，会将事件作为所有地区的日志文件进行 CloudTrail 处理。每个日志文件可以包含一个或多个事件。

以下示例显示了演示AWS合作伙伴中心账户 API 上`CreatePartner`操作的 CloudTrail 日志条目：

```
{
    "eventVersion": "1.11",
    "userIdentity": {
```

```
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/CloudTrailTestUser",
        "accountId": "123456789012",
        "accessKeyId": "ABCDEFGHIJKLMNOP1234",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAEXAMPLE52AGFT725JGDZ",
                "arn": "arn:aws:iam::123456789012:role/ExampleRole",
                "accountId": "123456789012",
                "userName": "ExampleRole"
            },
            "attributes": {
                "creationDate": "2025-11-07T19:45:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-11-07T19:45:58Z",
    "eventSource": "partnercentral-account.amazonaws.com",
    "eventName": "CreatePartner",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "PostmanRuntime/7.18.0",
    "requestParameters": {
        "catalog": "AWS",
        "clientToken": "abcdef12-3456-7890-bcde-f123456789ab",
        "legalName": "ExampleLegalName",
        "primarySolutionType": "VALUE_ADDED_RESALE_AWS_SERVICES",
        "allianceLeadContact": {
            "firstName": "ExampleFirstName",
            "lastName": "ExampleLastName",
            "email": "example@domain.com",
            "businessTitle": "ExampleBusinessTitle"
        },
        "emailVerificationCode": "ExampleVerificationCode",
        "tags": [
            {
                "key": "office",
                "value": "1"
            }
        ]
    },
```

```
    "responseElements": {
        "catalog": "AWS",
        "arn": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/partner/
EXAMPLE_PARTNER_ID",
        "id": "EXAMPLE_PARTNER_ID",
        "legalName": "ExampleLegalName",
        "createdAt": "2025-11-07T19:45:57.867007329Z",
        "profile": {
            "primarySolutionType": "VALUE_ADDED_RESALE_AWS_SERVICES"
        },
        "allianceLeadContact": {
            "firstName": "ExampleFirstName",
            "lastName": "ExampleLastName",
            "email": "example@domain.com",
            "businessTitle": "ExampleBusinessTitle"
        }
    },
    "requestID": "12345678-1234-5678-9abc-def012345678",
    "eventID": "87654321-4321-8765-cba9-fed098765432",
    "readOnly": false,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::PartnerCentral::Partner",
            "ARN": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/partner/
EXAMPLE_PARTNER_ID"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "partnercentral-account.partner.us-east-1.api.aws"
    }
}
```


在此示例中，该CreatePartner操作由名为的 IAM 用户调用 CloudTrailTestUser。该请求是在世界标准时间2025年11月7日 19:45:58 提出的。该请求为 AWS 目录创建了一个新的合作伙伴，其合法名称为 “ExampleLegalName”。EXAMPLE_PARTNER_ID

中的字段AWS合作伙伴中心账户 API 日志文件条目

CloudTrail 日志文件中的每个条目都包含有关谁提出了请求、请求中处理的资源以及AWS合作伙伴中心账户 API 返回的响应元素的信息。日志条目中的字段列表（例如`eventVersion`、`userIdentity`、`eventTime`和）提供了有关操作的详细信息。例如，该`sourceIPAddress`字段显示发出请求的 IP 地址。

正在记录AWS Partner Central Selling API

AWS CloudTrail是一项支持对您的进行治理、合规、运营审计和风险审计的服务AWS 账户。借AWS CloudTrail助，您可以记录、持续监控和保留与整个AWS基础架构中的操作相关的账户活动。AWS合作伙伴中心 API 活动在中记录为事件 CloudTrail。您可以创建跟踪，该配置允许将事件作为日志文件传输到 Amazon S3 存储桶。

概述

P AWS artner Central API 与AWS CloudTrail一项服务集成，该服务提供用户、角色或AWS服务在AWS合作伙伴中心中采取的操作的记录。 CloudTrail 将AWS合作伙伴平台的所有 API 调用记录为事件。捕获的呼叫包括来自AWS合作伙伴中心的调用以及对AWS合作伙伴中心 API 操作的代码调用。

如果您创建跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括AWS合作伙伴中心的事件。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的 “事件历史记录” 中查看最新的事件。

使用收集的信息 CloudTrail，您可以确定向 P AWS artner Central 发出的请求、发出请求的 IP 地址、谁提出了请求、何时提出请求以及其他详细信息。

了解AWS合作伙伴中心销售 API 日志文件条目

跟踪是一种允许将事件作为日志文件传输到 Amazon S3 存储桶的配置。当您的跟踪跟踪AWS合作伙伴中心事件时，会将事件作为所有地区的日志文件进行 CloudTrail 处理。每个日志文件可以包含一个或多个事件。

以下示例显示了演示AWS合作伙伴中心`ListOpportunities`操作的 CloudTrail 日志条目：

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDEFGHIJKLMNOP12345",
        "arn": "arn:aws:iam::123456789010:user/CloudTrailTestUser",
```

```
        "accountId": "123456789010",
        "accessKeyId": "ABCDEFGHIJKLMNOP1234",
        "userName": "CloudTrailTestUser"
    },
    "eventTime": "2023-10-17T21:49:23Z",
    "eventSource": "partnercentral-selling.amazonaws.com",
    "eventName": "ListOpportunities",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "PostmanRuntime/7.18.0",
    "requestParameters": {
        "MaxResults": 20
    },
    "responseElements": null,
    "requestID": "fEXAMPLE-cb3e-4e21-86fd-6b3EXAMPLEd1",
    "eventID": "7EXAMPLE-97d6-4139-91e3-01aEXAMPLE48",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789010"
}
```


在此示例中，该`ListOpportunities`操作由名为的 IAM 用户调用 CloudTrailTestUser。该行动是在 us-AWS 区域 ea st-1 中发起的，该请求是在世界标准时间2023年10月17日 21:49:23 提出的。

中的字段AWS合作伙伴中心销售 API 日志文件条目

CloudTrail 日志文件中的每个条目都包含有关谁提出了请求、请求中处理的资源以及 P AWS artner Central 返回的响应元素的信息。日志条目中的字段列表（例如`eventVersion`、`userIdentity`、`eventTime`和）提供了有关操作的详细信息。例如，该`sourceIPAddress`字段显示发出请求的 IP 地址。

正在记录AWS合作伙伴中心权益 API

AWS CloudTrail是一项支持对您的进行治理、合规、运营审计和风险审计的服务AWS 账户。借AWS CloudTrail助，您可以记录、持续监控和保留与整个AWS基础架构中的操作相关的账户活动。AWS合作伙伴中心权益 API 活动在中记录为事件 CloudTrail。您可以创建跟踪，这是一种允许将事件作为日志文件传输到 Amazon S3 存储桶的配置。

概述

P AWS artner Central Benefits API 与AWS CloudTrail一项服务集成，该服务提供用户、角色或AWS服务在AWS合作伙伴中心中采取的操作的记录。 CloudTrail 将AWS合作伙伴中心权益 API 的所有 API

调用作为事件捕获。捕获的呼叫包括来自合作伙伴中心的调用以及对AWS合作伙伴中心 Benefits AWS API 操作的代码调用。

如果您创建跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括AWS合作伙伴中心权益 API 的事件。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的 “事件历史记录” 中查看最新的事件。

使用收集的信息 CloudTrail，您可以确定向 P AWS artner Central Benefits API 发出的请求、发出请求的 IP 地址、谁提出了请求、何时提出请求以及其他详细信息。

了解AWS合作伙伴中心权益 API 日志文件条目

跟踪是一种允许将事件作为日志文件传输到 Amazon S3 存储桶的配置。当您的跟踪跟踪 P AWS artner Central Benefits API 事件时，会将这些事件作为所有地区的日志文件进行 CloudTrail 处理。每个日志文件可以包含一个或多个事件。

以下示例显示了一个 CloudTrail 日志条目，它演示了在 P AWS artner Central Benefits API 上`ListBenefitApplications`执行的操作

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/CloudTrailTestUser",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EX_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/TestRole",
                "accountId": "123456789012",
                "userName": "TestRole"
            },
            "attributes": {
                "creationDate": "2025-10-21T03:08:23Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-10-21T03:10:59Z",
```

```
    "eventSource": "partnercentral-benefits.amazonaws.com",
    "eventName": "ListBenefitApplications",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "python-requests/2.32.4",
    "requestParameters": {
        "catalog": "AWS"
    },
    "responseElements": null,
    "requestID": "12345678-1234-5678-9abc-def012345678",
    "eventID": "87654321-4321-8765-cba9-fed098765432",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```


在此示例中，该`ListBenefitApplications`操作是由名为 Alice 的 IAM 用户调用的。该请求是在世界标准时间 2025 年 10 月 21 日 03:10:59 提出的。该请求列出了 AWS 目录的福利申请。

中的字段AWS合作伙伴中心权益 API 日志文件条目

CloudTrail 日志文件中的每个条目都包含有关谁提出了请求、请求中处理的资源以及 P AWS artner Central Benefits API 返回的响应元素的信息。日志条目中的字段列表（例如`eventVersion`、`userIdentity`和`eventTime`）提供了有关操作的详细信息。例如，该`sourceIPAddress`字段显示发出请求的 IP 地址。

正在记录AWS合作伙伴中心渠道 API

AWS CloudTrail是一项支持对您的进行治理、合规、运营审计和风险审计的服务AWS 账户。借AWS CloudTrail助，您可以记录、持续监控和保留与整个AWS基础架构中的操作相关的账户活动。AWS合作伙伴中心渠道 API 活动在中记录为事件 CloudTrail。您可以创建跟踪，该配置允许将事件作为日志文件传输到 Amazon S3 存储桶。

概述

P AWS artner Central 渠道 API 与AWS CloudTrail一项服务集成，该服务提供用户、角色或AWS服务在 P AWS artner Central 中采取的操作的记录。 CloudTrail 将AWS合作伙伴中心渠道 API 的所有 API 调用捕获为事件。捕获的呼叫包括来自AWS合作伙伴中心的调用以及对AWS合作伙伴中心渠道 API 操作的代码调用。

如果您创建跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括AWS合作伙伴中心渠道 API 的事件。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的 “事件历史记录” 中查看最新的事件。

使用收集的信息 CloudTrail，您可以确定向 P AWS artner Central Channel API 发出的请求、发出请求的 IP 地址、谁提出了请求、何时发出请求以及其他详细信息。

了解AWS合作伙伴中心渠道 API 日志文件条目

跟踪是一种允许将事件作为日志文件传输到 Amazon S3 存储桶的配置。当您的跟踪跟踪AWS合作伙伴中心渠道 API 事件时，会将这些事件作为所有地区的日志文件进行 CloudTrail 处理。每个日志文件可以包含一个或多个事件。

以下示例显示了演示AWS合作伙伴中心渠道 API 上`CreateProgramManagementAccount`操作的 CloudTrail 日志条目：

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAEXAMPLE52AGFT725JGDZ:example-user-Isengard",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/example-user-Isengard",
        "accountId": "123456789012",
        "accessKeyId": "ASIAEXAMPLE52AGL6IXQLZ5",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAEXAMPLE52AGFT725JGDZ",
                "arn": "arn:aws:iam::123456789012:role/ExampleRole",
                "accountId": "123456789012",
                "userName": "ExampleRole"
            },
            "attributes": {
                "creationDate": "2025-10-21T17:06:47Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-10-21T17:07:26Z",
    "eventSource": "partnercentral-channel.amazonaws.com",
    "eventName": "CreateProgramManagementAccount",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
```

```
    "userAgent": "PostmanRuntime/7.18.0",
    "requestParameters": {
        "catalog": "AWS",
        "program": "SOLUTION_PROVIDER",
        "displayName": "ExampleDisplayName",
        "accountId": "987654321098",
        "clientToken": "abcdef12-3456-7890-bcde-f123456789ab"
    },
    "responseElements": {
        "programManagementAccountDetail": {
            "id": "pma-example123456789",
            "arn": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/program-
management-account/pma-example123456789"
        }
    },
    "requestID": "12345678-1234-5678-9abc-def012345678",
    "eventID": "87654321-4321-8765-cba9-fed098765432",
    "readOnly": false,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::PartnerCentralChannel::ProgramManagementAccount",
            "ARN": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/program-
management-account/pma-example123456789"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "clientProvidedHostHeader": "partnercentral-channel.global.api.aws"
    }
}
```


在此示例中，该`CreateProgramManagementAccount`操作是由 ExampleRole 通过代入角色会话命名的 IAM 角色调用的。该请求是在世界标准时间2025年10月21日 17:07:26 提出的。该请求为解决方案提供商计划创建了一个新的计划管理账户，账号`pma-example123456789`为 “解决方案提供商”。

中的字段AWS合作伙伴中心渠道 API 日志文件条目

CloudTrail 日志文件中的每个条目都包含有关谁提出了请求、请求中处理的资源以及 P AWS artner Central Channel API 返回的响应元素的信息。日志条目中的字段列表（例

如eventVersionuserIdentityeventTime、和）提供了有关操作的详细信息。例如，该sourceIPAddress字段显示发出请求的 IP 地址。

正在记录AWS合作伙伴中心收入归因 API

AWS CloudTrail是一项支持对您的进行治理、合规、运营审计和风险审计的服务AWS 账户。借AWS CloudTrail助，您可以记录、持续监控和保留与整个AWS基础架构中的操作相关的账户活动。AWS合作伙伴中心收入衡量 API 活动在中记录为事件 CloudTrail。您可以创建跟踪，该配置允许将事件作为日志文件传输到 Amazon S3 存储桶。

概述

P AWS artner Central 收入衡量 API 与AWS CloudTrail一项服务集成，该服务提供用户、角色或AWS服务在 P AWS artner Central 中采取的操作的记录。 CloudTrail 将AWS合作伙伴中心收入归因的所有 API 调用记录为事件。捕获的呼叫包括来自AWS合作伙伴中心控制台的调用以及对AWS合作伙伴中心收入衡量 API 操作的代码调用。

如果您创建跟踪，则可以将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括用于衡量AWS合作伙伴中心收入的事件。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的 “事件历史记录” 中查看最新的事件。

使用收集的信息 CloudTrail，您可以确定向AWS合作伙伴中心收入衡量机构发出的请求、提出请求的 IP 地址、谁提出了请求、何时提出请求以及其他详细信息。

以下AWS合作伙伴中心收入衡量 API 操作已登录 CloudTrail：

- CreateRevenueAttribution
- UpdateRevenueAttribution
- GetRevenueAttribution
- ListRevenueAttributions
- TagResource
- UntagResource
- ListTagsForResource
- CreateMarketplaceRevenueShare
- GetMarketplaceRevenueShare
- ListMarketplaceRevenueShares
- CreateMarketplaceRevenueShareAllocation

- GetMarketplaceRevenueShareAllocation
- UpdateMarketplaceRevenueShareAllocation
- ListMarketplaceRevenueShareAllocations
- StartRevenueAttributionAllocationsTask
- GetRevenueAttributionAllocationsTask
- ListRevenueAttributionAllocations
- GetRevenueAttributionAllocation

了解AWS合作伙伴中心收入衡量 API 日志文件条目

跟踪是一种允许将事件作为日志文件传输到 Amazon S3 存储桶的配置。当您的跟踪跟踪AWS合作伙伴中心收入衡量事件时，会将这些事件作为所有地区的日志文件进行 CloudTrail处理。每个日志文件可以包含一个或多个事件。

以下示例显示了一个 CloudTrail 日志条目，该条目演示了对AWS合作伙伴中心收入归因的CreateRevenueAttribution操作：

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAEXAMPLEID",
        "arn": "arn:aws:iam::123456789012:user/CloudTrailTestUser",
        "accountId": "123456789012",
        "accessKeyId": "AKIAEXAMPLEKEY",
        "userName": "CloudTrailTestUser"
    },
    "eventTime": "2026-06-02T11:53:54Z",
    "eventSource": "partnercentral-prm.amazonaws.com",
    "eventName": "CreateRevenueAttribution",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "aws-sdk-java/2.20.0",
    "requestParameters": {
        "catalog": "AWS",
        "name": "my-revenue-attribution",
        "marketplaceProduct": {
            "tenancyModel": "MULTI_TENANT"
        }
    },
```

```
    "responseElements": {
        "id": "ra-0123456789abcdef0",
        "arn": "arn:aws:partnercentral:us-east-1:123456789012:catalog/AWS/revenue-
attribution/ra-0123456789abcdef0",
        "name": "my-revenue-attribution",
        "revision": "1"
    },
    "requestID": "5ce2f907-3850-412a-b1a4-r012r1234567",
    "eventID": "80b39b72-eefe-4578-8db0-01ee2e34ee56",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

在此示例中，该CreateRevenueAttribution操作是由名为的 IAM 用户调用的CloudTrailTestUser。该行动是在us-AWS 区域 ea st-1 中发起的，该请求是在世界标准时间2026年6月2日 11:53:54 提出的。已使用该ID创建了一个新的收入归因资源ra-0123456789abcdef0。

中的字段AWS合作伙伴中心收入衡量 API 日志文件条目

CloudTrail 日志文件中的每个条目都包含有关谁提出了请求、请求中处理的资源以及AWS合作伙伴中心收入衡量返回的响应元素的信息。日志条目中的字段列表（例如eventVersionuserIdentityeventTime、和）提供了有关操作的详细信息。例如，该sourceIPAddress字段显示发出请求的 IP 地址。

的通知AWS合作伙伴中心 API

以下各节提供有关通知的信息。

主题

- 的通知AWS合作伙伴中心账户 API
- 的通知AWS Partner Central Selling API
- 的通知AWS合作伙伴中心权益 API
- 的通知AWS合作伙伴中心渠道 API

的通知AWS合作伙伴中心账户 API

Partner Account Connection 通知使合作伙伴能够随时了解直接影响其业务关系和运营工作流程的连接生命周期事件。这些事件对于合作伙伴来说至关重要：

- 对新的协作迅速做出反应
- 随时了解他们的连接组合状态
- 建立或终止连接时采取适当的措施
- 通过了解关系何时发生变化来确保业务连续性

主题

完成监控事件的先决条件

用户需要相应的 IAM 权限才能访问和管理账户连接 API 发布的事件。有关可用操作、资源和条件键的更多信息 EventBridge，请参阅亚马逊 EventBridge 用户指南中的在亚马逊 EventBridge中使用 IAM 策略条件。其中一个条件键是`events:detail-type`，它可用于将权限范围限定为特定事件类型。

以下示例策略演示了如何为建议的事件自定义权限并确定其权限范围。该`AllowPutRuleForPartnercentralAccountEvents`语句允许创建规则，但仅适用于来自`aws.partnercentral-account`源的事件。

有关详细的 IAM 策略示例，请参阅有关 EventBridge 权限的 AWS 文档。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralAccountEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-account.connection"
        }
      }
    }
  ]
}
```

将 Amazon 配置 EventBridge 为监控事件

要监控账户连接 API 事件，您需要创建一个与要捕获的事件相匹配的 EventBridge 规则。您可以使用 AWS 管理控制台或 S AWS DK 来创建和管理规则。以下各节说明如何使用这两种方法创建规则。无论使用哪种方法，都必须在美国东部（弗吉尼亚北部）us-east-1地区创建规则。

AWS 管理控制台设置

要使用设置 EventBridge 规则AWS 管理控制台，请按照《亚马逊 EventBridge 用户指南》中创建响应亚马逊事件的规则 EventBridge中的步骤进行操作。创建规则时，必须将事件总线设置为默认值，并在美国东部（弗吉尼亚北部）us-east-1区域创建规则。

以下是事件规则的示例：

```
{
    "source": ["aws.partnercentral-account"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS软件开发工具包设置

您可以使用AWS软件开发工具包以编程方式创建和管理 EventBridge 规则。有关更多信息，请参阅 Amazon EventBridge API 参考PutRule中的。

以下示例使用适用于 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
    Name='MyConnectionInvitationReceivedRule',
    EventPattern=
    '{
        "source": ["aws.partnercentral-account"],
        "detail-type": ["Partner Connection Invitation Received"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
```

```
print('Rule ARN:', response['RuleArn'])
```

详细了解账号关联 API 事件

以下各节描述了账号连接 API 事件类型、触发这些事件的场景以及事件示例。

事件类型

以下是事件类型及其触发器。

- 已收到合作伙伴连接邀请：通知接收者另一位合作伙伴想要建立连接
- P@@ artner Connection 邀请已接受：通知发件人其邀请已被接受并且连接现已激活
- 合作伙伴连接邀请被拒绝：通知发件人其邀请已被拒绝
- 合作伙伴连接已取消：当活动连接终止时，通知其他已连接的参与者
- P@@ artner Connection 邀请已取消：通知接收者在采取任何操作之前发件人已撤回邀请
- P@@ artner Connection 邀请已过期：通知发送者和接收者邀请已过期

示例事件

以下各节提供了前面部分中列出的事件的示例。

主题

- 已收到合作伙伴连接邀请
- 合作伙伴连接邀请已接受
- 合作伙伴连接邀请被拒绝
- 合作伙伴连接已取消
- 合作伙伴连接邀请已取消
- 合作伙伴连接邀请已过期

已收到合作伙伴连接邀请

触发上下文

此事件是在成功创建连接邀请并通过 CreateConnectionInvitation API 保存在PAC的数据存储中时生成的。该事件在成功创建邀请后立即发送，并保留在 PAC 的数据存储中，而不仅仅是在发出 API 调用时。

创建新的连接邀请后，该事件将自动发送。每份邀请创建一个活动，同一邀请中没有重复的事件。

收件人

连接邀请中接收者的AWS账户。

预期的处理方式

典型的客户行为：

- 即时通知：向业务团队发出有关新合作机会的警报
- 工作流程自动化：使用待处理的邀请自动更新合作伙伴管理系统
- Decision Support：根据连接类型向相应的决策者发送邀请
- 审核日志：记录邀请收据以进行合规和合作伙伴关系跟踪
- 响应自动化：对于值得信赖的合作伙伴，可能会自动接受某些类型的邀请

常见的集成模式：

- Lambda 函数 → 更新合作伙伴门户控制面板
- SQS 队列 → Batch 处理审阅邀请
- SNS 主题 → 给业务团队的 Email/Slack 通知
- API 目标 → 连接到外部 CRM/partner 管理系统的 Webhook

示例

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

合作伙伴连接邀请已接受

触发上下文

当成功接受连接邀请，并且通过 AcceptConnectionInvitation API 创建连接并将其保留在 PAC 的数据存储中时，就会生成此事件。成功接受邀请并建立连接后，将立即发送该事件。

当连接邀请被接受时，该事件将自动发送。每份邀请接受一个活动，同一份接受的活动不能重复进行。

收件人

连接邀请中涉及的两个AWS账户：最初发送邀请的发送者账户和接受邀请的接收者账户。

预期的处理方式

典型的客户行为：

- 激活伙伴关系：触发工作流程以启用第 2 层业务活动
- 状态更新：使用活动连接状态更新合作伙伴管理系统
- 通知：提醒业务团队成功建立合作伙伴关系
- 访问配置：自动授予适当的协作权限
- 分析：跟踪合作伙伴转化率和成功指标

常见的集成模式：

- Lambda 函数 → 启用数据共享权限并更新合作伙伴门户
- SQS 队列 → Batch 处理业务设置的连接激活
- SNS 主题 → 给业务和技术团队的 Email/Slack 通知

- API 目的地 → 通过 Webhook 到 CRM 系统以更新合作伙伴关系状态

示例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type":  "Partner Connection Invitation Accepted",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
  "resources": [ "<<Connection Invitation ARN>>" , "<<Connection ARN>>" ],
  "account": "<corresponding partner AWS account>",
  "detail": {
    "catalog": "AWS",
    "connectionInvitation" :{
        "arn": "<<Connection Invitation ARN>>",
        "id": "pacinv-****",
        "connectionType": "OpportunityCollaboration"
    },
    "connection": {
        "arn": "<<Connection ARN>>",
         "id": "pac-*****",
         "Participant1AccountId": "<<Sender AWS AccountId>>",
         "Participant2AccountId": "<<Receiver AWS AccountId>>",
         "status": "ACTIVE"
     }
  }
}
```

合作伙伴连接邀请被拒绝

触发上下文

当通过 RejectConnectionInvitation API 成功拒绝连接邀请时，就会生成此事件。该事件在处理邀请拒绝后立即发送，并保留在 PAC 的数据存储中。

当连接邀请被拒绝时，该事件将自动发送。每个邀请都拒绝一个事件，同一拒绝事件没有重复事件。

收件人

最初发送连接邀请的AWS账户（发件人账户）。

预期的处理方式

典型的客户行为：

- 状态更新：使用拒绝状态更新合作伙伴管理系统
- Follow-up 行动：触发替代伙伴关系方法的工作流程
- 通知：提醒业务团队注意合作决策
- 清理：从内部系统中删除待处理的邀请引用

常见的集成模式：

- Lambda 函数 → 更新合作伙伴门户并触发后续工作流程
- SQS 队列 → Batch 处理拒绝以进行业务分析
- SNS 主题 → 给业务开发团队的 Email/Slack 通知
- API 目的地 → 通过 Webhook 到 CRM 系统以更新机会状态

示例

```
{
  "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-****"
    }
  }
}
```

合作伙伴连接已取消

触发上下文

当成功取消活动连接并且通过 CancelConnection API 在 PAC 的数据存储中将活动状态更新为已终止时，就会生成此事件。处理连接取消操作并更新连接状态后，将立即发送该事件。

连接终止时，该事件将自动发送。每个连接取消一个事件，同一取消时没有重复的事件。

收件人

连接中其他参与者的AWS账户（不是发起取消的参与者）。

预期的处理方式

典型的客户行为：

- 撤销访问权限：立即撤消权限并禁用数据共享
- 状态更新：使用已终止的连接状态更新合作伙伴管理系统
- 通知：提醒业务和技术团队注意合作关系终止
- 清理：移除连接引用并清理共享资源
- 分析：跟踪连接持续时间和终止模式

常见的集成模式：

- Lambda 函数 → 撤消权限并更新合作伙伴门户网站状态
- SQS 队列 → 用于清理工作流的 Batch 处理连接终止
- SNS 主题 → 给业务和技术团队的 Email/Slack 通知
- API 目的地 → 通过 Webhook 到 CRM 系统以更新合作伙伴关系状态

示例

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

合作伙伴连接邀请已取消

触发上下文

当通过 CancelConnectionInvitation API 成功取消待处理的连接邀请时，就会生成此事件。活动将在处理取消邀请且邀请状态更新为 “已取消” 后立即发送。

取消连接邀请后，该事件将自动发送。每个连接邀请取消一个事件，同一取消时没有重复的事件。

收件人

本应收到连接邀请的AWS账户（接收者账户）。

预期的处理方式

典型的客户行为：

- 状态更新：更新合作伙伴管理系统以删除待处理的邀请引用
- 通知：提醒业务团队潜在的合作机会已被撤回
- 清理：从内部跟踪系统中删除邀请引用
- 分析：跟踪邀请取消模式以获取合作伙伴见解

常见的集成模式：

- Lambda 函数 → 更新合作伙伴门户并删除待处理的邀请通知
- SQS 队列 → 取消批处理以进行业务分析
- SNS 主题 → 给业务开发团队的 Email/Slack 通知
- API 目的地 → 通过 Webhook 到 CRM 系统以更新机会状态

示例

```
{
  "version": "1",
  "id": "<event id>",
  "detail-type": "Partner Connection Invitation Cancelled",
  "source": "aws.partnercentral-account",
  "time": "<ISO 8601 date time>",
  "region": "us-east-1",
  "account": "<corresponding partner AWS account>",
  "resources": [ "<<Connection Invitation ARN>>" ],
  "detail": {
    "catalog": "AWS",
    "connectionInvitation" :{
        "arn": "<<Connection Invitation ARN>>",
        "id": "pacinv-****",
        "connectionType": "OpportunityCollaboration",
        "invitationMessage": "We'd like to collaborate on a joint solution for cloud
 security. Please accept this connection invitation to proceed.",
        "senderProfileId": "pprofile-****"
    }
  }
}
```

合作伙伴连接邀请已过期

触发上下文

当待处理的连接邀请在到达其 ExpiresAt 时间戳后自动过期而没有被接收者接受或拒绝时，就会生成此事件。当邀请到期时间时，系统会自动触发该事件。

连接邀请到期后，将自动发送该事件。每份邀请都有一个活动已过期，同一有效期内没有重复的事件。

收件人

连接邀请中涉及的两个AWS账户：最初发送邀请的发送者账户和收到邀请的接收者账户。

预期的处理方式

典型的客户行为：

- 状态更新：更新合作伙伴管理系统以将邀请标记为已过期
- Follow-up 行动：触发重新参与的工作流程或替代伙伴关系方法

- 通知：提醒业务团队注意错过的合作机会
- 清理：从内部跟踪系统中删除过期的邀请引用

常见的集成模式：

- Lambda 函数 → 更新合作伙伴门户并触发后续工作流程
- SQS 队列 → Batch 处理过期时间以进行业务分析
- SNS 主题 → 给业务开发团队的 Email/Slack 通知
- API 目的地 → 通过 Webhook 到 CRM 系统以更新机会状态

示例

```
{
  "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-****"
    }
  }
}
```

的通知AWS Partner Central Selling API

销售 API 的事件提供有关机会状态变化或详细信息的实时通知。这些事件有助于使您的系统与AWS合作伙伴中心保持同步，并有助于确保及时响应和更新。

主题

完成监控事件的先决条件

用户需要相应的 IAM 权限才能访问和管理合作伙伴中心销售 API 发布的活动。有关可用操作、资源和条件键的更多信息 EventBridge，请参阅亚马逊 EventBridge 用户指南中的在亚马逊 EventBridge中使用 IAM 策略条件。其中一个条件键是`events:detail-type`，它可用于将权限范围限定为特定事件类型。

以下示例策略演示了如何自定义和限定建议事件的权限范围。
该`AllowPutRuleForPartnercentralSellingEvents`语句允许创建规则，但仅适用于来自`aws.partnercentral-selling`源的事件。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AllowPutRuleForPartnercentralSellingEvents",
    "Effect": "Allow",
    "Action": "events:PutRule",
    "Resource": "*",
    "Condition": {
      "ForAllValues:StringEquals": {
        "events:source": "aws.partnercentral-selling"
      }
    }
  }
}
```

将 Amazon 配置 EventBridge 为监控事件

要监控销售 API 事件，您需要创建一条与要捕获的事件相匹配的 EventBridge 规则。您可以使用AWS管理控制台或 S AWS DK 来创建和管理规则。以下各节说明如何使用这两种方法创建规则。无论使用哪种方法，都必须在美国东部（弗吉尼亚北部）`us-east-1`地区创建规则。

AWS 管理控制台设置

要使用设置 EventBridge 规则AWS 管理控制台，请按照《亚马逊 EventBridge 用户指南》中创建响应亚马逊事件的规则 EventBridge中的步骤进行操作。创建规则时，必须将事件总线设置为默认值，并在美国东部（弗吉尼亚北部）us-east-1区域创建规则。

以下是事件规则的示例：

```
{
    "source": ["aws.partnercentral-selling"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS软件开发工具包设置

您可以使用AWS软件开发工具包以编程方式创建和管理 EventBridge 规则。有关更多信息，请参阅 Amazon EventBridge API 参考PutRule中的。

以下示例使用适用于 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
    Name='MyOpportunityCreatedRule',
    EventPattern=
    '{
        "source": ["aws.partnercentral-selling"],
        "detail-type": ["Opportunity Created"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
print('Rule ARN:', response['RuleArn'])
```

了解有关销售 API 活动的更多信息

以下各节介绍销售 API 事件类型、触发这些事件的场景以及事件示例。

事件类型

以下是事件类型及其触发器。

- 机会已创建：创建新机会时触发。
- 机会已更新：当商机（机会或其对应的AWS机会摘要）更新时触发。
- 已创建参与邀请：创建AWS推荐邀请时触发。
- 已接受参与邀请：当合作伙伴接受AWS参与邀请，确认他们有兴趣AWS就该机会与之合作时触发。
- 参与邀请被拒绝：当合作伙伴拒绝AWS参与邀请时触发。
- 参与邀请已过期：当合作伙伴拒绝邀请时会触发此事件。 EngagementInvitation它会通知发送方和接收方合作伙伴邀请已过期。
- 已添加参与成员：当新成员在接受邀请后加入互动时触发此事件。它会通知参与的所有当前成员有新成员被添加到参与中。
- 项目资源快照已创建：创建与项目关联的资源（例如机会）快照的新版本时，会触发此事件。它会将相关资源 Snapshot 的更改通知参与的所有当前成员。
- 已创建互动：创建新交互时触发。
- 互动已更新：交互更新时触发。

事件场景

下表描述了事件在不同场景下的运作方式。针对这些场景做出了以下假设：

- AWS也是这些场景中的合作伙伴。
- `ResourceSnapshotJob`已由合作伙伴正确配置。
- 在机会上更新的字段也出现在资源快照的机会模板上。

您作为合作伙伴采取的行动	收到的事件	参与者类型
你创造了一个机会	机会已创建	
你`StartEngagementFromOpportunityTask` 用来提交机会	机会已更新、参与已创建、参与资源快照已创建、参与成员已添加、参与邀请已创建	

您作为合作伙伴采取的行动	收到的事件	参与者类型
AWS批准您提交的机会	参与邀请已接受，参与成员已添加，机会已更新，参与资源快照已创建	
AWS拒绝您提交的机会	参与邀请被拒绝、机会已更新、参与资源快照已创建	
您将报AWS Marketplace价与机会相关联	机会已更新	
您将解决方案与机会相关联	机会已更新，项目资源快照已创建	
你更新了机会	机会已更新，项目资源快照已创建（如果已设置ResourceSnapshotJob 且模板上有字段更新，则为可选）	
AWS更新您的机会	机会已更新，项目资源快照已创建	
您邀请其他合作伙伴参与互动	订婚邀请已创建	发件人
您的参与邀请已被合作伙伴接受	已接受订婚邀请，已添加参与成员	发件人
您的参与邀请被合作伙伴拒绝	订婚邀请被拒绝	发件人
合作伙伴在 15 天内未对你的参与邀请采取行动	订婚邀请已过期	发件人
您会收到其他合作伙伴发出的加入互动的邀请	订婚邀请已创建	接收方

您作为合作伙伴采取的行动	收到的事件	参与者类型
您接受其他合作伙伴的邀请，通过以下方式加入互动 `StartEngagementByAcceptingInvitationTask`	已接受参与邀请，已添加参与成员，已创建机会，已更新机会，已创建参与资源快照	接收方
您拒绝了来自其他合作伙伴的参与邀请	订婚邀请被拒绝	接收方
在 15 天内，您不会根据其他合作伙伴的邀请采取行动	订婚邀请已过期	接收方
你创建了互动	参与度已创建	发件人
你更新了互动	参与度已更新	发送者、接收者

示例事件

以下各节提供了前面部分所列事件的示例。

主题

- 机会已创建
- 机会已更新
- 订婚邀请已创建
- 订婚邀请已接受
- 订婚邀请被拒绝
- 订婚邀请已过期
- 已添加参与成员
- 参与资源快照已创建
- 参与度已创建
- 参与度已更新

机会已创建

合作伙伴利用此活动来：

- 通知他们的用户一个新的机会已经创建。
- 触发自动化工作流程，例如更新 CRM 系统或通知销售团队有关新机会创造的信息。

以下示例显示了一个典型的机会创建事件。

```
{
    "version": "1",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "source": "aws.partnercentral-selling",
    "detail-type": "Opportunity Created",
    "time": "2023-04-16T15:23:45Z",
    "region": "us-east-1",
    "account": "123456789012",
    "detail": {
        "schemaVersion": "<version number>",
        "catalog": "<Sandbox | AWS>",
        "opportunity": {
            "identifier": "String"
        }
    }
}
```

机会已更新

合作伙伴利用此活动来：

- 通知其用户现有机会已更新。
- 触发自动化工作流程，例如更新 CRM 系统或通知销售团队。

以下示例显示了一个典型的机会更新事件。

```
{
    "version": "1",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "source": "aws.partnercentral-selling",
    "detail-type": "Opportunity Updated",
    "time": "2023-04-16T15:23:45Z",
```

```
    "region": "us-east-1",
    "account": "123456789012",
    "detail": {
        "schemaVersion": "<version number>",
        "catalog": "<Sandbox | AWS>",
        "opportunity": {
             "identifier": "String"
        }
    }
}
```

订婚邀请已创建

合作伙伴利用此活动来：

- 通知用户 EngagementInvitation 他们收到的新消息。
- 触发自动工作流程以接受或拒绝邀请，例如更新 CRM 系统或通知销售团队。

以下示例显示了一个典型的互动邀请创建活动。

```
{
    "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
      }
    }
}
```

订婚邀请已接受

合作伙伴利用此活动来：

- 通知其用户 EngagementInvitation 已被接受。
- 更新他们的内部记录，以反映项目中的新合作伙伴。
- 触发自动化工作流程以同步数据或通知其他团队成员有关新参与成员的信息。

以下示例显示了一个典型的 “已接受参与邀请” 事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Invitation Accepted",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-16T15:23:45Z",
    "region": "us-east-1",
    "resources": ["arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/
engi-v7p8z56whnauo"],
    "detail": {
        "catalog": "AWS",
        "engagementInvitation": {
            "arn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-
invitation/engi-v7p8z56whnauo",
            "id": "engi-v7p8z56whnauo",
            "engagementId": "eng-12356whnauo",
            "participantType": "Enum", //Sender/Receiver
            "senderAccountId": "string",
            "receiverAccountId": "string",
            "payloadType": "LeadInvitation" //OpportunityInvitation|LeadInvitation
        }
    }
}
```

订婚邀请被拒绝

合作伙伴利用此活动来：

- 通知其用户 EngagementInvitation 已被拒绝。
- 更新他们的内部记录以反映被拒绝的邀请。
- 触发任何必要的工作流程，例如将拒绝通知销售团队。

以下示例显示了一个典型的参与邀请被拒绝事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Invitation Rejected",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-17T09:48:27Z",
    "region": "us-east-1",
    "resources": ["arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/
engi-v7p8z56whnauo"],
    "detail": {
        "catalog": "AWS",
        "engagementInvitation": {
            "arn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-
invitation/engi-v7p8z56whnauo",
            "id": "engi-v7p8z56whnauo",
            "engagementId": "eng-12356whnauo",
            "participantType": "Enum", //Sender/Receiver
            "senderAccountId": "string",
            "receiverAccountId": "string",
            "payloadType": "LeadInvitation" //OpportunityInvitation|LeadInvitation
        }
    }
}
```

订婚邀请已过期

合作伙伴利用此活动来：

- 通知其用户 EngagementInvitation 已过期，无法再对其采取行动。
- 更新他们的内部记录以反映已过期的邀请。

- 触发任何必要的工作流程，例如将已过期的邀请通知销售团队。

以下示例显示了一个典型的参与邀请已过期事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Invitation Expired",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/engi-
v7p8z56whnauo"
    ],
    "detail": {
      "catalog" : "AWS",
      "engagementInvitation": {
        "arn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement-invitation/
engi-v7p8z56whnauo",
        "id": "engi-v7p8z56whnauo",
        "engagementId": "eng-12356whnauo",
        "participantType": "Enum", //Sender/Receiver
        "senderAccountId": "string",
        "receiverAccountId": "string",
        "payloadType": "LeadInvitation" //OpportunityInvitation|LeadInvitation
      }
    }
}
```

已添加参与成员

合作伙伴利用此活动来：

- 将参与成员资格变更通知其用户。
- 更新他们的内部记录以反映新的参与成员。
- 触发任何必要的工作流程，例如将更改通知团队成员。

以下示例显示了一个典型的参与成员添加事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Member Added",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-16T15:23:45Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-v7p8z56whnauo"
    ],
    "detail": {
      "catalog" : "AWS",
      "engagement": {
        "id": "eng-v7p8z56whnauo"
      },
      "engagementMember": {
        "accountId": "string",
        "companyName": "string"
      }
    }
}
```

参与资源快照已创建

合作伙伴使用此活动来跟踪与项目相关的资源变更并触发自动化工作流程，例如更新 CRM 系统或通知销售团队。快照模板决定了更新的类型：

`OpportunitySummaryView`

- 通知参与的所有成员该活动已更新。

`AwsOpportunitySummaryFullView`

- 跟踪 AWS 专门针对项目中的机会所做的更新。
- 只有机会所有者才能看到此通知和快照。
- 包括交易规模更新，这些更新无法通过`OpportunitySummaryView`模板获得。

以下示例显示了一个典型的互动资源快照创建事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
```

```
    "detail-type": "Engagement Resource Snapshot Created",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-v7p8z56whnauo"
    ],
    "detail": {
      "catalog" : "AWS",
      "resourceSnapshot": {
        "arn": "arn:aws:partnercentral-selling:us-east-1::catalog/AWS/engagement/eng-
v7p8z56whnauo/resource/Opportunity/o-12312/template/temp-name/snapshot/snapshot-19232",
        "engagementId": "eng-v7p8z56whnauo",
        "resourceType": "Opportunity",
        "resourceId" : "O123211231",
        "createdBy": "123123123123",
        "targetMemberAccounts": ["123123123123", "aws"],
        "resourceSnapshotTemplateName": "temp-name"
      }
    }
}
```

参与度已创建

合作伙伴利用此活动来：

- 通知其用户已创建新的互动。
- 触发自动化工作流程，例如更新 CRM 系统或通知销售团队有关新的互动创建。

以下示例显示了一个典型的互动创建事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Created",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-567a1j5hxp35lo"
```

```
    ],
    "detail": {
        "catalog": "AWS",
        "engagement": {
            "engagementArn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/
eng-567a1j5hxp35lo",
            "engagementId": "eng-567a1j5hxp35lo",
            "CreatedAt": "2023-04-18T18:20:15Z",
            "CreatedBy": "aws",
            "ContextTypes": ["Lead"]

        }
    }
}
```

参与度已更新

合作伙伴利用此活动来：

- 通知其用户现有互动已更新。
- 触发自动化工作流程，例如更新 CRM 系统或通知销售团队。

以下示例显示了一个典型的互动更新事件。

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Engagement Updated",
    "source": "aws.partnercentral-selling",
    "account": "123456789012",
    "time": "2023-04-18T18:20:15Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/eng-567a1j5hxp35lo"
    ],
    "detail": {
        "catalog": "AWS",
        "engagement": {
            "engagementArn": "arn:aws:partnercentral:us-east-1::catalog/AWS/engagement/
eng-567a1j5hxp35lo",
            "engagementId": "eng-567a1j5hxp35lo",
            "LastModifiedAt": "2023-04-18T18:20:15Z",
```

```
            "LastModifiedBy": "aws",
            "ContextTypes": ["Lead", "CustomerProject"]
        }
    }
}
```

的通知AWS合作伙伴中心权益 API

福利服务为两者提供 EventBridge BenefitApplications 活动 BenefitAllocations。这些事件使客户能够构建全面的事件驱动架构，以响应整个资源生命周期中的变化，从最初的应用程序到分配再到最终的执行。

主题

- 完成监控事件的先决条件
- 将 Amazon 配置 EventBridge 为监控事件
- 详细了解福利 API 事件

完成监控事件的先决条件

用户需要相应的 IAM 权限才能访问和管理由福利 API 发布的事件。有关可用操作、资源和条件键的更多信息 EventBridge，请参阅亚马逊 EventBridge 用户指南中的在亚马逊 EventBridge中使用 IAM 策略条件。其中一个条件键是`events:detail-type`，它可用于将权限范围限定为特定事件类型。

以下示例策略演示了如何自定义和限定建议事件的权限范围。
该`AllowPutRuleForPartnercentralBenefitsEvents`语句允许创建规则，但仅适用于来自`aws.partnercentral-benefits`源的事件。

有关详细的 IAM 策略示例，请参阅有关 EventBridge 权限的 AWS 文档。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralBenefitsEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-benefits"
```

```
        }
      }
    }
  ]
}
```

将 Amazon 配置 EventBridge 为监控事件

要监控福利 API 事件，您需要创建一个与要捕获的事件相匹配的 EventBridge 规则。您可以使用AWS管理控制台或 S AWS DK 来创建和管理规则。以下各节说明如何使用这两种方法创建规则。无论使用哪种方法，都必须在美国东部（弗吉尼亚北部）us-east-1地区创建规则。

AWS 管理控制台设置

要使用设置 EventBridge 规则AWS 管理控制台，请按照《亚马逊 EventBridge 用户指南》中创建响应亚马逊事件的规则 EventBridge中的步骤进行操作。创建规则时，必须将事件总线设置为默认值，并在美国东部（弗吉尼亚北部）us-east-1区域创建规则。

以下是事件规则的示例：

```
{
    "source": ["aws.partnercentral-benefits"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS软件开发工具包设置

您可以使用AWS软件开发工具包以编程方式创建和管理 EventBridge 规则。有关更多信息，请参阅 Amazon EventBridge API 参考PutRule中的。

以下示例使用适用于 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
    Name='MyBenefitApplicationCreatedRule',
    EventPattern=
    '{
```

```
        "source": ["aws.partnercentral-benefits"],
        "detail-type": ["Benefit Application Created"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
print('Rule ARN:', response['RuleArn'])
```

详细了解福利 API 事件

以下各节描述了 API 事件类型的好处、触发它们的场景以及事件示例。

事件类型

以下是事件类型及其触发器。

福利申请

- 福利申请已创建：何时创建福利申请
- 福利申请正在审核中：当福利申请提交或申请从 “需要采取行动” 转换回审中时，将触发此事件。
- 需要采取福利申请行动：当内部审核者更新申请状态以表明合作伙伴需要提供其他信息、澄清或修改才能继续审核时，就会发生此事件。
- 福利申请被拒绝：当内部审核者更新申请状态以表明该申请不符合福利标准或要求并已被拒绝时，就会发生此事件
- 福利申请已批准：当内部审核者更新申请状态以表明该申请已获得福利分配批准时，就会发生此事件。
- 福利申请阶段已更改：当内部审核者在业务审查、AWS审核、财务审查等审核过程中更新申请阶段时，就会发生此事件。

福利分配

- 福利分配已激活：当福利分配创建或更新为由内部服务激活时。
- 福利分配已停用：当内部审核者更新分配状态或福利分配自动到期时，就会发生此事件。
- 福利分配已完成：当内部审核者更新分配状态以表明分配已被完全利用时，就会发生此事件。

示例事件

以下各节提供了前面部分中列出的事件的示例。

主题

福利申请已创建

触发上下文

当通过福利服务成功创建新的福利申请 EventBridge 时，“福利申请已创建” 事件将发布到亚马逊。当 CreateBenefitApplication API 调用成功完成时，会发生此事件，表示合作伙伴已发起特定权益的申请。

此事件代表福利申请生命周期的开始，当向客户提交新申请时，该事件会立即通知客户。

收件人

拥有者的账户将收到福利分配的活动。

示例

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

福利申请正在审核中

触发上下文

当福利申请通过福利服务转换为 IN_REVIEW 状态 EventBridge 时，“福利申请审核中” 事件将发布到亚马逊。在以下情况下会发生此事件：

- 合作伙伴通过 SubmitBenefitApplication API 调用提交申请
- 合作伙伴回复反馈后，应用程序将从 ACTION_REQUIRED 转换回 IN_REVIEW
- 申请被撤回，然后重新提交

此事件表示该申请已进入正式审核流程，并且正在接受AWS内部团队的评估。

收件人

拥有者的账户将收到福利分配的活动。

示例

```
{
  "id": "8hi99876-l1ki-e19h-j303-3gg2807n141o",
  "detail-type": "Benefit Application In Review",
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-02-10T16:00:00Z",
  "region": "us-east-1",
  "resources": [
```

```
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-application/
benappl-12345abcdef123"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitApplication": {
      "id": "benappl-12345abcdef123",
      "fulfillmentTypes": ["CREDITS"],
      "status": "IN_REVIEW",
      "stage": "BUSINESS_REVIEW",
      "revision": "ii8f9511-f30c-52e5-b827-557766551123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

需要采取福利申请行动

触发上下文

当福利申请通过福利服务转换为 ACTION_REQUIRED 状态 EventBridge 时，“需要福利申请操作” 事件就会发布到亚马逊。当内部审阅者通过 UpdateBenefitApplicationInternal API 更新申请状态以表明合作伙伴需要提供其他信息、澄清或修改才能继续审核时，就会发生此事件。

此事件代表了应用程序生命周期中的一个关键时刻，需要合作伙伴干预，以推动申请在审核过程中向前发展。

收件人

持有者账户将收到福利分配活动

示例

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

福利申请被拒绝

触发上下文

当福利申请通过福利服务转换为 “已拒绝” 状态 EventBridge 时，“福利申请被拒绝” 事件将发布到亚马逊。当内部审阅者通过 UpdateBenefitApplicationInternal API 更新申请状态以表明该申请不符合福利标准或要求并已被拒绝时，就会发生此事件。

此事件表示应用程序生命周期中的一种终止状态，即申请已被正式拒绝，并且不会继续进行福利分配。

收件人

拥有者的账户将收到福利分配的活动。

示例

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

福利申请已获批准

触发上下文

当福利申请通过福利服务转换为 “已批准” 状态 EventBridge 时，“福利申请已批准” 事件将发布到亚马逊。当内部审核者通过 UpdateBenefitApplicationInternal API 更新申请状态以表明该申请符合所有福利标准和要求并且已获得福利分配批准时，就会发生此事件。

此事件表示成功完成申请审核流程，通常会触发相应的福利分配的创建。

收件人

拥有者的账户将收到福利分配的活动。

示例

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

福利申请阶段已更改

触发上下文

当福利申请的审核阶段通过福利服务更新 EventBridge 时，“福利申请阶段已更改”事件将发布到 Amazon。当内部审阅者将应用程序的阶段更新为可能的枚举值之一（FINANCE_REVIEW、BUSINESS_REVIEW、TECH_REVIEW 和 _REVIEW）时，就会发生此事件。AWS

与状态更改不同，阶段变更是只读的信息更新，无需合作伙伴采取任何措施即可查看内部审查工作流程的进展。

收件人

拥有者的账户将收到福利分配的活动。

示例

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

福利分配已激活

触发上下文

当福利分配通过福利服务转换为 “有效” 状态 EventBridge 时，“激活福利分配” 事件将发布给 Amazon。当内部审核者通过 UpdateBenefitAllocationInternal API 更新分配状态以重新激活之前处于非活动状态的分配时，就会发生此事件。

此事件通常发生在暂时停用（由于到期、政策变更或管理原因）的分配恢复为有效状态，从而使合作伙伴可以再次使用福利时。

收件人

属于合作伙伴的AWS账户将收到消息。换句话说，就是拥有资源的AWS账户。

示例事件

```
{
  "id": "3no44321-q6pn-j64m-o858-8ll7352s696t",
  "detail-type": "Benefit Allocation Activated",
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-03-01T08:15:00Z",
```

```
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-allocation/benalloc-
abc123def456"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitAllocation": {
      "id": "benalloc-abc123def456",
      "fulfillmentType": "CREDITS",
      "status": "ACTIVE"
    },
    "benefitApplication": {
      "id": "benappl-12345abcdef123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

福利分配已停用

触发上下文

当福利分配通过福利服务转换为 “不活动” 状态 EventBridge 时，“福利分配失效” 事件将发布给 Amazon。当内部审核者通过 UpdateBenefitAllocationInternal API 更新分配状态或福利分配自动到期时，就会发生此事件。

此事件通常发生在分配被暂时暂停时（由于到期、政策变更、合规性问题或管理原因），使合作伙伴在重新激活之前无法使用这些权益。

收件人

属于合作伙伴的AWS账户将收到消息。换句话说，就是拥有资源的AWS账户。

示例事件

```
{
  "id": "4op55432-r7qo-k75n-p969-9mm8463t707u",
  "detail-type": "Benefit Allocation Inactivated",
  "source": "aws.partnercentral-benefits",
```

```
  "account": "123456789012",
  "time": "2025-06-30T23:59:59Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-allocation/benalloc-
abc123def456"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitAllocation": {
      "id": "benalloc-abc123def456",
      "fulfillmentType": "CREDITS",
      "status": "INACTIVE"
    },
    "benefitApplication": {
      "id": "benappl-12345abcdef123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

福利分配已完成

触发上下文

当福利分配通过福利服务转换为 “已发货” 状态 EventBridge 时，“福利分配已完成” 事件将发布给 Amazon。当内部审核者通过 UpdateBenefitAllocationInternal API 更新分配状态以表明分配已完全使用或福利条款已得到满足时，就会发生此事件。

此事件代表福利生命周期的结束，表示合作伙伴已成功利用分配的福利或分配已自然结束。

收件人

属于合作伙伴的AWS账户将收到消息。换句话说，就是拥有资源的AWS账户。

示例事件

```
{
  "id": "5pq66543-s8rp-l86o-q070-0nn9574u818v",
  "detail-type": "Benefit Allocation Fulfilled",
```

```
  "source": "aws.partnercentral-benefits",
  "account": "123456789012",
  "time": "2025-12-15T17:30:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:partnercentral:us-east-1:123456789012:benefit-allocation/benalloc-
abc123def456"
  ],
  "detail": {
    "catalog": "AWS",
    "benefitAllocation": {
      "id": "benalloc-abc123def456",
      "fulfillmentType": "CREDITS",
      "status": "FULFILLED"
    },
    "benefitApplication": {
      "id": "benappl-12345abcdef123"
    },
    "benefit": {
      "id": "ben-xyz789uvw012",
      "name": "AWS Partner Credits",
      "program": ["AWS Partner Network"]
    }
  }
}
```

的通知AWS合作伙伴中心渠道 API

频道 API 的事件提供有关频道相关活动的状态变化或详细信息的实时通知。这些事件有助于使您的系统与AWS合作伙伴中心保持同步，并有助于确保及时响应和更新。

先决条件

您需要相应的 IAM 权限才能通过AWS合作伙伴中心渠道 API 访问和管理活动。有关可用操作、资源和条件键的信息 EventBridge，请参阅亚马逊 EventBridge 用户指南中的在亚马逊 EventBridge中使用 IAM 策略条件。其中一个条件键是`events:detail-type`，您可以使用它来将权限范围限定为特定的事件类型。

以下示例策略演示了如何自定义事件的权限并确定其范围。
该`AllowPutRuleForPartnercentralChannelEvents`语句允许创建规则，但仅适用于来自`aws.partnercentral-channel`源的事件。

有关详细的 IAM 策略示例，请参阅有关 EventBridge 权限的 AWS 文档。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPutRuleForPartnerCentralChannelEvents",
      "Effect": "Allow",
      "Action": "events:PutRule",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:source": "aws.partnercentral-channel"
        }
      }
    }
  ]
}
```

将 Amazon 配置 EventBridge 为监控事件

要监控频道 API 事件，您需要创建一个与要捕获的事件相匹配的 EventBridge 规则。您可以使用AWS管理控制台或 S AWS DK 来创建和管理规则。以下各节说明如何使用这两种方法创建规则。无论使用哪种方法，都必须在美国东部（弗吉尼亚北部）us-east-1地区创建规则。

AWS 管理控制台设置

要使用设置 EventBridge 规则AWS 管理控制台，请按照《亚马逊 EventBridge 用户指南》中创建响应亚马逊事件的规则 EventBridge中的步骤进行操作。创建规则时，必须将事件总线设置为默认值，并在美国东部（弗吉尼亚北部）us-east-1区域创建规则。

以下是事件规则的示例：

```
{
    "source": ["aws.partnercentral-channel"],
    "detail": {
        "catalog": ["AWS"]
    }
}
```

AWS软件开发工具包设置

您可以使用AWS软件开发工具包以编程方式创建和管理 EventBridge 规则。有关更多信息，请参阅 Amazon EventBridge API 参考PutRule中的。

以下示例使用适用于 Python 的AWS SDK (Boto3)：

```
import boto3

client = boto3.client('events', region_name='us-east-1')

response = client.put_rule(
    Name='ChannelHandshakeCreatedRule',
    EventPattern=
    '{
        "source": ["aws.partnercentral-channel"],
        "detail-type": ["Channel Handshake Created"],
        "detail": {"catalog": ["AWS"]}
    }',
    State='ENABLED'
)
print('Rule ARN:', response['RuleArn'])
```

了解有关频道 API 事件的更多信息

以下各节描述了渠道 API 事件类型、触发这些事件的场景以及事件示例。

事件类型

以下是事件类型及其触发器。

- 频道握手已创建：创建新频道握手时触发。
- 频道握手已接受：接受新的频道握手时触发。
- 频道握手被拒绝：当新的频道握手被拒绝时触发。

频道握手已创建

以下示例显示了针对不同握手类型的典型Channel Handshake创建事件。

开始服务期握手：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Created",
    "source": "aws.partnercentral-channel",
    "account": "789789789789",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-abc123def456g"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "START_SERVICE_PERIOD",
        "id": "ch-abc123def456g",
        "receiverAccountId": "789789789789",
        "ownerAccountId": "123123123123",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "startServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

撤销服务期限握手：

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

项目管理账户握手：

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

频道握手已接受

以下示例显示了不同握手类型的典型频道握手接受事件。

开始服务期握手：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Accepted",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-abc123def456g"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "START_SERVICE_PERIOD",
        "id": "ch-abc123def456g",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "789789789789",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
```

```
        "handshakeDetail": {
            "startServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

撤销服务期限握手：

```
{
    "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"
```

```
            }
        }
    }
}
```

项目管理账户握手：

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

频道握手被拒绝

以下示例显示了不同握手类型的典型频道握手被拒绝事件。

开始服务期握手：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Rejected",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-abc123def456g"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
        "catalog": "AWS",
        "associatedResourceIdentifier": "rs-jkl012mno345p",
        "handshakeType": "START_SERVICE_PERIOD",
        "id": "ch-abc123def456g",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "789789789789",
        "senderAccountId": "456456456456",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "startServicePeriodHandshakeDetail": {
                "servicePeriodType": "MINIMUM_NOTICE_PERIOD",
                "minimumNoticeDays": "14",
                "endDate": null,
                "startDate": "2025-02-02T00:00:00Z",
                "note": "Test note example"
            }
        }
    }
}
```

撤销服务期限握手：

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

项目管理账户握手：

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "Channel Handshake Rejected",
    "source": "aws.partnercentral-channel",
    "account": "123123123123",
    "time": "2025-02-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:partnercentral:us-east-1:123123123123:catalog/AWS/channel-handshake/
ch-ghi789jkl012m"
    ],
    "detail": {
        "requestId": "12345678-1234-1234-1234-123456789abc",
```

```
        "catalog": "AWS",
        "associatedResourceIdentifier": "pma-abc123def456g",
        "handshakeType": "PROGRAM_MANAGEMENT_ACCOUNT",
        "id": "ch-ghi789jkl012m",
        "ownerAccountId": "123123123123",
        "receiverAccountId": "456456456456",
        "senderAccountId": "123123123123",
        "senderDisplayName": "TestDisplayName",
        "handshakeDetail": {
            "programManagementAccountHandshakeDetail": {
                "program": "SOLUTION_PROVIDER"
            }
        }
    }
}
```

在沙盒中测试

合作伙伴可以使用沙盒在安全和隔离的环境中测试和验证他们的 P AWS artner Central API 交互，从而确保在将解决方案推广到生产环境之前顺利运行。AWS为 Partn AWS er Central API 用户提供了一个动态沙箱，该沙箱返回的响应与生产环境类似。AWS不为沙盒环境提供用户界面。因此，合作伙伴需要依靠计划对策来测试其解决方案。

访问沙盒环境

合作伙伴只要将其关联到合作伙伴中心账户，AWS 账户就可以访问测试环境。有关更多信息，请参阅将您的AWS账户关联到AWS合作伙伴中心账户。每个请求都包含一个用于确定数据环境的`catalog`参数。当设置`catalog`为时`AWS`，它引用生产数据；当设置为时`Sandbox`，它会引用沙盒数据。

有关沙盒环境的重要细节

1. 数据刷新：每年AWS刷新一次沙盒环境中的数据（通常在年初）。刷新后，您可能会丢失测试环境中的一些数据。
2. 测试范围：沙盒环境通常用于功能测试，不用于测试可扩展性或性能。

主题

- 在沙箱中测试AWS合作伙伴中心账户 API
- 在沙箱中测试AWS Partner Central Selling API

- 在沙箱中测试AWS合作伙伴中心权益 API
- 在沙箱中测试AWS合作伙伴中心渠道 API
- 在沙箱中测试AWS合作伙伴中心收入衡量 API

在沙箱中测试AWS合作伙伴中心账户 API

如何使用沙箱

要使用沙箱，请完成以下步骤：

1. 创建一个 IAM 角色：

   在与您的AWS合作伙伴中心账户AWS 账户关联的 IAM 角色中创建。

2. 分配策略：

   将以下策略分配给 IAM 角色。有关更多信息，请参阅添加和删除 IAM 身份权限。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountManagement",
      "Effect": "Allow",
      "Action": [
        "partnercentral:AcceptConnectionInvitation",
        "partnercentral:AssociateAwsTrainingCertificationEmailDomain",
        "partnercentral:CancelConnection",
        "partnercentral:CancelConnectionInvitation",
        "partnercentral:CancelProfileUpdateTask",
        "partnercentral:CreateConnectionInvitation",
        "partnercentral:CreatePartner",
        "partnercentral:DisassociateAwsTrainingCertificationEmailDomain",
        "partnercentral:GetAllianceLeadContact",
        "partnercentral:GetConnection",
        "partnercentral:GetConnectionInvitation",
        "partnercentral:GetConnectionPreferences",
        "partnercentral:GetPartner",
        "partnercentral:GetProfileUpdateTask",
        "partnercentral:GetProfileVisibility",
  "partnercentral:GetVerification",
        "partnercentral:ListConnectionInvitations",
```

```
        "partnercentral:ListConnections",
        "partnercentral:ListPartners",
        "partnercentral:PutAllianceLeadContact",
        "partnercentral:PutProfileVisibility",
        "partnercentral:RejectConnectionInvitation",
        "partnercentral:SendEmailVerificationCode",
        "partnercentral:StartProfileUpdateTask",
  "partnercentral:StartVerification",
        "partnercentral:UpdateConnectionPreferences"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "Sandbox"
          ]
        }
      }
    },
    {
      "Sid": "TaggingAccess",
      "Effect": "Allow",
      "Action": [
        "partnercentral:TagResource",
        "partnercentral:UntagResource",
        "partnercentral:ListTagsForResource"
      ],
      "Resource": [
        "arn:aws:partnercentral:*:*:catalog/Sandbox/partner/*"
      ],
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "Sandbox"
          ]
        }
      }
    }
  ]
}
```

3. 使用 IAM 角色证书：

   在您的解决方案中使用此 IAM 角色的证书（私有密钥和访问密钥）来执行 API 操作。

测试AWS actions

在测试阶段，通常需要模拟AWS动作。该模拟使合作伙伴能够全面测试其与AWS服务集成的完整端到端流程。每个请求都包含一个用于确定数据环境的目录参数。

模拟合作伙伴的创建

要模拟伙伴的创建，请在CreatePartner操作的有效载荷"catalog": "Sandbox"中包含在有效载荷中。

例如：

```
{
  "ClientToken": "client-generated-uuid",
  "Catalog": "Sandbox",
  "LegalName": "Your Name",
  "PrimarySolutionType": "SOLUTION_TYPE",
  "AllianceLeadContact": {
    "FirstName": "Example First Name",
    "LastName": "Example Last Name",
    "BusinessTitle": "Example Title",
    "Email": "example@domain.com"
  },
  "EmailVerificationCode": "123456"
}
```

注意：在沙盒中，电子邮件验证已禁用。您可以使用任何六位数 EmailVerificationCode 进行测试。另外，请勿在请求中使用 “标签”。

其他测试说明

1. 要测试其他操作，请在有效载荷"catalog": "Sandbox"中进行设置。
2. 要在沙盒中测试合作伙伴账户连接，您必须在沙盒目录中创建合作伙伴后执行StartProfileUpdateTask操作。
3. 要移至生产环境，请将目录值从更改Sandbox为AWS。

在沙盒环境中测试事件

合作伙伴可以使用沙盒环境中的事件，以帮助测试基于事件的实现。通过 EventBridge 在事件详细信息中指定来设置AWS 账户与监听沙盒事件catalog: Sandbox的规则相同。有关更多信息，请参阅 的通知AWS合作伙伴中心账户 API。

事件规则示例：

```
{
  "source": ["aws.partnercentral-account"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

指定catalog为的事件规则Sandbox将仅匹配来自沙箱的事件，这些事件是由您在沙盒环境中执行的操作生成的。

在沙箱中测试AWS Partner Central Selling API

如何使用沙箱

1. 创建一个 IAM 角色：

   在与您的AWS合作伙伴中心账户AWS 账户关联的 IAM 角色中创建。

2. 分配策略：

   将以下策略分配给 IAM 角色。有关更多信息，请参阅添加和删除 IAM 身份权限。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "partnercentral:CreateOpportunity",
                "partnercentral:UpdateOpportunity",
                "partnercentral:ListOpportunities",
                "partnercentral:GetOpportunity",
                "partnercentral:GetAwsOpportunitySummary",
                "partnercentral:ListSolutions",
                "partnercentral:AssociateOpportunity",
                "partnercentral:DisassociateOpportunity",
                "partnercentral:AssignOpportunity",
                "partnercentral:SubmitOpportunity",
                "partnercentral:AcceptEngagementInvitation",
```

```
                "partnercentral:CreateEngagementInvitation",
                "partnercentral:RejectEngagementInvitation",
                "partnercentral:GetEngagementInvitation",
                "partnercentral:ListEngagementInvitations",
                "partnercentral:StartEngagementFromOpportunityTask",
                "partnercentral:StartEngagementByAcceptingInvitationTask",
                "partnercentral:CreateResourceSnapshotJob",
                "partnercentral:StartResourceSnapshotJob",
                "partnercentral:CreateEngagement"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": "Sandbox"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:ListEntities",
                "aws-marketplace:DescribeEntity"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:SearchAgreements",
                "aws-marketplace:DescribeAgreement"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws-marketplace:PartyType": "Proposer"
                }
            }
        }
    ]
}
```

3. 使用 IAM 角色证书：

   在您的解决方案中使用此 IAM 角色的证书（私有密钥和访问密钥）来执行 API 操作。

测试AWS actions

在测试阶段，通常需要模拟AWS动作。该模拟使合作伙伴能够全面测试其与AWS服务集成的完整端到端流程。

模拟创建AWS起源机会

要模拟AWS产生机会的创建，请在CreateOpportunity操作的有效载荷中包括"Catalog": "Sandbox"和"Origin": "AWS Referral"。

例如：

```
{
    "Catalog": "Sandbox",
    "Origin": "AWS Referral",
    "OpportunityIdentifier": "O123456",
    "Title": "Test Opportunity",
    ...
}
```

模拟合作伙伴发起的机会的更新

要模拟合作伙伴发起的机会的更新或其他操作，请在有效载荷“Action Required”中使用带"Catalog": "Sandbox"和Lifecycle.ReviewStatus: “Approved”或的UpdateOpportunity操作。

如果您要使用GetOpportunity操作中的有效负载进行更新，请确保将 ID 更改为Identifier，然后移除以下字段：

- CreatedDate
- OpportunityTeam
- RelatedEntityIdentifiers

例如：

```
{
    "Catalog": "Sandbox",
    "Identifier": "O123456",
    "Title": "Updated Test Opportunity",
    "Lifecycle": {
```

```
        "ReviewStatus": "Approved"
    }
    ...
}
```

在沙盒环境中测试事件

合作伙伴可以利用沙盒环境中的机会事件来帮助测试基于事件的实施。通过 EventBridge 在事件详细信息中指定来设置AWS 账户与监听沙盒事件`catalog: sandbox`的规则相同。有关更多信息，请参阅[的通知AWS Partner Central Selling API]。

事件规则示例：

```
{
    "source": ["aws.partnercentral-selling"],
    "detail": {
        "catalog": ["Sandbox"]
    }
}
```

指定`catalog`为的事件规则`Sandbox`将仅匹配来自沙箱的事件，这些事件是由您在沙盒环境中执行的操作生成的。

其他测试注意事项

1. 测试`AssociateOpportunity`操作：
   a. 使用默认解决方案`"S-1234567"`来测试`AssociateOpportunity`操作。
   b. 要测试 Marketplace 报价，请使用您账户中的真实报价 ID。
2. 要移至生产，请将`catalog`值从更改`Sandbox`为`AWS`。

获取帮助

如果您在将 CRM 与 CRM 集成时遇到困难AWS，或者需要测试此处未涵盖的特定场景，请通过以下步骤提出案例来寻求支持：

1. 使用您的[AWS合作伙伴网络凭据登录AWS合作伙伴中心]。
2. 在[合作伙伴中心的 Supp] ort Center 上`Open New Case`，选择记录新案例。完成字段，如下所示：
   a. 支持案例的类型:`AWS Partner Central`.

b. 有关的问题:Partner Central Tools or ACE leads and opportunities.

c. 具体说明：选择最合适的 CRM 集成案例类型。

d. 主题：包括对请求的简要描述。

e. 描述：提供问题、问题、错误和故障排除步骤的详细描述。

f. 附件：包括日志和屏幕截图（如果适用）。

在沙箱中测试AWS合作伙伴中心权益 API

如何使用沙箱

要使用沙箱，请完成以下步骤：

1. 创建一个 IAM 角色：

   在与您的AWS合作伙伴中心账户AWS 账户关联的 IAM 角色中创建。

2. 分配策略：

   将以下策略分配给 IAM 角色。有关更多信息，请参阅添加和删除 IAM 身份权限。

```
{
  "Version": "2012-10-17",
    "Statement": [{
            "Sid": "BenefitsManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListBenefits",
                "partnercentral:GetBenefit",
                "partnercentral:CreateBenefitApplication",
                "partnercentral:AmendBenefitApplication",
                "partnercentral:UpdateBenefitApplication",
                "partnercentral:SubmitBenefitApplication",
                "partnercentral:GetBenefitApplication",
                "partnercentral:CancelBenefitApplication",
                "partnercentral:RecallBenefitApplication",
                "partnercentral:ListBenefitApplications",
                "partnercentral:AssociateBenefitApplicationResource",
                "partnercentral:DisasssociateBenefitApplicationResource",
                "partnercentral:ListBenefitAllocations",
                "partnercentral:GetBenefitAllocation",
                "partnercentral:TagResource",
```

```
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "AWSPartnerOpportunityAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetAwsOpportunitySummary",
                "partnercentral:GetOpportunity",
                "partnercentral:ListOpportunities"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "ListingAWSMarketplaceEntities",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:ListEntities"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSMarketplaceOffersAccess",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:DescribeEntity"
            ],
```

```
            "Resource": [
                "arn:aws:aws-marketplace:::AWSMarketplace/Offer/*"
            ]
        },
        {
            "Sid": "S3PutObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": ["arn:aws:s3:::aws-partner-central-marketplace-ephemeral-
writeonly-files/*"]
        },
        {
            "Sid": "TaggingAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:TagResource",
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": [
                "arn:aws:partnercentral:us-east-1:*:catalog/Sandbox/benefit-
application/*",
                "arn:aws:partnercentral:us-east-1:*:catalog/Sandbox/benefit-
allocation/*"
            ],
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        }
    ]
}
```

3. 使用 IAM 角色证书：

   在您的解决方案中使用此 IAM 角色的证书（密钥和访问密钥）执行 API 操作。

测试AWS actions

在测试阶段，通常需要模拟AWS动作。该模拟使合作伙伴能够全面测试其与AWS服务集成的完整端到端流程。每个请求都包含一个用于确定数据环境的目录参数。

模拟福利申请的创建

要模拟福利应用程序的创建，请在CreateBenefitApplication操作的有效载荷"catalog": "Sandbox"中包含在有效载荷中。

例如：

```
{
  "Catalog": "Sandbox",
  "ClientToken": "String",
  "BenefitIdentifier": "String",
  "FulfillmentTypes": ["String"],
  "BenefitApplicationDetails": JsonDocument,
  "Name": "String",
  "Description": "String",
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ],
  "PartnerContacts": [
    {
      "BusinessTitle": "string",
      "Email": "string",
      "FirstName": "string",
      "LastName": "string",
      "Phone": "string"
    }
  ],
  "FileDetails": [
    {
      "FileURI": "String",
      "BusinessUseCase": "String"
    }
  ],
  "AssociatedResources": [
    {
```

```
      "ResourceType": "BENEFIT_ALLOCATION | OPPORTUNITY",
      "ResourceArn": "ARN"
    }
  ]
}
```

其他测试说明

1. 要测试其他操作，请在有效载荷"catalog": "Sandbox"中进行设置。
2. 沙盒目录和 AWS 目录中的权益 ID 不同。使用进行 GetBenefit API 调用"catalog": "Sandbox"以在沙盒中获取福利标识符。
3. 沙盒目录和 AWS 目录中的福利分配不同。
4. 要移至生产环境，请将目录值从更改Sandbox为AWS。

在沙盒环境中测试事件

合作伙伴可以使用沙盒环境中的事件，以帮助测试基于事件的实现。通过 EventBridge 在事件详细信息中指定来设置监听沙盒事件catalog: Sandbox的规则。AWS 账户有关更多信息，请参阅 的通知 AWS合作伙伴中心权益 API。

事件规则示例：

```
{
  "source": ["aws.partnercentral-benefits"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

指定catalog为的事件规则Sandbox将仅匹配来自沙箱的事件，这些事件是由您在沙盒环境中执行的操作生成的。

在沙箱中测试AWS合作伙伴中心渠道 API

沙盒中的账户不符合渠道权益的资格，也未根据所有计划要求进行验证。

如何使用沙箱

要使用沙箱，请完成以下步骤：

1. 创建一个 IAM 角色：

   在与您的AWS合作伙伴中心账户AWS 账户关联的 IAM 角色中创建。

2. 分配策略：

   将以下策略分配给 IAM 角色。有关更多信息，请参阅添加和删除 IAM 身份权限。

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

```
        "partnercentral:TagResource",
        "partnercentral:UntagResource",
        "partnercentral:ListTagsForResource"
      ],
      "Resource": [
        "arn:aws:partnercentral:*:*:catalog/Sandbox/program-management-account/*",
        "arn:aws:partnercentral:*:*:catalog/Sandbox/channel-handshake/*"
      ],
      "Condition": {
        "StringEquals": {
          "partnercentral:Catalog": [
            "Sandbox"
          ]
        }
      }
    }
  ]
}
```

3. 使用 IAM 角色证书：

   在您的解决方案中使用此 IAM 角色的证书（密钥和访问密钥）执行 API 操作。

测试AWS actions

在测试阶段，通常需要模拟AWS动作。该模拟使合作伙伴能够全面测试其与AWS服务集成的完整端到端流程。每个请求都包含一个用于确定数据环境的目录参数。

模拟项目管理账户的创建

要模拟程序管理帐户的创建，请在`CreateProgramManagementAccount`操作的有效载荷`"catalog": "Sandbox"`中包含在有效载荷中。

例如：

```
{
  "catalog": "Sandbox",
  "accountId": "123456789012",
  "displayName": "Test PMA Name",
  "clientToken": "123abc456def789ghi",
  "program": "SOLUTION_PROVIDER",
  "tags": [
    {
```

```
      "key": "testkey",
      "value": "testvalue"
    }
  ]
}
```

模拟关系的更新

要模拟关系的更新，请在负载`"catalog": "Sandbox"`中使用带的`UpdateRelationship`操作。

例如：

```
{
  "catalog": "Sandbox",
  "displayName": "Updated Test PMA",
  "identifier": "rs-abc123def456g",
  "programManagementAccountIdentifier": "pma-123abc456def7"
}
```

其他测试说明

1. 要测试其他操作，请在有效载荷`"catalog": "Sandbox"`中进行设置。
2. 要移至生产环境，请将目录值从更改`Sandbox`为`AWS`。

在沙盒环境中测试事件

合作伙伴可以使用沙盒环境中的事件，以帮助测试基于事件的实现。通过 EventBridge 在事件详细信息中指定来设置监听沙盒事件`catalog: Sandbox`的规则。AWS 账户有关更多信息，请参阅 [的通知 AWS合作伙伴中心渠道 API]。

事件规则示例：

```
{
  "source": ["aws.partnercentral-channel"],
  "detail": {
    "catalog": ["Sandbox"]
  }
}
```

指定`catalog`为的事件规则`Sandbox`将仅匹配来自沙箱的事件，这些事件是由您在沙盒环境中执行的操作生成的。

获取帮助

如果您在测试中遇到挑战，或者需要测试此处未涵盖的特定场景，请通过以下步骤提出案例来寻求支持：

1. 使用您的AWS合作伙伴网络凭据登录AWS合作伙伴中心。
2. 在合作伙伴中心的 Supp ort Center 上Open New Case，选择记录新案例。完成字段，如下所示：
   a. 支持案例类型：Channel Program
   b. 关于以下的问题：General Program Inquiry
   c. 具体说明：选择最合适的渠道案例类型。
   d. 主题：包括对请求的简要描述。
   e. 描述：详细描述问题、问题、错误和故障排除步骤。
   f. 附件：包括日志和屏幕截图（如果适用）。

在沙箱中测试AWS合作伙伴中心收入衡量 API

如何使用沙箱

要使用沙箱，请完成以下步骤：

1. 创建一个 IAM 角色：

   在与您的AWS合作伙伴中心账户AWS 账户关联的 IAM 角色中创建。
2. 分配策略：

   将以下策略分配给 IAM 角色。有关更多信息，请参阅添加和删除 IAM 身份权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RevenueMeasurementCreation",
            "Effect": "Allow",
            "Action": [
                "partnercentral:CreateRevenueAttribution",
                "partnercentral:CreateMarketplaceRevenueShare",
                "partnercentral:CreateMarketplaceRevenueShareAllocation"
            ],
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueMeasurementListing",
            "Effect": "Allow",
            "Action": [
                "partnercentral:ListRevenueAttributions",
                "partnercentral:ListRevenueAttributionAllocations",
                "partnercentral:ListMarketplaceRevenueShares",
                "partnercentral:ListMarketplaceRevenueShareAllocations"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "RevenueAttributionManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetRevenueAttribution",
                "partnercentral:UpdateRevenueAttribution",
                "partnercentral:GetRevenueAttributionAllocation",
                "partnercentral:StartRevenueAttributionAllocationsTask",
                "partnercentral:GetRevenueAttributionAllocationsTask"
            ],
            "Resource": "arn:aws:partnercentral:*:*:catalog/Sandbox/revenue-
attribution/*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
```

```
            }
        },
        {
            "Sid": "MarketplaceRevenueShareManagement",
            "Effect": "Allow",
            "Action": [
                "partnercentral:GetMarketplaceRevenueShare",
                "partnercentral:GetMarketplaceRevenueShareAllocation",
                "partnercentral:UpdateMarketplaceRevenueShareAllocation"
            ],
            "Resource": "arn:aws:partnercentral:*:*:catalog/Sandbox/marketplace-
revenue-share/*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "TaggingAccess",
            "Effect": "Allow",
            "Action": [
                "partnercentral:TagResource",
                "partnercentral:UntagResource",
                "partnercentral:ListTagsForResource"
            ],
            "Resource": [
                "arn:aws:partnercentral:*:*:catalog/Sandbox/revenue-attribution/*",
                "arn:aws:partnercentral:*:*:catalog/Sandbox/marketplace-revenue-
share/*"
            ],
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "OpportunityAccess",
            "Effect": "Allow",
```

```
            "Action": [
                "partnercentral:ListOpportunities",
                "partnercentral:GetOpportunity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "partnercentral:Catalog": [
                        "Sandbox"
                    ]
                }
            }
        },
        {
            "Sid": "ListingAWSMarketplaceEntities",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:ListEntities"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSMarketplaceEntityAccess",
            "Effect": "Allow",
            "Action": [
                "aws-marketplace:DescribeEntity"
            ],
            "Resource": [
                "arn:aws:aws-marketplace:*:*:AWSMarketplace*/Offer/*"
            ]
        }
    ]
}
```

3. 使用 IAM 角色证书：

   在您的解决方案中使用此 IAM 角色的证书（私有密钥和访问密钥）来执行 API 操作。

测试AWS actions

在测试阶段，通常需要模拟AWS动作。该模拟使合作伙伴能够全面测试其与AWS服务集成的完整端到端流程。每个请求都包含一个用于确定数据环境的目录参数。

模拟收入归因 ID 的创建

要模拟收入归因 ID 的创建，请在CreateRevenueAttribution操作的负载"Catalog": "Sandbox"中包含在有效载荷中。

例如：

```
{
  "Catalog": "Sandbox",
  "ClientToken": "String",
  "Name": "String",
  "Description": "String",
  "TenancyModel": "MULTI_TENANT | SINGLE_TENANT",
  "ProductIdentifier": "String",
  "Tags": [
    {
      "Key": "String",
      "Value": "String"
    }
  ]
}
```

其他测试说明

1. 要测试其他操作，请在有效载荷"Catalog": "Sandbox"中进行设置。
2. 要移至生产环境，请将目录值从更改Sandbox为AWS。

合作伙伴中心使用的代码示例AWS软件开发工具包

以下代码示例展示了如何将合作伙伴中心与AWS软件开发套件 (SDK) 配合使用。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您展示如何通过在一个服务中调用多个函数或与其他AWS 服务服务结合来完成特定任务的代码示例。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

代码示例

- 合作伙伴中心使用的基本示例AWS软件开发工具包
  - 合作伙伴中心使用的操作AWS软件开发工具包
    - AssignOpportunity搭配使用AWS SDK
    - AssociateOpportunity搭配使用AWS SDK
    - CreateOpportunity搭配使用AWS SDK
    - DisassociateOpportunity搭配使用AWS SDK
    - GetAwsOpportunitySummary搭配使用AWS SDK
    - GetEngagementInvitation搭配使用AWS SDK
    - GetOpportunity搭配使用AWS SDK
    - ListEngagementInvitations搭配使用AWS SDK
    - ListOpportunities搭配使用AWS SDK
    - ListSolutions搭配使用AWS SDK
    - RejectEngagementInvitation搭配使用AWS SDK
    - StartEngagementByAcceptingInvitationTask搭配使用AWS SDK
    - StartEngagementFromOpportunityTask搭配使用AWS SDK
    - UpdateOpportunity搭配使用AWS SDK
- 合作伙伴中心使用的场景AWS软件开发工具包
  - 更新机会的关联实体

合作伙伴中心使用的基本示例AWS软件开发工具包

以下代码示例展示了如何将 P AWS artner Central 的基础知识与AWS SDK 结合使用。

示例

- 合作伙伴中心使用的操作AWS软件开发工具包
  - AssignOpportunity搭配使用AWS SDK
  - AssociateOpportunity搭配使用AWS SDK
  - CreateOpportunity搭配使用AWS SDK
  - DisassociateOpportunity搭配使用AWS SDK
  - GetAwsOpportunitySummary搭配使用AWS SDK
  - GetEngagementInvitation搭配使用AWS SDK
  - GetOpportunity搭配使用AWS SDK
  - ListEngagementInvitations搭配使用AWS SDK
  - ListOpportunities搭配使用AWS SDK
  - ListSolutions搭配使用AWS SDK
  - RejectEngagementInvitation搭配使用AWS SDK
  - StartEngagementByAcceptingInvitationTask搭配使用AWS SDK
  - StartEngagementFromOpportunityTask搭配使用AWS SDK
  - UpdateOpportunity搭配使用AWS SDK

合作伙伴中心使用的操作AWS软件开发工具包

以下代码示例演示如何使用AWS软件开发工具包执行各个合作伙伴中心操作。每个示例都包含一个指向的链接 GitHub，您可以在其中找到有关设置和运行代码的说明。

这些代码节选调用了 Partner Central API，是必须在上下文中运行的较大型程序的代码节选。您可以在合作伙伴中心使用的场景AWS软件开发工具包中结合上下文查看操作。

以下示例仅包括最常用的操作。有关完整列表，请参阅 AWS Partner Central API 参考。

示例

- AssignOpportunity搭配使用AWS SDK
- AssociateOpportunity搭配使用AWS SDK

- CreateOpportunity搭配使用AWS SDK
- DisassociateOpportunity搭配使用AWS SDK
- GetAwsOpportunitySummary搭配使用AWS SDK
- GetEngagementInvitation搭配使用AWS SDK
- GetOpportunity搭配使用AWS SDK
- ListEngagementInvitations搭配使用AWS SDK
- ListOpportunities搭配使用AWS SDK
- ListSolutions搭配使用AWS SDK
- RejectEngagementInvitation搭配使用AWS SDK
- StartEngagementByAcceptingInvitationTask搭配使用AWS SDK
- StartEngagementFromOpportunityTask搭配使用AWS SDK
- UpdateOpportunity搭配使用AWS SDK

`AssignOpportunity`搭配使用AWS SDK

以下代码示例演示如何使用 `AssignOpportunity`。

Java

适用于 Java 的 SDK 2.x

将现有机会重新分配给其他用户。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考AssignOpportunity中的。

Python

适用于 Python 的 SDK（Boto3）

将现有机会重新分配给其他用户。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用AssignOpportunity于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含 AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`AssociateOpportunity`搭配使用AWS SDK

以下代码示例演示如何使用 `AssociateOpportunity`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 更新机会的关联实体

Java

适用于 Java 的 SDK 2.x

在机会与各种相关实体之间创建正式关联。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[AssociateOpportunity](#)中的。

Python

适用于 Python 的 SDK（Boto3）

在机会与各种相关实体之间创建正式关联。

```
#!/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 = "05465588"

    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()
```

- 有关 API 的详细信息，请参阅适用AssociateOpportunity于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含 AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`CreateOpportunity`搭配使用AWS SDK

以下代码示例演示如何使用 `CreateOpportunity`。

.NET

适用于 .NET 的 SDK

创建机会。

```
// 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.");
            }
        }
    }
}
```

- 有关 API 的详细信息，请参阅 适用于 .NET 的 AWS SDK API 参考[CreateOpportunity](#)中的。

Java

适用于 Java 的 SDK 2.x

创建机会。

```
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;
    }

}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考CreateOpportunity中的。

Python

适用于 Python 的 SDK（Boto3）

创建机会。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用CreateOpportunity于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`DisassociateOpportunity`搭配使用AWS SDK

以下代码示例演示如何使用 `DisassociateOpportunity`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 更新机会的关联实体

Java

适用于 Java 的 SDK 2.x

移除机会与相关实体之间的现有关联。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[DisassociateOpportunity](#)中的。

Python

适用于 Python 的 SDK（Boto3）

移除机会与相关实体之间的现有关联。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用DisassociateOpportunity于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`GetAwsOpportunitySummary`搭配使用AWS SDK

以下代码示例演示如何使用 `GetAwsOpportunitySummary`。

Java

适用于 Java 的 SDK 2.x

检索AWS机会摘要。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
```

```
import
 software.amazon.awssdk.services.partnercentralselling.model.GetAwsOpportunitySummaryRequ
import
 software.amazon.awssdk.services.partnercentralselling.model.GetAwsOpportunitySummaryResp

/*
 * Purpose
 * PC-API-25 Retrieves a summary of an AWS Opportunity.
 */

public class GetAwsOpportunitySummary {

 static PartnerCentralSellingClient client =
 PartnerCentralSellingClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(DefaultCredentialsProvider.create())
            .httpClient(ApacheHttpClient.builder().build())
            .build();

    public static void main(String[] args) {

     String opportunityId = args.length > 0 ? args[0] : OPPORTUNITY_ID;

     GetAwsOpportunitySummaryResponse response = getResponse(opportunityId);

     ReferenceCodesUtils.formatOutput(response);
    }

 public static GetAwsOpportunitySummaryResponse getResponse(String opportunityId)
 {

  GetAwsOpportunitySummaryRequest getOpportunityRequest =
 GetAwsOpportunitySummaryRequest.builder()
    .catalog(Constants.CATALOG_TO_USE)
          .relatedOpportunityIdentifier(opportunityId)
          .build();

  GetAwsOpportunitySummaryResponse response =
 client.getAwsOpportunitySummary(getOpportunityRequest);

        return response;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[GetAwsOpportunitySummary](#)中的。

Python

适用于 Python 的 SDK（Boto3）

检索AWS机会摘要。

```
#!/usr/bin/env python

"""
Purpose
PC-API-25 Retrieves a summary of an AWS Opportunity.
 LifeCycle.ReviewStatus=Approved
"""
import logging
import boto3
import utils.helpers as helper
from botocore.client import ClientError

from utils.constants import CATALOG_TO_USE

serviceName = "partnercentral-selling"

partner_central_client = boto3.client(
        service_name=serviceName,
        region_name='us-east-1'
)

def get_opportunity(identifier):
    get_opportunity_request ={
        "Catalog": CATALOG_TO_USE,
     "RelatedOpportunityIdentifier": identifier
    }
    try:
        # Perform an API call
        response =
 partner_central_client.get_aws_opportunity_summary(**get_opportunity_request)
        return response
```

```
    except ClientError as err:
        # Catch all client exceptions
        print(err.response)

def usage_demo():
    identifier = "05465588"

    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()
```

- 有关 API 的详细信息，请参阅适用GetAwsOpportunitySummary于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`GetEngagementInvitation`搭配使用AWS SDK

以下代码示例演示如何使用 `GetEngagementInvitation`。

Java

适用于 Java 的 SDK 2.x

检索与合作伙伴共享的参与邀请AWS的详细信息。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考GetEngagementInvitation中的。

Python

适用于 Python 的 SDK（Boto3）

检索与合作伙伴共享的参与邀请AWS的详细信息。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用GetEngagementInvitation于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含 AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`GetOpportunity`搭配使用AWS SDK

以下代码示例演示如何使用 `GetOpportunity`。

.NET

适用于 .NET 的 SDK

获得机会。

```
// 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.");
            }
        }
    }
}
```

- 有关 API 的详细信息，请参阅 适用于 .NET 的 AWS SDK API 参考[GetOpportunity](#)中的。

Go

适用于 Go 的 SDK V2

获得机会。

```
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))
}
```

- 有关 API 的详细信息，请参阅 适用于 Go 的 AWS SDK API 参考GetOpportunity中的。

Java

适用于 Java 的 SDK 2.x

获得机会。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考GetOpportunity中的。

Python

适用于 Python 的 SDK（Boto3）

获得机会。

```
#!/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 = "05465588"

    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()
```

- 有关 API 的详细信息，请参阅适用GetOpportunity于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`ListEngagementInvitations`搭配使用AWS SDK

以下代码示例演示如何使用 `ListEngagementInvitations`。

Java

适用于 Java 的 SDK 2.x

检索发送给合作伙伴的互动邀请列表。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考ListEngagementInvitations中的。

Python

适用于 Python 的 SDK（Boto3）

检索发送给合作伙伴的互动邀请列表。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用ListEngagementInvitations于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`ListOpportunities`搭配使用AWS SDK

以下代码示例演示如何使用 `ListOpportunities`。

.NET

适用于 .NET 的 SDK

列出机会。

```
// 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.");
            }
        }
    }
}
```

- 有关 API 的详细信息，请参阅 适用于 .NET 的 AWS SDK API 参考ListOpportunities中的。

Go

适用于 Go 的 SDK V2

列出机会。

```
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))
}
```

- 有关 API 的详细信息，请参阅 适用于 Go 的 AWS SDK API 参考ListOpportunities中的。

Java

适用于 Java 的 SDK 2.x

列出机会。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[ListOpportunities](#)中的。

Python

适用于 Python 的 SDK（Boto3）

列出机会。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用ListOpportunities于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`ListSolutions`搭配使用AWS SDK

以下代码示例演示如何使用 `ListSolutions`。

Java

适用于 Java 的 SDK 2.x

检索合作伙伴在合作中心上注册的合作伙伴解决方案的列表。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考ListSolutions中的。

Python

适用于 Python 的 SDK（Boto3）

检索合作伙伴在合作中心上注册的合作伙伴解决方案的列表。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用ListSolutions于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`RejectEngagementInvitation`搭配使用AWS SDK

以下代码示例演示如何使用 `RejectEngagementInvitation`。

Java

适用于 Java 的 SDK 2.x

拒绝AWS共享 EngagementInvitation 的内容。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[RejectEngagementInvitation](#)中的。

Python

适用于 Python 的 SDK（Boto3）

拒绝AWS共享 EngagementInvitation 的内容。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用RejectEngagementInvitation于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

StartEngagementByAcceptingInvitationTask搭配使用AWS SDK

以下代码示例演示如何使用 StartEngagementByAcceptingInvitationTask。

Java

适用于 Java 的 SDK 2.x

通过接受 a 开始订婚 EngagementInvitation。

```
package org.example;

import static org.example.utils.Constants.*;

import org.example.utils.Constants;
import org.example.utils.ReferenceCodesUtils;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.regions.Region;
```

```
import
 software.amazon.awssdk.services.partnercentralselling.PartnerCentralSellingClient;
import
 software.amazon.awssdk.services.partnercentralselling.model.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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[StartEngagementByAcceptingInvitationTask](#)中的。

Python

适用于 Python 的 SDK（Boto3）

通过接受 a 开始订婚 EngagementInvitation。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用StartEngagementByAcceptingInvitationTask于 Python 的 AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含 AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

StartEngagementFromOpportunityTask搭配使用AWS SDK

以下代码示例演示如何使用 StartEngagementFromOpportunityTask。

Java

适用于 Java 的 SDK 2.x

通过接受互动邀请并在合作伙伴的系统中创建相应的机会，从现有机会启动互动流程。

```
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;
 }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[StartEngagementFromOpportunityTask](#)中的。

Python

适用于 Python 的 SDK（Boto3）

通过接受互动邀请并在合作伙伴的系统中创建相应的机会，从现有机会启动互动流程。

```
#!/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 = "05465588"

    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()
```

- 有关 API 的详细信息，请参阅适用StartEngagementFromOpportunityTask于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

`UpdateOpportunity`搭配使用AWS SDK

以下代码示例演示如何使用 `UpdateOpportunity`。

Java

适用于 Java 的 SDK 2.x

更新机会。

```
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;
  }
    }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考UpdateOpportunity中的。

Python

适用于 Python 的 SDK（Boto3）

更新机会。

```
#!/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()
```

- 有关 API 的详细信息，请参阅适用UpdateOpportunity于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

合作伙伴中心使用的场景AWS软件开发工具包

以下代码示例向您展示了如何使用AWS软件开发工具包在 Partner Central 中实现常见场景。这些场景向您展示了如何通过调用 Partner Central 中的多个函数或与其他AWS 服务结合来完成特定任务。每个场景都包含完整源代码的链接，您可以在其中找到有关如何设置和运行代码的说明。

场景以中等水平的经验为目标，可帮助您结合具体环境了解服务操作。

示例

- 更新机会的关联实体

更新机会的关联实体

以下代码示例演示了如何：

- 取消与旧实体的关联。
- 关联新实体。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 还有更多相关信息 GitHub。在 Scenarios 存储库中查找完整示例，了解如何进行设置和运行。

更新机会的关联实体

```
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
disassociateOppornitityResponse(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;
```

```
 }
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的以下主题。
  - AssociateOpportunity
  - DisassociateOpportunity

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 还有更多相关信息 GitHub。在 AWS代码示例存储库中查找完整示例，了解如何进行设置和运行。

更新机会的关联实体

```
#!/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()
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API Reference》中的以下主题。
  - AssociateOpportunity
  - DisassociateOpportunity

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅使用AWS合作伙伴中心 API，其中包含 AWS SDK。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

发行说明

本页概述了AWS合作伙伴中心 API 文档的重大更改，首先是最新更新。

修订历史记录

更改	描述	日期
RC 6.0	• AWS 合作伙伴中心添加了新的合作伙伴收入衡量 API。	2026-06-30
RC 5.3	• ACE 共鸣更新：GetAwsOpportunitySummary 响应已更新。	2026-06-16
RC 5.2	• 欧元货币支持：CreateOpportunity UpdateOpportunity 、 ，CreateEngagementInvitation 现在除了 MRR 货币代码外，还接受EUR。USD当 AWS 分区为aws-eusc（AWS 欧洲主权云）时，货币代码必须为EUR。 • 读取操作中的 EUR: GetOpportunity ListOpportunities 、 ，GetAwsOpportunitySummary 现在返回EURCurrencyCode 字段以寻找aws-eusc分区中的机会。	2026-03-30
RC 5.1	• 欧元货币支持：CreateOpportunity UpdateOpportunity 、 ，CreateEngagementInvitation 现	2026-03-30

更改	描述	日期
	在除了 MRR 货币代码外，还接受EUR。USD当 AWS 分区为aws-eusc（AWS 欧洲主权云）时，货币代码必须为EUR。 • 读取操作中的 EUR: GetOpportunity ListOpportunities 、 ，GetAwsOpportunitySummary 现在返回EURCurrencyCode 字段以寻找aws-eusc分区中的机会。	

更改	描述	日期
RC 5	• 引入账户 API：向AWS合作伙伴中心添加了新的账户 API，使合作伙伴能够管理其账户信息、注册、个人资料、域名管理和合作伙伴连接。 • 账户 API 参考文档：为账户 API 添加了全面的 API 参考文档，包括合作伙伴注册、个人资料管理、域名验证和账户连接工作流程。 • 账户 API 日志：为账户 API 添加了日志记录支持和文档，以帮助审计和跟踪 API 的使用情况。 • 账户 API 通知：为账户 API 添加了事件通知支持，使合作伙伴能够接收账户相关活动的实时更新，包括合作伙伴账户关联。 • 账户 API 配额：为账户 API 添加了服务配额文档。 • 账户 API 的沙盒测试：为账户 API 增加了沙盒测试功能，允许合作伙伴在非生产环境中测试账户管理工作流程。 • 引入福利 API：向AWS合作伙伴中心添加了新的福利 API，使合作伙伴能够通过集中式界面发现、申请和管理AWS合作伙伴计划中的福利。	2025-11-30

更改	描述	日期
	• 福利 API 参考文档：为福利 API 添加了全面的 API 参考文档，涵盖福利申请、分配和配送流程。 • 好处 API 日志：为福利 API 添加了日志记录支持和文档，以帮助审计和跟踪 API 的使用情况。 • 福利 API 通知：增加了对 Benefits API 的事件通知支持，使合作伙伴能够接收福利申请和分配的实时更新。 • 福利 API 配额：添加了福利 API 的服务配额文档。 • Benefits API 的沙盒测试：为 Benefits API 增加了沙盒测试功能，允许合作伙伴在非生产环境中测试福利管理工作流程。 • 更新了销售 API：增强了销售 API 互动操作以支持销售线索管理工作流程。 • 销售 API 文档更新：增强的销售 API 文档，改进了销售线索管理的结构和更新的工作流程。 • 销售 API 通知更新：针对参与和参与邀请活动的增强版事件通知文档。 • API 结构扩展：扩展了文档结构，支持四个不同的 API（账户、福利、销售和渠道），每个 API 的权限、日	

更改	描述	日期
	志、通知、配额和沙盒测试都有专门的章节。 • 增强的 API 使用文档：为账户和福利 API 添加了全面的使用指南，其中包含详细的工作流程和集成模式。 • 支持的地区文档：为账户和福利 API 的支持AWS区域添加了文档。 • 新的筛选条件`ListOpportunities` ：添加了`TargetCloseDate` 筛选条件，允许使用`AfterTargetCloseDate` 和 YYYY-MM-DD 格式`BeforeTargetCloseDate` 字段按预期的截止日期筛选机会。	

更改	描述	日期
RC 4	• 引入渠道 API：向AWS合作伙伴中心添加了新的渠道 API，使合作伙伴能够管理渠道关系和合作伙伴之间的协作。 • 频道 API 参考文档：为频道 API 添加了全面的 API 参考文档，包括所有可用的操作和数据模型。 • 频道 API 权限：添加了特定于频道 API 的 IAM 权限策略和访问控制文档。 • 频道 API 日志：为频道 API 添加了日志记录支持和文档，以帮助审计和跟踪 API 的使用情况。 • 渠道 API 通知：增加了对频道 API 的事件通知支持，使合作伙伴能够接收有关频道相关活动的实时更新。 • 频道 API 配额：为频道 API 添加了服务配额文档。 • Channel API 的沙盒测试：为频道 API 增加了沙盒测试功能，允许合作伙伴在非生产环境中测试频道工作流程。 • API 结构重组：重新组织了文档，以区分销售 API 和渠道 API，每个 API 的权限、日志、通知和配额都有专门的章节。	2025-11-19

更改	描述	日期
	• 支持的地区文档：为销售和渠道 API 的支持AWS区域添加了文档。	

更改	描述	日期
RC 3	• 更新了 API 权限策略：政策已更新，以反映此版本中引入的新操作。确保相应调整您的 IAM 角色和权限。 • 自定义标题使用详情：添加了有关如何使用自定义标头`X-Amzn-User-Agent`来帮助审计和衡量 API 使用情况的详细信息。 • 将操作替换为异步操作：已替换`AcceptOpportunityEngagementInvitation` 为异步操作，`StartEngagementByAcceptingInvitationTask` 并且与 RC2 相比，有效负载格式已更改。 • 实体和动作重命名：已更改`OpportunityEngagementInvitation` 为。`EngagementInvitation` 这些操作的属性也已更改，以与新实体保持一致。以下操作已被替换： • `GetOpportunityEngagementInvitation` 现在是`GetEngagementInvitation` 。邀请中的多个属性已被更改、添加或删除。	2024-10-17

更改	描述	日期
	• ListOpportunityEngagementInvitations 现在是ListEngagementInvitations 。 • RejectOpportunityEngagementInvitation 现在是RejectEngagementInvitation 。 • AssignOpportunity 参数更新：AssignOpportunity 现在取Assignee代了AssigneeEmail 。相应地更新您的实现。 • SubmitOpportunity 功能更改：SubmitOpportunity 现在由异步操作执行StartEngagementFromOpportunityTask 。 SubmitOpportunity 现在接受InvolvementType 和Visibility 。 • 属性重命名和扩展：EstimatedAwsMonthlyRevenue 已更改为。ExpectedCustomerSpend ExpectedCustomerSp	

更改	描述	日期

	end 现在包括新属性，例如Frequency 和TargetCompany 。
	• AWS机会摘要的新操作：机会可见性现在通过一个名为的单独操作提供GetAwsOpportunitySummary 。
	• 处理机会的工作流程更新：使用此版本中引入的新操作更新了处理您的机会的工作流程。
	• 处理机会的工作流程更新：“处理机会” 的工作流程已更新为使用EngagementInvitations ，这改进了 AWS 提供的处理机会。
	• 跟踪更新的工作流程更新：跟踪更新的工作流程现在是处理机会更新，从而提高了监控机会状态的清晰度和效率。
	• 用于列出操作的新过滤器：为ListSolutions 和ListOpportunities 操作提供了新的过滤器。
	  • 对于ListSolutions ，新的过滤器是FilterCategory FilterIdentifier 、和FilterStatus 。
	  • 对于ListOpportunities ，新的过	

更改	描述	日期
	滤器是FilterCustomerCompanyName FilterIdentifier 、FilterLastModifiedDate 、FilterLifeCycleAppRovalStatus 、FilterLifeCycleStage 、和FilterOrigin 。 • 移除AccessControl 子对象：机会模型中的AccessControl 子对象已删除。 NationalSecurity 现在是顶级属性。 • 属性已重新定位到GetAwsOpportunitySummary ：诸如AWSOpportunityTeam 见解 (EngagementScore 、NextBestAction)、来源和InvolvementType 之类的属性现在可通过GetAwsOpportunitySummary 而不是GetOpportunity 。 • InvolvementType 领域简介：引入该InvolvementType 字段以区分 cosell 和仅限可见性的机会。	

更改	描述	日期
	• 活动名称更新：事件已更新为以反映更改。例如，创建的机会参与邀请现在已创建参与邀请。 • 响应负载更新：所有响应负载现在都包含 ID AR and/or N。 • 请求有效载荷更新：所有请求负载现在都使用标识符作为输入，这些标识符接受 ID 或 ARN。 • 互动实体更新：互动实体中的标题属性已重命名`EngagementTitle` 。 • 删除过滤器前缀：所有过滤器都不再包含过滤器前缀，从而简化了筛选机制。 • `StartEngagementFromOpportunity` 操作更新：属性`AwsInvolvement` 已更改为`AwsSubmission` 。 • 邀请项目详情更新：`Invitation.ProjectDetails.TargetCompletionDate` 已重命名`Invitation.ProjectDetails.EstimatedCompletionDate` 。 • 收入估算更新：现在`Invitation.PartnerRevenueEstimate` 是`ExpectedC`	

更改	描述	日期
	ustomerSpend 。数据类型从一个对象更改为包含一个对象的数组。 • 收款人账户更新：该字段ReceiverAccount 已重命名AccountReceiver 。 • 参与邀请新SenderContact 属性：为参与邀请引入了属性。 • Customer.Contact 现在是一个数组而不是一个对象。 OpportunityOwner ，PartnerAccountManager 支持的值为BusinessTitle 。 • PartnerOpportunityTeam 现在称为OpportunityTeam 。 • APNPrograms 现在是ApnPrograms • 参与邀请状态已更新。 • ListEngagementInvitations 现在需要ParticipantType as RECEIVER。	

更改	描述	日期
RC 2	• 引入了一个名为的新实体 OpportunityEngagementInvitation。 • 引入了一个新动作: ListOpportunityEngagementInvitations. • 引入了一个新动作: GetOpportunityEngagementInvitation. • 推出了一项新活动：“已创建机会参与邀请”。 • 替换 AcceptOpportunity 为 AcceptOpportunityEngagementInvitation。 • 替换 RejectOpportunity 为 RejectOpportunityEngagementInvitation。 • 将 “机会已接受” 替换为 “已接受机会参与邀请”。 • 将 “机会被拒绝” 替换为 “机会参与邀请被拒绝”。 • PrimaryNeedsFromAWS 改为 PrimaryNeedsFromAws。 • AWSOpportunityTeam 改为AwsOpportunityTeam 。 • AWSSalesLifeCycle 改为AwsSalesLifeCycle 。 • PrimaryNeedsFromAWSChangeReason 改	2024-08-07

更改	描述	日期
	为PrimaryNeedsFromAwsChangeReason 。 • AWSAccountId 改为AwsAccountId 。 • ExpectedMonthlyAWSRevenue 改为ExpectedMonthlyAwsRevenue 。 • AWSFundingUsed 改为AwsFundingUsed 。 • AWSMarketplaceOffers 改为AwsMarketplaceOffers 。 • Country改为CountryCode 。 • PartnerOpportunityContact 改为PartnerOpportunityTeam 。 • Contact.Title 将字段更新为Contact.BusinessTitle 。 • ContactInfo 将字段更新为OwnerInfo 。 • 将AWS队伍从地图更改为列表。 • 添加了有关已接受名单的详细信息 RejectionReasons。 • 提供了有关如何使用客户令牌的信息。 • 包括有关如何使用的详细信息 UpdateOpportunity。 • 添加了有关使用AWS产品和合作伙伴解决方案的指南。	

更改	描述	日期
	• 提供了有关该 OpportunityEngagementInvitation实体的详细信息。 • 已更新 StateOrRegion ，包括修订后的可接受值列表。 • 修复了多个错误，以改善性能和错误消息。	

更改	描述	日期
RC 1	• [Breaking] 重命名为 ReviewStatus： ApprovalStatus 为了更好地反映其用途，我们已 ReviewStatus 将该 ApprovalStatus 字段重命名为。此外，我们还引入了一种新状态，即 “待提交”，以表示有机会参加选秀。现在 ReviewStatus 将接受以下值：“待提交”、“已提交”、“审核中”、“已批准”、“已拒绝” 和 “需要采取行动”。以前的草稿状态将替换为 “待提交”。 • [Breaking SubmitOpportunity] Action 简介：引入了新的动作。 SubmitOpportunity与当前 CreateOpportunity 同时提交机会的行为不同，您现在可以创建处于 “待提交” 状态的商机，而无需立即关联解决方案或AWS产品。要完成提交，请使用 AssociateOpportunity 操作链接解决方案或产品，然后使用 SubmitOpportunity。这种分离提供了更好的 API 响应性能，并在提交之前提供了灵活的准备阶段。 • [Breaking] 对目录参数的要求和沙盒端点的弃用：“gamma” 端点 (partnercentral-selling-gamma.us-east-1.amazonaws.com) 将在	2024-06-21

更改	描述	日期
	2024 年之后不可用。 7/30 现在，所有 API 操作都需要一个 Catalog 参数来指定该操作是在沙盒还是AWS目录上执行。通知规则现在将根据目录而不是环境名称进行筛选。 • [Breaking] 重命名 OpportunityIdentifier 为 IdentifiAssignOpportunity er：Actions AcceptOpportunity， RejectOpportunity 现在使用标识符而不是与 UpdateOpportunity之 OpportunityIdentifier 保持一致。 • 对于 ApnProgram CustomerUseCase，从 Enum 类型更改为 String，以及 UseCase：我们将将 Project.APNPrograms Marketing.Activity UseCases、、以及 Project.CustomerUseCase 从枚举类型转换为字符串类型，以降低与更改值相关的风险。这些字段将仅接受来自预定义列表的值，但不接受软件开发工具包中原生可用的值。 • 增强的错误处理：API 中的错误处理已得到显著改进，便于更轻松地调试和更快地解决问题。新的错误处理将	

更改	描述	日期
	错误分为两个阶段：1. 请求验证失败：检查数据类型和格式，或 2. 业务验证失败：有效负载中的所有问题都将在一个响应中列出，并指定有错误的字段。此综合反馈使您能够更有效地进行故障排除和更正错误。错误代码和格式已经标准化。 • 客户来源标识符：合作伙伴能够在其请求中包含 X-Amzn-User-Agent 标头，这有助于识别流量来源。使用 [解决方案名称（包括解决方案提供商）] | [版本] 格式，例如：Partn AWS er CRM 连接器 | v2.0.1 • 错误修复，已部署，SDK] 机会创建和AWS账户 ID 没有变化：合作伙伴现在可以启动机会，同时将AWS账户 ID 从 null 更改为有效值，而不会遇到错误。以前，一个错误使合作伙伴无法单独执行这些操作。 • 错误修复，已部署，未更改 SDK] 解决方案与新机会的关联：合作伙伴可以将解决方案与新创建的机会关联起来。以前，当合作伙伴创造新的机会并同时关联解决方案时，解决方案信息就会丢失。此问题已得到解决。	

更改	描述	日期
	• 错误修复，已部署，SDK 未更改] 机会所有者的详细信息显示：已更正某些机会在组合的 “LastName” 字段中而不是预期的 “电子邮件”、“” 和 FirstName “LastName” 字段中显示所有者的详细信息的问题。 • 错误修复，已部署，SDK 未更改] Customer.Account.WebsiteUrl 字段可选：当 Customer.Account.WebsiteUrl 字段为 “是” 时，该 AccessControls.NationalSecurity 字段已变为可选，从而使 API 行为与现有 UI 行为保持一致。如果该 AccessControls.NationalSecurity 字段为 “否” 或为空，则需要该 WebsiteUrl 字段作为业务验证。 • 错误修复，已部署，SDK] 咨询合作伙伴的AWS账户 ID 要求没有变化：用户界面中显示咨询合作伙伴的AWS账户 ID 字段为可选字段的不一致问题已得到解决。根据文档，它现在是一个有条件的必填字段。 • 错误修复，待部署 6/19，SDK] Co-sell 机会阶段改为 “已关闭丢失”：只要提供已关闭的丢失原因，合作伙伴现在就可以成功地将	

更改	描述	日期
	Co-sell 机会阶段更新为 “已关闭丢失”。这解决了之前的错误，上面写着 “缺少必填字段：关闭机会时必须填写已关闭的丢失原因”。 • 文档] 对文档的改进：对文档进行了多项改进以反映这些更改。 • 待修复的已知问题和即将推出的功能列表： • 在对解决方案或产品执行 AssociateOpportunity 操作时，“选件” 中的错误消息显示不正确。 • 能够在沙盒中远程创造机会并模拟AWS验证过程 • 如果邮政编码的格式 XXXXX-XXXX 为美国，则返回 null。 • “后续步骤” 和 “下一步历史记录” 不适用于标记为 “不需要来自的支持” 的机会 AWS。 • 无法获取客户业务问题描述超过 255 个字符的历史机会。 • 能够删除待提交的商机。 • LeadSource 在数据模型中不可用。	

更改	描述	日期
测试版 5	• 已更改 LifeCycle.Approval Status 为 LifeCycle .ReviewStatus • 已更改 Insights.ReviewCom ments 为 LifeCycle .ReviewComments • 已添加 LifeCycle.ReviewSt atusReason （新属性） • 在 “待定” 中添加了新值 PartnerAcceptance.Status • 发布了 dotnet、ja va、javascript、pyth on、ruby 的开发工具包 • 更新了 API 参考指南以反映上述更改。	2024-04-19
测试版 4	• 添加了沙盒测试：能够在沙盒中测试机会事件通知。 • 新增文档：引入了机会事件通知示例规则。 • 添加了 SDK 版本：已发布适用于 Java (V2)、Java script (V2)、(V3) 和 Python DotNet (V3) 的软件开发工具包。 • 更新日期格式：已更新以符合 ISO 标准。	2024-03-04

更改	描述	日期
测试版 3	• 添加了更改日志文件：引入了更改日志文件，以便更好地跟踪更新和更改。 • 更新了 API 参考指南 (PDF)：发布了 PDF 格式的 API 参考指南的新版本。 • 更新了 API 参考指南（网络版）：更新了 API 参考指南的网页版。 • 新增 DotNet SDK：发布了 DotNet SDK，扩展了我们对不同编程环境的支持。 • 更新了 Python Boto 3 SDK (V2)：发布了 Python Boto 3 SDK 的第 2 版，引入了新功能和改进。	2024-01-24
测试版 2	• 新增 API 参考指南（网页版）：发布了新版本的 API 参考指南（网页版）。注意：此版本需要在解压后运行本地 Web 服务器才能正常运行。	2024-01-07

更改	描述	日期
测试版 1	• 新增 Boto3（初始版本）：推出了 Boto3 的初始版本，这是一款用于我们服务的 Python SDK。 • 添加了 Beta 版计划指南：为我们的 Beta 版测试计划的参与者引入了指南。 • 添加了 API 参考指南 (PDF)：上传了 PDF 格式的 API 参考指南的第一个版本，为我们的 API 提供了全面的文档。	2023-12-15
测试版 0	此初始版本提供了一套功能，用于管理和优化合作伙伴中心中的AWS合作伙伴互动和机会。	2023-11-15

文档历史记录

以下是此 API 参考的文档版本列表。

变更	说明	日期
为合作伙伴收入衡量 API 添加了文档	AWS合作伙伴中心为合作伙伴收入衡量 (PRM) 添加了新的 API。	2026 年 6 月 30 日
更新了 ACE Resonate 的文档	AWS合作伙伴中心更新了 API 响应，以便 GetAwsOpportunitySummary 采取行动支持 Cosell Motion。	2026 年 6 月 16 日
为合作伙伴中心 MCP 服务器添加了文档	AWS合作伙伴中央代理 MCP 服务器通过模型上下文协议 (MCP) 提供合作伙伴中心工具。	2026 年 3 月 16 日
RC 5 发布	AWS合作伙伴中心 API 的候选版本 5 已发布。在 Partner Central 中引入了新的账户 API，使合作伙伴能够管理其账户信息、注册、个人资料、域名管理和合作伙伴关系。AWS 在 Partner Central 中引入了新的 Benefits API，使合作伙伴能够通过集中式界面发现、申请和管理跨AWS合作伙伴计划的福利。AWS增强了销售 API 互动操作以支持销售线索管理工作流程。为账户、福利 API 添加了全面的使用指南和 API 文档，其中包含详细的工作流程和集成模式。增强的销	2025 年 11 月 30 日

	售 API 文档，用于使用潜在客户情境进行交互管理。	
RC 4 发布	AWS合作伙伴中心 API 的候选 4 版本已发布。推出了用于管理渠道关系和合作伙伴间合作的渠道 API，使合作伙伴在使用账单转账与最终客户进行交易时有资格获得计划权益。对文档进行了重新整理，以支持多个AWS合作伙伴中心 API。	2025 年 11 月 19 日
添加了有关多合作伙伴机会的文档	Multi-partner 机会（预览版）是 Partner Central 中的AWS集成体验，它允许AWS合作伙伴查找其他合作伙伴并与之建立联系，共同销售客户交易。	2024 年 12 月 4 日
RC 3 发布	AWS合作伙伴中心 API 的候选版本 3 已发布	2024 年 10 月 17 日
RC 2 发布	AWS合作伙伴中心 API 的候选版本 2 已发布	2024 年 8 月 7 日
RC 1 发布	AWS合作伙伴中心 API 的候选版本 1 已发布	2024 年 6 月 21 日
测试版 5 版本	AWS合作伙伴中心 API 的测试版 5 已发布	2024 年 4 月 19 日
测试版 4 版本	AWS合作伙伴中心 API 的测试版 4 已发布	2024 年 3 月 4 日
测试版 3 发布	AWS合作伙伴中心 API 的测试版 3 已发布	2024 年 1 月 24 日
测试版 2 版本	AWS合作伙伴中心 API 的测试版 2 已发布	2024年1月7日

测试版 1 版本	AWS合作伙伴中心 API 的测试版 1 已发布	2023 年 12 月 15 日
测试版 0 版本	AWS合作伙伴中心 API 初始版本	2023 年 11 月 15 日