aws

用户指南

AWS 上的 Claude 平台

AWS 上的 Claude 平台: 用户指南

Table of Contents

AWS 上的 Claude 平台是什么？

借助 Anthropic-managed 基础设施通过 AWS 访问 Claude 的全部平台功能。

AWS 上的 Claude 平台为您提供完整的 Anthropic 平台体验，包括消息 API、代理技能、代码执行和测试功能，可通过您的 AWS 账户访问。与 AWS 运营推理堆栈的 Amazon Bedrock 不同，Anthropic 在 AWS 上运行 Claude 平台。AWS 通过 AWS Marketplace 提供身份验证层（Sigv4 或 API 密钥）、IAM-based 访问控制和账单集成。

> **Note**
>
> Anthropic 软件开发工具包支持 AWS 上的 Claude 平台。要了解每种语言的客户端可用性，请参阅 Anthrop ic 客户端 SDK 文档。

平台集成的工作原理

Claude 模型在 Anthropic-managed 基础设施上运行。这是一种商业集成，用于通过 AWS 进行计费和访问。Anthropic 和 AWS 都充当独立的数据处理器。AWS 服务条款适用于 AWS 上的 Claude 平台。Anthropic 的商业服务条款、数据处理附录和使用政策也适用。

请注意以下操作特征。数据可能不存在于 AWS 中。推理可能会传递到 Anthropic 的主云。子服务可能会更改，恕不另行通知。设置每个请求的`inference_geo`参数以将推理固定到特定的地理位置。有关详细信息，请参阅数据驻留。

AWS 上的 Claude 平台遵循与第一方 Claude API 相同的数据保留政策。可根据要求提供零数据保留 (ZDR)。请联系您的 Anthropic 客户代表，为您的账户启用该功能。

AWS 上的 Claude 平台的特点

AWS 上的 Claude 平台提供以下功能：

- Same-day 模型和功能访问权限 — 全新 Claude 模型和 API 功能在第一方 Claude API 上线的当天即可使用。
- 代理技能 — 使用预先构建的技能生成文档（PowerPoint、Excel、Word、PDF）并创建自定义技能。
- 代码执行 — 在 Anthropic 的托管沙箱中运行代码。

- 扩展思维 — 为复杂的推理任务启用扩展思维。
- 流式处理和批处理-使用实时 SSE 流式传输或提交批处理请求以处理高吞吐量工作负载。
- 提示缓存-缓存工具、系统提示和消息历史记录，以减少延迟和成本。
- 文件 API-跨请求上传和引用文件。
- AWS IAM 集成 — 使用标准 IAM 策略和 Sigv4 身份验证控制访问权限。
- 统一 AWS 账单 — 使用基于消费的计量（Marketplace 作为账单后端）通过现有 AWS 账户付款。

有关支持的功能的完整列表，请参阅功能支持。

本指南的组织方式

本指南涵盖以下主题：

- 入门-账户设置、先决条件、身份验证和 SDK 安装。
- 使用 API — 可用模型、提出请求和功能支持。
- 管理您的环境 — 数据驻留、工作空间、Claude 控制台、速率限制和账单。
- 监控和操作 — CloudTrail 记录和请求 ID 跟踪。
- 迁移 — 在 AWS 上从 Amazon Bedrock 迁移到 Claude 平台。
- 安全和访问控制 — IAM 策略和操作参考。

AWS 上的 Claude 平台与亚马逊 Bedrock

这两种产品都允许您通过 AWS 使用 Claude，但它们在架构、API 表面和功能可用性方面有很大不同。

	AWS 上的 Claude 平台	Amazon Bedrock
谁在操作堆栈	Anthropic	采买
API 表面	人类信息 API () `/v1/messages`	Bedrock API（匡威/） InvokeModel
功能可用性	Same-day 作为克劳德 API	取决于 AWS 集成时间表
特工技能	可用	不可用（需要执行代码）
测试版功能	使用`anthropic-beta` 标题传递（请参阅功能支持）	需要 AWS 支持
身份验证	AWS IAM/sigv4 或 API 密钥	AWS IAM/sigv4 或不记名令牌（仅限 C#、Go 和 Java 软件开发工具包）
基本网址	`aws-external-anthropic.{region}.api.aws`	`bedrock-runtime.{region}.amazonaws.com`
SDK 客户端	Platform-specific 客户端类（例如，`AnthropicAWS` 在 Python 中），测试版	`AnthropicBedrock` /Bedrock SDK
控制台	Claude 控制台（`platform.claude.com` 通过 AWS 控制台访问）	基岩控制台
速率限制和配额	由 Anthropic 管理	由 AWS 管理
数据处理器	Anthropic 和 AWS	采买

如果你需要使用 Bedrock API 的 AWS-operated Claude，请参阅 A ma zon Bedrock。

> **Important**
>
> Anthropic 在 AWS 上提供 Claude 平台作为第三方产品。它不受标准的 AWS 合规计划、认证或审计报告（例如 SOC、ISO 或 HIPAA 资格）的保护。客户全权负责自行进行尽职调查，以确保该第三方产品符合其监管、法律和合规要求。

何时选择 Bedrock：如果您的组织需要 AWS-operated 推理、AWS 作为唯一的数据处理者，或者需要符合 AWS 合规计划（包括 FedRamp High、IL4、IL5、SOC、ISO 和 HIPAA 资格），请选择 Amazon Bedrock。Bedrock 完全在 AWS-controlled 基础设施上运行，以 AWS 为运营方。

设置您的 账户

在 AWS 上设置 Claude 平台分为四个阶段：在 AWS 控制台中注册、设置您的 Anthropic 组织、记下您的工作空间 ID 以及登录 Claude 控制台。

> **Note**
>
> 通过 AWS 控制台注册即可配置一个与您的 AWS 账户关联的新 Anthropic 组织。该组织与贵公司在 Anthropic 上的任何现有组织是分开的，包括通过 AWS Marketplace 采购的 Claude Enterprise 组织。来自第一方 Anthropic 组织的 API 密钥、工作空间和 Claude Console 设置不会延续。
>
> 如果您已有 Amazon Bedrock 私人优惠，请在注册前联系您的 Anthropic 或 AWS 客户经理，这样您的折扣将从您第一次申请起生效。折扣不能追溯适用于接受您的私人优惠之前产生的使用量。查看私人优惠。

步骤 1：在 AWS 控制台中注册

1. 打开 AWS 控制台并导航到 AW S 服务页面上的 Claude 平台。
2. 选择注册
3. 选择继续。

该页面会显示一个正在Sign-up 进行中的横幅。留在页面上。 Sign-up AWS 会花几分钟时间为您处理 AWS Marketplace 订阅，然后自动重定向您。

如果您的组织有 Anthropic 的私有报价，控制台会在 AWS Marketplace 中查找并提示您接受该报价。详情请参阅私人优惠。

> **Note**
>
> 如果您在 AWS 上使用 Claude 平台，则您的内容（例如提示和完成）将由 AWS 之外的 Anthropic 处理。有关如何处理和存储内容和元数据的详细信息，请参阅 Anthropic 的数据使用政策。

第 2 步：设置你的 Anthropic 组织

注册完成后，您将被重定向到。`platform.claude.com/partner-signup`

1. 输入组织所有者的电子邮件地址，然后选择开始。
2. 查看电子邮件收件箱中是否有设置链接，然后点击。如果您的浏览器显示 “以其他帐户身份登录” 页面，请选择注销并继续。
3. 填写组织详细信息表（组织名称、实体类型、国家、预期用途），然后选择完成设置。

完成设置即可创建您的 Anthropic 组织。AWS 服务条款适用于 AWS 上的 Claude 平台。Anthropic 的商业服务条款、数据处理附录和使用政策也适用。AWS 控制台服务页面现在显示左侧导航，包括主页、API 密钥、快速入门和工作空间。

第 3 步：记下您的工作空间 ID

注册时会自动配置默认工作空间。可以为每个区域创建其他工作空间；有关区域绑定、工作空间管理和 IAM 资源范围界定的详细信息，请参阅工作空间。

在 AWS 服务页面的 AWS 控制台 Claude 平台的工作空间下或在 Claude 控制台中找到工作空间 ID（参见使用 Clau de 控制台）。工作空间 ID 使用后`wrkspc_`跟字母数字标识符的格式。

第 4 步：登录 Claude 控制台

对 Claude 控制台的访问是通过 AWS IAM 进行联合的：

1. 扮演具有`aws-external-anthropic:AssumeConsole`权限的 IAM 角色。请参阅 IAM 操作。
2. 在 A WS 上的 Claude 平台服务页面上，选择登录。AWS 控制台发出 JWT 并将您重定向到。`platform.claude.com`
3. 首次登录时，系统会提示您输入电子邮件地址。输入您的工作电子邮件。平台会及时配置您的 Claude 控制台用户。

当您通过 AWS 控制台登录时，Claude 控制台的范围将限定在 AWS 上的 Claude 平台组织中。“由 AWS 管理的账户” 指示器显示在 Claude 控制台侧栏中。

账户设置疑难解答

- OutboundWebIdentityFederation“Sign-up 失败：无法启用”：如果您在首次提交时看到此红色横幅，请再次选择 “继续”。IAM 启用可能需要一点时间才能传播。
- 注册期间没有进度指示器： Sign-up 需要几分钟。当 AWS 预配置您的账户时，该页面会显示一个不带进度条的静态Sign-up 进度横幅。
- 点击设置链接后 “以其他帐户登录”：选择注销并继续。该页面使用您输入的电子邮件地址重新对您进行身份验证。
- 登录时出现 “未找到” 消息：此消息可能会在重定向期间短暂出现。你可以把它拒之门外。
- 首次调用 API 后，使用情况页面不显示任何数据：使用情况数据可能需要几分钟才能显示在 Claude 控制台中。

先决条件

在进行 API 调用之前，请确保您已经：

1. 订阅 AWS 上的 Claude 平台的活跃 AWS 账户（参见设置账户）。
2. 已安装并配置了 AWS CLI。
3. 在您的 AWS 账户上@@ 启用出站 Web 联合身份验证（一次性设置步骤；请参阅启用出站 Web 联合身份验证）。
4. 您的工作空间 ID（请参阅获取您的工作空间 ID）。

启用出站 Web 身份联合验证

AWS 网关上的 Claude 平台调用`sts:GetWebIdentityToken`服务器端来创建转发给 Anthropic 的 JWT。默认情况下，每个 AWS 账户都禁用此 STS 功能。每个账户启用一次：

```
aws iam enable-outbound-web-identity-federation
```

如果回复为`[ERROR] (FeatureEnabled) … already enabled`，则说明您的账户已开启该设置，您可以继续前进。验证并检索您账户的发卡机构 URL：

```
aws iam get-outbound-web-identity-federation-info
```

> **Warning**
>
> 如果不执行此步骤，则每个请求都会返回`"Outbound web identity federation is disabled for your account"`。这是最常见的设置错误。

获取您的工作空间 ID

注册时，系统会在每个区域自动配置默认工作空间（请参阅设置账户）。工作空间绑定到单个 AWS 区域。您可以在 Claude 控制台的 “工作空间” 下或 AWS 控制台服务页面的 “工作空间” 部分中找到工作空间 ID。有关区域绑定和 IAM 资源范围界定，请参阅工作空间。

设置`ANTHROPIC_AWS_WORKSPACE_ID`和`AWS_REGION`环境变量，以便 SDK 客户端自动读取它们：

```
export ANTHROPIC_AWS_WORKSPACE_ID='wrkspc_01AbCdEf23GhIj'
export AWS_REGION='us-west-2'
```

该区域为必填字段。如果未设置区域，SDK 客户端会抛出。将`aws_region`/传递`awsRegion`给构造函数，或者设置`AWS_REGION`（或`AWS_DEFAULT_REGION`）。支持所有 AWS 商业区域；选择加入区域要求您先选择该区域的账户。

身份验证

AWS 上的 Claude 平台支持两种身份验证方法：带有 Sigv4 请求签名（主要）的 AWS IAM 和 API 密钥身份验证。两者都使用相同的基本 URL 和请求格式。

Anthropic SDK（请参阅安装 SDK）实现了如下所述的身份验证流程和凭据解析。如果您不使用 Anthropic SDK，则必须针对区域端点构建 SigV4-signed 请求（或将您的 API 密钥作为持有者令牌出示）。`https://aws-external-anthropic.<region>.api.aws`有关 SDK-specific 客户端配置和权威证书解析顺序，请参阅 Anthropic 文档中的 Claude on AWS 设置指南。

SigV4 身份验证

Sigv4 是企业原生路径，可与您现有的 AWS IAM 策略、角色和审计功能集成。使用 AWS 默认凭证提供商链支持的任何方法配置 AWS 证书：

- 环境变量 (`AWS_ACCESS_KEY_ID`、`AWS_SECRET_ACCESS_KEY`、`AWS_SESSION_TOKEN`)
- 共享凭证文件 (`~/.aws/credentials`)
- 共享配置文件 (`~/.aws/config`)，包括 SSO 和 `credential_process`
- IRSA `AWS_WEB_IDENTITY_TOKEN_FILE` 和操作的网络身份（和 `AWS_ROLE_ARN` GitHub ）
- ECS 容器凭证
- EC2 实例元数据服务 (IMDS)

要验证 Anthropic SDK 正在获取您的凭据并解析到正确的区域端点，请按照提出请求中的示例进行测试请求。成功的响应确认证书、区域、基本 URL 和工作空间 ID 均已正确配置。

API 密钥身份验证

要获得更简单的集成路径（本地开发和脚本），您可以使用 API 密钥而不是 Sigv4 进行身份验证。设置`ANTHROPIC_AWS_API_KEY`环境变量或传递`apiKey`给 SDK 构造函数。Anthropic SDK 将密钥作为持有者令牌发送到 AWS 终端节点上的 Claude 平台；IAM 通过`aws-external-anthropic:CallWithBearerToken`操作对请求进行授权（参见 IAM 政策）。

在 AWS 控制台中 AWS 上的 Claude Platform 下生成 API 密钥 → API 密钥。选择生成密钥，然后复制密钥值。将 `aws-external-anthropic:CallWithBearerToken` IAM 操作授予本应允许使用 API 密钥身份验证的委托人。

> **Note**
>
> 只有在 AWS 的 Claude 平台下的 AWS 控制台中创建的 API 密钥才能使用此服务。
>
> - 在标准 Claude 控制台中创建的用于第一方 API 访问的密钥不适用于 AWS 终端节点上的 Claude 平台。
> - Amazon Bedrock API 密钥也不起作用 — Bedrock 使用单独的终端节点、身份验证流程和 IAM 命名空间。

Short-term API 密钥

如果您想要 API 密钥身份验证但没有长期有效密钥的生命周期，Anthropic 会发布令牌生成器库，这些库可以从您的 AWS IAM 证书中生成短期 API 密钥。代币的生命周期默认为 12 小时，并且可以限定为特定的工作空间和操作。`aws-external-anthropic:CallWithBearerToken`安装适用于您的语言的生成器，将 IAM 凭证交换 API 密钥，然后将密钥传递给 Anthropic SDK。

- Python：aws/token-aws-external-anthropic - python 的生成器
- JavaScript / TypeScript: aws/token-aws-external-anthropic -js 的生成器
- Java：aws/token-aws-external-anthropic - java 的生成器

Short-term API 密钥仍要求调用方的 IAM 委托人保留`aws-external-anthropic:CallWithBearerToken`在目标工作空间中。它们会自行过期，无需明确轮换或撤销。

凭证优先级和区域解析

Anthropic SDK 在 AWS 客户端上的 Claude 平台使用定义的优先顺序解析证书和区域。参数名称遵循每种语言的约定：CamelCase 代表 TypeScript 和 PHP，snake_case 代表Python和Ruby PascalCase ，用于 Go，以及 C# 和 Java 的属性或生成器模式。

有关每个软件开发工具包的权威优先顺序、支持的环境变量和构造函数参数，请参阅 Anthropic 文档中关于 AWS 设置的 Claude。通常，显式构造函数参数优先于环境变量，并且`ANTHROPIC_AWS_API_KEY`优先于默认的 AWS 凭证提供者链。区域是必填的；与`AnthropicBedrock`（后退为`us-east-1`）不同，如果构造函数或/没有提供任何区域，AWS 客户端上的 Claude 就会抛出`AWS_REGION`。`AWS_DEFAULT_REGION`

工作空间 ID

每个数据平面请求的`anthropic-workspace-id`标题中都必须包含工作空间 ID。默认情况下，Anthropic SDK 会从`ANTHROPIC_AWS_WORKSPACE_ID`环境变量中读取它，或者你可以将`workspaceId`/`workspace_id`传递给客户端构造函数。如果您使用基础`Anthropic`客户端（不是客户 Claude-on-AWS 端），请显式传递标头。有关如何找到您的工作空间 ID，请参阅为每种语言的示例提出请求，请参阅工作空间。

安装 开发工具包

Anthropic 的客户软件开发工具包支持 AWS 上的 Claude 平台。所有七个软件开发工具包都提供特定于平台的客户端类；或者，您可以使用 AWS 上的 Claude 平台基本 URL 配置基本客户端。

Python

```
pip install -U "anthropic[aws]"
```

> **Tip**
>
> 在装有 Homebrew Python 或其他外部管理的 Python `pip install` 环境的 macOS 上，可能会失败并出现 PEP 668 错误`externally-managed-environment`。首先创建并激活虚拟环境:`python3 -m venv .venv && source .venv/bin/activate`.

TypeScript

```
npm install @anthropic-ai/aws-sdk
```

C#

```
dotnet add package Anthropic.Aws
```

Go

```
go get github.com/anthropics/anthropic-sdk-go
```

Java

```
implementation("com.anthropic:anthropic-java-aws:2.27.0")
```

```
<dependency>
  <groupId>com.anthropic</groupId>
  <artifactId>anthropic-java-aws</artifactId>
  <version>2.27.0</version>
</dependency>
```

PHP

```
composer require anthropic-ai/sdk aws/aws-sdk-php
```

Ruby

```
gem install anthropic aws-sdk-core
```

> **Note**
>
> AWS 上的 Claude 平台的软件开发工具包客户端处于测试阶段。

可用的模型

AWS 上的 Claude Platform 提供与第一方 Claude API 相同的 Claude 模型集。

有关模型、模型 ID、上下文窗口大小和功能的当前列表，请参阅 Anthropic 文档中的模型概述。

模型 ID

AWS 上的 Claude 平台上的型号 ID 与第一方 Claude API 上使用的型号 ID 相同（例如、`claude-opus-4-6`、`claude-sonnet-4-6`）。`claude-haiku-4-5`没有 Bedrock-style ARN 或`anthropic.`前缀 — 使用与 Anthropic 发布的模型 ID 完全相同。

提出请求

AWS 上的 Claude 平台使用 Anthropic Messages API (`/v1/messages`)，这与 Claude 第一方平台的 API 表面相同。区别在于基本 URL、身份验证方法和`anthropic-workspace-id`标识请求目标工作空间的必填标头。

壳牌

```
curl "https://aws-external-anthropic.us-west-2.api.aws/v1/messages" \
  --aws-sigv4 "aws:amz:us-west-2:aws-external-anthropic" \
  --user "$AWS_ACCESS_KEY_ID:$AWS_SECRET_ACCESS_KEY" \
  -H "x-amz-security-token: $AWS_SESSION_TOKEN" \
  -H "content-type: application/json" \
  -H "anthropic-version: 2023-06-01" \
  -H "anthropic-workspace-id: $ANTHROPIC_AWS_WORKSPACE_ID" \
  -d '{
    "model": "claude-sonnet-4-6",
    "max_tokens": 1024,
    "messages": [
      {"role": "user", "content": "Hello!"}
    ]
  }'
```

Python

```
from anthropic import AnthropicAWS

client = AnthropicAWS()

message = client.messages.create(
    model="claude-sonnet-4-6",
    max_tokens=1024,
    messages=[{"role": "user", "content": "Hello!"}],
)
print(message)
```

TypeScript

```
import AnthropicAws from "@anthropic-ai/aws-sdk";

const client = new AnthropicAws();
```

```
const message = await client.messages.create({
  model: "claude-sonnet-4-6",
  max_tokens: 1024,
  messages: [{ role: "user", content: "Hello!" }]
});
console.log(message);
```

C#

```
using Anthropic;
using Anthropic.Aws;

var client = new AnthropicAwsClient();

var message = await client.Messages.Create(new()
{
    Model = "claude-sonnet-4-6",
    MaxTokens = 1024,
    Messages = [new() { Role = "user", Content = "Hello!" }]
});

Console.WriteLine(message);
```

Go

```
client, err := anthropicaws.NewClient(context.Background(),
 anthropicaws.ClientConfig{})
if err != nil {
    panic(err)
}

message, err := client.Messages.New(context.Background(), anthropic.MessageNewParams{
    Model:     anthropic.ModelClaudeSonnet4_6,
    MaxTokens: 1024,
    Messages: []anthropic.MessageParam{
        anthropic.NewUserMessage(anthropic.NewTextBlock("Hello!")),
    },
})
if err != nil {
    panic(err)
}
```

```
fmt.Println(message.Content[0].Text)
```

Java

```
import com.anthropic.aws.backends.AwsBackend;
import com.anthropic.client.AnthropicClient;
import com.anthropic.client.okhttp.AnthropicOkHttpClient;
import com.anthropic.models.messages.Message;
import com.anthropic.models.messages.MessageCreateParams;
import com.anthropic.models.messages.Model;

AnthropicClient client = AnthropicOkHttpClient.builder()
    .backend(AwsBackend.fromEnv())
    .build();

Message message = client.messages().create(
    MessageCreateParams.builder()
        .model(Model.CLAUDE_SONNET_4_6)
        .maxTokens(1024)
        .addUserMessage("Hello!")
        .build()
);

IO.println(message);
```

PHP

```
use Anthropic\Aws\Client;

$client = new Client();

$message = $client->messages->create(
    model: 'claude-sonnet-4-6',
    maxTokens: 1024,
    messages: [['role' => 'user', 'content' => 'Hello!']],
);

echo $message;
```

Ruby

```
require "anthropic"
```

```
client = Anthropic::AWSClient.new

message = client.messages.create(
  model: "claude-sonnet-4-6",
  max_tokens: 1024,
  messages: [{ role: "user", content: "Hello!" }]
)

puts message
```

客户端ANTHROPIC_AWS_WORKSPACE_ID从环境中读取AWS_REGION（或AWS_DEFAULT_REGION）和。你可以通过将aws_region/awsRegion或workspace_id/传递给构造函数workspaceId来覆盖。区域和工作空间 ID 均为必填项；如果无法从这些源中解析出任何一个，则构造函数会抛出。

> **Note**
>
> 只有临时证书（例如 IAM 角色、SSO 或 STS）才需要x-amz-security-token标头 (curl)。使用长期 IAM 用户证书时，请将其省略。SDK 客户端会根据凭证来源自动处理此问题。

该--aws-sigv4值遵循格式aws:amz:<region>:<service>。Sigv4 服务名称为aws-external-anthropic，并且该区域必须与您的终端节点 URL 中的区域匹配。两者中的不匹配都会产生一般的签名拒绝错误，而不是特定的诊断。

使用基本的 Anthropic 客户端

如果您不想添加 AWS 开发工具包依赖项，则可以使用基础Anthropic客户端。此路径使用 vanilla SDK 环境变量ANTHROPIC_AWS_*名称，而不是专用客户端读取的名称：

```
export ANTHROPIC_API_KEY='your-api-key'
export ANTHROPIC_BASE_URL='https://aws-external-anthropic.us-west-2.api.aws'
export ANTHROPIC_WORKSPACE_ID='your-workspace-id'
```

Python TypeScript、和 Go SDK ANTHROPIC_BASE_URL 会自动读取ANTHROPIC_API_KEY；对于其他语言，请将它们传递给构造函数。在每种语言中，在anthropic-workspace-id标题中传递工作空间 ID。

Python

```
import os
from anthropic import Anthropic

client = Anthropic(
    # ANTHROPIC_API_KEY and ANTHROPIC_BASE_URL are read automatically
    default_headers={"anthropic-workspace-id": os.environ["ANTHROPIC_WORKSPACE_ID"]},
)

message = client.messages.create(
    model="claude-sonnet-4-6",
    max_tokens=1024,
    messages=[{"role": "user", "content": "Hello!"}],
)
print(message.content[0].text)
```

TypeScript

```
import Anthropic from "@anthropic-ai/sdk";

const client = new Anthropic({
  // ANTHROPIC_API_KEY and ANTHROPIC_BASE_URL are read automatically
  defaultHeaders: { "anthropic-workspace-id": process.env.ANTHROPIC_WORKSPACE_ID }
});

const message = await client.messages.create({
  model: "claude-sonnet-4-6",
  max_tokens: 1024,
  messages: [{ role: "user", content: "Hello!" }]
});
console.log(message.content);
```

功能支持

AWS 上的 Claude 平台直接使用 Anthropic Messages API。您可以获得与第一方 Claude API 相同的完整功能，除非在 “当前不可用的功能” 中另有说明：

- 测试版功能：传递标准`anthropic-beta`标题即可访问测试版功能，就像使用 Claude API 一样。
- 特工技能：使用预建和自定义的特工技能，其`container.skills`参数和测试标题与 Claude API 相同。所有预先构建的技能（PowerPoint、Excel、Word、PDF）都是开箱即用的。
- 代码执行：使用代码执行工具在 Anthropic 的托管沙箱中运行代码。
- 工具使用：计算机使用和所有其他工具使用功能都可用。
- 扩展思维：使用与 Claude API 相同的参数启用扩展思维。
- 直播：完整的 SSE 流媒体支持，可实现实时响应。
- 批处理：为高吞吐量工作负载提交批处理请求。
- 提示缓存：缓存工具、系统提示和消息历史记录，以减少延迟和成本。所有提示缓存功能（5 分钟 TTL、1 小时 TTL 和自动缓存）均可用。
- 文件 API：跨请求上传和引用文件。
- 集成了 IAM 的工作空间标签：可以对工作空间进行标记，标签流向 IAM 进行基于属性的访问控制，并流向 AWS 成本和使用率报告进行成本分配。工作区标签是 AWS 上的 Claude 平台上的一项正式发布功能（它是第一方 Claude API 的测试版功能）。

Claude Managed Agents

Claude 托管代理可在 AWS 的 Claude 平台上使用。代理、会话、环境、凭证库和内存存储是一流的 IAM 资源。有关操作参考，请参阅 IAM 操作，有关相应页面，请参阅 “使用 Claude 控制台”。Anthropic Agent 软件开发工具包可与 AWS 上的 Claude 平台配合使用，无需额外 将其指向 AWS 基本 URL 上的 Claude 平台，然后使用 sigv4 或 API 密钥进行身份验证。

AWS 上的 Claude 平台上的自主会话要求每 6 小时重新进行一次身份验证。 Long-running 代理运行必须在该窗口内刷新其 Sigv4 凭据或 API 密钥，否则会话就会结束。

以下 Claude 托管代理功能目前在 AWS 的 Claude 平台上不可用：

- 结果：无法跟踪代理会话的结果。
- Multi-agent 会话：与多个交互代理的会话不可用。

合规

> **Important**
>
> Anthropic 将此服务作为第三方产品提供。它不在标准 AWS 合规计划、认证或审计报告（例如 SOC、ISO 或 HIPAA 资格）的涵盖范围内。客户全权负责自行进行尽职调查，以确保该第三方产品符合其监管、法律和合规要求。在处理敏感或受监管的数据之前，您应通过 Anthropic 信任中心查看 Anthropic 的官方合规文件。有关更多信息，请参阅 AWS 服务条款。

如何建模工作空间成员资格

在第一方 Claude API 上，访问权限受用户到工作空间的成员资格控制，即用户被添加到工作区并继承工作区的访问权限。AWS 上的 Claude Platform 用 IAM 授权取代了该模式：没有每个工作空间的用户列表。相反，IAM 委托人（用户或角色）持有授予针对特定工作空间 ARN 的`aws-external-anthropic`操作的策略。IAM 策略评估确定每位委托人在每个工作空间中可以做什么。

通过修改 IAM 策略（SCP、权限边界、身份附加策略或资源级条件）来授予和撤消工作空间访问权限，而不是在 Claude 控制台中管理每个工作区的成员资格。有关示例，请参阅 IAM 政策。

功能目前不可用

以下功能目前在 AWS 的 Claude 平台上不可用：

- HIPAA 准备就绪：Anthropic 的 HIPAA-ready 计划不在 AWS 的 Claude 平台上可用。具有 HIPAA 要求的买家应改为在 Amazon Bedrock 中评估 Claude。
- “标准” 和 “Batch：优先级” 之外的服务等级不可用。
- 管理员 API — 组织成员、工作空间成员、邀请、API 密钥、使用情况报告、费用报告和速率限制报告端点：这些管理员 API 端点目前不可用。工作区 CRUD 和重命名终端节点 (`/v1/organizations/workspaces`) 可通过`aws-external-anthropic`命名空间获得；请参阅 IAM 操作。成员资格是通过 AWS IAM 而非工作空间成员列表建模的（参见前一节）。API 密钥生命周期在 AWS 控制台的 Claude Platform 下管理 AWS → API 密钥。有关使用情况、成本和速率限制数据，请使用 Claude 控制台视图（请参阅监控和记录）或 Anthropic 使用情况和成本 API。
- 支出限额：不可用。改用 AWS 账单控制。
- Claude Code 工作区和 Analytics API：具有自动速率限制的 Claude Code 工作区不可用。Claude Code 的使用情况显示在常规用法视图中，而不是显示在专用屏幕中。

- OAuth 身份验证：不支持。使用 Sigv4 或 API 密钥身份验证。
- OpenAI-compatible API 终端节点：在 AWS 的 Claude 平台上不可用。
- Workspace-level 推理地理位置控件:`allowed_inference_geos`和`default_inference_geo`不可用。改为`inference_geo`在每个请求上进行设置。

数据驻留

AWS 上的 Claude 平台支持以下推理区域：

- 美国：推理保留在美国数据中心内。适用 1.1 倍的定价乘数。
- 全局：推理可以路由到全球任何 Anthropic-operated 数据中心。标准定价适用。

使用以下`inference_geo`参数设置每个请求的推理地理位置：

> **Note**
>
> Claude Opus 4.6、Claude Sonnet 4.6 及更高版本的机型支持该`inference_geo`参数。`inference_geo`在 Claude Opus 4.5、Claude Sonnet 4.5 或 Claude Haiku 4.5 上的请求会返回 400 错误。有关型号可用性的详细信息，请参阅数据驻留。

壳牌

```
curl "https://aws-external-anthropic.us-west-2.api.aws/v1/messages" \
  --aws-sigv4 "aws:amz:us-west-2:aws-external-anthropic" \
  --user "$AWS_ACCESS_KEY_ID:$AWS_SECRET_ACCESS_KEY" \
  -H "x-amz-security-token: $AWS_SESSION_TOKEN" \
  -H "content-type: application/json" \
  -H "anthropic-version: 2023-06-01" \
  -H "anthropic-workspace-id: $ANTHROPIC_AWS_WORKSPACE_ID" \
  -d '{
    "model": "claude-sonnet-4-6",
    "max_tokens": 1024,
    "inference_geo": "us",
    "messages": [
      {"role": "user", "content": "Hello!"}
    ]
  }'
```

Python

```
from anthropic import AnthropicAWS

client = AnthropicAWS(aws_region="us-west-2")
```

```
message = client.messages.create(
    model="claude-sonnet-4-6",
    max_tokens=1024,
    inference_geo="us",
    messages=[{"role": "user", "content": "Hello!"}],
)
print(message)
```

TypeScript

```
import AnthropicAws from "@anthropic-ai/aws-sdk";
const client = new AnthropicAws({ awsRegion: "us-west-2" });
const message = await client.messages.create({
  model: "claude-sonnet-4-6",
  max_tokens: 1024,
  inference_geo: "us",
  messages: [{ role: "user", content: "Hello!" }]
});
console.log(message);
```

C#

```
using Anthropic;
using Anthropic.Aws;

var client = new AnthropicAwsClient(new() { AwsRegion = "us-west-2" });

var message = await client.Messages.Create(new()
{
    Model = "claude-sonnet-4-6",
    MaxTokens = 1024,
    InferenceGeo = "us",
    Messages = [new() { Role = "user", Content = "Hello!" }]
});

Console.WriteLine(message);
```

Go

```
client, err := anthropicaws.NewClient(context.Background(),
 anthropicaws.ClientConfig{})
if err != nil {
```

```
    panic(err)
}

message, err := client.Messages.New(context.Background(), anthropic.MessageNewParams{
    Model:        anthropic.ModelClaudeSonnet4_6,
    MaxTokens:    1024,
    InferenceGeo: anthropic.String("us"),
    Messages: []anthropic.MessageParam{
        anthropic.NewUserMessage(anthropic.NewTextBlock("Hello!")),
    },
})
if err != nil {
    panic(err)
}

fmt.Println(message)
```

Java

```
import com.anthropic.aws.backends.AwsBackend;
import com.anthropic.client.AnthropicClient;
import com.anthropic.client.okhttp.AnthropicOkHttpClient;
import com.anthropic.models.messages.Message;
import com.anthropic.models.messages.MessageCreateParams;
import com.anthropic.models.messages.Model;
import software.amazon.awssdk.regions.Region;

AnthropicClient client = AnthropicOkHttpClient.builder()
    .backend(AwsBackend.builder().region(Region.of("us-west-2")).build())
    .build();

Message message = client.messages().create(
    MessageCreateParams.builder()
        .model(Model.CLAUDE_SONNET_4_6)
        .maxTokens(1024)
        .inferenceGeo("us")
        .addUserMessage("Hello!")
        .build()
);

IO.println(message);
```

PHP

```php
use Anthropic\Aws\Client;

$client = new Client(awsRegion: 'us-west-2');

$message = $client->messages->create(
    model: 'claude-sonnet-4-6',
    maxTokens: 1024,
    inferenceGeo: 'us',
    messages: [['role' => 'user', 'content' => 'Hello!']],
);

echo $message;
```

Ruby

```ruby
require "anthropic"

client = Anthropic::AWSClient.new(aws_region: "us-west-2")

message = client.messages.create(
  model: "claude-sonnet-4-6",
  max_tokens: 1024,
  inference_geo: "us",
  messages: [{ role: "user", content: "Hello!" }]
)

puts message
```

如果省略`inference_geo`，则请求默认为。`global`

Workspace-level 推理地理位置控制（`allowed_inference_geos`和`default_inference_geo`）在 AWS 的 Claude 平台上不可用。改为`inference_geo`在每个请求上进行设置。

Workspaces

AWS 上的 Claude 平台上的推理和资源请求以工作空间为目标。您可以在这些 API 调用的`anthropic-workspace-id`标题中传递工作区的 ID。工作区 ID 使用带标签的格式，`wrkspc_`后跟字母数字标识符（例如，`wrkspc_01AbCdEf23GhIj`）。如果您还没有，请参阅获取您的工作空间 ID。

工作空间范围界定

工作空间绑定到单个 AWS 区域。在中创建的工作空间`us-west-2`只能通过`us-west-2`端点进行访问。使用情况、配额、成本、文件、批次和技能都会汇总每个工作空间，从而在 Claude 控制台中为您提供按区域划分的细分。

工作空间还是 AWS 上 Claude 平台的主要 IAM 资源。您可以使用工作空间 ARN 通过 AWS IAM 策略授予或拒绝访问特定工作区的权限。ARN 的资源段与您在标题中传递`wrkspc_`的带前缀的 ID 相同：`anthropic-workspace-id`

```
arn:aws:aws-external-anthropic:{region}:{account-id}:workspace/{workspace-id}
```

例如：`arn:aws:aws-external-anthropic:us-west-2:123456789012:workspace/wrkspc_01AbCdEf23GhIj`

有关策略示例，请参阅 IAM 政策。

创建工作空间

注册时会自动配置默认工作空间。其他工作空间可以通过`/v1/organizations/workspaces`终端节点创建、更新、重命名和存档，并由相应的`aws-external-anthropic`操作（、`CreateWorkspace`、`UpdateWorkspace`、`ArchiveWorkspace`）授权；请参阅 IAM 操作。

在客户端中设置工作空间 ID

每个数据平面请求的`anthropic-workspace-id`标题中都必须包含工作空间 ID。当你使用 Anthropic SDK 的 Claude-on-AWS 客户端时，有三种方式可以提供它：

1. 环境变量 — 设置`ANTHROPIC_AWS_WORKSPACE_ID`，客户端会自动读取它。

```
export ANTHROPIC_AWS_WORKSPACE_ID='wrkspc_01AbCdEf23GhIj'
```

2. 构造函数参数 — 在实例化客户端时传递`workspaceId`/`workspace_id`（名称遵循 SDK 的语言惯例）。
3. Per-request 覆盖 — 如果您需要处理来自同一客户端的多个工作空间，请直接在单个请求上传递标头。

如果您使用基础`Anthropic`客户端（没有 AWS 后端），请为每个请求明确设置`anthropic-workspace-id`标头 — 有关每种语言的示例，请参阅提出请求。有关 SDK-specific 参数名称，请参阅 Ant hropic 客户端软件开发工具包文档。

标记工作区

工作区标签是附加到工作空间的键值对。它们与 AWS IAM 集成以实现基于属性的访问控制，并与 AWS 成本和使用率报告集成以进行成本分配（CUR 详情请参阅监控和记录）。在 AWS 的 Claude 平台上，标签已正式发布。

使用命名空间`TagResource`和`UntagResource`命名空间更新现有工作`aws-external-anthropic`空间上的标签。相同的标签密钥无需额外配置即可推动 IAM 条件和成本分配。

Tag-based 访问控制

您可以使用`aws:ResourceTag/<key>`条件上下文密钥对工作区标签值进行门禁`aws-external-anthropic`操作。以下策略仅允许对已标记的工作空间进行推理：`Environment=production`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "aws-external-anthropic:CreateInference",
        "aws-external-anthropic:CreateBatchInference",
        "aws-external-anthropic:CountTokens"
      ],
      "Resource": "arn:aws:aws-external-anthropic:*:*:workspace/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Environment": "production"
```

```
        }
      }
    }
  ]
}
```

使用aws:RequestTag/<key>和aws:TagKeys条件键TagResource来强制执行标记策略（例如，要求工作空间携带CostCenter和Owner标记）。有关更多示例，请参阅 IAM 政策。

使用 Claude 控制台

AWS 上的 Claude 平台使用 platform.claude.com 上的标准克劳德控制台。当您从 AWS 控制台登录时，Claude 控制台侧栏中会显示 “由 AWS 管理的账户” 指示器。控制台的范围适用于您的 AWS 组织上的 Claude 平台。它提供使用情况分析、成本明细、速率限制可见性、工作空间可见性以及用于管理文件、代理技能、批处理作业、代理、会话、环境、凭据库和内存存储的页面。

登录

对 Claude 控制台的访问是通过 AWS IAM 进行联合的。有关完整的首次登录流程，请参阅设置您的账户。简而言之：

1. 扮演具有`aws-external-anthropic:AssumeConsole`权限的 IAM 角色。请参阅 IAM 操作。
2. 在 AWS 控制台中导航到 AWS 上的 Claude 平台页面。
3. 选择 “打开 Claude 控制台”。AWS 控制台发出 JWT 并将您重定向到。`platform.claude.com`
4. 首次登录时，系统会提示您输入电子邮件地址；请输入您的工作电子邮件。平台会及时配置您的 Claude 控制台用户。

有两个 Claude 控制台角色可用：管理员和开发人员。管理员角色授予访问所有 Claude 控制台页面和可用于 AWS 上的 Claude 平台的设置的权限。开发者角色授予对使用情况、成本、速率限制和工作空间信息的读取权限。请联系您的 Anthropic 客户代表，将管理员或开发者角色分配给委托人。

可用页面

通过 AWS 网关列指示该页面是否通过 AWS 网关读取和写入数据（因此受 IAM 操作的约束）。标有 “否” 的页面直接通过 Anthropic API 读取组织级别的元数据，并绕过 IAM 操作检查。

页面	可用	通过 AWS 网关	注意
用法	是	否	按型号、工作空间和维度查看代币使用情况。收到请求后，数据可能需要几分钟才能显示。
成本	是	否	按模型和工作空间查看成本明细。AWS Cost Explorer 显示了汇总的 CCU 行项目。

页面	可用	通过 AWS 网关	注意
限制	是	否	查看速率限制（只读）。
工作空间	是	否	查看每个区域的工作空间（只读）。
文件	支持	是	查看和管理上传的文件。
技能	支持	是	查看和管理特工技能。
批次	支持	是	查看和管理批处理作业。
座席	支持	是	查看和管理代理定义。
会话	支持	是	查看代理会话和事件历史记录。
环境	支持	是	查看和管理会话的云容器配置。
凭证保管库	支持	是	查看和管理用于会话身份验证的凭据库。
内存存储	支持	是	查看和管理永久代理内存。
API 密钥	否	N/A	在 AWS 控制台中管理 API 密钥（AW S 上的 Claude 平台 → API 密钥）。请参阅身份验证。
成员	否	N/A	不适用。AWS IAM 负责管理访问权限。
Billing	否	N/A	不适用。AWS Marketplace 管理账单和发票。在 “成本” 页面上查看成本明细。
克劳德密码	否	N/A	在 “使用情况” 页面上查看 Claude 代码的使用情况。

切换组织

Claude 控制台不支持在 AWS 上切换组织 Claude 平台。要访问其他组织，请使用该组织的 AWS 账户的 IAM 角色登出并通过 AWS 控制台重新进行身份验证。

速率限制和配额

AWS 上的 Claude 平台会在您订阅时分配第 1 级速率限制。Anthropic 直接管理速率限制，而不是通过 AWS 配额系统进行管理。

默认限制

AWS 上的 Claude Platform 使用 Anthropic 的标准等级计划，与第一方 Claude API 相同。每个工作空间都适用第 1 层限制。限额按模型系列汇总。例如，所有 Opus 型号共享一个组合限制，所有 Sonnet 型号共享另一个限制，所有 Haiku 型号共享三分之一。

有关当前的 1 级值（RPM、ITPM、OTPM）和更高级别的阈值，请参阅 Anthropi c 文档网站上的速率限制。Anthropic 页面是真实信息来源，当限制发生变化时会更新。

速率限制标题

每个响应都包含报告您当前速率限制状态的标题。关键标题：

- `anthropic-ratelimit-requests-limit`— 每分钟的最大请求数
- `anthropic-ratelimit-requests-remaining`— 当前窗口中剩余的请求
- `anthropic-ratelimit-requests-reset`— 请求限制重置的时间 (RFC 3339)
- `anthropic-ratelimit-tokens-limit`— 每分钟最大组合代币（输入 + 输出）
- `anthropic-ratelimit-tokens-remaining`— 当前窗口中剩余的组合代币
- `anthropic-ratelimit-tokens-reset`— 组合代币限额重置的时间 (RFC 3339)
- `anthropic-ratelimit-input-tokens-limit/-remaining/-reset`— Input-token-specific 标题
- `anthropic-ratelimit-output-tokens-limit/-remaining/-reset`— Output-token-specific 标题
- `retry-after`— 在 429 响应中，重试之前要等待的秒数

有关完整内容，请参阅 Anthropic 文档网站上的响应标头。

申请更高的限额

与第一方 Claude API 不同，自动升级等级不适用于 AWS 上的 Claude 平台。要申请更高的限制，请联系您的 Anthropic 客户代表，告知您的工作空间 ID 和所需的吞吐量。有关等级阈值和其他详细信息，请参阅 Anthropic 文档网站上的速率限制。

速率限制错误

当您超过速率限制时，API 会返回带有`rate_limit_error`类型的 HTTP 429。在重试逻辑中使用抖动实现指数退避。标`retry-after`题指示在重试之前要等待多少秒。

计费

AWS 上的 Claude Platform 使用 AWS Marketplace 作为基于消费的计量的账单后端。Anthropic 通过 AWS Marketplace 按小时计量使用情况，AWS 会根据您的现有 AWS 账单开具月度发票。

账单仅为拖欠款（您按实际用量付费）和税前（AWS 应用您账户的税务设置）。

使用量以 Claude 消费单位 (CCU) 计价，每个 CCU 0.01 美元。CCU 价格是固定的，从不打折。Anthropic 按每个型号、每个功能的标准费率对您的代币使用量进行美元评级，应用任何协议折扣，然后将结果转换为每个 CCU 0.01 美元的 CCU。折扣会减少计量的 CCU，而不是降低 CCU 价格。CCU 不是预付积分；没有 CCU 余额或承诺。有关 CCU 定义和每种型号的代币费率，请参阅定价。

监控和日志记录

AWS CloudTrail 可以在 AWS 上捕获对 Claude 平台的所有请求。默认情况下，工作空间和文件库操作被记录为管理事件。所有其他操作（推理、批处理、文件、技能、模型、用户配置文件和 Claude Managed Agent，保管库除外）均为数据事件。记录它们需要进行明确的配置，并且会产生额外 CloudTrail 费用。有关完整的事件类型分类，请参阅 IAM 操作；有关配置的详细信息，请参 CloudTrail 阅 AWS 文档。

配置 CloudTrail 数据事件记录时，请选择资源类型`AWS::AWSExternalAnthropic::Workspace`。这是 AWS 工作空间上的 Claude Platform 的 CloudTrail 资源类型，是捕获数据平面事件（推理、批处理、文件和其他按工作空间执行的操作）所必需的。如果您想记录工作空间子集而不是整个账户的活动，请将数据事件选择器范围限定为特定的工作空间 ARN。

请求 ID

每个响应的响应标头中都包含两个请求 ID：

- AWS 请求 ID (**`x-amzn-requestid`**)：主编号，已编入索引。 CloudTrail在通过 AWS 工具调查请求或联系 AWS 支持人员时，请使用此选项。
- Anthropic 请求 ID (**`request-id`**)：次要 ID。在联系 Anthropic 支持人员时使用此选项。

Python

```
from anthropic import AnthropicAWS

client = AnthropicAWS(aws_region="us-west-2")

response = client.messages.with_raw_response.create(
    model="claude-sonnet-4-6",
    max_tokens=1024,
    messages=[{"role": "user", "content": "Hello!"}],
)

print(response.headers.get("x-amzn-requestid"))  # AWS request ID
print(response.headers.get("request-id"))  # Anthropic request ID

message = response.parse()
print(message.content)
```

TypeScript

```
import AnthropicAws from "@anthropic-ai/aws-sdk";

const client = new AnthropicAws({ awsRegion: "us-west-2" });

const { data: message, response } = await client.messages
  .create({
    model: "claude-sonnet-4-6",
    max_tokens: 1024,
    messages: [{ role: "user", content: "Hello!" }]
  })
  .withResponse();

console.log(response.headers.get("x-amzn-requestid")); // AWS request ID
console.log(response.headers.get("request-id")); // Anthropic request ID
console.log(message.content);
```

C#

```
using Anthropic;
using Anthropic.Aws;

var client = new AnthropicAwsClient(new() { AwsRegion = "us-west-2" });

var response = await client.WithRawResponse.Messages.Create(new()
{
    Model = "claude-sonnet-4-6",
    MaxTokens = 1024,
    Messages = [new() { Role = "user", Content = "Hello!" }]
});

Console.WriteLine(response.Headers.GetValues("x-amzn-requestid").First()); // AWS
 request ID
Console.WriteLine(response.Headers.GetValues("request-id").First()); // Anthropic
 request ID
Console.WriteLine(response.Value.Content);
```

Go

```
client, err := anthropicaws.NewClient(context.Background(),
 anthropicaws.ClientConfig{})
```

```
if err != nil {
    panic(err)
}

var response *http.Response
message, err := client.Messages.New(
    context.Background(),
    anthropic.MessageNewParams{
        Model:     anthropic.ModelClaudeSonnet4_6,
        MaxTokens: 1024,
        Messages: []anthropic.MessageParam{
            anthropic.NewUserMessage(anthropic.NewTextBlock("Hello!")),
        },
    },
    option.WithResponseInto(&response),
)
if err != nil {
    panic(err)
}

fmt.Println(response.Header.Get("x-amzn-requestid")) // AWS request ID
fmt.Println(response.Header.Get("request-id"))       // Anthropic request ID
fmt.Println(message.Content[0].Text)
```

Java

```
import com.anthropic.aws.backends.AwsBackend;
import com.anthropic.client.AnthropicClient;
import com.anthropic.client.okhttp.AnthropicOkHttpClient;
import com.anthropic.core.http.HttpResponseFor;
import com.anthropic.models.messages.Message;
import com.anthropic.models.messages.MessageCreateParams;
import com.anthropic.models.messages.Model;

AnthropicClient client = AnthropicOkHttpClient.builder()
    .backend(AwsBackend.fromEnv())
    .build();

HttpResponseFor<Message> response = client.messages().withRawResponse().create(
    MessageCreateParams.builder()
        .model(Model.CLAUDE_SONNET_4_6)
        .maxTokens(1024)
        .addUserMessage("Hello!")
```

```
        .build()
);

IO.println(response.headers().values("x-amzn-requestid")); // AWS request ID
IO.println(response.requestId()); // Anthropic request ID
IO.println(response.parse().content());
```

PHP

```
use Anthropic\Aws\Client;

$client = new Client();

$response = $client->messages->raw->create(
    model: 'claude-sonnet-4-6',
    maxTokens: 1024,
    messages: [['role' => 'user', 'content' => 'Hello!']],
);

echo $response->getHeaderLine('x-amzn-requestid') . "\n"; // AWS request ID
echo $response->getHeaderLine('request-id') . "\n"; // Anthropic request ID
echo $response->parse();
```

Ruby

Ruby SDK 目前未公开原始响应访问器，因此无法从 Ruby 中的响应标头中读取请求 ID。如果您需要 request-ID 日志记录，请使用上述任何其他语言，或者在 HTTP 代理层捕获 ID。

Anthropic 建议至少连续记录 30 天的活动，以了解使用模式并调查任何潜在问题。

> **Note**
>
> AWS CloudTrail 是在您的 AWS 账户中配置的。除了账单和服务运营所需的权限外，启用日志记录不会让 AWS 或 Anthropic 访问您的内容。

使用情况指标和仪表板

AWS 上的 Claude Platform 不会向亚马逊发布每个工作空间的使用指标。 CloudWatch代币计数、请求量和每个模型的使用情况可通过 Anthropic 的使用界面获得：

- Claude Console 使用视图 — 当委托人通过`aws-external-anthropic:AssumeConsole`（参见 IAM 政策）联合到 Claude 控制台时，使用情况控制面板会显示可访问工作空间的请求量、令牌数量和每个模型的细分。 Account-wide 使用情况视图需要管理员控制台功能。
- Anthropic 使用情况和成本 API — 要以编程方式访问使用量和成本数据，包括每个工作空间和每个模型的聚合，请参阅 Anthropic 文档中的使用情况和成本 API。

要进行操作监控（错误率、延迟、速率限制标头），请使用每个请求返回的响应标头（参见速率限制和配额）以及中捕获的 AWS 请求 ID CloudTrail。

成本分配和监控

Claude Platform on AWS 费用显示在 AWS 服务上的 Claude 平台下的 AWS 账单上，并附有每个型号的使用维度。将 AWS 成本和使用率报告 (CUR) 与工作空间标签结合使用，进行精细的成本分配：

1. 使用您关心的分配维度（团队、应用程序、环境、成本中心）标记您的工作空间。有关标签的详细信息，请参阅工作区。
2. 在 AWS 账单控制台中，将每个标签密钥激活为成本分配标签。激活后，激活日期当天或之后产生的使用量将按照 CUR 中的标签进行拆分。
3. 在 CUR 或 AWS Cost Explorer 中，按`resourceTags/user:<tag-key>`列分组以按工作空间标签细分支出。

工作区标签会流向 AWS 上所有 Claude 平台使用情况的 CUR 行项目。相同的标签密钥可以驱动 IAM-based 访问控制和成本分配。有关设置步骤和激活延迟，请参阅 AWS 账单文档中的成本分配标签。

从 Amazon Bedrock 迁移

如果您目前在 Bedrock 上使用 Claude，则迁移到 AWS 上的 Claude 平台需要在整个集成过程中进行更改。仍然支持 Sigv4 签名，但签名上下文、基本 URL、API 格式、型号 ID、SDK 客户端和软件包、直播格式、请求标头和区域可用性都发生了变化。下表汇总了差异。

发生了什么变化

	Amazon Bedrock	AWS 上的 Claude 平台
基本网址	`bedrock-runtime.{region}.amazonaws.com`	`aws-external-anthropic.{region}.api.aws`
API 格式	基岩匡威/ InvokeModel	人类信息 API () `/v1/messages`
模型 ID	`anthropic.claude-opus-4-7-v1`	`claude-opus-4-7`
SDK 客户端	`AnthropicBedrock` /Bedrock SDK	Platform-specific 客户端（`AnthropicAWS` `AnthropicAws` 、`AnthropicAwsClient` 、等），测试版
SDK 软件包	`anthropic[bedrock]` `@anthropic-ai/bedrock-sdk`、等	`anthropic[aws]` `@anthropic-ai/aws-sdk` 、等（请参阅安装 SDK）
sigv4 服务名称	`bedrock`	`aws-external-anthropic`
直播格式	AWS EventStream	SSE（与 Claude API 相同）
工作区标题	不适用	`anthropic-workspace-id` 必需
区域可用性	多个商业区域	所有 AWS 商业区域（选择加入区域需要账户选择加入）

你会得到什么

- 通常当天即可访问新模型和功能，无需单独的合作伙伴集成步骤
- 生成文档的代理技能（PowerPoint、Excel、Word、PDF）
- 在 Anthropic 的托管沙箱中执行代码
- `anthropic-beta`标题中的测试版功能（请参阅当前不可用的功能）
- 用于配额可见性和使用情况分析的 Claude 控制台
- 直接的人类支持
- API 密钥身份验证作为 Sigv4 的替代方案（请参阅身份验证）

什么保持不变

- AWS IAM 身份验证 (sigv4)
- 通过您的 AWS 账户计费
- AWS 承诺退休

迁移陷阱

> **Warning**
>
> 首先启用出站 Web 联合身份验证。如果您的 AWS 账户之前未在 AWS 上使用过 Claude 平台，则在发出请求之前，必须为每个账户启用一次出站 Web 联合身份验证。如果不执行此步骤，则所有请求都将失败，并出现联盟错误。对于 Bedrock 来说，此步骤不是必需的。

> **Warning**
>
> 在 AWS 的 Claude 平台上，可以选择启用零数据保留 (ZDR)。在 Bedrock 上，AWS 是数据处理器，Anthropic 不保留推理输入或输出；Anthropic 的 ZDR 计划不适用于此。在 AWS 的 Claude Platform 上，Anthropic 是数据处理者，ZDR 遵循第一方 Claude API 模型：可通过您的 Anthropic 客户代表要求提供。在迁移依赖数据保留保证的生产工作负载之前，请确认 ZDR 注册。

商业考虑

- 服务@@ 条款：AWS 服务条款适用于 AWS 上的 Claude 平台。Anthropic 的商业服务条款、数据处理附录和使用政策也适用。
- 折扣和私人优惠：协议折扣和 AWS Marketplace 私有优惠不会在 AWS 上的 Bedrock 和 Claude 平台之间自动转移。与您的 Anthropic 客户代表合作，为 AWS 上的 Claude 平台制定商业条款。

IAM 策略

AWS 上的 Claude 平台与 AWS IAM 集成以实现访问控制。您可以使用标准的 IAM 策略语法授予或拒绝对特定工作空间上的特定 API 操作的访问权限。

Sigv4 服务名称和 IAM 操作命名空间为。`aws-external-anthropic`操作遵循模式`aws-external-anthropic:<Action>`（例如，`aws-external-anthropic:CreateInference`）。

示例：拒绝批量推理

以下策略允许在阻止批处理的同时进行实时推理，这是 ZDR-sensitive 工作负载的常见要求：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "aws-external-anthropic:CreateInference",
        "aws-external-anthropic:CountTokens",
        "aws-external-anthropic:GetModel",
        "aws-external-anthropic:ListModels",
        "aws-external-anthropic:GetWorkspace",
        "aws-external-anthropic:ListWorkspaces"
      ],
      "Resource": "arn:aws:aws-external-anthropic:*:*:workspace/*"
    },
    {
      "Effect": "Deny",
      "Action": [
        "aws-external-anthropic:CreateBatchInference",
        "aws-external-anthropic:GetBatchInference",
        "aws-external-anthropic:ListBatchInferences"
      ],
      "Resource": "*"
    }
  ]
}
```

该`GetBatchInference`操作同时授权批次元数据路径和批处理结果路径。同时拒绝它`ListBatchInferences`，同时会阻止读取和批量枚举。

该Allow语句列举了具体的Get*和List*操作，而不是使用通配符。通配符会授予GetFile（下载文件字节）和其他你可能不打算的读取；Allow无论如何都会Deny覆盖，但显式形式是更安全的建模模式。

示例：在单个工作空间上进行同步推理

为针对一个生产工作空间运行推理的 IAM 委托人授予最低权限：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "aws-external-anthropic:CreateInference",
        "aws-external-anthropic:CountTokens",
        "aws-external-anthropic:Get*",
        "aws-external-anthropic:List*"
      ],
      "Resource": "arn:aws:aws-external-anthropic:us-west-2:123456789012:workspace/
wrkspc_01AbCdEf23GhIj"
    }
  ]
}
```

> **Note**
>
> 此政策中的List*通配符也匹配ListWorkspaces，这是账户范围的。工作空间 ARN 约束会将其静默过滤掉，因此此策略不授权列出工作空间。如果您的服务帐号需要枚举工作空间，请为 with 添加单独的Allow语句。ListWorkspaces Resource: "*"
> 此策略假设 AWS Sigv4 身份验证。如果委托人使用 API 密钥进行身份验证，则还要授权aws-external-anthropic:CallWithBearerToken（请参阅身份验证）。

示例：每个客户的工作空间隔离

将角色限制为单个工作区：

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "aws-external-anthropic:*",
      "Resource": "arn:aws:aws-external-anthropic:us-west-2:123456789012:workspace/
wrkspc_01AbCdEf23GhIj"
    },
    {
      "Effect": "Allow",
      "Action": [
        "aws-external-anthropic:CallWithBearerToken",
        "aws-external-anthropic:AssumeConsole"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 第一条语句中的`aws-external-anthropic:*`通配符包括账户范围的操作 (`CreateWorkspace`,`ListWorkspaces`)，工作空间 ARN 约束会静默过滤掉这些操作。这与“隔离”意图一致，即角色无法创建或枚举工作空间，但该策略包含的权限无效。有关账户范围的模式，请参阅配置自动化。
>
> 第二条语句`AssumeConsole`在所有资源上授予`CallWithBearerToken`和，因为两者都是不绑定到工作空间 ARN 的无路由操作。如果角色仅使用 SigV4 并且从未与 Claude 控制台联合，则省略第二条语句。

示例： ZDR-sensitive 工作空间的功能锁定

阻止特定工作空间上的批处理和文件上传，同时保留同步推理可用。当工作空间处理不得在服务器端保留的零数据保留 (ZDR) 数据时很有用。将此策略附加到“允许”策略旁边，例如上`AnthropicLimitedAccess`面的单一工作区示例；就其本身而言， Deny-only 策略不授予任何权限：

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

```
    {
      "Effect": "Deny",
      "Action": [
        "aws-external-anthropic:CreateBatchInference",
        "aws-external-anthropic:CreateFile"
      ],
      "Resource": "arn:aws:aws-external-anthropic:us-west-2:123456789012:workspace/
wrkspc_01AbCdEf23GhIj"
    }
  ]
}
```

> **Note**
>
> 这仅拒绝创建方块。除非您同时列出其他文件和批处理操作，否则它们不会被拒绝。要实现完全锁定，使工作区永远不能存放文件或批次，也可以拒绝aws-external-anthropic:GetFileaws-external-anthropic:ListFilesaws-external-anthropic:DeleteFile、aws-external-anthropic:GetBatchInference、aws-external-anthropic:ListBatchInferences、、aws-external-anthropic:CancelBatchInference、和aws-external-anthropic:DeleteBatchInference。

示例：配置自动化

向 CI/CD 角色授予创建和管理工作空间所需的操作，无需任何推理权限：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "aws-external-anthropic:CreateWorkspace",
        "aws-external-anthropic:GetWorkspace",
        "aws-external-anthropic:ListWorkspaces",
        "aws-external-anthropic:UpdateWorkspace",
        "aws-external-anthropic:ArchiveWorkspace"
      ],
      "Resource": "*"
    }
```

```
  ]
}
```

CreateWorkspace并且ListWorkspaces是账户范围内的操作。为这些操作指定工作空间 ARN 无效；使用。Resource: "*"

托管策略

AWS 为常见访问模式提供托管策略：

- **AnthropicFullAccess**: aws-external-anthropic:* 对所有资源的补助。
- **AnthropicReadOnlyAccess**: 补助金Get*List*、CallWithBearerToken和所有资源。
- **AnthropicInferenceAccess**：授予对所有资源的 ReadOnly 操作和推理操作（CreateInferenceCreateBatchInferenceCancelBatchInference、DeleteBatchInference
- **AnthropicLimitedAccess**：授予对所有AnthropicInferenceAccess资源的操作以及所有 Claude Managed Agents 操作（代理、会话、环境、文件库、内存存储）。
- **AnthropicSelfHostedEnvironmentAccess**：授予自托管沙盒工作人员轮询和处理环境工作项目、读取关联的环境、会话和技能资源以及更新会话状态所需的权限。将此策略附加到您的自托管环境工作人员担任的 IAM 角色。 AnthropicSelfHostedEnvironmentAccess是推荐的默认单角色；如果您在不同的 IAM 委托人下运行工作轮询器和每个会话的沙箱，则可以将这些权限分成两个范围较小的自定义策略，以获得最低权限。

AnthropicInferenceAccess是足以进行推理的最狭窄的托管策略。通过Get*和List*通配符，它授予对命名空间中每个 API 资源的读取权限，包括通过下载的文件内容GetFile和通过下载的内存内容。GetMemoryStore它不允许创建或删除文件或技能、用户配置文件管理、工作区变更或控制台联合。

> **Note**
>
> AnthropicReadOnlyAccess、AnthropicInferenceAccess、且AnthropicLimitedAccess不授予AssumeConsole。需要联合到 Claude Console 的委托人需要单独获得授权，aws-external-anthropic:AssumeConsole无论是通过策略AnthropicFullAccess还是自定义策略。参见联合到 Claude 控制台。

> **Note**
>
> `CreateInference`并且`CreateBatchInference`是单独的动作。否认一个并不能阻挡另一个。如果您打算阻止所有模型调用，请同时拒绝这两个调用。

联合到 Claude 控制台

`aws-external-anthropic:AssumeConsole`让 IAM 委托人联合进入 Anthropic-operated Claude 控制台。对于大多数操作，控制台内的访问权限仍受 IAM 控制，但是部分管理员操作（主要是没有相应 IAM API 的使用视图）受委托人联合功能的限制。

存在两种能力：

- **`developer`**— 允许 AWS 上的 Claude Platform 开发人员进行日常工作所需的操作：从控制台运行推理、读取工作空间数据、查看个人使用情况。
- **`admin`**— 此外还允许仅限管理员的控制台操作，包括账户范围内的使用情况视图和管理设置，这些设置不会通过 IAM 操作浮出水面。

使用`AssumeConsole`操作上的`aws-external-anthropic:Capability`条件键控制委托人可以请求哪些能力：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "aws-external-anthropic:AssumeConsole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws-external-anthropic:Capability": "admin"
        }
      }
    }
  ]
}
```

AnthropicFullAccess无能力限制AssumeConsole的授权。对于任何范围较窄的授权（仅限开发者的控制台访问权限或仅限管理员的访问权限），请附上以条件为范围的自定义策略，如上所AssumeConsoleaws-external-anthropic:Capability示。

使用不记名代币进行通话

aws-external-anthropic:CallWithBearerToken授权 API 密钥作为持有者令牌呈现时使用的 SigV4-free 请求路径。任何使用 API 密钥进行身份验证的委托人都需要在目标工作空间上执行此操作。无论您是直接使用ANTHROPIC_AWS_API_KEY还是设置Authorization: Bearer标题，这都适用。

除了推理操作（CreateInference、CreateBatchInference等）之外，API 密钥调用者也需要这样做。否则CallWithBearerToken，API 密钥请求会在到达推理授权检查之前被拒绝。Sigv4 呼叫者不需要此操作。

AnthropicReadOnlyAccess、AnthropicInferenceAccess、AnthropicLimitedAccess、Anthro部包含CallWithBearerToken。如果您为 API 密钥访问编写了自定义策略，请明确添加该策略。

出站 Web 联合身份验证（控制台访问所必需的）

Claude 控制台在 Anthropic 基础设施中运行，而不是 AWS。当 IAM 委托人调用时AssumeConsole，AWS STS 会发布一个限于 Anthropic 受众的网络身份令牌；然后 Anthropic 控制台接受该令牌并建立联合会话。为此，AWS 账户必须允许aws-external-anthropic受众进行出站 Web 联合身份验证。

除了aws-external-anthropic:AssumeConsole操作之外，调用的委托人还AssumeConsole需要以下 STS 权限：

- **sts:GetWebIdentityToken**— 允许发放 Anthropic 控制台消耗的 Web 身份令牌。
- **sts:TagGetWebIdentityToken**— 允许将会话标签附加到 Web 身份令牌。AWS 上的 Claude Platform 使用这些标签将委托人的能力和工作空间环境传达给控制台。

将两者都包含在控制台用户担任的任何角色的信任 inline/managed 策略中，或者包含在AssumeConsole直接调用的用户身份所附的策略中。 AnthropicFullAccess包括两个 STS 操作。在 SCP 或权限边界sts:*级别拒绝的环境必须明确允许这两个操作才能成功进行控制台联合。

AWS 上的 Claude 平台的 IAM 操作

IAM 操作参考，用于通过 AWS 策略控制 AWS 上的 Claude 平台的访问权限。

AWS 上的 Claude 平台使用 AWS IAM 进行访问控制。每个 API 路由都映射到`aws-external-anthropic`命名空间中的 IAM 操作。此页面列出了所有操作、每个操作授权的路由，以及适用于常见访问模式的托管策略。有关平台设置和身份验证的信息，请参阅 AWS 上的 Claude 平台是什么？ 。

服务详细信息

IAM 服务前缀	`aws-external-anthropic`
资源类型	`workspace`
工作空间 ARN	`arn:aws:aws-external-anthropic:{region}:{account-id}:workspace/{workspace-id}`

ARN 区域始终处于填充状态，并且与工作空间绑定到的区域相匹配。资源段是带标签的工作空间 ID (`wrkspc_…`)，与您在`anthropic-workspace-id`标题中传递的值相同。

操作

该服务在 15 个组中定义了 65 个操作。操作遵循 AWS `VerbNoun` 惯例，使用动词纪律，`Get*`这样`List*`通配符就会生成一个干净的只读边界。

推理

处理建议	已授权的路线
`CreateInference`	`POST /v1/messages`
`CountTokens`	`POST /v1/messages/count_tokens`

批处理

处理建议	已授权的路线
CreateBatchInference	POST /v1/messages/batches
GetBatchInference	GET /v1/messages/batches/{id} GET /v1/messages/batches/{id}/results
ListBatchInferences	GET /v1/messages/batches
CancelBatchInference	POST /v1/messages/batches/{id}/cancel
DeleteBatchInference	DELETE /v1/messages/batches/{id}

模型

处理建议	已授权的路线
GetModel	GET /v1/models/{id}
ListModels	GET /v1/models

文件

处理建议	已授权的路线
CreateFile	POST /v1/files
GetFile	GET /v1/files/{id} GET /v1/files/{id}/content
ListFiles	GET /v1/files
DeleteFile	DELETE /v1/files/{id}

> **Note**
>
> GetFile授权下载元数据和内容。具有只读访问权限的主体可以下载文件字节，而不仅仅是列出文件。

技能

处理建议	已授权的路线
CreateSkill	POST /v1/skills
GetSkill	GET /v1/skills/{id} GET /v1/skills/{id}/versions GET /v1/skills/{id}/versions/{version} GET /v1/skills/{id}/versions/{version}/content
ListSkills	GET /v1/skills
UpdateSkill	POST /v1/skills/{id}/versions DELETE /v1/skills/{id}/versions/{version}
DeleteSkill	DELETE /v1/skills/{id}

> **Note**
>
> 删除单个技能版本会映射到UpdateSkill，而不是DeleteSkill。拒绝的策略aws-external-anthropic:Delete*仍允许删除版本。拒绝 UpdateSkillCreateSkill，如果你需要防止任何技能变异，也可以拒绝。

用户配置文件

处理建议	已授权的路线
CreateUserProfile	POST /v1/user_profiles

处理建议	已授权的路线
GetUserProfile	GET /v1/user_profiles/{id}
ListUserProfiles	GET /v1/user_profiles
UpdateUserProfile	POST /v1/user_profiles/{id}

> **Warning**
>
> IAM 操作匹配不区分大小写。通配符aws-external-anthropic:*File匹配CreateFileGetFileDeleteFile、和，但不匹配ListFiles（以“文件”结尾，而不是“文件”）。它还会过度匹配CreateUserProfileGetUserProfile、和，UpdateUserProfile因为“配置文件”以“文件”结尾。如果您打算仅允许或拒绝 Files API 操作，请显式枚举它们（CreateFile、GetFile、ListFiles、DeleteFile），而不是使用*File后缀模式。

座席

Cla ude 托管代理的代理定义。

处理建议	已授权的路线
CreateAgent	代理创建路由
GetAgent	代理读取路由
ListAgents	代理列表路线
UpdateAgent	代理更新路线
ArchiveAgent	代理存档路由

会话

代理会话和事件历史记录。

处理建议	已授权的路线
CreateSession	会话创建路由
GetSession	会话读取路由
ListSessions	会话列表路由
UpdateSession	会话更新路线
ArchiveSession	会话存档路由
DeleteSession	会话删除路由

环境

会话的云容器配置。

处理建议	已授权的路线
CreateEnvironment	环境创建路线
GetEnvironment	环境读取路由
ListEnvironments	环境列出路由
UpdateEnvironment	环境更新路线
ArchiveEnvironment	环境存档路线
DeleteEnvironment	环境删除路由
ProcessEnvironmentWork	Self-hosted 环境工作者路线

保管库

用于会话身份验证的凭据库。

处理建议	已授权的路线
CreateVault	保管库创建路线
GetVault	保管库读取路由
ListVaults	保管库列表路线
UpdateVault	保管库更新路线
ArchiveVault	保管库存档路线
DeleteVault	保管库删除路线

> **Note**
>
> 文件库操作被归类为 CloudTrail 管理事件（而不是数据事件），因为文件库持有凭据并受益于默认开启的审核日志。其他 Claude 托管代理操作（代理、会话、环境、内存存储）是数据事件。

内存存储

永久代理内存。

处理建议	已授权的路线
CreateMemoryStore	存储器创建路由
GetMemoryStore	内存存储读取路由
ListMemoryStores	内存存储列表路由
UpdateMemoryStore	内存存储更新路线
ArchiveMemoryStore	内存存储存档路由
DeleteMemoryStore	存储器删除路由

> **Note**
>
> GetMemoryStore读取内存内容。因此，托管策略或自定义策略中的Get*通配符授予内存读取权限。如果必须限制内存内容，则明确规定策略的范围。

Webhook

会话的生命周期事件通知。

处理建议	已授权的路线
ListWebhooks	GET /v1/webhooks
GetWebhook	GET /v1/webhooks/{webhook_endpoint_id}
CreateWebhook	POST /v1/webhooks
UpdateWebhook	POST /v1/webhooks/{webhook_endpoint_id}
DeleteWebhook	DELETE /v1/webhooks/{webhook_endpoint_id}
RotateWebhookSecret	POST /v1/webhooks/{webhook_endpoint_id}/regenerate_signing_secret

Workspaces

处理建议	已授权的路线
CreateWorkspace	POST /v1/organizations/workspaces
GetWorkspace	GET /v1/organizations/workspaces/{id}
ListWorkspaces	GET /v1/organizations/workspaces
UpdateWorkspace	POST /v1/organizations/workspaces/{id}

处理建议	已授权的路线
ArchiveWorkspace	POST /v1/organizations/workspaces/{id}/archive

注册时会配置默认工作空间；请参阅工作空间。

身份验证

处理建议	已授权的路线
CallWithBearerToken	（无）

CallWithBearerToken是一种身份验证层权限，它授权委托人通过 API 密钥（持有者令牌）而不是 AWS Sigv4 进行身份验证。它不会映射到路线。将其与您希望 API 密钥持有者执行的路由映射操作一起授予。

控制台访问

处理建议	已授权的路线
AssumeConsole	（无）

AssumeConsole授权委托人通过 AWS 控制台联合流程在 AWS 工作空间上打开 Claude 平台的 Claude 控制台。它不会映射到路线。将其授予本应能够在 AWS 控制台的 AWS 服务页面上单击 Claude 平台上的 “打开 Claude 控制台” 的委托人。AssumeConsole操作上的aws-external-anthropic:Capability条件键控制委托人可以请求哪些控制台功能（开发者或管理员）。有关详细信息，请参阅 IAM 策略和使用 Claude 控制台了解登录流程。

> **Warning**
>
> aws-external-anthropic:AssumeConsole发出持续长达 12 小时的 Claude 控制台会话，与呼叫者自己的会话持续时间无关。IAM 会话短于 12 小时的呼叫者仍然可以获得超过其源证书寿命的控制台会话。将此权限限制为需要 Claude Console 访问权限的委托人，并在不再需要时立即撤销该权限。

> **Note**
>
> 联合后在 Claude 控制台内执行的操作不会记录在 AWS CloudTrail 中，也不会通过 IAM 进行归因。这包括工作空间范围内的全球活动，例如查看使用情况报告。如果您需要控制台活动的审计跟踪，请使用 Claude 控制台中提供的审计日志，而不 CloudTrail是。

Route-to-action 映射

下表列出了 AWS 上的 Claude 平台上的每条路由，以及调用该路由所需的 IAM 操作。稳定路由的 IAM 操作还会授权使用`anthropic-beta`标头的请求。 CloudTrail 将每个操作分类为数据事件（高容量、数据平面操作）或管理事件（控制平面操作）。

方法	路线	IAM 操作	CloudTrail 事件类型
`POST`	`/v1/messages`	`CreateInference`	数据
`POST`	`/v1/messages/count_tokens`	`CountTokens`	数据
`POST`	`/v1/messages/batches`	`CreateBatchInference`	数据
`GET`	`/v1/messages/batches`	`ListBatchInferences`	数据
`GET`	`/v1/messages/batches/{id}`	`GetBatchInference`	数据
`GET`	`/v1/messages/batches/{id}/results`	`GetBatchInference`	数据
`POST`	`/v1/messages/batches/{id}/cancel`	`CancelBatchInference`	数据
`DELETE`	`/v1/messages/batches/{id}`	`DeleteBatchInference`	数据
`GET`	`/v1/models`	`ListModels`	数据

方法	路线	IAM 操作	CloudTrail 事件类型
GET	/v1/models/{id}	GetModel	数据
POST	/v1/files	CreateFile	数据
GET	/v1/files	ListFiles	数据
GET	/v1/files/{id}	GetFile	数据
GET	/v1/files/{id}/content	GetFile	数据
DELETE	/v1/files/{id}	DeleteFile	数据
POST	/v1/skills	CreateSkill	数据
GET	/v1/skills	ListSkills	数据
GET	/v1/skills/{id}	GetSkill	数据
DELETE	/v1/skills/{id}	DeleteSkill	数据
POST	/v1/skills/{id}/versions	UpdateSkill	数据
GET	/v1/skills/{id}/versions	GetSkill	数据
GET	/v1/skills/{id}/versions/{version}	GetSkill	数据
GET	/v1/skills/{id}/versions/{version}/content	GetSkill	数据
DELETE	/v1/skills/{id}/versions/{version}	UpdateSkill	数据
POST	/v1/user_profiles	CreateUserProfile	数据
GET	/v1/user_profiles	ListUserProfiles	数据
GET	/v1/user_profiles/{id}	GetUserProfile	数据

方法	路线	IAM 操作	CloudTrail 事件类型
POST	/v1/user_profiles/{id}	UpdateUserProfile	数据
POST	/v1/organizations/workspaces	CreateWorkspace	管理
GET	/v1/organizations/workspaces	ListWorkspaces	管理
GET	/v1/organizations/workspaces/{id}	GetWorkspace	管理
POST	/v1/organizations/workspaces/{id}	UpdateWorkspace	管理
POST	/v1/organizations/workspaces/{id}/archive	ArchiveWorkspace	管理

Claude Managed Agents 路由（代理、会话、环境、文件库、内存存储）遵循相同的行动路径模式；有关每条路由的完整映射，请参阅 Anthropic IAM 操作参考。保管库路由是管理事件；代理、会话、环境和内存存储路由是数据事件。

默认情况下，网关会拒绝不在 AWS 的 Claude 平台上的路由。

另请参阅

- IAM 策略，例如策略和托管策略
- 有关 IAM 策略语法和评估逻辑的 AWS IAM 用户指南
- 用于审计日志配置的 AWS CloudTrail 用户指南

文档历史记录

下表描述了 AWS 上的 Claude 平台用户指南的重要更改。

更改	描述	日期
初次发布	在 AWS 上发布了 Claude 平台用户指南。	2026 年 5 月 7 日

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。