aws

ユーザーガイド

AWS での Claude プラットフォーム

AWS での Claude プラットフォーム: ユーザーガイド

Table of Contents

AWS での Claude プラットフォームとは

Anthropic が管理するインフラストラクチャを使用して、AWS を通じて Claude の完全なプラットフォーム機能にアクセスします。

Claude Platform on AWS では、Messages API、エージェントスキル、コード実行、ベータ機能など、Anthropic プラットフォームの完全なエクスペリエンスを AWS アカウントから利用できます。AWS が推論スタックを操作する Amazon Bedrock とは異なり、Anthropic は AWS で Claude Platform を操作します。AWS は、認証レイヤー (SigV4 または API キー) 、IAM ベースのアクセスコントロール、AWS Marketplace を介した請求統合を提供します。

> **Note**
>
> Anthropic SDKs、AWS での Claude プラットフォームをサポートしています。言語ごとのクライアントの可用性については、Anthropic Client SDKsドキュメントを参照してください。

プラットフォーム統合の仕組み

Claude モデルは Anthropic が管理するインフラストラクチャで実行されます。これは、AWS を介した請求とアクセスのための商用統合です。Anthropic と AWS はどちらも独立したデータ処理者として動作します。AWS のサービス条件は、AWS の Claude プラットフォームに適用されます。Anthropic の商用利用規約、データ処理補遺、および使用ポリシーも適用されます。

以下の動作特性に注意してください。データは AWS に存在しない場合があります。推論は Anthropic のプライマリクラウドにルーティングされる場合があります。サブサービスは予告なしに変更される場合があります。リクエストごとに `inference_geo`パラメータを設定して、推論を特定の地域に固定します。詳細については、「データレジデンシー」を参照してください。

AWS の Claude プラットフォームは、ファーストパーティ Claude API と同じデータ保持ポリシーに従います。ゼロデータ保持 (ZDR) はリクエストに応じて利用できます。Anthropic アカウント担当者に連絡して、アカウントで有効にしてください。

AWS での Claude プラットフォームの機能

AWS での Claude プラットフォームには、次の機能があります。

- 同日モデルと機能へのアクセス — 新しい Claude モデルと API 機能は、ファーストパーティ Claude API で起動したのと同じ日に利用できます。
- エージェントスキル — ドキュメント生成 (PowerPoint、Excel、Word、PDF) に構築済みのスキルを使用し、カスタムスキルを作成します。
- コード実行 — Anthropic のマネージドサンドボックスでコードを実行します。
- 拡張思考 — 複雑な推論タスクに対して拡張思考を有効にします。
- ストリーミングとバッチ処理 — リアルタイム SSE ストリーミングを使用するか、高スループットワークロードのバッチリクエストを送信します。
- プロンプトキャッシュ — キャッシュツール、システムプロンプト、メッセージ履歴を使用して、レイテンシーとコストを削減します。
- ファイル API — リクエスト間でファイルをアップロードして参照します。
- AWS IAM 統合 — 標準の IAM ポリシーと SigV4 認証を使用してアクセスを制御します。
- 統一された AWS 請求 — 消費ベースの計測 (Marketplace を請求バックエンド) で既存の AWS アカウントを通じて支払います。

サポートされている機能の完全なリストについては、「機能のサポート」を参照してください。

このガイドの整理方法

このガイドでは、以下のトピックについて説明します。

- 開始方法 — アカウントのセットアップ、前提条件、認証、SDK のインストール。
- API の使用 — 利用可能なモデル、リクエストの作成、機能サポート。
- 環境の管理 — データレジデンシー、ワークスペース、Claude コンソール、レート制限、請求。
- モニタリングとオペレーション — CloudTrail のログ記録とリクエスト ID の追跡。
- 移行 — Amazon Bedrock から AWS 上の Claude プラットフォームへの移行。
- セキュリティとアクセスコントロール — IAM ポリシーとアクションのリファレンス。

AWS での Claude プラットフォームと Amazon Bedrock の比較

どちらのサービスでも AWS を通じて Claude を使用できますが、アーキテクチャ、API サーフェス、機能の可用性が大きく異なります。

	AWS での Claude プラットフォーム	Amazon Bedrock
スタックを操作するユーザー	Anthropic	AWS
API サーフェス	Anthropic Messages API (`/v1/messages`)	Bedrock API (Converse/InvokeModel)
機能の可用性	Claude API と同じ日	AWS 統合タイムラインによって異なります
エージェントスキル	使用可能	使用不可 (コード実行が必要)
ベータ機能	`anthropic-beta` ヘッダーでパススルーする (「機能のサポート」を参照)	AWS サポートが必要
認証	AWS IAM / SigV4 または API キー	AWS IAM/SigV4 またはベアラートークン (C#、Go、Java SDKs のみ)
ベース URL	`aws-external-anthropic.{region}.api.aws`	`bedrock-runtime.{region}.amazonaws.com`
SDK クライアント	ベータ版`AnthropicAWS` のプラットフォーム固有のクライアントクラス (Python など)	`AnthropicBedrock` / Bedrock SDK
コンソール	Claude コンソール (`platform.claude.com` 、AWS コンソール経由でアクセス)	Bedrock コンソール

	AWS での Claude プラットフォーム	Amazon Bedrock
レート制限とクォータ	Anthropic による管理	AWS による管理
データ処理	Anthropic と AWS	AWS

Bedrock API で AWS が運営する Claude が必要な場合は、「Amazon Bedrock」を参照してください。

> **Important**
>
> Anthropic は、AWS での Claude プラットフォームをサードパーティーサービスとして提供しています。標準の AWS コンプライアンスプログラム、認定、または監査レポート (SOC、ISO、HIPAA 資格など) の対象ではありません。お客様は、このサードパーティーのサービスが規制、法律、コンプライアンスの要件を満たしていることを確認するために、独自のデューデリジェンスを実行する責任を単独で負います。

Bedrock を選択する場合: 組織が AWS が運営する推論、AWS を唯一のデータプロセッサ、または AWS コンプライアンスプログラム (FedRAMP High、IL4、IL5、SOC、ISO、HIPAA の適格性を含む) の対象となる必要がある場合は、Amazon Bedrock を選択します。Bedrock は AWS が管理するインフラストラクチャ上で完全に動作し、AWS が運用者として動作します。

アカウントの設定

AWS での Claude プラットフォームのセットアップには、AWS コンソールへのサインアップ、Anthropic 組織のセットアップ、ワークスペース ID のメモ、Claude コンソールへのサインインの 4 つのフェーズがあります。

> **Note**
>
> AWS コンソールからサインアップすると、AWS アカウントに関連付けられた新しい Anthropic 組織がプロビジョニングされます。この組織は、AWS Marketplace を通じて調達された Claude Enterprise 組織など、Anthropic の既存の組織とは別です。ファーストパーティの Anthropic 組織の API キー、ワークスペース、および Claude コンソール設定は引き継がれません。
>
> 既存の Amazon Bedrock プライベートオファーがある場合は、サインアップする前に Anthropic または AWS アカウントエグゼクティブに連絡して、最初のリクエストから割引を適用してください。割引は、プライベートオファーが受け入れられる前に発生した使用量に遡及して適用することはできません。「プライベートオファー」を参照してください。

ステップ 1: AWS コンソールにサインアップする

1. AWS コンソールを開き、AWS の Claude プラットフォームサービスページに移動します。
2. [サインアップ] を選択します。
3. [続行] をクリックしてください。

このページには、進行中のサインアップバナーが表示されます。ページにとどまります。AWS が AWS Marketplace サブスクリプションを処理してから自動的にリダイレクトされるまで、サインアップには数分かかります。

組織に Anthropic からのプライベートオファーがある場合、コンソールはそれを検索し、AWS Marketplace でそれを受け入れるように求めます。詳細については、「プライベートオファー」を参照してください。

> **Note**
>
> AWS で Claude Platform を使用する場合、コンテンツ (プロンプトや完了など) は AWS の外部で Anthropic によって処理されます。コンテンツとメタデータの処理と保存方法の詳細については、「Anthropic のデータ使用ポリシー」を参照してください。

ステップ 2: Anthropic 組織を設定する

サインアップが完了すると、 にリダイレクトされますplatform.claude.com/partner-signup。

1. 組織の所有者の E メールアドレスを入力し、開始方法を選択します。
2. E メールの受信トレイでセットアップリンクを確認し、それに従います。ブラウザに別のアカウントページとしてサインインが表示された場合は、ログアウトして続行を選択します。
3. 組織の詳細フォーム (組織名、エンティティタイプ、国、使用目的) に記入し、セットアップの完了を選択します。

セットアップを完了すると、Anthropic 組織が作成されます。AWS のサービス条件は、AWS での Claude プラットフォームに適用されます。Anthropic の商用利用規約、データ処理補遺、および使用ポリシーも適用されます。AWS コンソールサービスページに、ホーム、API キー、クイックスタート、ワークスペースを使用した左側のナビゲーションが表示されるようになりました。

ステップ 3: ワークスペース ID を書き留める

デフォルトのワークスペースは、サインアップ時に自動的にプロビジョニングされます。追加のワークスペースはリージョンごとに作成できます。リージョンバインディング、ワークスペース管理、IAM リソーススコープの詳細については、「Workspaces」を参照してください。

ワークスペース ID は、AWS サービスページの AWS コンソール Claude プラットフォームのワークスペースまたは Claude コンソールで検索します (「Claude コンソールの使用」を参照) 。WorkSpace IDs、 形式wrkspc_に続けて英数字識別子を使用します。

ステップ 4: Claude コンソールにサインインする

Claude コンソールへのアクセスは、AWS IAM を介してフェデレーションされます。

1. アクセスaws-external-anthropic:AssumeConsole許可を持つ IAM ロールを引き受けます。「IAM アクション」を参照してください。
2. AWS のサービスページの Claude プラットフォームから、サインインを選択します。AWS コンソールは JWT を発行し、 にリダイレクトしますplatform.claude.com。
3. 初回サインイン時に、E メールアドレスの入力を求められます。作業用 E メールを入力します。プラットフォームは Claude コンソールユーザーjust-in-timeプロビジョニングします。

AWS コンソールからサインインすると、Claude コンソールは AWS 上の Claude プラットフォーム組織にスコープします。AWS によって管理されるアカウントインジケータが Claude コンソールのサイドバーに表示されます。

アカウント設定のトラブルシューティング

- 「サインアップに失敗しました: OutboundWebIdentityFederation を有効にできませんでした」: 最初の送信時にこの赤いバナーが表示された場合は、もう一度続行を選択します。IAM 有効化の伝播には時間がかかる場合があります。
- サインアップ中の進行状況インジケータなし: サインアップには数分かかります。このページには、AWS がアカウントをプロビジョニングしている間、進行状況バーのない進行中の静的サインアップバナーが表示されます。
- セットアップリンクの後に「別のアカウントとしてサインインしました」: ログアウトを選択して続行します。このページでは、入力した E メールアドレスで再認証されます。
- サインイン中の「Not found」メッセージ: このメッセージはリダイレクト中に短時間表示される場合があります。却下できます。
- 使用状況ページには、最初の API 呼び出し後にデータが表示されません。使用状況データが Claude コンソールに表示されるまでに数分かかる場合があります。

前提条件

API コールを行う前に、以下を確認してください。

1. Claude Platform on AWS へのサブスクリプションを持つアクティブな AWS アカウント (「アカウントのセットアップ」を参照) 。
2. AWS CLI がインストールされ、設定されています。
3. AWS アカウントで有効になっているアウトバウンドウェブ ID フェデレーション (ワンタイムセットアップステップ。「アウトバウンドウェブ ID フェデレーションを有効にする」を参照) 。
4. ワークスペース ID (「ワークスペース ID を取得する」を参照) 。

アウトバウンドウェブ ID フェデレーションを有効にする

AWS ゲートウェイの Claude プラットフォームは`sts:GetWebIdentityToken`サーバー側を呼び出して、Anthropic に転送する JWT をミントします。この STS 機能は、すべての AWS アカウントでデフォルトで無効になっています。アカウントごとに 1 回有効にします。

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

レスポンスが の場合`[ERROR] (FeatureEnabled) … already enabled` 、アカウントの設定は既にオンになっているため、先に進むことができます。アカウントの発行者 URL を確認して取得します。

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

> **Warning**
>
> このステップがないと、すべてのリクエストは を返します`"Outbound web identity federation is disabled for your account"`。これは最も一般的なセットアップエラーです。

ワークスペース ID を取得する

デフォルトのワークスペースは、サインアップ時に各リージョンに自動的にプロビジョニングされます (「アカウントのセットアップ」を参照) 。WorkSpaces は 1 つの AWS リージョンにバインドさ

れます。ワークスペース ID は、Workspaces の Claude コンソール、または AWS コンソールサービスページの Workspaces セクションにあります。リージョンバインディングと IAM リソーススコープについては、「Workspaces」を参照してください。

SDK クライアントが自動的に読み取るように、 ANTHROPIC_AWS_WORKSPACE_IDおよび AWS_REGION環境変数を設定します。

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

リージョンは必須です。リージョンが設定されていない場合、SDK クライアントは をスローします。コンストラクタに aws_region/awsRegion を渡すか、 AWS_REGION (または) を設定しますAWS_DEFAULT_REGION。すべての AWS 商用リージョンがサポートされています。オプトインリージョンでは、まずアカウントをリージョンにオプトインする必要があります。

認証

AWS の Claude Platform は、SigV4 リクエスト署名 (プライマリ) を使用した AWS IAM と API キー認証の 2 つの認証方法をサポートしています。どちらも同じベース URL とリクエスト形式を使用します。

Anthropic SDKs「 SDK のインストール」を参照) は、以下で説明する認証フローと認証情報解決を実装します。Anthropic SDK を使用していない場合は、リージョンエンドポイント に対して SigV4-signedリクエストを構築 (または API キーをベアラートークンとして提示) する必要がありますhttps://aws-external-anthropic.<region>.api.aws。SDK 固有のクライアント設定と信頼できる認証情報解決の順序については、Anthropic ドキュメントの Claude on AWS セットアップガイドを参照してください。

SigV4 認証

SigV4 はエンタープライズネイティブパスであり、既存の AWS IAM ポリシー、ロール、監査と統合されます。AWS のデフォルトの認証情報プロバイダーチェーンでサポートされている任意の方法を使用して AWS 認証情報を設定します。

- 環境変数 (AWS_ACCESS_KEY_ID、AWS_SECRET_ACCESS_KEY、AWS_SESSION_TOKEN)
- 共有認証情報ファイル (~/.aws/credentials)
- SSO と を含む共有設定ファイル (~/.aws/config) credential_process
- IRSA および GitHub Actions のウェブ ID (AWS_WEB_IDENTITY_TOKEN_FILE および AWS_ROLE_ARN)
- ECS コンテナ認証情報
- EC2 インスタンスメタデータサービス (IMDS)

Anthropic SDK が認証情報を取得し、正しいリージョンエンドポイントに解決していることを確認するには、「リクエストの実行」の例に従ってテストリクエストを行います。レスポンスが成功すると、認証情報、リージョン、ベース URL、ワークスペース ID がすべて正しく設定されていることを確認します。

API キー認証

よりシンプルな統合パス (ローカル開発とスクリプト) では、SigV4 の代わりに API キーを使用して認証できます。ANTHROPIC_AWS_API_KEY 環境変数を設定するか、 SDK コ

ンストラクタapiKeyに渡します。Anthropic SDK はキーをベアラートークンとして AWS エンドポイントの Claude プラットフォームに送信します。IAM は aws-external-anthropic:CallWithBearerTokenアクションを通じてリクエストを承認します (IAM ポリシーを参照) 。

AWS コンソールの AWS → API キーの Claude Platform で API キーを生成します。キーの生成を選択し、キー値をコピーします。API aws-external-anthropic:CallWithBearerToken キー認証の使用を許可するプリンシパルに IAM アクションを付与します。

> **Note**
>
> AWS の Claude Platform で AWS コンソールで作成された API キーのみが、このサービスと連携します。
>
> - ファーストパーティー API アクセス用に標準の Claude コンソールで作成されたキーは、AWS の Claude プラットフォームエンドポイントでは機能しません。
> - Amazon Bedrock API キーも機能しません。Bedrock は別のエンドポイント、認証フロー、IAM 名前空間を使用します。

短期 API キー

API キー認証が必要なが、存続期間の長いキーの有効期間がない場合、Anthropic は AWS IAM 認証情報から短期 API キーを奪うトークン生成ライブラリを公開します。トークンのデフォルトの有効期間は 12 時間で、特定のワークスペースと aws-external-anthropic:CallWithBearerTokenアクションにスコープを設定できます。言語のジェネレーターをインストールし、IAM 認証情報を API キーと交換して、そのキーを Anthropic SDK に渡します。

- Python: aws/token-generator-for-aws-external-anthropic-python
- JavaScript / TypeScript: aws/token-generator-for-aws-external-anthropic-js
- Java: aws/token-generator-for-aws-external-anthropic-java

短期 API キーでは、引き続き呼び出し元の IAM プリンシパルがターゲットワークスペースaws-external-anthropic:CallWithBearerTokenに保持する必要があります。これらは単独で期限切れになるため、明示的にローテーションまたは取り消す必要はありません。

認証情報の優先順位とリージョン解決

Anthropic SDK の Claude Platform on AWS クライアントは、定義された優先順位を使用して認証情報とリージョンを解決します。引数名は各言語の規則に従います。TypeScript と PHP の場合は camelCase、Python と Ruby の場合は snake_case、Go の場合は PascalCase、C# と Java の場合はプロパティまたはビルダーパターンです。

各 SDK の権威優先順位、サポートされている環境変数、およびコンストラクタ引数については、Anthropic ドキュメントの「AWS での Claude のセットアップ」を参照してください。一般に、明示的なコンストラクタ引数は環境変数よりも優先され、デフォルトの AWS 認証情報プロバイダーチェーンよりもANTHROPIC_AWS_API_KEY優先されます。リージョンは必須です。AnthropicBedrock (にフォールバックするus-east-1) とは異なり、リージョンがコンストラクタまたは AWS_REGION / によって指定されていない場合、AWS クライアント上の Claude はスローしますAWS_DEFAULT_REGION。

ワークスペース ID

すべてのデータプレーンリクエストには、 anthropic-workspace-idヘッダーにワークスペース ID を含める必要があります。Anthropic SDKs、デフォルトでANTHROPIC_AWS_WORKSPACE_ID環境変数からこれを読み取るか、クライアントコンストラクタworkspace_idに workspaceId / を渡すことができます。ベースAnthropicクライアント (Claude-on-AWS クライアントではない) を使用する場合は、 ヘッダーを明示的に渡します。ワークスペース ID を見つける方法については、「言語ごとの例のリクエストを行う」および「Workspaces」を参照してください。

SDK をインストールします。

Anthropic のクライアント SDKsAWS での Claude プラットフォームをサポートしています。7 つの SDKs はすべてプラットフォーム固有のクライアントクラスを提供します。または、AWS の Claude Platform ベース URL を使用してベースクライアントを設定できます。

Python

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

> **Tip**
>
> Homebrew Python またはその他の外部管理の Python 環境を使用する macOS では、 が PEP 668 `externally-managed-environment` エラーで失敗`pip install`することがあります。最初に仮想環境を作成してアクティブ化します: `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 での Claude Platform の SDK クライアントはベータ版です。

利用可能なモデル

Claude Platform on AWS は、ファーストパーティ Claude API と同じ Claude モデルのセットを提供します。

モデル、モデル IDs、コンテキストウィンドウのサイズ、および機能の現在のリストについては、Anthropic ドキュメントの「モデルの概要」を参照してください。

モデル ID

AWS の Claude プラットフォームのモデル IDs は、ファーストパーティ Claude API で使用されるもの (、`claude-opus-4-6`、`claude-sonnet-4-6`、 など) と同じです`claude-haiku-4-5`。Bedrock スタイルの ARNs や`anthropic.`プレフィックスはありません。Anthropic が公開するのとまったく同じモデル ID を使用します。

リクエストの実行

AWS の Claude Platform は、Claude ファーストパーティープラットフォームと同じ API サーフェスである Anthropic Messages API (`/v1/messages`) を使用します。違いは、ベース URL、認証方法、およびリクエストがターゲットとするワークスペースを識別する必須`anthropic-workspace-id`ヘッダーです。

シェル

```
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
```

クライアントは環境ANTHROPIC_AWS_WORKSPACE_IDから AWS_REGION (または AWS_DEFAULT_REGION) と を読み取ります。/ awsRegionまたは aws_region / workspace_id をコンストラクタworkspaceIdに渡すことで上書きできます。リージョン ID とワークスペース ID の両方が必要です。どちらもこれらのソースから解決できない場合、コンストラクタはスローします。

> **Note**
>
> x-amz-security-token ヘッダー (curl) は、IAM ロール、SSO、STS などの一時的な認証情報にのみ必要です。長期 IAM ユーザー認証情報を使用する場合は省略します。SDK クライアントは、認証情報ソースに基づいてこれを自動的に処理します。

--aws-sigv4 値は の形式に従いますaws:amz:<region>:<service>。SigV4 サービス名はでaws-external-anthropic、リージョンはエンドポイント URL のリージョンと一致する必要があります。いずれかの の不一致により、特定の診断ではなく一般的な署名拒否エラーが発生します。

基本 Anthropic クライアントの使用

AWS SDK の依存関係を追加しない場合は、 ベースAnthropicクライアントを使用できます。このパスは、専用クライアントが読み取る名前ではなく、バニラ SDK 環境変数ANTHROPIC_AWS_*名を使用します。

```
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、Go SDKs読み取りANTHROPIC_API_KEYとANTHROPIC_BASE_URL自動で、他の言語の場合はコンストラクタに渡します。すべての言語で、 anthropic-workspace-idヘッダーにワークスペース ID を渡します。

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);
```

機能のサポート

AWS の Claude プラットフォームは Anthropic Messages API を直接使用します。ファーストパーティ Claude API との完全な機能パリティが得られます。ただし、「現在利用できない機能」に記載されている場合を除きます。

- ベータ機能: Claude API と同様に、標準`anthropic-beta`ヘッダーを渡してベータ機能にアクセスします。
- エージェントスキル: Claude API と同じ`container.skills`パラメータとベータヘッダーを持つ構築済みおよびカスタムエージェントスキルを使用します。構築済みのスキル (PowerPoint、Excel、Word、PDF) はすべてすぐに使用できます。
- コード実行: コード実行ツールを使用して、Anthropic のマネージドサンドボックスでコードを実行します。
- ツールの使用: コンピュータの使用およびその他のすべてのツールの使用機能を利用できます。
- 拡張思考: Claude API と同じパラメータで拡張思考を有効にします。
- ストリーミング: リアルタイムレスポンスの完全な SSE ストリーミングサポート。
- バッチ処理: 高スループットワークロードのバッチリクエストを送信します。
- プロンプトキャッシュ: キャッシュツール、システムプロンプト、メッセージ履歴を使用して、レイテンシーとコストを削減します。すべてのプロンプトキャッシュ機能 (5 分 TTL、1 時間 TTL、自動キャッシュ) を使用できます。
- ファイル API: リクエスト間でファイルをアップロードして参照します。
- IAM 統合による Workspace タグ: WorkSpaces にタグを付けることができ、タグは属性ベースのアクセスコントロールのために IAM に、コスト配分のために AWS コストと使用状況レポートに流れます。Workspace タグ付けは、AWS 上の Claude Platform の GA 機能です (ファーストパーティ Claude API のベータ機能です) 。

Claude Managed Agents

Claude マネージドエージェントは、AWS の Claude プラットフォームで使用できます。エージェント、セッション、環境、認証情報ボールト、メモリストアは、ファーストクラスの IAM リソースです。アクションリファレンスについては「IAM アクション」、対応するページについては「Claude コンソールの使用」を参照してください。Anthropic Agent SDK は、追加の統合なしで AWS の Claude プラットフォームと連携します。AWS ベース URL の Claude プラットフォームにポイントし、SigV4 または API キーで認証します。

AWS 上の Claude プラットフォームの自律セッションでは、6 時間ごとに再認証が必要です。長時間実行されるエージェントの実行は、そのウィンドウ内で SigV4 認証情報または API キーを更新する必要があります。更新しないと、セッションは終了します。

次の Claude マネージドエージェント機能は、現在 AWS の Claude プラットフォームでは使用できません。

- 結果: エージェントセッションの結果追跡は使用できません。
- マルチエージェントセッション: 複数のインタラクションエージェントを含むセッションは使用できません。

コンプライアンス

> **Important**
>
> Anthropic は、このサービスをサードパーティーサービスとして提供しています。標準の AWS コンプライアンスプログラム、認定、または監査レポート (SOC、ISO、HIPAA 適格性など) の対象ではありません。お客様は、このサードパーティーのサービスが規制、法律、コンプライアンスの要件を満たしていることを確認するために、独自のデューデリジェンスを実行する責任を単独で負います。機密データまたは規制対象データを処理する前に、Anthropic Trust Center を通じて Anthropic の公式コンプライアンスドキュメントを確認する必要があります。詳細については、AWS サービス条件を参照してください。

ワークスペースメンバーシップのモデル化方法

ファーストパーティ Claude API では、アクセスはusers-to-workspacesメンバーシップによって管理されます。ユーザーはワークスペースに追加され、ワークスペースのアクセス権を継承します。Claude Platform on AWS は、そのモデルを IAM 認可に置き換えます。ワークスペースごとのユーザーリストはありません。代わりに、IAM プリンシパル (ユーザーまたはロール) は、特定のワークスペース ARN に対するaws-external-anthropicアクションを付与するポリシーを保持します。 ARNs IAM ポリシー評価は、各プリンシパルが各ワークスペースで何ができるかを決定します。

Claude コンソールでワークスペースごとのメンバーシップを管理するのではなく、IAM ポリシー (SCPs、アクセス許可の境界、アイデンティティにアタッチされたポリシー、またはリソースレ

ベルの条件) を変更して、ワークスペースへのアクセスを許可および取り消します。例については、「IAM ポリシー」を参照してください。

現在利用できない機能

以下の機能は、現在 AWS の Claude プラットフォームでは使用できません。

- HIPAA の準備状況: Anthropic の HIPAA 対応プログラムは、AWS の Claude Platform では使用できません。HIPAA 要件があるお客様は、代わりに Amazon Bedrock で Claude を評価する必要があります。
- 標準およびバッチを超えるサービス階層: Priority Tier は使用できません。
- 管理者 API — 組織メンバー、ワークスペースメンバー、招待、API キー、使用状況レポート、コストレポート、レート制限レポートエンドポイント: これらの管理者 API エンドポイントは現在利用できません。Workspace CRUD と名前変更エンドポイント (`/v1/organizations/workspaces`) は、 `aws-external-anthropic`名前空間から使用できます。「IAM アクション」を参照してください。メンバーシップは、ワークスペースメンバーシップリストではなく AWS IAM を使用してモデル化されます (前のセクションを参照) 。API キーのライフサイクルは、AWS コンソールの Claude Platform on AWS → API キーで管理されます。使用量、コスト、レート制限データについては、Claude コンソールビュー (「モニタリングと口グ記録」を参照) または Anthropic Usage and Cost API を使用します。
- 使用制限: 利用できません。代わりに AWS の請求コントロールに依存します。
- Claude Code ワークスペースと Analytics API: 自動レート制限がある Claude Code ワークスペースは使用できません。Claude Code の使用状況は、専用画面ではなく一般的な使用状況ビューに表示されます。
- OAuth 認証: サポートされていません。SigV4 または API キー認証を使用します。
- OpenAI 互換 API エンドポイント: AWS の Claude プラットフォームでは使用できません。
- ワークスペースレベルの推論地理コントロール: `allowed_inference_geos`および `default_inference_geo`は使用できません。代わりに、各リクエスト`inference_geo`で を設定します。

データレジデンシー

AWS での Claude プラットフォームは、次の推論地域をサポートしています。

- 米国: 推論は米国のデータセンター内にとどまります。1.1 倍の料金乗数が適用されます。
- グローバル: 推論は、世界中の Anthropic が運営するデータセンターにルーティングされる場合があります。標準料金が適用されます。

`inference_geo` パラメータを使用して、リクエストごとに推論ジオグラフィを設定します。

> **Note**
>
> `inference_geo` パラメータは、Claude Opus 4.6、Claude Sonnet 4.6 以降のモデルでサポートされています。Claude Opus 4.5、Claude Sonnet 4.5、または Claude Haiku 4.5 `inference_geo`での リクエストは、400 エラーを返します。モデルの可用性の詳細については、「データレジデンシー」を参照してください。

シェル

```
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

```
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

```
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
```

を省略するとinference_geo、リクエストはデフォルトで になりますglobal。

ワークスペースレベルの推論地理コントロール (allowed_inference_geos および default_inference_geo) は、AWS の Claude プラットフォームでは使用できません。代わりに、各リクエストinference_geoで を設定します。

WorkSpaces

AWS の Claude プラットフォームでの推論リクエストとリソースリクエストは、ワークスペースをターゲットにします。これらの API コールの anthropic-workspace-idヘッダーでワークスペースの ID を渡します。WorkSpace IDsは、タグ付けされた形式wrkspc_に続けて英数字識別子 (例:) を使用しますwrkspc_01AbCdEf23GhIj。ワークスペース ID をまだお持ちでない場合は、「ワークスペース ID を取得する」を参照してください。

WorkSpace スコープ

WorkSpaces は 1 つの AWS リージョンにバインドされます。で作成されたワークスペースには、us-west-2エンドポイントを介してus-west-2のみアクセスできます。使用量、クォータ、コスト、ファイル、バッチ、スキルはすべてワークスペースごとにロールアップされるため、Claude コンソールでリージョンごとの内訳を確認できます。

WorkSpaces は、AWS 上の Claude プラットフォームのプライマリ IAM リソースとしても機能します。ワークスペース ARN を使用して、AWS IAM ポリシーを通じて特定のワークスペースへのアクセスを許可または拒否します。ARN のリソースセグメントは、anthropic-workspace-idヘッダーで渡す wrkspc_プレフィックス付き ID と同じです。

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

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

ポリシーの例については、「IAM ポリシー」を参照してください。

ワークスペースの作成

デフォルトのワークスペースはサインアップ時に自動的にプロビジョニングされます。追加のワークスペースは、対応するaws-external-anthropicアクション (、CreateWorkspace、ArchiveWorkspace) によって承認された/v1/organizations/workspacesエンドポイントを介して作成、更新、名前変更UpdateWorkspace、アーカイブできます。「IAM アクション」を参照してください。

クライアントでのワークスペース ID の設定

すべてのデータプレーンリクエストには、 anthropic-workspace-idヘッダーにワークスペース ID を含める必要があります。Anthropic SDK の Claude-on-AWS クライアントを使用する場合は、次の 3 つの方法があります。

1. 環境変数 — を設定するとANTHROPIC_AWS_WORKSPACE_ID、クライアントは自動的に読み取ります。

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

2. コンストラクタ引数 — クライアントのインスタンス化時に pass workspaceId / workspace_id (名前は SDK の言語規則に従います) 。
3. リクエストごとの上書き — 同じクライアントから複数のワークスペースに対処する必要がある場合は、個々のリクエストで ヘッダーを直接渡します。

ベースAnthropicクライアント (AWS バックエンドなし) を使用する場合は、すべてのリクエストで anthropic-workspace-idヘッダーを明示的に設定します。「言語ごとのリクエストの実行例」を参照してください。SDK 固有の引数名については、Anthropic Client SDKsを参照してください。

ワークスペースのタグ付け

ワークスペースタグは、ワークスペースにアタッチされたキーと値のペアです。属性ベースのアクセスコントロールのために AWS IAM と統合され、コスト配分のために AWS コストと使用状況レポートと統合されます (CUR の詳細については「モニタリングとログ記録」を参照) 。タグ付けは AWS の Claude プラットフォームでは GA です。

既存のワークスペースのタグをaws-external-anthropic名前空間の TagResourceと UntagResource で更新します。同じタグキーを使用すると、追加の設定なしで IAM 条件とコスト配分を駆動できます。

タグベースのアクセス制御

aws:ResourceTag/<key> 条件コンテキストキーを使用して、ワークスペースタグ値に対するaws-external-anthropicアクションをゲートできます。次のポリシーでは、 にタグ付けされたワークスペースでのみ推論を許可します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"
        }
      }
    }
  ]
}
```

で aws:RequestTag/<key>および aws:TagKeys条件キーTagResourceを使用して、タグ付けポリシーを適用します (ワークスペースに CostCenterおよび Owner タグを渡すように要求するなど)。その他の例については、「IAM ポリシー」を参照してください。

Claude コンソールの使用

AWS の Claude プラットフォームは、platform.claude.com で標準の Claude コンソールを使用します。AWS コンソールからサインインすると、AWS によって管理されるアカウントインジケータが Claude コンソールのサイドバーに表示されます。コンソールは、AWS 上の Claude プラットフォーム組織にスコープします。使用状況分析、コスト内訳、レート制限の可視性、ワークスペースの可視性、ファイルを管理するためのページ、エージェントスキル、バッチジョブ、エージェント、セッション、環境、認証情報ボールト、メモリストアを提供します。

サインイン

Claude コンソールへのアクセスは、AWS IAM を介してフェデレーションされます。完全な初回サインインフローについては、「アカウントのセットアップ」を参照してください。つまり、次のようになります。

1. アクセスaws-external-anthropic:AssumeConsole許可を持つ IAM ロールを引き受けます。「IAM アクション」を参照してください。
2. AWS コンソールの AWS の Claude プラットフォームページに移動します。
3. Open Claude Console を選択します。AWS コンソールは JWT を発行し、 にリダイレクトしますplatform.claude.com。
4. 最初のサインイン時に、E メールアドレスの入力を求められます。作業用 E メールを入力します。プラットフォームは Claude コンソールユーザーをjust-in-timeプロビジョニングします。

管理者と開発者の 2 つの Claude コンソールロールを使用できます。管理者ロールは、AWS 上の Claude プラットフォームで使用できるすべての Claude コンソールページと設定へのアクセスを許可します。開発者ロールは、使用量、コスト、レート制限、ワークスペース情報への読み取りアクセスを許可します。Anthropic アカウントの担当者に連絡して、管理者または開発者のロールをプリンシパルに割り当てます。

利用可能なページ

AWS ゲートウェイ経由の列は、ページが AWS ゲートウェイを介してデータの読み取りと書き込みを行うかどうかを示します (したがって、IAM アクションによって管理されます)。「No read organization-level metadata directly through Anthropic APIs and bypass IAM action checks」とマークされたページ。

ページ	使用可能	AWS ゲートウェイ経由	注意事項
使用方法	はい	いいえ	モデル、ワークスペース、ディメンションごとにトークンの使用状況を表示します。リクエスト後、データが表示されるまでに数分かかる場合があります。
コスト	はい	いいえ	モデルとワークスペース別のコスト内訳を表示します。AWS Cost Explorer は、集計された CCU 明細項目を表示します。
制限	はい	いいえ	レート制限を表示する (読み取り専用) 。
WorkSpaces	はい	いいえ	リージョンごとのワークスペース (読み取り専用) を表示します。
ファイル	はい	はい	アップロードされたファイルを表示および管理します。
スキル	はい	はい	エージェントスキルを表示および管理します。
バッチ	はい	はい	バッチ処理ジョブを表示および管理します。
[エージェント]	はい	はい	エージェント定義を表示および管理します。
セッション	はい	はい	エージェントセッションとイベント履歴を表示します。
環境	はい	はい	セッションのクラウドコンテナ設定を表示および管理します。
認証情報ボールト	はい	はい	セッション認証の認証情報ボールトを表示および管理します。
メモリストア	はい	はい	永続エージェントメモリを表示および管理します。

ページ	使用可能	AWS ゲートウェイ経由	注意事項
API キー	いいえ	該当なし	AWS コンソールで API キーを管理します (Claude Platform on AWS → API キー)。「認証」を参照してください。
メンバー	いいえ	該当なし	該当なし。AWS IAM はアクセスを管理します。
「請求」	いいえ	該当なし	該当なし。AWS Marketplace は請求と請求を管理します。Cost ページでコストの内訳を表示します。
Claude コード	いいえ	該当なし	使用状況ページで Claude コードの使用状況を表示します。

組織の切り替え

Claude コンソールは、AWS での Claude プラットフォームの組織切り替えをサポートしていません。別の組織にアクセスするには、サインアウトして、その組織の AWS アカウントの IAM ロールを使用して AWS コンソールから再認証します。

レート制限とクォータ

Claude Platform on AWS は、サブスクライブ時に階層 1 のレート制限を割り当てます。Anthropic は、AWS クォータシステムではなく、レート制限を直接管理します。

デフォルトの制限

AWS の Claude Platform は、ファーストパーティ Claude API と同じ Anthropic の標準階層スケジュールを使用します。ワークスペースごとに階層 1 の制限が適用されます。制限はモデルファミリーごとにプールされます。たとえば、すべての Opus モデルは 1 つの結合制限を共有し、すべての Sonnet モデルは別のモデルを共有し、すべての Haiku モデルは 3 番目のモデルを共有します。

現在の階層 1 の値 (RPM、ITPM、OTPM) および上位階層のしきい値については、Anthropic ドキュメントウェブサイトの「レート制限」を参照してください。Anthropic ページは信頼できる情報源であり、制限が変更されると更新されます。

レート制限ヘッダー

すべてのレスポンスには、現在のレート制限ステータスを報告するヘッダーが含まれます。キーヘッダー:

- `anthropic-ratelimit-requests-limit` — 1 分あたりの最大リクエスト数
- `anthropic-ratelimit-requests-remaining` — 現在のウィンドウに残っているリクエスト
- `anthropic-ratelimit-requests-reset` — リクエスト制限がリセットされた時刻 (RFC 3339)
- `anthropic-ratelimit-tokens-limit` — 1 分あたりの合計トークン (入力 + 出力) の最大数
- `anthropic-ratelimit-tokens-remaining` — 現在のウィンドウに残っているトークンの組み合わせ
- `anthropic-ratelimit-tokens-reset` — 結合トークン制限がリセットされる時間 (RFC 3339)
- `anthropic-ratelimit-input-tokens-limit` / `-remaining` / `-reset` — Input-token-specificヘッダー
- `anthropic-ratelimit-output-tokens-limit` /`-remaining`/ `-reset` — Output-token-specificヘッダー
- `retry-after` — 429 レスポンスで、再試行するまで待機する秒数

完全なセットについては、Anthropic ドキュメントウェブサイトの「レスポンスヘッダー」を参照してください。

制限の引き上げをリクエストする

ファーストパーティ Claude API とは異なり、階層の自動昇格は AWS の Claude プラットフォームには適用されません。制限の引き上げをリクエストするには、ワークスペース ID と希望するスループットで Anthropic アカウント担当者にお問い合わせください。階層のしきい値およびその他の詳細については、Anthropic ドキュメントウェブサイトの「レート制限」を参照してください。

レート制限エラー

レート制限を超えると、API は `rate_limit_error`型の HTTP 429 を返します。再試行ロジックにジッターを含むエクスポネンシャルバックオフを実装します。`retry-after` ヘッダーは、再試行するまでの待機秒数を示します。

料金

AWS の Claude Platform は、消費ベースの計測の請求バックエンドとして AWS Marketplace を使用します。Anthropic は AWS Marketplace を通じて 1 時間あたりの使用量を計測し、AWS は既存の AWS 請求書に対して毎月請求書を発行します。

請求は遅延のみ (使用した分に対して支払います) および税引き前 (AWS はアカウントの税設定を適用します) です。

使用量は、CCUs) で表されます。CCU 料金は固定であり、割引されることはありません。Anthropic は、トークンの使用量をモデルごとの標準レート、機能ごとのレートで USD で評価し、交渉された割引を適用し、その結果を CCUs あたり 0.01 USD の CCU に変換します。割引により、計測される CCUs が少なくなり、CCU 料金が低くなります。CCUs は前払いクレジットではなく、CCU の残高やコミットメントはありません。CCU 定義とモデルごとのトークンレートについては、「 料金表」を参照してください。

モニタリングとログ記録

AWS CloudTrail は、AWS 上の Claude プラットフォームへのすべてのリクエストをキャプチャできます。ワークスペースおよびボールトオペレーションは、デフォルトで管理イベントとして記録されます。他のすべてのオペレーション (推論、バッチ、ファイル、スキル、モデル、ユーザープロファイル、およびボールトを除く Claude マネージドエージェント) はデータイベントです。ログ記録には明示的な設定が必要であり、追加の CloudTrail 料金が発生します。完全なイベントタイプの分類については IAM アクション、設定の詳細については AWS CloudTrail ドキュメントを参照してください。

CloudTrail データイベントログ記録を設定するときは、リソースタイプ を選択しますAWS::AWSExternalAnthropic::Workspace。これは AWS ワークスペース上の Claude プラットフォームの CloudTrail リソースタイプであり、データプレーンイベント (推論、バッチ、ファイル、およびその他のワークスペースごとのオペレーション) をキャプチャするために必要です。アカウント全体ではなくワークスペースのサブセットのアクティビティをログに記録する場合は、データイベントセレクタを特定のワークスペース ARNs にスコープします。

リクエスト IDs

各レスポンスには、レスポンスヘッダーに 2 つのリクエスト IDs が含まれます。

- AWS リクエスト ID (**`x-amzn-requestid`**): CloudTrail でインデックス付けされたプライマリ ID。これは、AWS ツールを介してリクエストを調査する場合や、AWS サポートに問い合わせる場合に使用します。
- Anthropic リクエスト ID (**`request-id`**): セカンダリ ID。Anthropic サポートに連絡するときに使用します。

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 は現在 raw-response アクセサーを公開していないため、Ruby のレスポンスヘッダーからリクエスト IDs を読み取ることはできません。request-ID ログ記録が必要な場合は、上記の他の言語を使用するか、HTTP プロキシレイヤーで IDs をキャプチャします。

Anthropic では、使用パターンを理解し、潜在的な問題を調査するために、少なくとも 30 日間のローリングベースでアクティビティを記録することを推奨しています。

> **Note**
>
> AWS CloudTrail は AWS アカウント内で設定されます。ログ記録を有効にしても、AWS または Anthropic は請求およびサービスオペレーションに必要な範囲を超えてコンテンツにアクセスすることはできません。

使用状況メトリクスとダッシュボード

Claude Platform on AWS は、ワークスペースごとの使用状況メトリクスを Amazon CloudWatch に発行しません。トークン数、リクエスト量、モデルごとの使用量は、代わりに Anthropic の使用表面から使用できます。

- Claude コンソールの使用状況ビュー — プリンシパルが を使用して Claude コンソールにフェデレーションする場合 `aws-external-anthropic:AssumeConsole` (IAM ポリシーを参照) 、使用状況ダッシュボードには、アクセス可能なワークスペースのリクエストボリューム、トークン数、モデルごとの内訳が表示されます。アカウント全体の使用状況ビューには、管理者コンソール機能が必要です。
- Anthropic Usage and Cost API — ワークスペースごとやモデルごとの集計など、使用状況とコストデータへのプログラムによるアクセスについては、Anthropic ドキュメントの「Usage and Cost API」を参照してください。

運用モニタリング (エラー率、レイテンシー、レート制限ヘッダー) には、CloudTrail でキャプチャされた AWS リクエスト ID とともに、すべてのリクエストで返されるレスポンスヘッダーを使用します (「レート制限とクォータ」を参照) 。 IDs

コスト配分とモニタリング

AWS での Claude Platform の料金は、モデルごとの使用ディメンションとともに、AWS での Claude Platform on AWS サービスの AWS 請求書に表示されます。AWS コストと使用状況レポート (CUR) をワークスペースタグと組み合わせて使用して、きめ細かなコスト配分を行います。

1. ワークスペースに、関心のある割り当てディメンション (チーム、アプリケーション、環境、コストセンター) をタグ付けします。タグ付けの詳細については、「WorkSpaces」を参照してください。
2. AWS Billing コンソールで、各タグキーをコスト配分タグとしてアクティブ化します。アクティベーション後、アクティベーション日以降に発生した使用量は、CUR のタグで分割されます。

3. CUR または AWS Cost Explorer では、 `resourceTags/user:<tag-key>`列でグループ化して、ワークスペースタグ別に支出を分類します。

WorkSpace タグは、AWS でのすべての Claude プラットフォームの CUR 明細項目に流れます。同じタグキーは、IAM ベースのアクセスコントロールとコスト配分の両方を促進します。セットアップ手順とアクティベーションの遅延については、AWS Billing ドキュメントの「コスト配分タグ」を参照してください。

Amazon Bedrock からの移行

現在 Bedrock で Claude を使用している場合、AWS で Claude プラットフォームに移行するには、統合全体で変更が必要です。SigV4 署名は引き続きサポートされますが、署名コンテキスト、ベース URL、API 形式、モデル IDs、SDK クライアントとパッケージ、ストリーミング形式、リクエストヘッダー、リージョンの可用性はすべて変更されます。次の表は、違いをまとめたものです。

変更点

	Amazon Bedrock	AWS での Claude プラットフォーム
ベース URL	`bedrock-runtime.{region}.amazonaws.com`	`aws-external-anthropic.{region}.api.aws`
API 形式	Bedrock Converse/InvokeModel	Anthropic Messages API (`/v1/messages`)
モデル IDs	`anthropic.claude-opus-4-7-v1`	`claude-opus-4-7`
SDK クライアント	`AnthropicBedrock` / Bedrock SDK	ベータ版のプラットフォーム固有のクライアント (`AnthropicAWS` 、`AnthropicAwsClient` 、`AnthropicAws` など)
SDK パッケージ	`anthropic[bedrock]` 、`@anthropic-ai/bedrock-sdk`など。	`anthropic[aws]` 、`@anthropic-ai/aws-sdk` など (「SDK のインストール」を参照)
SigV4 サービス名	`bedrock`	`aws-external-anthropic`
ストリーミング形式	AWS EventStream	SSE (Claude API と同じ)
WorkSpace ヘッダー	該当しない	`anthropic-workspace-id` が必須

	Amazon Bedrock	AWS での Claude プラットフォーム
利用可能なリージョン	複数の商用リージョン	すべての AWS 商用リージョン (オプトインリージョンにはアカウントオプトインが必要です)

得られるもの

- 通常、別のパートナー統合ステップなしで、新しいモデルと機能に同日アクセスする
- ドキュメント生成のためのエージェントスキル (PowerPoint、Excel、Word、PDF)
- Anthropic のマネージドサンドボックスでのコード実行
- `anthropic-beta` ヘッダーを介したベータ機能 (「現在利用できない機能」を参照)
- クォータの可視性と使用状況分析のための Claude コンソール
- 直接 Anthropic サポート
- SigV4 に代わる API キー認証 (「認証」を参照)

変わらないもの

- AWS IAM 認証 (SigV4)
- AWS アカウントを介した請求
- AWS コミットメントの廃止

移行の落とし穴

> **Warning**
>
> 最初にアウトバウンドウェブ ID フェデレーションを有効にします。AWS アカウントで AWS で Claude プラットフォームを以前に使用したことがない場合は、リクエストを行う前に、アカウントごとに 1 回アウトバウンドウェブ ID フェデレーションを有効にする必要があります。このステップがないと、すべてのリクエストはフェデレーションエラーで失敗します。このステップは Bedrock には必要ありません。

> **Warning**
>
> ゼロデータ保持 (ZDR) は、AWS の Claude プラットフォームでオプトインされています。Bedrock では、AWS はデータプロセッサであり、Anthropic は推論の入力または出力を保持しません。Anthropic の ZDR プログラムは適用されません。AWS の Claude Platform では、Anthropic がデータ処理者であり、ZDR はファーストパーティの Claude API モデルに従います。これは、Anthropic アカウント担当者を通じてリクエストに応じて利用できます。データ保持の保証に依存する本番稼働用ワークロードを移行する前に、ZDR 登録を確認します。

商業上の考慮事項

- サービス条件: AWS のサービス条件は、AWS の Claude プラットフォームに適用されます。Anthropic の商用利用規約、データ処理補遺、および使用ポリシーも適用されます。
- 割引とプライベートオファー: 交渉済みの割引と AWS Marketplace プライベートオファーは、AWS の Bedrock プラットフォームと Claude プラットフォーム間で自動的に転送されません。Anthropic アカウント担当者と協力して、AWS で Claude Platform の商用条件を設定します。

IAM ポリシー

Claude Platform on AWS は、アクセスコントロールのために AWS IAM と統合されています。標準の IAM ポリシー構文を使用して、特定のワークスペースで特定の API アクションへのアクセスを許可または拒否します。

SigV4 サービス名と IAM アクション名前空間は ですaws-external-anthropic。アクションはパターンに従います aws-external-anthropic:<Action> (例: aws-external-anthropic:CreateInference)。

例: バッチ推論を拒否する

次のポリシーでは、ZDR の影響を受けやすいワークロードの一般的な要件であるバッチ処理をブロックしながら、リアルタイムの推論を許可します。

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

```
}
```

GetBatchInference アクションは、バッチメタデータルートとバッチ結果ルートの両方を承認します。とともに拒否すると、読み取りとバッチ列挙の両方がListBatchInferencesブロックされます。

Allow ステートメントは、ワイルドカードを使用するのではなく、特定の Get* および List*アクションを列挙します。ワイルドカードはGetFile、(ファイルバイトをダウンロードする) および意図していない可能性のあるその他の読み取りを許可します。 は、 Allow に関係なくDeny上書きしますが、明示的な形式の方がモデルにとってより安全なパターンです。

例: 単一のワークスペースでの同期推論

1 つの本稼働ワークスペースに対して推論を実行する IAM プリンシパルに最小限のアクセス許可を付与します。

```
{
  "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**
>
> このポリシーのList*ワイルドカードはListWorkspaces、アカウントスコープの ともー致します。ワークスペース ARN 制約はそれをサイレントに除外するため、このポリシーはワークスペースのー覧表示を許可しません。サービスアカウントでワークスペースを列挙する必要がある場合は、 ListWorkspacesで に別のAllowステートメントを追加しますResource: "*"。

このポリシーは、AWS SigV4 認証を前提としています。プリンシパルが API キーで認証する場合は、 も付与します aws-external-anthropic:CallWithBearerToken (「認証」を参照) 。

例: 顧客ごとのワークスペース分離

ロールを 1 つのワークスペースに制限します。

```
{
  "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

最初のステートメントのaws-external-anthropic:*ワイルドカードには、ワークスペース ARN 制約がサイレントに除外するアカウントスコープアクション (CreateWorkspace、ListWorkspaces) が含まれています。これは「分離」インテントと一致しています。ロールはワークスペースを作成または列挙できませんが、ポリシーには効果のないアクセス許可が含まれています。「アカウントスコープパターンのプロビジョニングの自動化」を参照してください。
2 番目のステートメントは、ワークスペース ARN にバインドされないルートレスアクションであるため、すべてのリソースAssumeConsoleに対して CallWithBearerTokenと を付

与します。ロールが SigV4 のみを使用し、Claude コンソールにフェデレーションしない場合は、2 番目のステートメントを省略します。

例: ZDR 対応ワークスペースの機能ロックダウン

同期推論を使用可能のまま、特定のワークスペースでのバッチ処理とファイルのアップロードをブロックします。ワークスペースがサーバー側で保持してはならないゼロデータ保持 (ZDR) データを処理する場合に便利です。このポリシーを、 などの許可ポリシーAnthropicLimitedAccessまたは上記の単一ワークスペースの例と一緒にアタッチします。単独では、拒否のみのポリシーはアクセス許可を付与しません。

```
{
  "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

この拒否は作成のみをブロックします。他のファイルおよびバッチアクションは、リストしない限り拒否されません。ワークスペースがファイルやバッチを保持してはならない完全なロックダウンの場合、aws-external-anthropic:GetFile、、aws-external-anthropic:ListFiles、aws-external-anthropic:DeleteFile、aws-external-anthropic:GetBatchInference、aws-external-anthropic:ListBatchInferencesaws-external-anthropic:CancelBatchInference、および も拒否しますaws-external-anthropic:DeleteBatchInference。

例: 自動化のプロビジョニング

推論アクセス許可なしで、ワークスペースの作成と管理に必要なアクションを CI/CD ロールに付与します。

```
{
  "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 および ListWorkspacesはアカウントスコープのオペレーションです。これらのアクションでワークスペース ARN を指定しても効果はありません。 を使用しますResource: "*"。

マネージドポリシー

AWS は、一般的なアクセスパターンの マネージドポリシーを提供します。

- **AnthropicFullAccess**: すべてのリソースaws-external-anthropic:*に を付与します。
- **AnthropicReadOnlyAccess**: すべてのリソースCallWithBearerTokenで Get*、List*、および を付与します。
- **AnthropicInferenceAccess**: すべてのリソースで ReadOnly アクションと推論アクション (CreateInference、CreateBatchInference、CancelBatchInferenceDeleteBatchInference を付与します。
- **AnthropicLimitedAccess**: AnthropicInferenceAccessアクションに加えて、すべてのリソースに対するすべての Claude Managed Agents アクション (エージェント、セッション、環境、ボールト、メモリストア) を付与します。

- **AnthropicSelfHostedEnvironmentAccess**: セルフホストサンドボックスワーカーが環境作業項目のポーリングと処理、関連する環境、セッション、スキルリソースの読み取り、セッション状態の更新を行うために必要なアクセス許可を付与します。セルフホスト環境ワーカーが引き受ける IAM ロールにこのポリシーをアタッチします。AnthropicSelfHostedEnvironmentAccessは推奨される単一ロールのデフォルトです。個別の IAM プリンシパルで作業ポーラーとセッションごとのサンドボックスを実行すると、これらのアクセス許可を 2 つの狭いカスタムポリシーに分割して、最小特権を確保できます。

AnthropicInferenceAccess は、推論を実行するのに十分な最も狭い管理ポリシーです。Get* および List*ワイルドカードを介して、 を介してダウンロードされるファイルコンテンツGetFileや を介してメモリコンテンツなど、名前空間内のすべての API リソースへの読み取りアクセスを許可しますGetMemoryStore。ファイルやスキルの作成や削除、ユーザープロファイル管理、ワークスペースミューテーション、コンソールフェデレーションは許可されません。

> **Note**
>
> AnthropicReadOnlyAccess、AnthropicInferenceAccess、および は を付与AnthropicLimitedAccessしませんAssumeConsole。Claude コンソールにフェデレーションする必要があるプリンシパルには、 aws-external-anthropic:AssumeConsole AnthropicFullAccessまたはカスタムポリシーのいずれかを使用して、 に対する個別の許可が必要です。「Claude コンソールへのフェデレーション」を参照してください。

> **Note**
>
> CreateInference と CreateBatchInferenceは個別のアクションです。一方を拒否しても、他方はブロックされません。すべてのモデル呼び出しを禁止する場合は、両方を拒否します。

Claude コンソールへのフェデレーション

aws-external-anthropic:AssumeConsole では、IAM プリンシパルを Anthropic が運営する Claude コンソールにフェデレーションできます。コンソール内のアクセスは、ほとんどのオペレーションで引き続き IAM によって管理されますが、主に対応する IAM API を持たない使用ビューである管理オペレーションのサブセットは、プリンシパルがフェデレーションした機能でゲートされます。

次の 2 つの機能があります。

- **developer** — は、AWS の Claude プラットフォーム開発者がday-to-dayで必要とするオペレーションを許可します。コンソールからの推論の実行、ワークスペースデータの読み取り、個人の使用状況の表示です。
- **admin** — さらに、アカウント全体の使用状況ビューや IAM アクションでは表示されない管理設定など、管理者専用のコンソールオペレーションも許可します。

AssumeConsole アクションの aws-external-anthropic:Capability条件キーを使用して、プリンシパルがリクエストできる機能を制御します。

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

AnthropicFullAccess 機能制限AssumeConsoleのない 許可。デベロッパーのみのコンソールアクセスまたは管理者のみのアクセスなど、より狭い権限には、上記のように aws-external-anthropic:Capability条件AssumeConsoleでスコープするカスタムポリシーをアタッチします。

ベアラートークンを使用した呼び出し

aws-external-anthropic:CallWithBearerToken は、API キーがベアラートークンとして提示されるときに使用される SigV4-freeリクエストパスを承認します。API キーで認証するプリンシパルには、ターゲットワークスペースでこのアクションが必要です。これは、 Authorization: Bearer ヘッダーを直接使用ANTHROPIC_AWS_API_KEYまたは設定する場合に適用されます。

これは、推論アクション (CreateInference、 CreateBatchInferenceなど) に加えて API キー発信者に必要です。がない場合CallWithBearerToken、API キーリクエストは推論認可チェックに到達する前に拒否されます。SigV4 発信者にはこのアクションは必要ありません。

AnthropicReadOnlyAccess、AnthropicInferenceAccess、AnthropicLimitedAccess、および AnthropicFullAccessはすべて を含みますCallWithBearerToken。API キーアクセスのカスタムポリシーを記述する場合は、明示的に追加します。

アウトバウンドウェブ ID フェデレーション (コンソールアクセスに必要)

Claude コンソールは、AWS ではなく Anthropic インフラストラクチャで実行されます。IAM プリンシパルが を呼び出すとAssumeConsole、AWS STS は Anthropic オーディエンスを対象とするウェブ ID トークンを発行します。その後、Anthropic コンソールはそのトークンを受け入れ、フェデレーティッドセッションを確立します。これを機能させるには、AWS アカウントがaws-external-anthropic対象者へのアウトバウンドウェブ ID フェデレーションを許可する必要があります。

を呼び出すプリンシパルには、 aws-external-anthropic:AssumeConsoleアクションに加えて次の STS アクセス許可AssumeConsoleが必要です。

- **sts:GetWebIdentityToken** — Anthropic コンソールが消費するウェブ ID トークンの発行を許可します。
- **sts:TagGetWebIdentityToken** — は、セッションタグをウェブ ID トークンにアタッチすることを許可します。AWS の Claude Platform は、これらのタグを使用して、プリンシパルの機能とワークスペースコンテキストをコンソールに伝えます。

コンソールユーザーが引き受けるロールの信頼ポリシー、または AssumeConsole を直接呼び出すユーザー ID にアタッチされたインライン/管理ポリシーの両方に を含めます。 には、両方の STS アクションAnthropicFullAccessが含まれます。SCP またはアクセス許可sts:*の境界レベルで拒否する環境では、コンソールフェデレーションを成功させるために、これら 2 つのアクションを明示的に許可する必要があります。

AWS での Claude プラットフォームの IAM アクション

AWS ポリシーを通じて AWS 上の Claude プラットフォームへのアクセスを制御するための IAM アクションリファレンス。

Claude Platform on AWS は、アクセスコントロールに AWS IAM を使用します。すべての API ルートは、 `aws-external-anthropic`名前空間の IAM アクションにマッピングされます。このページには、すべてのアクション、各アクションが承認するルート、一般的なアクセスパターンで使用できる管理ポリシーが一覧表示されます。プラットフォームのセットアップと認証については、「AWS での Claude プラットフォームとは」を参照してください。

サービスの詳細

IAM サービスプレフィックス	`aws-external-anthropic`
リソースタイプ:	`workspace`
Workspace ARN	`arn:aws:aws-external-anthropic:{region}:{account-id}:workspace/{workspace-id}`

ARN リージョンは常に入力され、ワークスペースがバインドされているリージョンと一致します。リソースセグメントは、タグ付けされたワークスペース ID (`wrkspc_…`) です。これは、`anthropic-workspace-id` ヘッダーで渡す値と同じです。

アクション

このサービスは、15 グループの 65 のアクションを定義します。アクションは AWS `VerbNoun`の規則に従い、動詞規律を使用して、 `Get*`および `List*`ワイルドカードがクリーンな読み取り専用境界を生成するようにします。

推測

アクション	承認されたルート
`CreateInference`	`POST /v1/messages`

アクション	承認されたルート
CountTokens	POST /v1/messages/count_tokens

Batch 処理

アクション	承認されたルート
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}

モデル

アクション	承認されたルート
GetModel	GET /v1/models/{id}
ListModels	GET /v1/models

ファイル

アクション	承認されたルート
CreateFile	POST /v1/files
GetFile	GET /v1/files/{id} GET /v1/files/{id}/content
ListFiles	GET /v1/files

アクション	承認されたルート
DeleteFile	DELETE /v1/files/{id}

> **Note**
>
> GetFile はメタデータとコンテンツのダウンロードの両方を承認します。読み取り専用アクセス権を持つプリンシパルは、ファイルを一覧表示するだけでなく、ファイルバイトをダウンロードできます。

スキル

アクション	承認されたルート
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**
>
> 個々のスキルバージョンを削除するとUpdateSkill、 ではなく にマッピングされますDeleteSkill。を拒否するポリシーaws-external-anthropic:Delete*では、バージョンの削除が引き続き許可されます。スキルのミューテーションを防ぐ必要がある場合CreateSkillは、 UpdateSkillと も拒否します。

ユーザープロファイル

アクション	承認されたルート
CreateUserProfile	POST /v1/user_profiles
GetUserProfile	GET /v1/user_profiles/{id}
ListUserProfiles	GET /v1/user_profiles
UpdateUserProfile	POST /v1/user_profiles/{id}

> **Warning**
>
> IAM アクションマッチングでは、大文字と小文字は区別されません。ワイルドカードは CreateFile、GetFile、 aws-external-anthropic:*Fileに一致しますがDeleteFile、一致しません ListFiles (「files」ではなく「files」で終わります) 。またCreateUserProfile、「Profile」は「file」で終わるUpdateUserProfileためGetUserProfile、、「」、「」ともオーバーマッチします。Files API アクションのみを許可または拒否する場合は、*Fileサフィックスパターンを使用するのではなく、明示的に (CreateFile、GetFile、ListFiles、DeleteFile) 列挙します。

エージェント

Claude マネージドエージェントのエージェント定義。

アクション	承認されたルート
CreateAgent	エージェント作成ルート
GetAgent	エージェント読み取りルート
ListAgents	エージェントリストルート
UpdateAgent	エージェント更新ルート
ArchiveAgent	エージェントアーカイブルート

セッション

エージェントセッションとイベント履歴。

アクション	承認されたルート
CreateSession	セッションルートの作成
GetSession	セッション読み取りルート
ListSessions	セッションリストルート
UpdateSession	セッション更新ルート
ArchiveSession	セッションアーカイブルート
DeleteSession	セッション削除ルート

環境

セッションのクラウドコンテナ設定。

アクション	承認されたルート
CreateEnvironment	環境作成ルート
GetEnvironment	環境読み取りルート
ListEnvironments	環境リストルート
UpdateEnvironment	環境更新ルート
ArchiveEnvironment	環境アーカイブルート
DeleteEnvironment	環境削除ルート
ProcessEnvironmentWork	セルフホスト環境ワーカールート

ボールト

セッション認証の認証情報ボールト。

アクション	承認されたルート
`CreateVault`	ボールト作成ルート
`GetVault`	ボールト読み取りルート
`ListVaults`	ボールトリストルート
`UpdateVault`	ボールト更新ルート
`ArchiveVault`	ボールトアーカイブルート
`DeleteVault`	ボールト削除ルート

> **Note**
>
> ボールトオペレーションは (データイベントではなく) CloudTrail 管理イベントに分類されます。ボールトは認証情報を保持し、デフォルトオンの監査ログ記録の恩恵を受けるためです。その他の Claude マネージドエージェントのオペレーション (エージェント、セッション、環境、メモリストア) はデータイベントです。

メモリストア

永続エージェントメモリ。

アクション	承認されたルート
`CreateMemoryStore`	メモリストア作成ルート
`GetMemoryStore`	メモリストアの読み取りルート
`ListMemoryStores`	メモリストアリストルート
`UpdateMemoryStore`	メモリストアの更新ルート

アクション	承認されたルート
ArchiveMemoryStore	メモリストアアーカイブルート
DeleteMemoryStore	メモリストアの削除ルート

> **Note**
>
> GetMemoryStore はメモリコンテンツを読み取ります。したがって、 マネージドポリシーまたはカスタムポリシーのGet*ワイルドカードは、メモリ読み取りアクセスを付与します。メモリコンテンツを制限する必要がある場合は、スコープポリシーを明示的に指定します。

ウェブフック

セッションのライフサイクルイベント通知。

アクション	承認されたルート
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

アクション	承認されたルート
CreateWorkspace	POST /v1/organizations/workspaces

アクション	承認されたルート
GetWorkspace	GET /v1/organizations/workspaces/{id}
ListWorkspaces	GET /v1/organizations/workspaces
UpdateWorkspace	POST /v1/organizations/workspaces/{id}
ArchiveWorkspace	POST /v1/organizations/workspaces/{id}/archive

デフォルトのワークスペースはサインアップ時にプロビジョニングされます。「Workspaces」を参照してください。

認証

アクション	承認されたルート
CallWithBearerToken	(none)

CallWithBearerToken は、プリンシパルが AWS SigV4 ではなく API キー (ベアラートークン) を介して認証することを許可する認証レイヤーアクセス許可です。ルートにはマッピングされません。API キーホルダーが実行するルートマップアクションと一緒に付与します。

コンソールアクセス

アクション	承認されたルート
AssumeConsole	(none)

AssumeConsole は、AWS コンソールフェデレーションフローを通じて、AWS ワークスペース上の Claude プラットフォームの Claude コンソールを開くことをプリンシパルに許可します。ルートにはマッピングされません。AWS コンソールの Claude プラットフォームの AWS サービスページで Open Claude コンソールをクリックできるプリンシパルに付与します。AssumeConsole アクションaws-external-anthropic:Capabilityの条件キーは、プリンシパルがリクエストできる

コンソール機能 (開発者または管理者) を制御します。詳細については、「IAM ポリシー」およびサインインフローの Claude コンソールの使用を参照してください。

> **Warning**
>
> `aws-external-anthropic:AssumeConsole` は、発信者自身のセッション期間に関係なく、最大 12 時間持続する Claude コンソールセッションを発行します。IAM セッションが 12 時間未満の発信者は、ソース認証情報を期限切れにするコンソールセッションを取得できます。Claude コンソールへのアクセスを必要とするプリンシパルにこのアクセス許可を制限し、不要になったらすぐに取り消します。

> **Note**
>
> フェデレーション後に Claude コンソール内で実行されたアクションは、AWS CloudTrail に記録されず、IAM を通じて帰属しません。これには、使用状況レポートの表示など、ワークスペース範囲のグローバルアクティビティが含まれます。コンソールアクティビティの監査証跡が必要な場合は、CloudTrail ではなく Claude コンソールで利用可能な監査ログを使用します。

Route-to-action マッピング

次の表に、AWS 上の Claude Platform のすべてのルートと、その呼び出しに必要な IAM アクションを示します。安定ルートの IAM アクションは、 `anthropic-beta`ヘッダーを使用するリクエストも承認します。CloudTrail は、各アクションをデータイベント (大容量、データプレーンオペレーション) または管理イベント (コントロールプレーンオペレーション) として分類します。

方法	ルート	IAM アクション	CloudTrail イベントタイプ
`POST`	`/v1/messages`	`CreateInference`	データ
`POST`	`/v1/messages/count_tokens`	`CountTokens`	データ
`POST`	`/v1/messages/batches`	`CreateBatchInference`	データ

方法	ルート	IAM アクション	CloudTrail イベントタイプ
GET	/v1/messages/batches	ListBatchInferences	データ
GET	/v1/messages/batches/{id}	GetBatchInference	データ
GET	/v1/messages/batches/{id}/results	GetBatchInference	データ
POST	/v1/messages/batches/{id}/cancel	CancelBatchInference	データ
DELETE	/v1/messages/batches/{id}	DeleteBatchInference	データ
GET	/v1/models	ListModels	データ
GET	/v1/models/{id}	GetModel	データ
POST	/v1/files	CreateFile	データ
GET	/v1/files	ListFiles	データ
GET	/v1/files/{id}	GetFile	データ
GET	/v1/files/{id}/content	GetFile	データ
DELETE	/v1/files/{id}	DeleteFile	データ
POST	/v1/skills	CreateSkill	データ
GET	/v1/skills	ListSkills	データ
GET	/v1/skills/{id}	GetSkill	データ
DELETE	/v1/skills/{id}	DeleteSkill	データ
POST	/v1/skills/{id}/versions	UpdateSkill	データ

方法	ルート	IAM アクション	CloudTrail イベントタイプ
GET	/v1/skills/{id}/versions	GetSkill	データ
GET	/v1/skills/{id}/versions/{version}	GetSkill	データ
GET	/v1/skills/{id}/versions/{version}/content	GetSkill	データ
DELETE	/v1/skills/{id}/versions/{version}	UpdateSkill	データ
POST	/v1/user_profiles	CreateUserProfile	データ
GET	/v1/user_profiles	ListUserProfiles	データ
GET	/v1/user_profiles/{id}	GetUserProfile	データ
POST	/v1/user_profiles/{id}	UpdateUserProfile	データ
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 ルート (エージェント、セッション、環境、ボールト、メモリストア) は、同じ route-to-action パターンに従います。完全なルートごとのマッピングについては、Anthropic IAM アクションリファレンスを参照してください。ボールトルートは管理イベント、エージェント、セッション、環境、メモリストアルートはデータイベントです。

AWS の Claude プラットフォーム上にないルートは、デフォルトでゲートウェイで拒否されます。

関連情報

- ポリシーの例と管理ポリシーの IAM ポリシー
- IAM ポリシー構文と評価ロジックの AWS IAM ユーザーガイド
- 監査ログ記録設定用の AWS CloudTrail ユーザーガイド

ドキュメント履歴

次の表は、AWS ユーザーガイドの Claude プラットフォームの重要な変更点を示しています。

変更	説明	日付
初版発行	Claude Platform on AWS ユーザーガイドを公開しました。	2026 年 5 月 7 日

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