ユーザーガイド

AWS Deadline クラウド

Version latest

AWS Deadline クラウド: ユーザーガイド

Table of Contents

AWS デッドラインクラウドとは？

Deadline Cloud は、デジタルコンテンツ作成パイプラインやワークステーションから直接 Amazon Elastic Compute Cloud (Amazon EC2) インスタンス上でレンダリングプロジェクトやジョブを作成および管理するために使用できます。 AWS のサービス

Deadline Cloud には、コンソールインターフェイス、ローカルアプリケーション、コマンドラインツール、および API が用意されています。Deadline Cloudでは、ファーム、フリート、ジョブ、ユーザーグループ、ストレージを作成、管理、監視できます。ハードウェア要件を指定したり、特定のワークロード用の環境を作成したり、制作に必要なコンテンツ作成ツールをDeadline Cloudパイプラインに統合したりすることもできます。

Deadline Cloud は、すべてのレンダリングプロジェクトを 1 か所で管理できる統合インターフェイスを提供します。ユーザーの管理、ユーザーへのプロジェクトの割り当て、職務の権限の付与を行うことができます。

トピック

Deadline クラウドの機能

Deadline Cloudがビジュアルコンピューティングワークロードの実行と管理に役立つ主な方法をいくつかご紹介します。

- ファーム、キュー、フリートを素早く作成できます。それらの状況を監視し、ファームの運営や仕事に関する洞察を得ましょう。
- Deadline Cloudのユーザーとグループを一元管理し、権限を割り当てます。
- を使用して、プロジェクトユーザーと外部の ID プロバイダーのサインインセキュリティを管理します。 AWS IAM Identity Center

- AWS Identity and Access Management (IAM) のポリシーとロールを使用して、プロジェクトリソースへのアクセスを安全に管理します。
- タグを使用してプロジェクトリソースを整理し、すばやく見つけることができます。
- プロジェクトのリソース使用量と見積もりコストを管理できます。
- クラウドまたは対面でのレンダリングをサポートする、幅広いコンピューティング管理オプションを提供します。

Deadline クラウドの概念と用語

このトピックでは、 AWS Deadline Cloudを使い始めるのに役立つように、主要な概念と用語をいくつか説明します。

予算マネージャー

予算マネージャーは Deadline クラウドモニターの一部です。予算マネージャーを使用して予算を作成および管理します。また、予算内に収まるようにアクティビティを制限することもできます。

Deadline Cloud クライアントライブラリ

クライアントライブラリには、Deadline Cloud を管理するためのコマンドラインインターフェースとライブラリが含まれています。機能には、Open Job Description 仕様に基づくジョブバンドルのDeadline Cloudへの送信、ジョブ添付ファイルの出力のダウンロード、コマンドラインインターフェイスを使用したファームの監視が含まれます。

デジタルコンテンツ作成アプリケーション (DCC)

デジタルコンテンツ作成アプリケーション (DCC) は、デジタルコンテンツを作成するサードパーティ製品です。DCC の例としては、、Maya、などがあります。Nuke HoudiniDeadline Cloud には、特定の DCC 用のジョブ送信者統合プラグインが用意されています。

ファーム

ファームはプロジェクトのリソースが置かれる場所です。キューとフリートで構成されています。

フリート

フリートはレンダリングを行うワーカーノードのグループです。ワーカーノードはジョブを処理します。1 つのフリートを複数のキューに関連付けることができ、1 つのキューを複数のフリートに関連付けることができます。

ジョブ

ジョブはレンダリングリクエストです。ユーザーはジョブをサブミットします。ジョブには、ステップとタスクとして概説された特定のジョブプロパティが含まれています。

Job 添付ファイル

ジョブアタッチメントは、ジョブの入力と出力を管理するために使用できるDeadline Cloudの機能です。Job ファイルは、レンダリング処理中にジョブの添付ファイルとしてアップロードされます。これらのファイルには、テクスチャ、3D モデル、照明リグ、その他の類似アイテムがあります。

ジョブプロパティ

Job プロパティは、レンダリングジョブの送信時に定義する設定です。例としては、フレームレンジ、出力パス、ジョブアタッチメント、レンダリング可能なカメラなどがあります。プロパティは、レンダリングの送信元の DCC によって異なります。

ジョブテンプレート

ジョブテンプレートは、Deadline Cloud ジョブの一部として実行されるランタイム環境とすべてのプロセスを定義します。

キュー

キューは送信されたジョブが保存され、レンダリングがスケジュールされる場所です。レンダリングを正常に行うには、キューをフリートに関連付ける必要があります。キューは複数のフリートに関連付けることができます。

キューとフリートの関連付け

キューがフリートと関連付けられると、キューとフリートが関連付けられます。アソシエーションを使用して、フリートのワーカーをそのキュー内のジョブにスケジュールします。アソシエーションを開始および停止して、作業のスケジュールを管理できます。

[ステップ]

ステップはジョブ内で実行する 1 つの特定のプロセスです。

Deadline クラウド送信者

Deadline Cloud サブミッターはデジタルコンテンツ作成 (DCC) プラグインです。アーティストはこのプラグインを使用して、使い慣れたサードパーティの DCC インターフェースからジョブを送信します。

タグ

AWS タグはリソースに割り当てることができるラベルです。各タグは、お客様が定義するキーとオプション値で構成されています。

タグを使用すると、 AWS リソースをさまざまな方法で分類できます。例えば、各インスタンスの所有者とスタックレベルを追跡しやすくするため、アカウントの Amazon EC2 インスタンスに対してタグセットを定義できます。

また、 AWS リソースを目的、所有者、または環境別に分類することもできます。この方法は、同じ種類のリソースが多数ある場合に役立ちます。割り当てたタグに基づいて、特定のリソースをすばやく識別できます。

タスク

タスクはレンダリングステップの 1 つのコンポーネントです。

使用量ベースのライセンス (UBL)

従量制ライセンス (UBL) は、一部のサードパーティ製品で利用できるオンデマンドライセンスモデルです。このモデルは従量課金制で、使用した時間と分数に応じて課金されます。

使用状況エクスプローラー

ユースエクスプローラーはDeadline クラウドモニターの機能です。コストと使用量の概算値が表示されます。

ワーカー

ワーカーはフリートに所属し、Deadline Cloud が割り当てたタスクを実行してステップとジョブを完了します。ワーカーはタスクオペレーションのログを Amazon CloudWatch Logs に保存します。ワーカーはジョブ添付機能を使用して、入力と出力を Amazon Simple Storage Service (Amazon S3) バケットに同期することもできます。

Deadline クラウドを使い始めましょう

Deadline Cloud を使用すると、Amazon EC2 インスタンス設定や Amazon Simple Storage Service (Amazon S3) バケットなどのデフォルト設定とリソースを使用してレンダーファームをすばやく作成できます。

レンダーファームを作成するときにも設定とリソースを定義できます。この方法は、デフォルトの設定やリソースを使用するよりも時間がかかりますが、より細かく制御できます。

Deadline Cloud の概念と用語を理解したら、「はじめに」を参照して、ファームの作成、ユーザーの追加、役立つ情報へのリンクを確認してください。 step-by-step

Deadline クラウドへのアクセス

Deadline Cloud には、以下のいずれかの方法でアクセスできます。

- Deadline Cloud コンソール — ブラウザーでコンソールにアクセスしてファームとそのリソースを作成し、ユーザーアクセスを管理します。詳細については、「開始する」を参照してください。
- Deadline Cloud モニター — 優先順位やジョブステータスの更新など、レンダリングジョブを管理します。ファームを監視し、ログとジョブのステータスを表示します。所有者権限を持つユーザーには、Deadline Cloud モニターから使用状況の確認や予算の作成もできます。Deadline Cloud モニターは、Web ブラウザーとデスクトップアプリケーションの両方で使用できます。
- AWS SDK と AWS CLI — AWS Command Line Interface (AWS CLI) を使用して、ローカルシステムのコマンドラインから Deadline Cloud API オペレーションを呼び出します。詳細については、「開発者ワークステーションの設定」を参照してください。

関連サービス

Deadline Cloud は以下と連携します。 AWS のサービス

- Amazon CloudWatch — を使用すると CloudWatch、 AWS プロジェクトと関連リソースを監視できます。詳細については、Amazon CloudWatch ユーザーガイドを参照してください。
- Amazon EC2 — AWS のサービス クラウドでアプリケーションを実行する仮想サーバーを提供します。ワークロードに Amazon EC2 インスタンスを使用するようにプロジェクトを設定できます。詳細については、「Amazon EC2 インスタンス」を参照してください。
- Amazon EC2 Auto Scaling — Auto Scaling を使用すると、インスタンスに対する需要の変化に応じて、インスタンス数を自動的に増減できます。Auto Scaling は、インスタンスに障害が発生した場合でも、必要な数のインスタンスを実行していることを確認するのに役立ちます。Deadline Cloud でAuto Scaling を有効にすると、Auto Scaling によって起動されたインスタンスは自動的にワークロードに登録されます。同様に、Auto Scaling によって終了されたインスタンスは、自動的にワークロードから登録解除されます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドを参照してください。
- AWS PrivateLink— トラフィックをパブリックインターネットに公開することなく AWS のサービス、仮想プライベートクラウド (VPC) AWS PrivateLink とオンプレミスネットワーク間のプライ

ベート接続を提供します。 AWS PrivateLink 異なるアカウントや VPC にまたがるサービスを簡単に接続できます。詳細については、「AWS PrivateLink」を参照してください。

- Amazon S3 — Amazon S3 はオブジェクトストレージサービスです。Deadline クラウドは Amazon S3 バケットを使用してジョブの添付ファイルを保存します。
- IAM Identity Center — IAM Identity Center は、ユーザーに割り当てられたすべてのアカウントとアプリケーションへのシングルサインオンアクセスを 1 AWS のサービス か所から提供できる場所です。また、 AWS Organizationsのすべてのアカウントへのマルチアカウントアクセスとユーザーのアクセス許可を、一元的に管理することも可能です。詳細については、「AWS IAM Identity Center に関するよくある質問」を参照してください。

Deadline Cloud の仕組み

Deadline Cloud を使用すると、デジタルコンテンツ作成 (DCC) パイプラインとワークステーションから直接レンダリングプロジェクトとジョブを作成および管理できます。

AWS SDK、 AWS Command Line Interface (AWS CLI) 、または Deadline Cloud ジョブ送信者を使用して Deadline Cloud にジョブを送信します。Deadline Cloud は、ジョブテンプレート仕様の Open Job Description (OpenJD) をサポートしています。詳細については、 GitHubウェブサイトの「オープンジョブの説明」を参照してください。

Deadline Cloud はジョブ送信者を提供します。ジョブ送信者は、 Mayaや などのサードパーティーの DCC インターフェイスからレンダージョブを送信するための DCC プラグインですNuke。送信者を使用すると、アーティストはサードパーティーのインターフェイスから Deadline Cloud にレンダリングジョブを送信できます。Deadline Cloud では、プロジェクトリソースが管理され、ジョブがモニタリングされます。

Deadline Cloud ファームを使用すると、キューとフリートの作成、ユーザーの管理、プロジェクトリソースの使用状況とコストの管理を行うことができます。ファームはキューとフリートで構成されます。キューは、送信されたジョブが配置され、レンダリングがスケジュールされている場所です。フリートは、タスクを実行してジョブを完了するワーカーノードのグループです。ジョブをレンダリングするには、キューをフリートに関連付ける必要があります。1 つのフリートで複数のキューをサポートでき、1 つのキューを複数のフリートでサポートできます。

ジョブはステップで構成され、各ステップは特定のタスクで構成されます。Deadline Cloud モニターを使用すると、ジョブ、ステップ、タスクのステータス、ログ、その他のトラブルシューティングメトリクスにアクセスできます。

Deadline Cloud のアクセス許可

Deadline Cloud は以下をサポートしています。

- AWS Identity and Access Management (IAM) を使用した API オペレーションへのアクセスの管理
- との統合を使用したワークフォースユーザーのアクセスの管理 AWS IAM Identity Center

誰でもプロジェクトに取り組む前に、そのプロジェクトと関連するファームにアクセスできる必要があります。Deadline Cloud は IAM Identity Center と統合され、ワークフォースの認証と認可を管理します。ユーザーを IAM Identity Center に直接追加することも、 や などの既存の ID プロバイダー (IdP) Okta に接続することもできますActive Directory。IT 管理者は、さまざまなレベルでユーザーやグループにアクセス許可を付与できます。後続の各レベルには、前のレベルのアクセス許可が含まれます。次のリストは、最低レベルから最高レベルまでの 4 つのアクセスレベルを示しています。

- ビューワー - アクセスできるファーム、キュー、フリート、ジョブ内のリソースを表示するアクセス許可。ビューワーはジョブを送信または変更できません。
- Contributor – ビューワーと同じですが、キューまたはファームにジョブを送信するアクセス許可があります。
- マネージャー – 寄稿者と同じですが、アクセスできるキュー内のジョブを編集し、アクセスできるリソースに対するアクセス許可を付与するアクセス許可があります。
- 所有者 – マネージャーと同じですが、予算を表示および作成し、使用状況を確認できます。

> **Note**
>
> これらのアクセス許可は、Deadline Cloud インフラストラクチャを変更するための AWS Management Console または アクセス許可をユーザーに付与しません。

ユーザーは、関連するキューとフリートにアクセスする前に、ファームにアクセスできる必要があります。ユーザーアクセスは、ファーム内でキューとフリートに個別に割り当てられます。

ユーザーを個人として追加することも、グループの一部として追加することもできます。ファーム、フリート、またはキューにグループを追加すると、大規模なグループのアクセス許可を簡単に管理できます。例えば、特定のプロジェクトに取り組んでいるチームがある場合は、各チームメンバーをグループに追加できます。その後、対応するファーム、フリート、またはキューのグループ全体にアクセス許可を付与できます。

Deadline Cloud でのソフトウェアサポート

Deadline Cloud は、コマンドラインインターフェイスから実行され、パラメータ値を使用して制御できるすべてのソフトウェアアプリケーションと連携します。Deadline Cloud は、パラメータ化されたソフトウェアスクリプトステップ (フレーム範囲全体など) を使用して、作業をジョブとしてタスク に記述するためのOpenJD仕様をサポートしています。Deadline Cloud のツールと機能を使用してOpenJDジョブ指示書をジョブバンドルにアセンブルし、サードパーティーのソフトウェアアプリケーションからステップを作成、実行、ライセンスします。

ジョブをレンダリングするにはライセンスが必要です。Deadline Cloud は、使用状況に基づいて分単位で時間単位で請求されるソフトウェアアプリケーションライセンスの選択に対して、使用状況ベースのライセンス (UBL) を提供します。Deadline Cloud では、必要に応じて独自のソフトウェアライセンスを使用することもできます。ジョブがライセンスにアクセスできない場合、レンダリングされず、Deadline Cloud モニターのタスクログに表示されるエラーが生成されます。

Deadline Cloud の開始方法

AWS Deadline Cloud でファームを作成するには、Deadline Cloud コンソールまたは AWS Command Line Interface () を使用できますAWS CLI。コンソールを使用して、キューやフリートを含むファームの作成をガイド付きで体験できます。を使用して AWS CLI 、サービスを直接操作したり、Deadline Cloud で動作する独自のツールを開発したりできます。

ファームを作成し、Deadline Cloud モニターを使用するには、Deadline Cloud のアカウントを設定します。Deadline Cloud モニターインフラストラクチャは、アカウントごとに 1 回だけセットアップする必要があります。ファームから、ファームとそのリソースへのユーザーアクセスを含むプロジェクトを管理できます。

Deadline Cloud モニターインフラストラクチャを設定せずにファームを作成するには、Deadline Cloud 用のデベロッパーワークステーションを設定します。

ジョブを受け入れる最小限のリソースでファームを作成するには、コンソールのホームページでクイックスタートを選択します。 では、これらの手順Deadline Cloud モニターをセットアップするについて説明します。これらのファームは、キューと自動的に関連付けられるフリートから始まります。このアプローチは、実験するサンドボックススタイルのファームを作成する便利な方法です。

トピック

AWS アカウントのセットアップ

Deadline Cloud AWS を使用する AWS アカウント ように を設定します。

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。

2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用してルートユーザーアクセスが必要なタスクを実行してください。

を初めて作成するときは AWS アカウント、アカウント内のすべての およびリソースへの AWS のサービス 完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。

> **Important**
>
> 日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

Deadline Cloud モニターをセットアップする

開始するには、Deadline Cloud モニターインフラストラクチャを作成し、ファームを定義する必要があります。グループとユーザーの追加、サービスロールの選択、リソースへのタグの追加など、追加のオプション手順を実行することもできます。

ステップ 1: モニターを設定する

Deadline Cloud モニターは、 を使用してユーザーを承認 AWS IAM Identity Center します。Deadline Cloud に使用する IAM Identity Center インスタンスは、モニター AWS リージョン と同じ にある必要があります。モニターの作成時にコンソールで別のリージョンを使用している場合は、IAM Identity Center リージョンに変更するよう求めるメッセージが表示されます。

モニタのインフラストラクチャは、次のコンポーネントで構成されます。

- Monitor の表示名 : Monitor の表示名は、モニターを識別する方法です。例えば、AnyCompany モニター です。モニタの名前によってもモニタ URL が決まります。

> **Important**
> セットアップが完了したら、モニタの表示名を変更することはできません。

- モニター URL : モニター URL を使用してモニターにアクセスできます。URL は Monitor の表示名、例えば https://anycompanymonitor.awsapps.com に基づいています。

> **Important**
> セットアップが完了したら、 Monitor URL を変更することはできません。

- AWS リージョン: AWS リージョンは、 AWS データセンターの集合の物理的な場所です。モニタを設定すると、リージョンはデフォルトで最も近い場所に設定されます。ユーザーに最も近い場所に配置されるようにリージョンを変更することをお勧めします。これにより、遅延が軽減され、データ転送速度が向上します。 は Deadline Cloud AWS リージョン と同じ で有効に AWS IAM Identity Center する必要があります。

> **Important**
> Deadline Cloud の設定が完了したら、リージョンを変更することはできません。

このセクションのタスクを完了して、モニタのインフラストラクチャを設定します。

モニタのインフラストラクチャを設定するには

1. にサインインAWS Management Consoleして「Deadline クラウドへようこそ」のセットアップを開始し、「次へ」を選択します。
2. Monitor の表示名を入力します。例えば、 です**`AnyCompany Monitor`**。
3. （オプション） Monitor 名 を変更するには、URL の編集 を選択します。
4. （オプション) ユーザーに最も近いAWS リージョンように を変更するには、「リージョンの変更」を選択します。
   a. ユーザーに最も近いリージョンを選択します。
   b. [リージョンを適用] を選択します。

- （オプション）グループとユーザーを追加するには、 を選択します（オプション) グループとユーザーを追加する。
- （オプション）モニタの設定をさらにカスタマイズするには、 を選択します詳細設定。

5. の準備ができたらステップ 2: ファームの詳細を定義する、次へを選択します。

（オプション) グループとユーザーを追加する

Deadline Cloud モニターの設定を完了する前に、モニターユーザーを追加してグループに追加できます。

セットアップが完了したら、新しいユーザーとグループを作成し、 などのユーザーを管理してグループ、アクセス許可、アプリケーションを割り当てるか、モニターからユーザーを削除できます。

詳細設定

Deadline Cloud のセットアップには、追加の設定が含まれています。これらの設定を使用すると、Deadline Cloud のセットアップによって に加えられたすべての変更を表示したり AWS アカウント、モニターユーザーロールを設定したり、暗号化キータイプを変更したりできます。

AWS IAM Identity Center

AWS IAM Identity Center は、ユーザーとグループを管理するためのクラウドベースのシングルサインオンサービスです。IAM Identity Center をエンタープライズシングルサインオン (SSO) プロバイダーと統合して、ユーザーが会社のアカウントでサインインできるようにすることも可能です。

Deadline Cloud はデフォルトで IAM Identity Center を有効にし、Deadline Cloud をセットアップして使用する必要があります。Deadline Cloud に使用する IAM Identity Center インスタンスは、モニター AWS リージョン と同じ にある必要があります。詳細については、「 とは AWS IAM Identity Center」を参照してください。

サービスアクセスロールを設定する

AWS サービスは、ユーザーに代わってアクションを実行するサービスロールを引き受けることができます。Deadline Cloud では、モニター内のリソースへのアクセス権をユーザーに付与するために、モニターユーザーロールが必要です。

AWS Identity and Access Management (IAM) 管理ポリシーをモニターユーザーロールにアタッチできます。このポリシーにより、ユーザーは特定の Deadline Cloud アプリケーションでジョブを作成

するなど、特定のアクションを実行できます。アプリケーションは管理ポリシーの特定の条件に依存するため、管理ポリシーを使用しないと、アプリケーションが期待どおりに動作しない可能性があります。

モニターユーザーロールは、セットアップ完了後にいつでも変更できます。ユーザーロールの詳細については、「IAM ロール」を参照してください。

以下のタブには、2 つの異なるユースケースの説明が含まれています。新しいサービスロールを作成して使用するには、[新しいサービスロール] タブを選択します。既存のサービスロールを使用するには、[既存のサービスロール] タブを選択します。

New service role

新しいサービスロールを作成して使用するには

1. [新しいサービスロールを作成し使用する] を選択します。
2. (オプション) サービスユーザーロール名を入力します。
3. ロールの詳細については、[許可の詳細を表示] を選択します。

Existing service role

既存のサービスロールを使用するには

1. [既存のサービスロールを使用する] を選択します。
2. ドロップダウンリストを開いて既存のサービスロールを選択します。
3. (オプション) ロールの詳細については、[IAM コンソールで表示] を選択してください。

ステップ 2: ファームの詳細を定義する

Deadline Cloud コンソールに戻り、次のステップを実行してファームの詳細を定義します。

1. Farm の詳細 で、ファームの名前を追加します。
2. 説明 にファームの説明を入力します。わかりやすい説明は、ファームの目的をすばやく特定するのに役立ちます。
3. （オプション）デフォルトでは、データはセキュリティのために が AWS 所有および管理するキーで暗号化されます。暗号化設定をカスタマイズ (アドバンスト) を選択して、既存のキーを使用するか、管理する新しいキーを作成できます。

チェックボックスを使用して暗号化設定をカスタマイズする場合は、ARN AWS KMS を入力するか、新しい KMS キーの作成 AWS KMS を選択して新しい を作成します。

4. （オプション）新しいタグを追加 を選択して、ファームに 1 つ以上のタグを追加します。
5. 以下のオプションのいずれかを選択します。
   - 「スキップ」を選択してレビューし、「作成」を選択してファーム「」を確認して作成します。
   - 次へ を選択して、追加のオプションステップに進みます。

(オプション) ステップ 3: キューの詳細を定義する

キューは、ジョブの進行状況を追跡し、作業をスケジュールします。

1. キューの詳細から、キューの名前を指定します。
2. 説明 に、キューの説明を入力します。明確な説明は、キューの目的をすばやく特定するのに役立ちます。
3. ジョブアタッチメント では、新しい Amazon S3 バケットを作成するか、既存の Amazon S3 バケットを選択できます。既存の Amazon S3 バケットがない場合は、バケットを作成する必要があります。

   a. 新しい Amazon S3 バケットを作成するには、新しいジョブバケットの作成 を選択します。ルートプレフィックスフィールドでジョブバケットの名前を定義できます。バケットを呼び出すことをお勧めします**deadlinecloud-job-attachments-[MONITORNAME]**。

   小文字とダッシュのみを使用できます。スペースや特殊文字は使用できません。

   b. 既存の Amazon S3 バケットを検索して選択するには、既存の Amazon S3 バケットから選択 を選択します。次に、S3 を参照 を選択して既存のバケットを検索します。使用可能な Amazon S3 バケットのリストが表示されたら、キューに使用する Amazon S3 バケットを選択します。
4. カスタマーマネージドフリートを使用している場合は、カスタマーマネージドフリートとの関連付けを有効にする を選択します。
   - カスタマーマネージドフリートの場合は、キュー設定のユーザー を追加し、POSIX および/または Windows 認証情報を設定します。または、チェックボックスをオンにして run-as 機能をバイパスすることもできます。

5. キューには、ユーザーに代わって Amazon S3 にアクセスするためのアクセス許可が必要です。キューごとに新しいサービスロールを作成することをお勧めします。

    a. 新しいロールの場合は、次のステップを実行します。

        i. [新しいサービスロールを作成し使用する] を選択します。

        ii. キューロールのロール名を入力するか、指定されたロール名を使用します。

        iii. （オプション）キューロールの説明 を追加します。

        iv. キューロールの IAM アクセス許可を表示するには、アクセス許可の詳細を表示 を選択します。

    b. または、既存のサービスロールを選択することもできます。

6. （オプション）名前と値のペアを使用して、キュー環境の環境変数を追加します。

7. （オプション）キーと値のペアを使用してキューのタグを追加します。

キューの詳細をすべて入力したら、次へ を選択します。

(オプション) ステップ 4: フリートの詳細を定義する

フリートは、レンダリングタスクを実行するワーカーを割り当てます。レンダリングタスクにフリートが必要な場合は、「フリートの作成」のチェックボックスをオンにします。

1. フリートの詳細

    a. フリートの名前とオプションの説明の両方を指定します。

    b. コンピューティングリソースのスケーリング方法を選択します。サービスマネージドオプションを使用すると、Deadline Cloud はコンピューティングリソースを自動スケーリングできます。カスタマー管理オプションを使用すると、独自のコンピューティングスケーリングを制御できます。

2. インスタンスオプションセクションで、スポットまたはオンデマンドの を選択します。Amazon EC2 オンデマンドインスタンスは可用性を向上させ、Amazon EC2 スポットインスタンスはコスト削減の取り組みに適しています。

3. フリート内のインスタンス数を自動スケーリングするには、最小インスタンス数と最大インスタンス数の両方を選択します。

    追加コストが発生しないように、常にインスタンスの最小数`0`を に設定することを強くお勧めします。

4. フリートには、 CloudWatch ユーザーに代わって に書き込むためのアクセス許可が必要です。フリートごとに新しいサービスロールを作成することをお勧めします。

    a. 新しいロールの場合は、次のステップを実行します。

        i. [新しいサービスロールを作成し使用する] を選択します。

        ii. フリートロールのロール名を入力するか、指定されたロール名を使用します。

        iii. （オプション） フリートロールの説明 を追加します。

        iv. フリートロールの IAM アクセス許可を表示するには、アクセス許可の詳細を表示 を選択します。

    b. または、既存のサービスロールを使用することもできます。

5. （オプション）キーと値のペアを使用してフリートのタグを追加します。

すべてのフリートの詳細を入力したら、次へ を選択します。

(オプション) ステップ 5: ワーカーの要件を設定する

ワーカーインスタンスの要件を定義します。

1. オペレーティングシステム (OS) と CPU アーキテクチャの設定を確認して認識してください。
2. ハードウェア要件に合わせて vCPUs の最小数と最大数を更新します。
3. ハードウェア要件に合わせてメモリの最小数と最大数 (GiB) を更新します。
4. ワーカーインスタンスのタイプを許可または除外することで、インスタンスタイプをフィルタリングできます。どちらのフィルタリングオプションでも、最大 10 個の Amazon EC2 インスタンスタイプをフィルタリングできます。
5. 追加要件 (オプション） では、サイズ (GiB)、IOPS、スループット (MiB /s) でルート EBS ボリュームを定義できます。
6. すべてのワーカー要件を設定したら、次へ を選択してグループのアクセスレベルを定義します。

(オプション) ステップ 6: アクセスレベルを定義する

モニターに接続されているグループがある場合は、そのアクセスレベルを定義できます。Deadline Cloud の機能を使用するアクセス許可は、アクセスレベルによって管理されます。ユーザーのグループに異なるアクセスレベルを割り当てることができます。

1. Deadline Cloud ファームのアクセスレベルメニューを使用して、グループのアクセス許可のレベルを選択します。
2. 次へ を選択して続行し、入力したすべてのファームの詳細を確認します。

ステップ 7: 確認して作成する

入力されたすべての情報を確認してファームを作成します。準備ができたら、ファームの作成 を選択します。

ファームの作成の進行状況が Farms ページに表示されます。ファームが使用可能になると、成功メッセージが表示されます。

Deadline Cloud 用のデベロッパーワークステーションのセットアップ

このチュートリアルでは、 を使用して AWS CloudShell シンプルなデベロッパーファームを作成し、ワーカーエージェントを実行します。その後、パラメータとアタッチメントを使用してシンプルなジョブを送信して実行し、サービスマネージドフリートを追加し、完了したらファームリソースをクリーンアップできます。

以下のセクションでは、Deadline Cloud のさまざまな機能と、それらがどのように機能し、連携するかについて説明します。これらのステップに従うことは、新しいワークロードとカスタマイズの開発とテストに役立ちます。

トピック

ステップ 1: Deadline Cloud ファームを作成する

AWS Deadline Cloud でデベロッパーファームとキューリソースを作成するには、次の手順に示すように AWS Command Line Interface（AWS CLI) を使用します。また、 AWS Identity and Access

Management (IAM) ロールとカスタマーマネージドフリート (CMF) を作成し、フリートをキューに関連付けます。その後、 を設定し AWS CLI 、ファームが指定されたとおりに設定され、動作していることを確認します。

このファームを使用して Deadline Cloud の機能を調べ、新しいワークロード、カスタマイズ、パイプライン統合を開発およびテストできます。

ファームを作成するには

1. まだインストールしていない場合は、 AWS Command Line Interface （ AWS CLI) をインストールして設定します。詳細については、[「 の最新バージョンをインストールまたは更新する AWS CLI」](#)を参照してください。
2. ファームの名前を作成し、そのファーム名を に追加します`~/.bashrc`。これにより、他のターミナルセッションで使用できるようになります。

```
echo "DEV_FARM_NAME=DeveloperFarm" >> ~/.bashrc
source ~/.bashrc
```

3. ファームリソースを作成し、そのファーム ID を に追加します`~/.bashrc`。

```
aws deadline create-farm \
    --display-name "$DEV_FARM_NAME"

echo "DEV_FARM_ID=\$(aws deadline list-farms \
        --query \"farms[?displayName=='\$DEV_FARM_NAME'].farmId \
        | [0]\" --output text)" >> ~/.bashrc
source ~/.bashrc
```

4. キューリソースを作成し、そのキュー ID を に追加します。 `~/.bashrc`.

```
aws deadline create-queue \
    --farm-id $DEV_FARM_ID \
    --display-name "$DEV_FARM_NAME Queue" \
    --job-run-as-user '{"posix": {"user": "job-user", "group": "job-group"},
 "runAs":"QUEUE_CONFIGURED_USER"}'

echo "DEV_QUEUE_ID=\$(aws deadline list-queues \
        --farm-id \$DEV_FARM_ID \
        --query \"queues[?displayName=='\$DEV_FARM_NAME Queue'].queueId \
        | [0]\" --output text)" >> ~/.bashrc
source ~/.bashrc
```

5. フリートの IAM ロールを作成します。このロールは、フリートのワーカーホストに、キューからジョブを実行するために必要なセキュリティ認証情報を提供します。

```
aws iam create-role \
    --role-name "${DEV_FARM_NAME}FleetRole" \
    --assume-role-policy-document \
        '{
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "credentials.deadline.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }'
aws iam put-role-policy \
    --role-name "${DEV_FARM_NAME}FleetRole" \
    --policy-name WorkerPermissions \
    --policy-document \
        '{
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": [
                        "deadline:AssumeFleetRoleForWorker",
                        "deadline:UpdateWorker",
                        "deadline:DeleteWorker",
                        "deadline:UpdateWorkerSchedule",
                        "deadline:BatchGetJobEntity",
                        "deadline:AssumeQueueRoleForWorker"
                    ],
                    "Resource": "*",
                    "Condition": {
                        "StringEquals": {
                            "aws:PrincipalAccount": "${aws:ResourceAccount}"
                        }
                    }
                },
                {
```

```
                "Effect": "Allow",
                "Action": [
                    "logs:CreateLogStream"
                ],
                "Resource": "arn:aws:logs:*:*:*:/aws/deadline/*",
                "Condition": {
                    "StringEquals": {
                        "aws:PrincipalAccount": "${aws:ResourceAccount}"
                    }
                }
            },
            {
                "Effect": "Allow",
                "Action": [
                    "logs:PutLogEvents",
                    "logs:GetLogEvents"
                ],
                "Resource": "arn:aws:logs:*:*:*:/aws/deadline/*",
                "Condition": {
                    "StringEquals": {
                        "aws:PrincipalAccount": "${aws:ResourceAccount}"
                    }
                }
            }
        ]
    }'
```

6. カスタマーマネージドフリート (CMF) を作成し、そのフリート ID を に追加します~/.bashrc。

```
FLEET_ROLE_ARN="arn:aws:iam::$(aws sts get-caller-identity \
        --query "Account" --output text):role/${DEV_FARM_NAME}FleetRole"
aws deadline create-fleet \
    --farm-id $DEV_FARM_ID \
    --display-name "$DEV_FARM_NAME CMF" \
    --role-arn $FLEET_ROLE_ARN \
    --max-worker-count 5 \
    --configuration \
        '{
            "customerManaged": {
                "mode": "NO_SCALING",
                "workerCapabilities": {
                    "vCpuCount": {"min": 1},
```

```
                    "memoryMiB": {"min": 512},
                    "osFamily": "linux",
                    "cpuArchitectureType": "x86_64"
                }
            }
        }'

echo "DEV_CMF_ID=\$(aws deadline list-fleets \
        --farm-id \$DEV_FARM_ID \
        --query \"fleets[?displayName=='\$DEV_FARM_NAME CMF'].fleetId \
        | [0]\" --output text)" >> ~/.bashrc
source ~/.bashrc
```

7. Deadline Cloud にアクセスできることを確認します。

```
pip install deadline
```

8. CMF をキューに関連付けます。

```
aws deadline create-queue-fleet-association \
    --farm-id $DEV_FARM_ID \
    --queue-id $DEV_QUEUE_ID \
    --fleet-id $DEV_CMF_ID
```

9. デフォルトのファームをファーム ID に設定し、キューを以前に作成したキュー ID に設定するには、次のコマンドを使用します。

```
deadline config set defaults.farm_id $DEV_FARM_ID
deadline config set defaults.queue_id $DEV_QUEUE_ID
```

10. (オプション) ファームが仕様に従って設定されていることを確認するには、次のコマンドを使用します。

   - すべてのファームを一覧表示する – **`deadline farm list`**
   - デフォルトファーム内のすべてのキューを一覧表示する – **`deadline queue list`**
   - デフォルトファーム内のすべてのフリートを一覧表示する – **`deadline fleet list`**
   - デフォルトのファームを取得する – **`deadline farm get`**
   - デフォルトのキューを取得する – **`deadline queue get`**
   - デフォルトキューに関連付けられているすべてのフリートを取得する – **`deadline fleet get`**

ステップ 2: Deadline Cloud で開発者モードでワーカーエージェントを実行する

デベロッパーファームのキューに送信するジョブを実行する前に、ワーカーホストでデベロッパーモードで AWS Deadline Cloud ワーカーエージェントを実行する必要があります。

このチュートリアルの残りの部分では、2 つの AWS CloudShell タブを使用してデベロッパーファームでオペレーションを実行します AWS CLI 。最初のタブでは、ジョブを送信できます。2 番目のタブでは、ワーカーエージェントを実行できます。

> **Note**
>
> CloudShell セッションを 20 分以上アイドル状態のままにすると、ワーカーエージェントはタイムアウトして停止します。ワーカーエージェントを再起動するには、以下の手順に従います。

ワーカーエージェントをデベロッパーモードで実行するには

1. まだインストールしていない場合は、 AWS Command Line Interface （ AWS CLI) をインストールして設定します。詳細については、「 の最新バージョンをインストールまたは更新する AWS CLI」を参照してください。
2. ファームを最初の CloudShell タブで開いたままにして、2 番目の CloudShell タブを開き、demoenv-logsおよび demoenv-persist ディレクトリを作成します。

```
mkdir ~/demoenv-logs
mkdir ~/demoenv-persist
```

3. PyPI から Deadline Cloud ワーカーエージェントパッケージをダウンロードしてインストールします。

> **Note**
>
> ではWindows、エージェントファイルが Python のグローバル site-packages ディレクトリにインストールされている必要があります。Python 仮想環境は現在サポートされていません。

```
python -m pip install deadline-cloud-worker-agent
```

4. ワーカーエージェントが実行中のジョブの一時ディレクトリを作成できるようにするには、ディレクトリを作成します。

```
sudo mkdir /sessions
sudo chmod 750 /sessions
sudo chown cloudshell-user /sessions
```

5. Deadline Cloud ワーカーエージェントを、 に追加DEV_CMF_IDした変数 DEV_FARM_IDと を使用してデベロッパーモードで実行します~/.bashrc。

```
deadline-worker-agent \
    --farm-id $DEV_FARM_ID \
    --fleet-id $DEV_CMF_ID \
    --run-jobs-as-agent-user \
    --logs-dir ~/demoenv-logs \
    --persistence-dir ~/demoenv-persist
```

ワーカーエージェントが初期化して UpdateWorkerSchedule API オペレーションをポーリングすると、次の出力が表示されます。

```
INFO    Worker Agent starting
[2024-03-27 15:51:01,292][INFO    ] # Worker Agent starting
[2024-03-27 15:51:01,292][INFO    ] AgentInfo
Python Interpreter: /usr/bin/python3
Python Version: 3.9.16 (main, Sep  8 2023, 00:00:00)  - [GCC 11.4.1 20230605 (Red
 Hat 11.4.1-2)]
Platform: linux
...
[2024-03-27 15:51:02,528][INFO    ] # API.Resp # [deadline:UpdateWorkerSchedule]
(200) params={'assignedSessions': {}, 'cancelSessionActions': {},
 'updateIntervalSeconds': 15} ...
[2024-03-27 15:51:17,635][INFO    ] # API.Resp # [deadline:UpdateWorkerSchedule]
(200) params=(Duplicate removed, see previous response) ...
[2024-03-27 15:51:32,756][INFO    ] # API.Resp # [deadline:UpdateWorkerSchedule]
(200) params=(Duplicate removed, see previous response) ...
...
```

6. 最初の CloudShell タブを選択し、フリート内のワーカーを一覧表示します。

```
deadline worker list --fleet-id $DEV_CMF_ID
```

次のような出力が表示されます。

```
Displaying 1 of 1 workers starting at 0

- workerId: worker-8c9af877c8734e89914047111f
  status: STARTED
  createdAt: 2023-12-13 20:43:06+00:00
```

本番稼働用設定では、Deadline Cloud ワーカーエージェントは、ホストマシン上の管理ユーザーとして複数のユーザーと設定ディレクトリを設定する必要があります。これらの設定を上書きできます。これは、自分だけがアクセスできる独自の開発ファームでジョブを実行しているためです。

ステップ 3: Deadline Cloud でジョブを送信して実行する

AWS Deadline Cloud を使用してジョブを実行するには、次の手順を使用します。最初の AWS CloudShell タブを使用して、デベロッパーファームにジョブを送信します。2 番目の CloudShell タブを使用して、ワーカーエージェントの出力を表示します。

トピック

- simple_job サンプルを送信する
- パラメータsimple_jobを使用して を送信する
- ファイル I/O を使用して simple_file_job ジョブバンドルを作成する

simple_job サンプルを送信する

ファームを作成してワーカーエージェントを実行したら、simple_jobサンプルを Deadline Cloud に送信できます。

Deadline Cloud にsimple_jobサンプルを送信するには

1. まだインストールしていない場合は、 AWS Command Line Interface（ AWS CLI) をインストールして設定します。詳細については、「 の最新バージョンをインストールまたは更新する AWS CLI」を参照してください。
2. からサンプルをダウンロードします GitHub。

```
cd ~
git clone https://github.com/aws-deadline/deadline-cloud-samples.git
```

3. 最初の CloudShell タブを選択し、ジョブバンドルサンプルディレクトリに移動します。

```
cd ~/deadline-cloud-samples/job_bundles/
```

4. simple_job サンプルを送信します。

```
deadline bundle submit simple_job
```

5. 2 番目の CloudShell タブを選択すると、 の呼び出しBatchGetJobEntities、セッションの取得、およびセッションアクションの実行に関するログ記録出力が表示されます。

```
...
[2024-03-27 16:00:21,846][INFO    ] # Session.Starting
 # [session-053d77cef82648fe2] Starting new Session.
 [queue-3ba4ff683ff54db09b851a2ed8327d7b/job-d34cc98a6e234b6f82577940ab4f76c6]
[2024-03-27 16:00:21,853][INFO    ] # API.Req # [deadline:BatchGetJobEntity]
 resource={'farm-id': 'farm-3e24cfc9bbcd423e9c1b6754bc1',
 'fleet-id': 'fleet-246ee60f46d44559b6cce010d05', 'worker-id':
 'worker-75e0fce9c3c344a69bff57fcd83'} params={'identifiers': [{'jobDetails':
 {'jobId': 'job-d34cc98a6e234b6f82577940ab4'}}]} request_url=https://
scheduling.deadline.us-west-2.amazonaws.com/2023-10-12/farms/
farm-3e24cfc9bbcd423e /fleets/fleet-246ee60f46d44559b1 /workers/worker-
 75e0fce9c3c344a69b /batchGetJobEntity
[2024-03-27 16:00:22,013][INFO    ] # API.Resp # [deadline:BatchGetJobEntity](200)
 params={'entities': [{'jobDetails': {'jobId': 'job-d34cc98a6e234b6f82577940ab6',
 'jobRunAsUser': {'posix': {'user': 'job-user', 'group': 'job-group'},
 'runAs': 'QUEUE_CONFIGURED_USER'}, 'logGroupName': '/aws/deadline/
farm-3e24cfc9bbcd423e9c1b6754bc1/queue-3ba4ff683ff54db09b851a2ed83', 'parameters':
 '*REDACTED*', 'schemaVersion': 'jobtemplate-2023-09'}}], 'errors': []}
 request_id=a3f55914-6470-439e-89e5-313f0c6
[2024-03-27 16:00:22,013][INFO    ] # Session.Add #
 [session-053d77cef82648fea9c69827182] Appended new SessionActions.
 (ActionIds: ['sessionaction-053d77cef82648fea9c69827182-0'])
 [queue-3ba4ff683ff54db09b851a2ed8b/job-d34cc98a6e234b6f82577940ab6]
[2024-03-27 16:00:22,014][WARNING ] # Session.User #
 [session-053d77cef82648fea9c69827182] Running as the Worker Agent's
 user. (User: cloudshell-user) [queue-3ba4ff683ff54db09b851a2ed8b/job-
d34cc98a6e234b6f82577940ac6]
```

```
[2024-03-27 16:00:22,015][WARNING ] # Session.AWSCreds #
 [session-053d77cef82648fea9c69827182] AWS Credentials are not available: Queue has
 no IAM Role. [queue-3ba4ff683ff54db09b851a2ed8b/job-d34cc98a6e234b6f82577940ab6]
[2024-03-27 16:00:22,026][INFO    ] # Session.Logs #
 [session-053d77cef82648fea9c69827182] Logs streamed to: AWS CloudWatch
 Logs. (LogDestination: /aws/deadline/farm-3e24cfc9bbcd423e9c1b6754bc1/
queue-3ba4ff683ff54db09b851a2ed83/session-053d77cef82648fea9c69827181)
 [queue-3ba4ff683ff54db09b851a2ed83/job-d34cc98a6e234b6f82577940ab4]
[2024-03-27 16:00:22,026][INFO    ] # Session.Logs #
 [session-053d77cef82648fea9c69827182] Logs streamed to: local
 file. (LogDestination: /home/cloudshell-user/demoenv-logs/
queue-3ba4ff683ff54db09b851a2ed8b/session-053d77cef82648fea9c69827182.log)
 [queue-3ba4ff683ff54db09b851a2ed83/job-d34cc98a6e234b6f82577940ab4]
...
```

> **Note**
>
> ワーカーエージェントからのログ記録出力のみが表示されます。ジョブを実行するセッションには別のログがあります。

6. 最初のタブを選択し、ワーカーエージェントが書き込むログファイルを検査します。

   a. ワーカーエージェントのログディレクトリに移動し、その内容を表示します。

   ```
   cd ~/demoenv-logs
   ls
   ```

   b. ワーカーエージェントが作成した最初のログファイルを出力します。

   ```
   cat worker-agent-bootstrap.log
   ```

   このファイルには、Deadline Cloud API を呼び出してフリートにワーカーリソースを作成し、フリートロールを引き受けた方法に関するワーカーエージェントの出力が含まれています。

   c. ワーカーエージェントがフリートに参加するときに、ログファイルの出力を出力します。

   ```
   cat worker-agent.log
   ```

このログには、ワーカーエージェントが実行するすべてのアクションに関する出力が含まれますが、それらのリソースの IDs を除き、ジョブを実行するキューに関する出力は含まれません。

d. キューリソース ID と同じ名前のディレクトリに、各セッションのログファイルを出力します。

```
cat $DEV_QUEUE_ID/session-*.log
```

ジョブが成功すると、ログファイルの出力は次のようになります。

```
 cat $DEV_QUEUE_ID/$(ls -t $DEV_QUEUE_ID | head -1)
2024-03-27 16:00:22,026 WARNING Session running with no AWS Credentials.
2024-03-27 16:00:22,404 INFO
2024-03-27 16:00:22,405 INFO ==============================================
2024-03-27 16:00:22,405 INFO --------- Running Task
2024-03-27 16:00:22,405 INFO ==============================================
2024-03-27 16:00:22,406 INFO ----------------------------------------------
2024-03-27 16:00:22,406 INFO Phase: Setup
2024-03-27 16:00:22,406 INFO ----------------------------------------------
2024-03-27 16:00:22,406 INFO Writing embedded files for Task to disk.
2024-03-27 16:00:22,406 INFO Mapping: Task.File.runScript -> /sessions/
session-053d77cef82648fea9c698271812a/embedded_fileswa_gj55_/tmp2u9yqtsz
2024-03-27 16:00:22,406 INFO Wrote: runScript -> /sessions/
session-053d77cef82648fea9c698271812a/embedded_fileswa_gj55_/tmp2u9yqtsz
2024-03-27 16:00:22,407 INFO ----------------------------------------------
2024-03-27 16:00:22,407 INFO Phase: Running action
2024-03-27 16:00:22,407 INFO ----------------------------------------------
2024-03-27 16:00:22,407 INFO Running command /sessions/
session-053d77cef82648fea9c698271812a/tmpzuzxpslm.sh
2024-03-27 16:00:22,414 INFO Command started as pid: 471
2024-03-27 16:00:22,415 INFO Output:
2024-03-27 16:00:22,420 INFO Welcome to AWS Deadline Cloud!
2024-03-27 16:00:22,571 INFO
2024-03-27 16:00:22,572 INFO ==============================================
2024-03-27 16:00:22,572 INFO --------- Session Cleanup
2024-03-27 16:00:22,572 INFO ==============================================
2024-03-27 16:00:22,572 INFO Deleting working directory: /sessions/
session-053d77cef82648fea9c698271812a
```

7. ジョブに関する情報を出力します。

```
deadline job get
```

ジョブを送信すると、システムはそれをデフォルトとして保存するため、ジョブ ID を入力する必要はありません。

パラメータsimple_jobを使用して を送信する

パラメータを使用してジョブを送信できます。次の手順では、simple_jobテンプレートを編集してカスタムメッセージを含め、 を送信しsimple_job、セッションログファイルを出力してメッセージを表示します。

パラメータを使用してsimple_jobサンプルを送信するには

1. 最初の CloudShell タブを選択し、ジョブバンドルのサンプルディレクトリに移動します。

```
cd ~/deadline-cloud-samples/job_bundles/
```

2. simple_job テンプレートの内容を出力します。

```
cat simple_job/template.yaml
```

`Message` パラメータを含む `parameterDefinitions`セクションは次のようになります。

```
parameterDefinitions:
- name: Message
  type: STRING
  default: Welcome to AWS Deadline Cloud!
```

3. パラメータ値を使用してsimple_jobサンプルを送信し、ジョブの実行が完了するまで待ちます。

```
deadline bundle submit simple_job \
    -p "Message=Greetings from the developer getting started guide."
```

4. カスタムメッセージを表示するには、最新のセッションログファイルを表示します。

```
cd ~/demoenv-logs
cat $DEV_QUEUE_ID/$(ls -t $DEV_QUEUE_ID | head -1)
```

ファイル I/O を使用して simple_file_job ジョブバンドルを作成する

レンダリングジョブは、シーン定義を読み取ってからイメージをレンダリングし、そのイメージを出力ファイルに保存する必要があります。このアクションをシミュレートするには、イメージをレンダリングする代わりに、ジョブに入力のハッシュを計算します。

ファイル I/O を使用して simple_file_job ジョブバンドルを作成するには

1. 最初の CloudShell タブを選択し、ジョブバンドルのサンプルディレクトリに移動します。

```
cd ~/deadline-cloud-samples/job_bundles/
```

2. 新しい名前`simple_job`で のコピーを作成します`simple_file_job`。

```
cp -r simple_job simple_file_job
```

3. ジョブテンプレートを次のように編集します。

> **Note**
>
> これらのステップnanoでは、 を使用することをお勧めします。を使用する場合はVim、を使用して貼り付けモードを設定する必要があります:`set paste`。

   a. テキストエディタでテンプレートを開きます。

   ```
   nano simple_file_job/template.yaml
   ```

   b. 次の `type`、`objectType`、および `dataFlow` を追加します`parameterDefinitions`。

   ```
   - name: InFile
     type: PATH
     objectType: FILE
     dataFlow: IN
   - name: OutFile
     type: PATH
     objectType: FILE
     dataFlow: OUT
   ```

   c. 入力ファイルから読み取り、出力ファイルに書き込むファイルの末尾に、次のbashスクリプトコマンドを追加します。

```
        # hash the input file, and write that to the output
        sha256sum "{{Param.InFile}}" > "{{Param.OutFile}}"
```

更新された template.yamlは、以下と完全に一致する必要があります。

```
specificationVersion: 'jobtemplate-2023-09'
name: Simple File Job Bundle Example
parameterDefinitions:
- name: Message
  type: STRING
  default: Welcome to AWS Deadline Cloud!
- name: InFile
  type: PATH
  objectType: FILE
  dataFlow: IN
- name: OutFile
  type: PATH
  objectType: FILE
  dataFlow: OUT
steps:
- name: WelcomeToDeadlineCloud
  script:
    actions:
      onRun:
        command: '{{Task.File.runScript}}'
    embeddedFiles:
    - name: runScript
      type: TEXT
      runnable: true
      data: |
        #!/usr/bin/env bash
        echo "{{Param.Message}}"

        # hash the input file, and write that to the output
        sha256sum "{{Param.InFile}}" > "{{Param.OutFile}}"
```

Note

の間隔を調整する場合はtemplate.yaml、必ずインデントではなくスペースを使用してください。

d. ファイルを保存し、テキストエディタを終了します。

4. simple_file_job を送信する入出力ファイルのパラメータ値を指定します。

```
deadline bundle submit simple_file_job \
    -p "InFile=simple_job/template.yaml" \
    -p "OutFile=hash.txt"
```

5. ジョブに関する情報を出力します。

```
deadline job get
```

- 次のような出力が表示されます。

```
parameters:
  Message:
    string: Welcome to AWS Deadline Cloud!
  InFile:
    path: /local/home/cloudshell-user/BundleFiles/JobBundle-Examples/simple_job/
template.yaml
  OutFile:
    path: /local/home/cloudshell-user/BundleFiles/JobBundle-Examples/hash.txt
```

- 相対パスのみを指定しましたが、パラメータにはフルパスが設定されています。は、現在の作業ディレクトリを、パスのタイプが の場合にパラメータとして指定されたパス AWS CLI に結合しますPATH。
- 他のターミナルウィンドウで実行されているワーカーエージェントは、ジョブを取得し、実行します。このアクションにより hash.txt ファイルが作成され、次のコマンドで表示できます。

```
cat hash.txt
```

このコマンドは、次のような出力を出力します。

```
eaa2df5d34b54be5ac34c56a24a8c237b8487231a607eaf530a04d76b89c9cd3 /local/home/
cloudshell-user/BundleFiles/JobBundle-Examples/simple_job/template.yaml
```

ステップ 4: Deadline Cloud でジョブアタッチメントを使用してジョブを実行する

多くのファームは、共有ファイルシステムを使用して、ジョブを送信するホストとジョブを実行するホスト間でファイルを共有します。例えば、前の`simple_file_job`例では、ローカルファイルシステムはターミナルウィンドウ間で共有されます。 AWS CloudShell ターミナルウィンドウは、ジョブを送信するタブ 1 とワーカーエージェントを実行するタブ 2 で実行されます。

共有ファイルシステムは、送信者ワークステーションとワーカーホストが同じローカルエリアネットワーク上にある場合に便利です。それにアクセスするワークステーションの近くでオンプレミスにデータを保存する場合、クラウドベースのファームを使用すると、高レイテンシー VPN でファイルシステムを共有するか、クラウドでファイルシステムを同期する必要があります。これらのオプションはいずれも、設定や運用が容易ではありません。

AWS Deadline Cloud は、E メールの添付ファイルに似たジョブの添付ファイル を含むシンプルなソリューションを提供します。ジョブアタッチメントでは、データをジョブにアタッチします。その後、Deadline Cloud は、ジョブデータの転送と Amazon Simple Storage Service (Amazon S3) バケットへの保存の詳細を処理します。

コンテンツ作成ワークフローは、多くの場合反復的です。つまり、ユーザーは変更されたファイルの小さなサブセットを使用してジョブを送信します。Amazon S3 バケットはジョブアタッチメントをコンテンツアドレス可能なストレージに保存するため、各オブジェクトの名前はオブジェクトのデータのハッシュに基づいており、ディレクトリツリーの内容はジョブにアタッチされたマニフェストファイル形式で保存されます。

ジョブアタッチメントを使用してジョブを実行するには、次のステップを実行します。

トピック

ジョブアタッチメント設定をキューに追加する

キューでジョブアタッチメントを有効にするには、アカウントのキューリソースにジョブアタッチメント設定を追加します。

ジョブアタッチメント設定をキューに追加するには

1. まだインストールしていない場合は、 AWS Command Line Interface （ AWS CLI) をインストールして設定します。詳細については、「 の最新バージョンをインストールまたは更新する AWS CLI」を参照してください。
2. 最初の CloudShell タブを選択し、次のいずれかのコマンドを入力して、ジョブアタッチメントに Amazon S3 バケットを使用します。
   - 既存のプライベート Amazon S3 バケットがない場合は、新しい S3 バケットを作成して使用できます。

```
DEV_FARM_BUCKET=$(echo $DEV_FARM_NAME \
    | tr '[:upper:]' '[:lower:]')-$(xxd -l 16 -p /dev/urandom)
if [ "$AWS_REGION" == "us-east-1" ]; then LOCATION_CONSTRAINT=
else LOCATION_CONSTRAINT="--create-bucket-configuration \
    LocationConstraint=${AWS_REGION}"
fi
aws s3api create-bucket \
    $LOCATION_CONSTRAINT \
    --acl private \
    --bucket ${DEV_FARM_BUCKET}
```

   - プライベート Amazon S3 バケットがすでにある場合は、 をバケットの名前*MY_BUCKET_NAME*に置き換えて使用できます。

```
DEV_FARM_BUCKET=MY_BUCKET_NAME
```

3. Amazon S3 バケットを作成または選択したら、バケット名を に追加~/.bashrcして、バケットを他のターミナルセッションで使用できるようにします。

```
echo "DEV_FARM_BUCKET=$DEV_FARM_BUCKET" >> ~/.bashrc
```

4. キューの AWS Identity and Access Management (IAM) ロールを作成します。

```
aws iam create-role --role-name "${DEV_FARM_NAME}QueueRole" \
    --assume-role-policy-document \
        '{
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
```

```
                        "Service": "credentials.deadline.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }'
aws iam put-role-policy \
    --role-name "${DEV_FARM_NAME}QueueRole" \
    --policy-name S3BucketsAccess \
    --policy-document \
            '{
                "Version": "2012-10-17",
                "Statement": [
                {
                    "Action": [
                        "s3:GetObject*",
                        "s3:GetBucket*",
                        "s3:List*",
                        "s3:DeleteObject*",
                        "s3:PutObject",
                        "s3:PutObjectLegalHold",
                        "s3:PutObjectRetention",
                        "s3:PutObjectTagging",
                        "s3:PutObjectVersionTagging",
                        "s3:Abort*"
                    ],
                    "Resource": [
                        "arn:aws:s3:::'$DEV_FARM_BUCKET'",
                        "arn:aws:s3:::'$DEV_FARM_BUCKET'/*"
                    ],
                    "Effect": "Allow"
                }
            ]
            }'
```

5. キューを更新して、ジョブアタッチメント設定と IAM ロールを含めます。

```
QUEUE_ROLE_ARN="arn:aws:iam::$(aws sts get-caller-identity \
        --query "Account" --output text):role/${DEV_FARM_NAME}QueueRole"
aws deadline update-queue \
    --farm-id $DEV_FARM_ID \
    --queue-id $DEV_QUEUE_ID \
    --role-arn $QUEUE_ROLE_ARN \
    --job-attachment-settings \
```

```
        '{
            "s3BucketName": "'$DEV_FARM_BUCKET'",
            "rootPrefix": "JobAttachments"
        }'
```

6. キューを更新したことを確認します。

   ```
   deadline queue get
   ```

   次のような出力が表示されます。

   ```
   ...
   jobAttachmentSettings:
     s3BucketName: DEV_FARM_BUCKET
     rootPrefix: JobAttachments
   roleArn: arn:aws:iam::ACCOUNT_NUMBER:role/DeveloperFarmQueueRole
   ...
   ```

ジョブアタッチメントsimple_file_jobで送信する

ジョブアタッチメントを使用する場合、ジョブバンドルは Deadline Cloud に、PATHパラメータの使用など、ジョブのデータフローを決定するのに十分な情報を提供する必要があります。の場合 simple_file_job、 `template.yaml` ファイルを編集して、データフローが入力ファイルと出力ファイルにあることを Deadline Cloud に伝えます。

ジョブアタッチメント設定をキューに追加したら、ジョブアタッチメントを含む simple_file_job サンプルを送信できます。これを行うと、ログ記録とジョブ出力を表示して、ジョブアタッチメント simple_file_jobを持つ が機能していることを確認します。

ジョブアタッチメントを使用して simple_file_job ジョブバンドルを送信するには

1. 最初の CloudShell タブを選択し、 `JobBundle-Samples` ディレクトリを開きます。
2.
   ```
   cd ~/AmazonDeadlineCloud-DocumentationAndSamples/JobBundle-Samples
   ```
3. simple_file_job をキューに送信します。アップロードを確認するプロンプトが表示されたら、と入力します**y**。

   ```
   deadline bundle submit simple_file_job \
       -p InFile=simple_job/template.yaml \
   ```

```
    -p OutFile=hash-jobattachments.txt
```

4. ジョブアタッチメントのデータ転送セッションログ出力を表示するには、2 番目の CloudShell タブを選択します。

```
JOB_ID=$(deadline config get defaults.job_id)
SESSION_ID=$(aws deadline list-sessions \
        --farm-id $DEV_FARM_ID \
        --queue-id $DEV_QUEUE_ID \
        --job-id $JOB_ID \
        --query "sessions[0].sessionId" \
        --output text)
cat ~/demoenv-logs/$DEV_QUEUE_ID/$SESSION_ID.log
```

5. セッション内で実行されたセッションアクションを一覧表示します。

```
aws deadline list-session-actions \
    --farm-id $DEV_FARM_ID \
    --queue-id $DEV_QUEUE_ID \
    --job-id $JOB_ID \
    --session-id $SESSION_ID
```

次のような出力が表示されます。

```
{
    "sessionactions": [
        {
            "sessionActionId": "sessionaction-123-0",
            "status": "SUCCEEDED",
            "startedAt": "<timestamp>",
            "endedAt": "<timestamp>",
            "progressPercent": 100.0,
            "definition": {
                "syncInputJobAttachments": {}
            }
        },
        {
            "sessionActionId": "sessionaction-123-1",
            "status": "SUCCEEDED",
            "startedAt": "<timestamp>",
            "endedAt": "<timestamp>",
            "progressPercent": 100.0,
```

```
                "definition": {
                    "taskRun": {
                        "taskId": "task-abc-0",
                        "stepId": "step-def"
                    }
                }
            }
        ]
}
```


最初のセッションアクションは入力ジョブアタッチメントをダウンロードし、2 番目のアクションは前のようにタスクを実行し、出力ジョブアタッチメントをアップロードします。

6. 出力ディレクトリを一覧表示します。

```
ls *.txt
```

などの出力`hash.txt`が表示されますが、`hash-jobattachments.txt`は存在しません。

7. 最新のジョブから出力をダウンロードします。

```
deadline job download-output
```

8. ダウンロードしたファイルの出力を表示します。

```
cat hash-jobattachments.txt
```

次のような出力が表示されます。

```
eaa2df5d34b54be5ac34c56a24a8c237b8487231a607eaf530a04d76b89c9cd3  /tmp/openjd/
session-123/assetroot-abc/simple_job/template.yaml
```

ジョブアタッチメントを Amazon S3 に保存する方法を理解する

(AWS CLI) を使用して AWS Command Line Interface 、Amazon S3 バケットに保存されているジョブアタッチメントのデータをアップロードまたはダウンロードできます。Deadline Cloud がジョブアタッチメントを Amazon S3 に保存する方法を理解することは、ワークロードとパイプラインの統合を開発する際に役立ちます。

Deadline Cloud ジョブアタッチメントが Amazon S3 にどのように保存されているかを調べるには

1. 最初の CloudShell タブを選択し、ジョブバンドルサンプルディレクトリを開きます。

```
cd ~/AmazonDeadlineCloud-DocumentationAndSamples/JobBundle-Samples
```

2. ジョブのプロパティを検査します。

```
deadline job get
```

次のような出力が表示されます。

```
parameters:
  Message:
    string: Welcome to Amazon Deadline Cloud!
  InFile:
    path: /home/cloudshell-user/AmazonDeadlineCloud-DocumentationAndSamples/
JobBundle-Samples/simple_job/template.yaml
  OutFile:
    path: /home/cloudshell-user/AmazonDeadlineCloud-DocumentationAndSamples/
JobBundle-Samples/hash-jobattachments.txt
attachments:
  manifests:
  - rootPath: /home/cloudshell-user/AmazonDeadlineCloud-DocumentationAndSamples/
JobBundle-Samples
    rootPathFormat: posix
    outputRelativeDirectories:
    - .
    inputManifestPath: farm-3040c59a5b9943d58052c29d907a645d/queue-
cde9977c9f4d4018a1d85f3e6c1a4e6e/Inputs/
f46af01ca8904cd8b514586671c79303/0d69cd94523ba617c731f29c019d16e8_input.xxh128
    inputManifestHash: f95ef91b5dab1fc1341b75637fe987ee
  fileSystem: COPIED
```

添付ファイルフィールドには、ジョブの実行時にジョブが使用する入出力データパスを記述するマニフェスト構造のリストが含まれています。ジョブを送信したマシンのローカルディレクトリパスrootPathを確認します。マニフェストファイルを含む Amazon S3 オブジェクトのサフィックスを確認するには、「」を参照してくださいinputManifestFile。マニフェストファイルには、ジョブの入力データのディレクトリツリースナップショットのメタデータが含まれています。

3. Amazon S3 マニフェストオブジェクトをプリティプリントして、ジョブの入力ディレクトリ構造を確認します。

```
MANIFEST_SUFFIX=$(aws deadline get-job \
     --farm-id $DEV_FARM_ID \
     --queue-id $DEV_QUEUE_ID \
     --job-id $JOB_ID \
     --query "attachments.manifests[0].inputManifestPath" \
     --output text)
 aws s3 cp s3://$DEV_FARM_BUCKET/JobAttachments/Manifests/$MANIFEST_SUFFIX - | jq .
```

次のような出力が表示されます。

```
{
     "hashAlg": "xxh128",
     "manifestVersion": "2023-03-03",
     "paths": [
     {
         "hash": "2ec297b04c59c4741ed97ac8fb83080c",
         "mtime": 1698186190000000,
         "path": "simple_job/template.yaml",
         "size": 445
     }
     ],
     "totalSize": 445
 }
```

4. 出力ジョブアタッチメントのマニフェストを保持する Amazon S3 プレフィックスを構築し、その下に オブジェクトを一覧表示します。

```
SESSION_ACTION=$(aws deadline list-session-actions \
    --farm-id $DEV_FARM_ID \
    --queue-id $DEV_QUEUE_ID \
    --job-id $JOB_ID \
    --session-id $SESSION_ID \
    --query "sessionActions[?definition.taskRun != null] | [0]")
STEP_ID=$(echo $SESSION_ACTION | jq -r .definition.taskRun.stepId)
TASK_ID=$(echo $SESSION_ACTION | jq -r .definition.taskRun.taskId)
TASK_OUTPUT_PREFIX=JobAttachments/Manifests/$DEV_FARM_ID/$DEV_QUEUE_ID/$JOB_ID/
$STEP_ID/$TASK_ID/
aws s3api list-objects-v2 --bucket $DEV_FARM_BUCKET --prefix $TASK_OUTPUT_PREFIX
```

出力ジョブアタッチメントは、ジョブリソースから直接参照されるのではなく、ファームリソース ID に基づいて Amazon S3 バケットに配置されます。 IDs

5. 特定のセッションアクション ID の最新マニフェストオブジェクトキーを取得し、マニフェストオブジェクトをプリティープリントします。

```
SESSION_ACTION_ID=$(echo $SESSION_ACTION | jq -r .sessionActionId)
 MANIFEST_KEY=$(aws s3api list-objects-v2 \
     --bucket $DEV_FARM_BUCKET \
     --prefix $TASK_OUTPUT_PREFIX \
     --query "Contents[*].Key" --output text \
     | grep $SESSION_ACTION_ID \
     | sort | tail -1)
 MANIFEST_OBJECT=$(aws s3 cp s3://$DEV_FARM_BUCKET/$MANIFEST_KEY -)
 echo $MANIFEST_OBJECT | jq .
```

`hash-jobattachments.txt` 出力には、次のようなファイルのプロパティが表示されます。

```
{
     "hashAlg": "xxh128",
     "manifestVersion": "2023-03-03",
     "paths": [
     {
         "hash": "f60b8e7d0fabf7214ba0b6822e82e08b",
         "mtime": 1698785252554950,
         "path": "hash-jobattachments.txt",
         "size": 182
     }
     ],
     "totalSize": 182
 }
```

ジョブにはタスク実行ごとに 1 つのマニフェストオブジェクトしかありませんが、一般的にタスク実行ごとにより多くのオブジェクトを持つことができます。

6. `Data` プレフィックスの下にコンテンツアドレス指定可能な Amazon S3 ストレージ出力を表示します。

```
 FILE_HASH=$(echo $MANIFEST_OBJECT | jq -r .paths[0].hash)
 FILE_PATH=$(echo $MANIFEST_OBJECT | jq -r .paths[0].path)
```

```
aws s3 cp s3://$DEV_FARM_BUCKET/JobAttachments/Data/$FILE_HASH -
```

次のような出力が表示されます。

```
eaa2df5d34b54be5ac34c56a24a8c237b8487231a607eaf530a04d76b89c9cd3  /tmp/openjd/
session-123/assetroot-abc/simple_job/template.yaml
```

ステップ 5: Deadline Cloud のデベロッパーファームにサービスマネージドフリートを追加する

AWS CloudShell は、大規模なワークロードをテストするのに十分なコンピューティング容量を提供しません。また、複数のワーカーホストにタスクを分散するジョブで動作するように設定されていません。

を使用する代わりに CloudShell、Auto Scaling サービスマネージドフリート (SMF) をデベロッパーファームに追加できます。SMF は、大規模なワークロードに十分なコンピューティング性能を提供し、複数のワーカーホストにジョブタスクを分散する必要があるジョブを処理できます。スケジューラは、CMF ワーカーをシャットダウンしない限り、SMF ワーカーと CMF ワーカーの両方を使用してジョブを実行します。

デベロッパーファームにサービスマネージドフリートを追加するには

1. まだインストールしていない場合は、 AWS Command Line Interface (AWS CLI) をインストールして設定します。詳細については、「 の最新バージョンをインストールまたは更新する AWS CLI」を参照してください。
2. 最初の AWS CloudShell タブを選択し、サービスマネージドフリートを作成し、そのフリート ID を に追加します.bashrc。このアクションにより、他のターミナルセッションで使用できるようになります。

```
FLEET_ROLE_ARN="arn:aws:iam::$(aws sts get-caller-identity \
         --query "Account" --output text):role/${DEV_FARM_NAME}FleetRole"
 aws deadline create-fleet \
     --farm-id $DEV_FARM_ID \
     --display-name "$DEV_FARM_NAME SMF" \
     --role-arn $FLEET_ROLE_ARN \
     --max-worker-count 5 \
     --configuration \
```

```
        '{
            "serviceManagedEc2": {
                "instanceCapabilities": {
                    "vCpuCount": {
                        "min": 2,
                        "max": 4
                    },
                    "memoryMiB": {
                        "min": 512
                    },
                    "osFamily": "linux",
                    "cpuArchitectureType": "x86_64"
                },
                "instanceMarketOptions": {
                    "type": "spot"
                }
            }
        }'

echo "DEV_SMF_ID=$(aws deadline list-fleets \
        --farm-id $DEV_FARM_ID \
        --query "fleets[?displayName=='$DEV_FARM_NAME SMF'].fleetId \
        | [0]" --output text)" >> ~/.bashrc
source ~/.bashrc
```

3. SMF をキューに関連付けます。

```
aws deadline create-queue-fleet-association \
     --farm-id $DEV_FARM_ID \
     --queue-id $DEV_QUEUE_ID \
     --fleet-id $DEV_SMF_ID
```

4. > **Note**
   >
   > スケジューラは、CMF ワーカーをシャットダウンしない限り、SMF ワーカーと CMF ワーカーの両方を使用してジョブを実行します。

   キューsimple_file_jobに送信します。アップロードを確認するプロンプトが表示されたら、 と入力します**y**。

```
deadline bundle submit simple_file_job \
```

```
    -p InFile=simple_job/template.yaml \
    -p OutFile=hash-jobattachments.txt
```

5. SMF が正しく動作していることを確認します。

```
deadline fleet get
```

   - ワーカーの起動には数分かかる場合があります。
   - カスタマーマネージドフリートとサービスマネージドフリートqueueFleetAssociationsStatusの は になりますACTIVE。
   - SMF は から GROWINGに変更autoScalingStatusされますSTEADY。

   ステータスは次のようになります。

```
fleetId: fleet-2cc78e0dd3f04d1db427e7dc1d51ea44
farmId: farm-63ee8d77cdab4a578b685be8c5561c4a
displayName: DeveloperFarm SMF
description: ''
status: ACTIVE
autoScalingStatus: STEADY
targetWorkerCount: 0
workerCount: 0
minWorkerCount: 0
maxWorkerCount: 5
```

6. 送信したジョブのログを表示します。このログは、 CloudShell ファイルシステムではなく Amazon CloudWatch Logs のログに保存されます。

```
JOB_ID=$(deadline config get defaults.job_id)
SESSION_ID=$(aws deadline list-sessions \
        --farm-id $DEV_FARM_ID \
        --queue-id $DEV_QUEUE_ID \
        --job-id $JOB_ID \
        --query "sessions[0].sessionId" \
        --output text)
aws logs tail /aws/deadline/$DEV_FARM_ID/$DEV_QUEUE_ID \
    --log-stream-names $SESSION_ID
```

ステップ 6: Deadline Cloud でファームリソースをクリーンアップする

新しいワークロードとパイプライン統合を開発してテストするには、このチュートリアル用に作成した Deadline Cloud デベロッパーファームを引き続き使用できます。デベロッパーファームが不要になった場合は、Amazon CloudWatch Logs でファーム、フリート、キュー、 AWS Identity and Access Management (IAM) ロール、ログなどのリソースを削除できます。これらのリソースを削除した後、リソースを使用するにはチュートリアルを再度開始する必要があります。詳細については、「Deadline Cloud 用のデベロッパーワークステーションのセットアップ」を参照してください。

デベロッパーファームリソースをクリーンアップするには

1. まだインストールしていない場合は、 AWS Command Line Interface （ AWS CLI) をインストールして設定します。詳細については、「 の最新バージョンをインストールまたは更新する AWS CLI」を参照してください。
2. 最初の CloudShell タブを選択し、キューのすべてのキューフリートの関連付けを停止します。

```
FLEETS=$(aws deadline list-queue-fleet-associations \
        --farm-id $DEV_FARM_ID \
        --queue-id $DEV_QUEUE_ID \
        --query "queueFleetAssociations[].fleetId" \
        --output text)
for FLEET_ID in $FLEETS; do
    aws deadline update-queue-fleet-association \
        --farm-id $DEV_FARM_ID \
        --queue-id $DEV_QUEUE_ID \
        --fleet-id $FLEET_ID \
        --status STOP_SCHEDULING_AND_CANCEL_TASKS
done
```

3. キューフリートの関連付けを一覧表示します。

```
aws deadline list-queue-fleet-associations \
     --farm-id $DEV_FARM_ID \
     --queue-id $DEV_QUEUE_ID
```

出力が と報告されるまでコマンドを再実行する必要がある場合があります。その後"status": "STOPPED"、次のステップに進むことができます。このプロセスは完了までに数分かかることがあります。

```
{
```

```
    "queueFleetAssociations": [
        {
            "queueId": "queue-abcdefgh01234567890123456789012id",
            "fleetId": "fleet-abcdefgh01234567890123456789012id",
            "status": "STOPPED",
            "createdAt": "2023-11-21T20:49:19+00:00",
            "createdBy": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/
MySessionName",
            "updatedAt": "2023-11-21T20:49:38+00:00",
            "updatedBy": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/
MySessionName"
        },
        {
            "queueId": "queue-abcdefgh01234567890123456789012id",
            "fleetId": "fleet-abcdefgh01234567890123456789012id",
            "status": "STOPPED",
            "createdAt": "2023-11-21T20:32:06+00:00",
            "createdBy": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/
MySessionName",
            "updatedAt": "2023-11-21T20:49:39+00:00",
            "updatedBy": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/
MySessionName"
        }
    ]
}
```

4. キューのすべてのキューフリートの関連付けを削除します。

```
for FLEET_ID in $FLEETS; do
     aws deadline delete-queue-fleet-association \
         --farm-id $DEV_FARM_ID \
         --queue-id $DEV_QUEUE_ID \
         --fleet-id $FLEET_ID
 done
```

5. キューに関連付けられているすべてのフリートを削除します。

```
for FLEET_ID in $FLEETS; do
     aws deadline delete-fleet \
         --farm-id $DEV_FARM_ID \
         --fleet-id $FLEET_ID
 done
```

6. キューを削除します。

```
aws deadline delete-queue \
     --farm-id $DEV_FARM_ID \
     --queue-id $DEV_QUEUE_ID
```

7. ファームを削除します。

```
aws deadline delete-farm \
     --farm-id $DEV_FARM_ID
```

8. ファームの他の AWS リソースを削除します。

   a. フリート AWS Identity and Access Management (IAM) ロールを削除します。

   ```
   aws iam delete-role-policy \
        --role-name "${DEV_FARM_NAME}FleetRole" \
        --policy-name WorkerPermissions
   aws iam delete-role \
        --role-name "${DEV_FARM_NAME}FleetRole"
   ```

   b. キューの IAM ロールを削除します。

   ```
   aws iam delete-role-policy \
        --role-name "${DEV_FARM_NAME}QueueRole" \
        --policy-name S3BucketsAccess
   aws iam delete-role \
        --role-name "${DEV_FARM_NAME}QueueRole"
   ```

   c. Amazon CloudWatch Logs ロググループを削除します。各キューとフリートには独自のロググループがあります。

   ```
   aws logs delete-log-group \
        --log-group-name "/aws/deadline/$DEV_FARM_ID/$DEV_QUEUE_ID"
   aws logs delete-log-group \
        --log-group-name "/aws/deadline/$DEV_FARM_ID/$DEV_CMF_ID"
   aws logs delete-log-group \
        --log-group-name "/aws/deadline/$DEV_FARM_ID/$DEV_SMF_ID"
   ```

Deadline Cloud 送信者を設定する

このプロセスは、Deadline Cloud AWS 送信者をインストール、セットアップ、起動したい管理者とアーティストを対象としています。Deadline Cloud 送信者は、デジタルコンテンツ作成 (DCC) プラグインです。アーティストはこれを使用して、使い慣れたサードパーティーの DCC インターフェイスからジョブを送信します。

> **Note**
> このプロセスは、アーティストがレンダリングの送信に使用するすべてのワークステーションで完了する必要があります。

トピック

ステップ 1: Deadline Cloud 送信者をインストールする

以下のセクションでは、Deadline Cloud 送信者をインストールする手順について説明します。

送信者インストーラをダウンロードする

Deadline Cloud 送信者をインストールする前に、送信者インストーラをダウンロードする必要があります。現在、Deadline Cloud 送信者インストーラは Windowsと のみをサポートしています Linux。

1. にサインイン AWS Management Console し、Deadline Cloud コンソール を開きます。
2. サイドナビゲーションペインから、ダウンロード を選択します。
3. Deadline Cloud 送信者インストーラーセクションを見つけます。
4. コンピュータのオペレーティングシステムのインストーラを選択し、 のダウンロードを選択します。

(オプション) ダウンロードしたソフトウェアの信頼性を検証する

ダウンロードしたソフトウェアが本物であることを確認するには、 Windowsまたは のどちらかに次の手順を使用しますLinux。

> **Note**
>
> これらの手順を使用して、まずインストーラを検証し、次のセクション (ステップ 2) でダウンロードした後で Deadline Cloud モニターを検証できます。

Windows

ダウンロードしたファイルの真正性を確認するには、次のステップを実行します。

1. 次のコマンド*file*で、 を、検証するファイルに置き換えます。例えば ***C:\PATH\TO\MY\DeadlineCloudSubmitter-windows-x64-installer.exe*** です。また、 を、インストールされている SignTool SDK のバージョン*signtool-sdk-version*に置き換えます。例えば **10.0.22000.0** です。

```
"C:\Program Files (x86)\Windows Kits\10\bin\signtool-sdk-version\x86\signtool.exe" verify /vfile
```

2. 例えば、次のコマンドを実行して、Deadline Cloud 送信者インストーラファイルを確認できます。

```
"C:\Program Files (x86)\Windows Kits\10\bin\10.0.22000.0\x86\signtool.exe" verify /v DeadlineCloudSubmitter-windows-x64-installer.exe
```

Linux

ダウンロードしたファイルの信頼性を確認するには、 gpg コマンドラインツールを使用します。

1. 次のコマンドを実行して、Deadline Cloud 送信者インストーラの OpenPGPキーをインポートします。

```
 gpg --import --armor <<EOF
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGX6GQsBEADduUtJgqSXI+q76O6fsFwEYKmbnlyL0xKvlq32EZuyv0otZo5L
```

```
le4m5Gg52AzrvPvDiUTLooAlvYeozaYyirIGsK08Ydz0Ftdjroiuh/mw9JSJDJRI
rnRn5yKet1JFezkjopA3pjsTBP6lW/mb1bDBDEwwwtH0x9lV7A03FJ9T7Uzu/qSh
qO/UYdkafro3cPASvkqgDt2tCvURfBcUCAjZVFcLZcVD5iwXacxvKsxxS/e7kuVV
I1+VGT8Hj8XzWYhjCZxOLZk/fvpYPMyEEujN0fYUp6RtMIXve0C9awwMCy5nBG2J
eE2Ol5DsCpTaBd4Fdr3LWcSs8JFA/YfP9auL3NczOozPoVJt+fw8CBlVIXOOJ7l5
hvHDjcC+5v0wxqAlMG6+f/SX7CT8FXK+L3iOJ5gBYUNXqHSxUdv8kt76/KVmQa1B
Akl+MPKpMq+lhw++S3G/lXqwWaDNQbRRw7dSZHymQVXvPp1nsqc3hV7KlOM+6s6g
1g4mvFY4lf6DhptwZLWyQXU8rBQpojvQfiSmDFrFPWFi5BexesuVnkGIolQoklKx
AVUSdJPVEJCteyy7td4FPhBaSqT5vW3+ANbr9b/uoRYWJvn17dN0cc9HuRh/Ai+I
nkfECo2WUDLZ0fEKGjGyFX+todWvJXjvc5kmE9Ty5vJp+M9Vvb8jd6t+mwARAQAB
tCxBV1MgRGVhZGxpbmUgQ2xvdWQgPGF3cy1kZWFkbGluZUBhbWF6b24uY29tPokC
VwQTAQgAQRYhBLhAwIwpqQeWoHH6pfbNPOa3bzzvBQJl+hkLAxsvBAUJA8JnAAUL
CQgHAgIiAgYVCgkICwIDFgIBAh4HAheAAAoJEPbNPOa3bzzvKswQAJXzKSAY8sY8
F6Eas2oYwIDDdDurs8FiEnFghjUEO6MTt9AykF/jw+CQg2UzFtEyObHBymhgmhXE
3buVeom96tgM3ZDfZu+sxi5pGX6oAQnZ6riztN+VpkpQmLgwtMGpSMLl3KLwnv2k
WK8mrR/fPMkfdaewB7A6RIUYiW33GAL4KfMIs8/vIwIJw99NxHpZQVoU6dFpuDtE
1OuxGcCqGJ7mAmo6H/YawSNp2Ns80gyqIKYo7o3LJ+WRroIRlQyctq8gnR9JvYXX
42ASqLq5+OXKo4qh81blXKYqtc176BbbSNFjWnzIQgKDgNiHFZCdcOVgqDhwO15r
NICbqqwwNLj/Fr2kecYx180Ktpl0jOOw5IOyh3bf3MVGWnYRdjvA1v+/CO+55N4g
z0kf50Lcdu5RtqV10XBCifn28pecqPaSdYcssYSRl5DLiFktGbNzTGcZZwITTKQc
af8PPdTGtnnb6P+cdbW3bt9MVtN5/dgSHLThnS8MPEuNCtkTnpXshuVuBGgwBMdb
qUC+HjqvhZzbwns8dr5WI+6HWNBFgGANn6ageYl58vVp0UkuNP8wcWjRARciHXZx
ku6W2jPTHDWGNrBQO2Fx7fd2QYJheIPPAShHcfJO+xgWCof45D0vAxAJ8gGg9Eq+
gFWhsx4NSHn2gh1gDZ41Ou/4exJ1lwPM
=uVaX
-----END PGP PUBLIC KEY BLOCK-----
EOF
```


2. OpenPGP キーを信頼するかどうかを決定します。上記のキーを信頼するかどうかを決定する際に考慮すべき要素には、次のようなものがあります。

   - このウェブサイトから GPG キーを取得するために使用したインターネット接続は安全です。
   - このウェブサイトにアクセスしているデバイスは安全です。
   - AWS は、このウェブサイトでOpenPGPパブリックキーのホスティングを保護するための対策を講じています。

3. OpenPGP キーを信頼する場合は、次の例gpgのように でキーを編集して信頼します。

```
$ gpg --edit-key 0xB840C08C29A90796A071FAA5F6CD3CE6B76F3CEF

    gpg (GnuPG) 2.0.22; Copyright (C) 2013 Free Software Foundation, Inc.
    This is free software: you are free to change and redistribute it.
```

```
   There is NO WARRANTY, to the extent permitted by law.


   pub  4096R/4BF0B8D2  created: 2023-06-23  expires: 2025-06-22  usage: SCEA
                        trust: unknown       validity: unknown
   [ unknown] (1). AWS Deadline Cloud example@example.com

   gpg> trust
   pub  4096R/4BF0B8D2  created: 2023-06-23  expires: 2025-06-22  usage: SCEA
                        trust: unknown       validity: unknown
   [ unknown] (1). AWS Deadline Cloud aws-deadline@amazon.com

   Please decide how far you trust this user to correctly verify other users'
keys
   (by looking at passports, checking fingerprints from different sources,
etc.)

     1 = I don't know or won't say
     2 = I do NOT trust
     3 = I trust marginally
     4 = I trust fully
     5 = I trust ultimately
     m = back to the main menu

   Your decision? 5
   Do you really want to set this key to ultimate trust? (y/N) y

   pub  4096R/4BF0B8D2  created: 2023-06-23  expires: 2025-06-22  usage: SCEA
                        trust: ultimate      validity: unknown
   [ unknown] (1). AWS Deadline Cloud aws-deadline@amazon.com
   Please note that the shown key validity is not necessarily correct
   unless you restart the program.

   gpg> quit
```

4. インストーラの検証

   インストーラを確認するには、次のステップを実行します。

   a. Deadline Cloud コンソールのダウンロードページに戻り、Deadline Cloud 送信者インストーラの署名ファイルをダウンロードします。

   b. 以下を実行して、Deadline Cloud 送信者インストーラの署名を確認します。

```
gpg --verify ./DeadlineCloudSubmitter-linux-x64-
installer.run.sig ./DeadlineCloudSubmitter-linux-x64-
installer.run
```

5. Deadline Cloud モニターを検証する

> **Note**
>
> 署名ファイルまたはプラットフォーム固有の方法を使用して、Deadline Cloud モニターのダウンロードを確認できます。プラットフォーム固有の方法については、Linux (DEB)タブまたはダウンロードしたファイルタイプに基づく Linux (AppImage) タブを参照してください。

署名ファイルを使用して Deadline Cloud Monitor デスクトップアプリケーションを検証するには、次のステップを実行します。

a. Deadline Cloud [コンソール](https://console.aws.amazon.com/deadlinecloud/home)のダウンロードページに戻り、対応する .sig ファイルをダウンロードして、 を実行します。

.deb の場合：

```
gpg --verify ./deadline-cloud-
monitor_<APP_VERSION>_amd64.deb.sig ./deadline-cloud-
monitor_<APP_VERSION>_amd64.deb
```

の場合AppImage：

```
gpg --verify ./deadline-cloud-
monitor_<APP_VERSION>_amd64.AppImage.sig ./deadline-cloud-
monitor_<APP_VERSION>_amd64.AppImage
```

b. 出力が次のようになっていることを確認します。

```
gpg: Signature made Mon Apr 1 21:10:14 2024 UTC
```

```
gpg: using RSA key B840C08C29A90796A071FAA5F6CD3CE6B7
```

出力に というフレーズが含まれている場合は`Good signature from "AWS Deadline Cloud"`、署名が正常に検証され、Deadline Cloud モニターのインストールスクリプトを実行できることを意味します。

Linux (DEB)

Linux .deb バイナリを使用するパッケージを確認するには、まず Linuxタブのステップ 1～3 を完了します。

dpkg は、ほとんどのdebianベースのLinuxディストリビューションのコアパッケージ管理ツールです。.deb ファイルは、 ツールを使用して検証できます。

1. Deadline Cloud コンソールのダウンロードページから、Deadline Cloud モニター .deb ファイルをダウンロードします。
2. を、検証する .deb ファイルのバージョン***<APP_VERSION>***に置き換えます。

   ```
   dpkg-sig --verify deadline-cloud-monitor_<APP_VERSION>_amd64.deb
   ```
3. 出力は次のようになります。

   ```
   Processing deadline-cloud-monitor_1.1.1_amd64.deb... GOODSIG
   _gpgbuilder B840C08C29A90796A071FAA5F6CD3C 171200
   ```
4. .deb ファイルを検証するには、 `GOODSIG` が出力に存在することを確認します。

Linux (AppImage)

.AppImage binary を使用するパッケージを確認するには、まず Linux Linuxタブのステップ 1～3 を完了します。

1. Deadline Cloud コンソールのダウンロードページから、Deadline Cloud モニター .AppImage file をダウンロードします。
2. <APP_VERSION> を、検証する .AppImage file のバージョンに置き換えるには、次の手順を実行します。

   a. .AppImage file の署名を .sig ファイルに書き込みます。

   ```
   ./deadline-cloud-monitor_<APP_VERSION>_amd64.AppImage
   --appimage-signature > ./deadline-cloud-
   monitor_<APP_VERSION>_amd64_.AppImage.sig
   ```

b. 生成された .sig ファイルを使用して、次のコマンドを使用して検証します。

```
gpg --verify ./deadline-cloud-
monitor_<APP_VERSION>_amd64.AppImage.sig
```

c. （オプション) アクセス許可拒否エラーが表示された場合は、次のコマンドを使用して実行アクセス許可を追加します。

```
chmod +x ./deadline-cloud-monitor_<APP_VERSION>_amd64.AppImage
```

d. 出力が次のようになっていることを確認します。

```
gpg: Signature made Mon Apr 1 21:10:14 2024 UTC

gpg: using RSA key B840C08C29A90796A071FAA5F6CD3CE6B7
```

出力に というフレーズが含まれている場合は`Good signature from "AWS Deadline Cloud"`、署名が正常に検証され、Deadline Cloud モニターのインストールスクリプトを実行できることを意味します。

Deadline Cloud 送信者をインストールする

Deadline Cloud 送信者は、 Windowsまたは を使用してインストールできますLinux。インストーラでは、次の送信者をインストールできます。

- 2024 年 5 月
- Nuke 14.0～15.0
- Houdini 19.5
- キーショット 12
- Blender 3.6
- Unreal エンジン 5

Windows

1. ファイルブラウザで、インストーラがダウンロードしたフォルダに移動し、 を選択します`DeadlineCloudSubmitter-windows-x64-installer.exe`。

   a. Windows で PC を保護しているポップアップが表示された場合は、詳細 を選択します。
   b. とにかく実行 を選択します。
2. Deadline Cloud Submitter Setup Wizard AWS が開いたら、次へ を選択します。
3. 次のいずれかのステップを実行して、インストール範囲を選択します。
   - 現在のユーザーのみに をインストールするには、ユーザー を選択します。
   - すべてのユーザーに をインストールするには、システム を選択します。

     システム を選択した場合は、インストーラを終了し、次の手順を実行して管理者として再実行する必要があります。

     a. を右クリックし**DeadlineCloudSubmitter-windows-x64-installer.exe**、管理者として実行 を選択します。
     b. 管理者認証情報を入力し、はい を選択します。
     c. インストールスコープのシステムを選択します。
4. インストールスコープを選択したら、次へ を選択します。
5. もう一度次へ を選択して、インストールディレクトリを受け入れます。
6. の統合送信者Nuke、またはインストールする送信者を選択します。
7. [次へ] をクリックします。
8. インストールを確認し、次へ を選択します。
9. 次へ をもう一度選択し、 の終了 を選択します。

Linux

> **Note**
>
> Linux および Deadline Cloud モニター用の Deadline Cloud 統合Nukeインストーラは、少なくとも GLIBC 2.31 のLinuxディストリビューションにのみインストールできます。

1. ターミナルウィンドウを開きます。
2. インストーラをシステムインストールするには、 コマンドを入力し**sudo -i**、Enter キーを押してルートにします。
3. インストーラをダウンロードした場所に移動します。

例えば **cd /home/*USER*/Downloads** です。

4. インストーラを実行可能にするには、 と入力します**chmod +x DeadlineCloudSubmitter-linux-x64-installer.run**。
5. Deadline Cloud 送信者インストーラーを実行するには、「」と入力します**./ DeadlineCloudSubmitter-linux-x64-installer.run**。
6. インストーラが開いたら、画面上のプロンプトに従ってセットアップウィザードを完了します。

ここに記載されていない他の送信者をインストールできます。Deadline Cloud ライブラリを使用して送信者を構築します。これらのライブラリと送信者のソースコードは、aws-deadline GitHub 組織にあります。

ステップ 2: Deadline Cloud モニターをインストールしてセットアップする

Deadline Cloud モニターデスクトップアプリケーションは、 Windowsまたは を使用してインストールできますLinux。

Windows

1. まだサインインしていない場合は、 にサインイン AWS Management Console し、Deadline Cloud コンソール を開きます。
2. 左側のナビゲーションペインから、ダウンロード を選択します。
3. Deadline Cloud Monitor セクションで、コンピュータのオペレーティングシステムの ファイルを選択します。
4. Deadline Cloud モニターをダウンロードするには、ダウンロード を選択します。

Linux

RPM ディストリビューション AppImage に Deadline Cloud モニターをインストールするには

1. 最新の Deadline Cloud モニター をダウンロードします AppImage。
2. AppImage 実行可能ファイルを作成するには、「」と入力します**chmod a+x deadline-cloud-monitor_*<APP_VERSION>*_amd64.AppImage**。
3. 正しい SSL 証明書パスを設定するには、「」と入力します**sudo ln -sf /etc/ssl/certs/ca-bundle.crt /etc/ssl/certs/ca-certificates.crt**。

Debian ディストリビューション AppImage に Deadline Cloud モニターをインストールするには

1. 最新の Deadline Cloud モニター をダウンロードします AppImage。
2. > Note
   >
   > このステップは Ubuntu 22 以降用です。他のバージョンの Ubuntu の場合は、このステップをスキップします。

   libfuse2 をインストールするには、「」と入力します。 **`sudo apt udpate`**

   `sudo apt install libfuse2`.
3. AppImage 実行可能ファイルを作成するには、「」と入力します**`chmod a+x deadline-cloud-monitor_<APP_VERSION>_amd64.AppImage`**。

Debian ディストリビューションに Deadline Cloud Monitor Debian パッケージをインストールするには

1. 最新の Deadline Cloud Monitor Debian パッケージをダウンロードします。
2. > Note
   >
   > このステップは Ubuntu 22 以降用です。他のバージョンの Ubuntu の場合は、このステップをスキップします。

   libssl1.1 をインストールするには、「」と入力します。 **`wget http://nz2.archive.ubuntu.com/ubuntu/pool/main/o/openssl/libssl1.<APP_VERSION>.1f-1ubuntu2.22_amd64.deb`**

   `sudo dpkg -i libssl1.<APP_VERSION>.1f-1ubuntu2.22_amd64.deb`.
3. Deadline Cloud Monitor Debian パッケージをインストールするには、「」と入力します。 **`sudo apt update`**

   `sudo apt install ./deadline-cloud-monitor_<APP_VERSION>_amd64.deb`.
4. 依存関係が満たされていないパッケージでインストールが失敗した場合、壊れたパッケージを修正し、次のコマンドを実行します。

```
sudo apt --fix-missing update

sudo apt update

sudo apt install -f
```

ダウンロードが完了したら、ダウンロードしたソフトウェアの信頼性を検証できます。ステップ 1 の「ダウンロードしたソフトウェアの信頼性を検証する」を参照してください。

Deadline Cloud モニターをダウンロードして信頼性を確認したら、次の手順を使用して Deadline Cloud モニターを設定します。

Deadline Cloud モニターを設定するには

1. Deadline Cloud モニター を開きます。
2. 新しいプロファイルを作成するように求められたら、次のステップを実行します。
    a. モニター URL を URL 入力に次のように入力します。 **https://*MY-MONITOR*.deadlinecloud.amazonaws.com/**
    b. プロファイル名を入力します。
    c. プロファイルの作成 を選択します。

    プロファイルが作成され、作成したプロファイル名を使用するソフトウェアと認証情報が共有されるようになりました。
3. Deadline Cloud モニタープロファイルを作成した後は、プロファイル名やスタジオ URL を変更することはできません。変更を加える必要がある場合は、代わりに次の操作を行います。
    a. プロファイルを削除します。左側のナビゲーションペインで、Deadline Cloud Monitor 、設定 、削除 を選択します。
    b. 必要な変更を含む新しいプロファイルを作成します。
4. 左側のナビゲーションペインで、>Deadline Cloud Monitor オプションを使用して以下を実行します。
    - Deadline Cloud モニタープロファイルを変更して、別のモニターにログインします。
    - Autologin を有効にすると、Deadline Cloud Monitor のその後のオープン時にモニター URL を入力する必要がなくなります。

5. Deadline Cloud モニターウィンドウを閉じます。バックグラウンドで実行され続け、15 分ごとに認証情報を同期します。
6. レンダリングプロジェクトに使用する予定のデジタルコンテンツ作成 (DCC) アプリケーションごとに、次の手順を実行します。
    a. Deadline Cloud 送信者から、Deadline Cloud ワークステーション設定を開きます。
    b. ワークステーション設定で、Deadline Cloud Monitor で作成したプロファイルを選択します。Deadline Cloud の認証情報がこの DCC と共有され、ツールは期待どおりに動作するはずです。

ステップ 3: Deadline Cloud 送信者を起動する

以下のセクションでは、、Blender、、NukeMayaおよび で Deadline Cloud 送信者プラグインを起動する手順について説明しますHoudini。

で Deadline Cloud 送信者を起動するには Blender

> **Note**
>
> のサポートBlenderは、サービスマネージドフリートの Conda環境を使用して提供されます。詳細については、「デフォルトのCondaキュー環境」を参照してください。

1. Blender を開きます。
2. アセットルートディレクトリ内に存在する依存関係を持つBlenderシーンを開きます。
3. レンダリングメニューで、Deadline Cloud Dialog を選択します。
    a. Deadline Cloud 送信者でまだ認証されていない場合、認証情報ステータスに NEEDS_LOGIN が表示されます。
    b. [ログイン] を選択します。
    c. ログインブラウザウィンドウが表示されます。ユーザー認証情報を使用してログインします。
    d. [Allow] (許可) を選択します。これでログインし、認証情報ステータスが認証済み と表示されます。
4. [送信] を選択します。

で Deadline Cloud 送信者を起動するには Foundry Nuke

> **Note**
>
> のサポートNukeは、サービスマネージドフリートの Conda環境を使用して提供されます。詳細については、「[デフォルトのCondaキュー環境](#)」を参照してください。

1. Nuke を開きます。
2. アセットルートディレクトリ内に存在する依存関係を持つNukeスクリプトを開きます。
3. Thinkbox を選択し、Deadline Cloud に送信 を選択して送信者を起動します。
    a. Deadline Cloud 送信者でまだ認証されていない場合、認証情報ステータスは NEEDS_LOGIN と表示されます。
    b. [ログイン] を選択します。
    c. ログインブラウザウィンドウで、ユーザー認証情報を使用してログインします。
    d. [Allow] (許可) を選択します。これでログインし、認証情報のステータスが認証済み と表示されます。
4. [送信] を選択します。

で Deadline Cloud 送信者を起動するには Maya

> **Note**
>
> Maya および のサポートArnold for Maya(MtoA)は、サービスマネージドフリートの Conda環境を使用して提供されます。詳細については、「[デフォルトのCondaキュー環境](#)」を参照してください。

1. Maya を開きます。
2. プロジェクトを設定し、アセットルートディレクトリ内に存在するファイルを開きます。
3. Windows → 設定/設定 → プラグインマネージャー を選択します。
4. DeadlineCloud送信者 を検索します。
5. Deadline Cloud 送信者プラグインをロードするには、ロード済み を選択します。

a. Deadline Cloud 送信者でまだ認証されていない場合、認証情報ステータスは NEEDS_LOGIN と表示されます。
b. [ログイン] を選択します。
c. ログインブラウザウィンドウが表示されます。ユーザー認証情報を使用してログインします。
d. [Allow] (許可) を選択します。これでログインし、認証情報ステータスが認証済み として表示されます。

6. （オプション) を開くたびに Deadline Cloud 送信者プラグインをロードするにはMaya、「自動ロード」を選択します。
7. Deadline Cloud シェルフを選択し、緑色のボタンを選択して送信者を起動します。

で Deadline Cloud 送信者を起動するには Houdini

> **Note**
> のサポートHoudiniは、サービスマネージドフリートの Conda環境を使用して提供されます。詳細については、「デフォルトのCondaキュー環境」を参照してください。

1. Houdini を開きます。
2. ネットワークエディタ で、/out ネットワークを選択します。
3. タブ を押し、 と入力します**deadline**。
4. Deadline Cloud オプションを選択し、既存のネットワークに接続します。
5. Deadline Cloud ノード をダブルクリックします。

で Deadline Cloud 送信者を起動するには KeyShot

これは、Deadline Cloud と PySide2 が既にダウンロードされていることを前提としています。

1. ファイル deadline-cloud-for-keyshot/keyshot_script/Submit を AWS Deadline Cloud.py にコピーまたはリンクして、KeyShotスクリプトフォルダに移動します。

   例えば、 ではWindows、スクリプトフォルダの場所は になります**C:/Users/*USER*/Documents/KeyShot 12/Scripts**。

2. 次の環境変数を設定します。

   a. deadline-cloud と PySide2 が配置されている Python インストールへのパス**DEADLINE_PYTHON**として環境変数を設定します。

      例えば、 ではWindows、Python 3.10 を使用している場合、コマンドは になります**set DEADLINE_PYTHON=C:/Users/*USER*/AppData/Local/Programs/Python/Python310/python**。

   b. keyshot_submitter フォルダへのパス**DEADLINE_KEYSHOT**として環境変数を設定します。

      例えば、 ではWindows、ソースがデスクトップ上にある場合、コマンドは である可能性があります**set DEADLINE_KEYSHOT=C:/Users/*USER*/Desktop/deadline-cloud-for-keyshot/src/deadline/keyshot_submitter**。

3. 環境変数を設定したら、 を起動しますKeyShot。
4. から送信者を起動するには KeyShot、、スクリプティングコンソール Windows、Deadline Cloud AWS への送信、および実行 を選択します。

で Deadline Cloud 送信者を起動するには Unreal Engine

これは、Deadline Cloud が既にダウンロードされていることを前提としています。

1. Unreal Engine プロジェクトに使用するフォルダを作成または開きます。
2. コマンドラインを開き、次のコマンドを実行します。

   - `git clone https://github.com/aws-deadline/deadline-cloud-for-unreal-engine`
   - `cd deadline-cloud-for-unreal/test_projects`
   - `git lfs fetch -all`

3. のプラグインをダウンロードするにはUnreal Engine、Unreal Engineプロジェクトフォルダを開き、deadline-cloud-forunreal/test_projects/pull_ue_plugin.bat を起動します。

   これにより、プラグインファイルは C:/LocalProjects/UnrealDeadlineCloudTest/Plugins/UnrealDeadlineCloudServiceに配置されます。

4. 送信者をダウンロードするには、 UnrealDeadlineCloudService フォルダを開き、を実行します**deadline-cloud-forunreal/ test_projects/Plugins/UnrealDeadlineCloudService/install_unreal_submitter.bat**。
5. から送信者を起動するにはUnreal Engine、次の手順を実行します。

a. 編集、 > プロジェクト設定 を選択します。
b. [Search] バーに「**movie render pipeline**」と入力します。
c. 次の映画レンダリングパイプライン設定を調整します。

    i. デフォルトのリモートエグゼキューター には、 と入力します**MoviePipelineDeadlineCloudRemote Executor**。
    ii. デフォルトのエグゼキュータージョブ には、「」と入力します。**MoviePipelineDeadlineCloudExecutorJob**
    iii. デフォルトのジョブ設定クラス で、プラス記号を選択し、 と入力します**DeadlineCloudRenderStepSetting**。

これらの設定では、 から Deadline Cloud プラグインを選択できますUnreal Engine。

ファームを使用する

すべての開始手順に従った場合は、ローカルワークステーションからファームへのジョブの送信を開始して、それらのジョブとリソースをモニタリングするために必要なものがすべて設定されています。あらゆる種類のジョブまたはモニタリングの送信の詳細については、以下の関連トピックを参照してください。

- ジョブ
- モニターの使用

Deadline Cloud モニターの使用

AWS Deadline Cloud モニターには、ビジュアルコンピューティングジョブの全体像が表示されます。これを使用して、ジョブのモニタリングと管理、フリートのワーカーアクティビティの表示、予算と使用状況の追跡、ジョブの結果のダウンロードを行うことができます。

各キューには、ジョブ、ステップ、タスクのステータスを示すジョブモニターがあります。モニターには、モニターから直接ジョブを管理する方法が用意されています。優先順位付けの変更、ジョブのキャンセル、ジョブの再キューを行うことができます。

Deadline Cloud モニターには、ジョブの概要ステータスを示すテーブルがあります。または、ジョブを選択して、ジョブに関する問題のトラブルシューティングに役立つ詳細なタスクログを表示できます。

Deadline Cloud モニターを使用して、ジョブの作成時に指定されたワークステーション上の場所に結果をダウンロードできます。

Deadline Cloud モニターは、使用状況のモニタリングやコスト管理にも役立ちます。詳細については、「Deadline Cloud の予算と使用状況の管理」を参照してください。

トピック

Deadline Cloud モニター URL を共有する

Deadline Cloud サービスをセットアップすると、デフォルトでアカウントの Deadline Cloud モニターを開く URL が作成されます。この URL を使用して、ブラウザまたはデスクトップでモニター

を開きます。他のユーザーが Deadline Cloud モニターにアクセスできるように、他のユーザーと URL を共有します。

ユーザーが Deadline Cloud モニターを開く前に、ユーザーにアクセス権を付与する必要があります。アクセスを許可するには、モニターの承認されたユーザーのリストにユーザーを追加するか、モニターにアクセスできるグループに追加します。詳細については、「Deadline Cloud でのユーザーの管理」を参照してください。

モニター URL を共有するには

1. Deadline Cloud コンソール を開きます。
2. 開始方法 から、「Deadline Cloud ダッシュボードに移動」を選択します。
3. ナビゲーションペインで、ダッシュボード を選択します。
4. アカウントの概要セクションで、アカウントの詳細 を選択します。
5. URL をコピーして、Deadline Cloud モニターにアクセスする必要があるすべてのユーザーに安全に送信します。

Deadline Cloud モニターを開く

Deadline Cloud モニターは、次のいずれかの方法で開くことができます。

- コンソール – にサインイン AWS Management Console し、Deadline Cloud コンソールを開きます。
- ウェブ – Deadline Cloud のセットアップ時に作成したモニター URL に移動します。
- Monitor – デスクトップの Deadline Cloud モニターを使用します。

コンソールを使用する場合、ID AWS を使用して AWS Identity and Access Management にサインインし、 AWS IAM Identity Center 認証情報を使用してモニターにサインインできる必要があります。IAM Identity Center の認証情報しかない場合は、モニター URL またはデスクトップアプリケーションを使用してサインインする必要があります。

Deadline Cloud モニターを開くには (ウェブ)

1. ブラウザを使用して、Deadline Cloud のセットアップ時に作成したモニター URL を開きます。
2. ユーザー認証情報を使用してサインインします。

Deadline Cloud モニターを開くには (コンソール)

1. Deadline Cloud コンソール を開きます。
2. ナビゲーションペインで、ファーム を選択します。
3. ファームを選択し、ジョブの管理を選択して Deadline Cloud モニターページを開きます。
4. ユーザー認証情報を使用してサインインします。

Deadline Cloud モニターを開くには (デスクトップ)

1. Deadline Cloud コンソール を開きます。

   -または-

   Deadline Cloud モニター - モニター URL からウェブを開きます。
2. • Deadline Cloud コンソールで、次の操作を行います。
     1. モニタで、「Deadline Cloud ダッシュボードに移動」を選択し、左側のメニューから「ダウンロード」を選択します。
     2. Deadline Cloud モニター から、デスクトップのモニターバージョンを選択します。
     3. [ダウンロード] を選択します。
   • Deadline Cloud モニター - ウェブで、次の操作を行います。
     • 左側のメニューから、ワークステーションのセットアップ を選択します。Workstation セットアップ項目が表示されない場合は、矢印を使用して左側のメニューを開きます。
     • [ダウンロード] を選択します。
     • OS の選択 から、オペレーティングシステムを選択します。
3. Deadline Cloud モニター - デスクトップをダウンロードします。
4. モニタをダウンロードしてインストールしたら、コンピュータで開きます。

   • Deadline Cloud モニターを初めて開く場合は、モニター URL を指定してプロファイル名を作成する必要があります。次に、Deadline Cloud 認証情報を使用してモニターにサインインします。
   • プロファイルを作成したら、プロファイルを選択してモニターを開きます。Deadline Cloud の認証情報の入力が必要になる場合があります。

Deadline Cloud でキューとフリートの詳細を表示する

Deadline Cloud モニターを使用して、ファーム内のキューとフリートの設定を表示できます。モニターを使用して、キュー内のジョブまたはフリート内のワーカーのリストを表示することもできます。

キューとフリートの詳細を表示するには、 アクセスVIEWING許可が必要です。詳細が表示されない場合は、管理者に連絡して正しいアクセス許可を取得してください。

キューの詳細を表示するには

1. Deadline Cloud モニターを開く.
2. ファームのリストから、関心のあるキューを含むファームを選択します。
3. キューのリストで、キューを選択してその詳細を表示します。2 つ以上のキューの設定を比較するには、複数のチェックボックスをオンにします。
4. キュー内のジョブのリストを表示するには、キューのリストからキュー名を選択するか、詳細パネルからキュー名を選択します。

モニターが既に開いている場合は、左側のナビゲーションペインのキューリストからキューを選択できます。

フリートの詳細を表示するには

1. Deadline Cloud モニターを開く.
2. ファームのリストから、関心のあるフリートを含むファームを選択します。
3. Farm リソース で、フリート を選択します。
4. フリートのリストで、フリートを選択してその詳細を表示します。2 つ以上のフリートの設定を比較するには、複数のチェックボックスをオンにします。
5. フリート内のワーカーのリストを表示するには、フリートのリストから、または詳細パネルからフリート名を選択します。

モニターが既に開いている場合は、左側のナビゲーションペインのフリートリストからフリートを選択できます。

Deadline Cloud でのジョブ、ステップ、タスクの表示と管理

キューを選択すると、Deadline Cloud モニターのジョブモニターセクションに、そのキュー内のジョブ、ジョブ内のステップ、および各ステップのタスクが表示されます。ジョブ、ステップ、またはタスクを選択すると、アクションメニューを使用してそれぞれを管理できます。

ジョブモニターを開くには、ステップに従って でキューを表示しDeadline Cloud でキューとフリートの詳細を表示する、使用するジョブ、ステップ、またはタスクを選択します。

ジョブ、ステップ、タスクの場合は、次の操作を実行できます。

- ステータスを Requeued 、Succeeded 、Failed 、Canceled に変更します。
- 処理された出力をジョブ、ステップ、またはタスクからダウンロードします。
- ジョブ、ステップ、またはタスクの ID をコピーします。

選択したジョブでは、次のことができます。

- ジョブをアーカイブします。
- 優先順位の変更やステップの表示からステップの依存関係の変更など、ジョブのプロパティを変更します。
- ジョブのパラメータを使用して追加の詳細を表示します。

詳細については、 Deadline Cloud でジョブの詳細を表示する を参照してください。

ステップごとに、次のことができます。

- ステップの依存関係を表示します。ステップの依存関係は、ステップを実行する前に完了する必要があります。

詳細については、「Deadline Cloud でステップを表示する」を参照してください。

タスクごとに、次のことができます。

- タスクのログを表示します。
- タスクパラメータを表示します。

詳細については、「Deadline Cloud でタスクを表示する」を参照してください。

Deadline Cloud でジョブの詳細を表示する

Deadline Cloud Monitor のジョブモニターページには、次の情報が表示されます。

- ジョブの進行状況の全体ビュー。
- ジョブを構成するステップとタスクのビュー。

リストからジョブを選択してジョブのステップのリストを表示し、ステップのリストからステップを選択してジョブのタスクを表示します。項目を選択したら、その項目のアクションメニューを使用して詳細を表示できます。

ジョブの詳細を表示するには

1. のステップに従って、キューを表示しますDeadline Cloud でキューとフリートの詳細を表示する。
2. ナビゲーションペインで、ジョブを送信したキューを選択します。
3. 次のいずれかの方法を使用してジョブを選択します。
   a. ジョブリストからジョブを選択して、その詳細を表示します。
   b. 検索フィールドから、ジョブ名やジョブを作成したユーザーなど、ジョブに関連付けられたテキストを入力します。表示される結果から、表示するジョブを選択します。

ジョブの詳細には、ジョブのステップと各ステップのタスクが含まれます。アクションメニューを使用して、次の操作を実行できます。

- ジョブのステータスを変更します。
- ジョブのプロパティを表示および変更します。ジョブ内のステップ間の依存関係を表示し、ジョブの優先度を変更できます。通常、優先度の高いジョブは早く完了します。
- ジョブの送信時に設定されたジョブのパラメータを表示します。
- ジョブの出力をダウンロードします。ジョブの出力をダウンロードすると、ジョブのステップとタスクによって生成されたすべての出力が含まれます。

Deadline Cloud でステップを表示する

Deadline Cloud AWS モニターを使用して、処理ジョブのステップを表示します。ジョブモニター では、ステップリストに、選択したジョブを構成するステップのリストが表示されます。ステップを選択すると、タスクリストにステップ内のタスクが表示されます。

ステップを表示するには

1. の手順に従って[Deadline Cloud でジョブの詳細を表示する](#)、ジョブのリストを表示します。
2. [ジョブ] リストからジョブを選択します。
3. ステップリストからステップを選択します。

アクションメニューを使用して、次の操作を実行できます。

- ステップのステータスを変更します。
- ステップの出力をダウンロードします。ステップの出力をダウンロードすると、ステップのタスクによって生成されたすべての出力が含まれます。
- ステップの依存関係を表示します。依存関係テーブルには、選択したステップを開始する前に完了する必要があるステップのリストと、このステップの完了を待っているステップのリストが表示されます。

Deadline Cloud でタスクを表示する

AWS Deadline Cloud モニターを使用して、処理ジョブのタスクを表示します。ジョブモニター のタスクリストには、ステップリストで選択されたステップを構成するタスクが表示されます。

タスクを表示するには

1. の手順に従って[Deadline Cloud でジョブの詳細を表示する](#)、ジョブのリストを表示します。
2. [ジョブ] リストからジョブを選択します。
3. ステップリストからステップを選択します。
4. タスクリストからタスクを選択します。

アクションメニューを使用して、次の操作を実行できます。

- タスクのステータスを変更します。

- タスクログを表示します。詳細については、「Deadline Cloud でログを表示する」を参照してください。
- タスクの作成時に設定されたパラメータを表示します。
- タスクの出力をダウンロードします。タスクの出力をダウンロードすると、選択したタスクによって生成された出力のみが含まれます。

Deadline Cloud でログを表示する

ログには、タスクのステータスと処理に関する詳細情報が表示されます。 AWS Deadline Cloud モニターには、次の 2 種類のログが表示されます。

- セッションログには、以下を含むアクションのタイムラインが詳しく記載されています。
  - アタッチメントの同期やソフトウェア環境のロードなどのセットアップアクション
  - タスクまたはタスクセットの実行
  - ワーカーの環境をシャットダウンするなどの閉鎖アクション

  セッションには少なくとも 1 つのタスクの処理が含まれ、複数のタスクを含めることができます。セッションログには、Amazon Elastic Compute Cloud (Amazon EC2) インスタンスタイプ、vCPU、メモリに関する情報も表示されます。セッションログには、セッションで使用されるワーカーのログへのリンクも含まれています。
- ワーカーログは、ワーカーがライフサイクル中に処理するアクションのタイムラインの詳細を提供します。ワーカーログには、複数のセッションに関する情報を含めることができます。

セッションログとワーカーログをダウンロードして、オフラインで調べることができます。

セッションログを表示するには

1. の手順に従ってDeadline Cloud でジョブの詳細を表示する、ジョブのリストを表示します。
2. [ジョブ] リストからジョブを選択します。
3. ステップリストからステップを選択します。
4. タスクリストからタスクを選択します。
5. アクションメニューから、ログの表示 を選択します。

タイムラインセクションには、タスクのアクションの概要が表示されます。セッションで実行されているタスクをさらに表示し、セッションのシャットダウンアクションを表示するには、すべてのタスクのログを表示する を選択します。

タスクからワーカーログを表示するには

1. の手順に従って[Deadline Cloud でジョブの詳細を表示する]、ジョブのリストを表示します。
2. [ジョブ] リストからジョブを選択します。
3. ステッププリストからステップを選択します。
4. タスクリストからタスクを選択します。
5. アクションメニューから、ログの表示 を選択します。
6. セッション情報 を選択します。
7. ワーカーログの表示 を選択します。

フリートの詳細からワーカーログを表示するには

1. の手順に従って[Deadline Cloud でキューとフリートの詳細を表示する]フリートを表示します。
2. ワーカーリストからワーカー ID を選択します。
3. アクションメニューから、ワーカーログの表示 を選択します。

Deadline Cloud で完成した出力をダウンロードする

ジョブが完了したら、Deadline Cloud AWS モニターを使用して結果をワークステーションにダウンロードできます。出力ファイルは、ジョブの作成時に指定した名前と場所とともに保存されます。

出力ファイルは無期限に保存されます。ストレージコストを削減するには、キューの Amazon S3 バケットの S3 ライフサイクル設定を作成することを検討してください。 Amazon S3 詳細については、「Amazon Simple Storage Service ユーザーガイド」の「ストレージライフサイクルの管理」を参照してください。

ジョブ、ステップ、またはタスクの完了出力をダウンロードするには

1. の手順に従って[Deadline Cloud でジョブの詳細を表示する]、ジョブのリストを表示します。
2. 出力をダウンロードするジョブ、ステップ、またはタスクを選択します。

- ジョブを選択すると、そのジョブのすべてのステップで、すべてのタスクのすべての出力をダウンロードできます。
- ステップを選択すると、そのステップのすべてのタスクのすべての出力をダウンロードできます。
- タスクを選択すると、その個々のタスクの出力をダウンロードできます。

3. アクションメニューから、出力のダウンロード を選択します。
4. 出力は、ジョブの送信時に設定された場所にダウンロードされます。

> **Note**
>
> メニューを使用した出力のダウンロードは、現在 Windowsおよび でのみサポートされていますLinux。がありMac、出力のダウンロードメニュー項目を選択すると、レンダリングされた出力のダウンロードに使用できる AWS CLI コマンドがウィンドウに表示されます。

デッドラインクラウドファーム

ファームは、ジョブとタスクを実行するコンピュートリソースの群を管理するキューのコンテナです。

トピック

- [ファームを作成します。](#)
- [ファームを削除します。](#)
- [ファームを編集します。](#)

ファームを作成します。

1. [Deadline Cloud コンソールから](#) [ダッシュボードに移動] を選択します。
2. Deadline Cloud ダッシュボードの「ファーム」セクションで、「アクション」→「ファームを作成」を選択します。

   - または、左側のパネルで [ファームとその他のリソース] を選択し、[ファームを作成] を選択します。
3. ファームの名前を追加します。
4. [説明] に、ファームの説明を入力します。わかりやすい説明があれば、ファームの目的がすぐにわかります。
5. (オプション) デフォルトでは、 AWS データはセキュリティのために所有および管理されるキーで暗号化されます。[暗号化設定のカスタマイズ (詳細)] を選択して、既存のキーを使用するか、自分で管理する新しいキーを作成できます。

   チェックボックスを使用して暗号化設定をカスタマイズする場合は、 AWS KMS ARN を入力するか、[Create new KMS key] AWS KMS を選択して新しい ARN を作成します。
6. (オプション) [Add new tag] を選択してファームに 1 つ以上のタグを追加します。
7. [ファームを作成] を選択します。作成後、ファームが表示されます。

ファームを削除します。

1. Deadline Cloud ダッシュボードから [ファームとその他のリソース] を選択します。
2. ファームリストで、削除する 1 つまたは複数のファームを選択し、[削除] を選択します。

ファームを編集します。

1. Deadline Cloud ダッシュボードから [ファームとその他のリソース] を選択します。
2. ファームリストで、削除する 1 つまたは複数のファームを選択し、[編集] を選択します。
3. 表示される編集ウィンドウで、ファーム名または説明を変更し、[Save changes] を選択します。

Deadline クラウドキュー

キューは、ジョブを管理および処理するファームリソースです。

キューを使用するには、モニターとファームが既にセットアップされている必要があります。

トピック

キューを作成する

1. Deadline Cloud コンソールダッシュボードから、キューを作成するファームを選択します。
   - または、左側のパネルで Farms およびその他のリソース を選択し、キューを作成するファームを選択します。
2. キュー タブで、キューの作成 を選択します。
3. キューの名前を入力します。
4. 説明 に、キューの説明を入力します。説明は、キューの目的を特定するのに役立ちます。
5. ジョブアタッチメント では、新しい Amazon S3 バケットを作成するか、既存の Amazon S3 バケットを選択できます。

   a. 新しい Amazon S3 バケットを作成するには

      i. 新しいジョブバケットの作成 を選択します。
      ii. バケットの名前を入力します。バケット に名前を付けることをお勧めします`deadlinecloud-job-attachments-[MONITORNAME]`。
      iii. ルートプレフィックスを入力して、キューのルートロケーションを定義または変更します。

   b. 既存の Amazon S3 バケットを選択するには

      i. 既存の S3 バケットを選択 > S3 を参照を選択します。

ii. 使用可能なバケットのリストからキューの S3 バケットを選択します。

6. （オプション）キューをカスタマーマネージドフリートに関連付けるには、カスタマーマネージドフリートとの関連付けを有効にする を選択します。

7. カスタマーマネージドフリートとの関連付けを有効にする場合は、次のステップを完了する必要があります。

> **Important**
>
> run-as 機能にユーザーとグループを指定することを強くお勧めします。そうしないと、ジョブはワーカーのエージェントが実行できるすべてのことを実行できるため、ファームのセキュリティ体制が低下します。潜在的なセキュリティリスクの詳細については、「ユーザーおよびグループとしてジョブを実行する」を参照してください。

a. ユーザーとして実行の場合：

キューのジョブの認証情報を指定するには、キュー設定ユーザー を選択します。

または、独自の認証情報の設定をオプトアウトし、ワーカーエージェントユーザーとしてジョブを実行するには、ワーカーエージェントユーザー を選択します。

b. （オプション）ユーザー認証情報として実行 で、ユーザー名とグループ名を入力して、キューのジョブの認証情報を指定します。

Windows フリートを使用している場合は、ユーザーとして実行のパスワードを含む シークレットを作成 AWS Secrets Manager する必要があります。シークレットを作成するには、次の手順に従います。*`jobuser`* を の名前に置き換えます`jobRunAsUser`。

i. を開くPowerShellか、管理者としてコマンドプロンプトを表示します。

ii. ユーザーを作成します。

```
net user jobuser /add
```

iii. パスワードを設定します。

```
net user jobuser *
```

iv. ユーザーのローカルプロファイルとホームディレクトリを作成します。次のコマンドを実行し、プロンプトが表示されたらユーザーのパスワードを入力します。

```
runas /profile /user:jobuser "cmd.exe /C"
```

8. 予算を義務付けると、キューのコストを管理するのに役立ちます。予算を必要としないか、予算が必要 を選択します。
9. キューには、ユーザーに代わって Amazon S3 にアクセスするためのアクセス許可が必要です。新しいサービスロールを作成することも、既存のサービスロールを使用することもできます。既存のサービスロールがない場合は、新しいサービスロールを作成して使用します。

   a. 既存のサービスロールを使用するには、サービスロールの選択 を選択し、ドロップダウンからロールを選択します。
   b. 新しいサービスロールを作成するには、新しいサービスロール を作成して使用し、ロール名と説明を入力します。
10. （オプション）キュー環境の環境変数を追加するには、新しい環境変数 を追加を選択し、追加する変数の名前と値を入力します。
11. （オプション）新しいタグを追加 を選択して、キューに 1 つ以上のタグを追加します。
12. デフォルトのCondaキュー環境を作成するには、チェックボックスをオンのままにします。キュー環境の詳細については、「キュー環境の作成」を参照してください。カスタマーマネージドフリートのキューを作成する場合は、チェックボックスをオフにします。
13. [キューの作成]を選択します。

キュー環境を作成する

キュー環境は、フリートワーカーを設定する一連の環境変数とコマンドです。キュー環境を使用して、ソフトウェアアプリケーション、環境変数、その他のリソースをキュー内のジョブに提供できます。

キューを作成するときは、デフォルトのCondaキュー環境を作成するオプションがあります。この環境では、サービスマネージドフリートがパートナー DCC アプリケーションとレンダラーのパッケージにアクセスできます。詳細については、「デフォルトのCondaキュー環境」を参照してください。

キュー環境は、コンソールを使用するか、json または YAML テンプレートを直接編集することで追加できます。この手順では、 コンソールを使用して環境を作成する方法について説明します。

1. キュー環境にキューを追加するには、キューに移動し、キュー環境タブ を選択します。
2. 「アクション」を選択し、「 で新規作成」を選択します。

3. キュー環境の名前と説明を入力します。
4. 新しい環境変数 を追加 を選択し、追加する変数の名前と値を入力します。
5. （オプション）キュー環境の優先度を入力します。優先度は、このキュー環境がワーカーで実行される順序を示します。優先度の高いキュー環境が最初に実行されます。
6. キュー環境の作成 を選択します。

デフォルトのCondaキュー環境

サービスマネージドフリートに関連付けられたキューを作成する場合、 がジョブの仮想環境にパッケージをダウンロードしてインストールCondaするデフォルトのキュー環境を追加するオプションがあります。

Conda は、チャネル からのパッケージを提供します。チャネルは、パッケージが保存される場所です。Deadline Cloud は、パートナーの DCC アプリケーションとレンダラーをサポートするパッケージ`deadline-cloud`をホストするチャネル を提供します。パッケージは以下のとおりです。

- Blender
  - `blender=3.6`
  - `blender-openjd`
- フーディニ語
  - `houdini=19.5`
  - `houdini-openjd`
- Maya
  - `maya=2024`
  - `maya-mtoa=2024.5.3`
  - `maya-openjd`
- Nuke
  - `nuke=15`
  - `nuke-openjd`

デフォルトのConda環境でジョブをキューに送信すると、環境はジョブに 2 つのパラメータを追加します。これらのパラメータは、タスクが処理される前にジョブの環境を設定するために使用するCondaパッケージとチャネルを指定します。パラメータは次のとおりです。

- CondaPackages – blender=3.6や など、パッケージ一致仕様のスペース区切りリストnumpy>1.22。仮想環境の作成をスキップするには、デフォルトは空です。
- CondaChannels – deadline-cloud、、conda-forgeまたは などのCondaチャネルのスペース区切りリストs3://*DOC-EXAMPLE-BUCKET*/conda/channel。デフォルトは です。これはdeadline-cloud、パートナー DCC アプリケーションとレンダラーを提供するサービスマネージドフリートで使用できるチャネルです。

統合された送信者を使用して DCC から Deadline Cloud にジョブを送信すると、送信者は DCC アプリケーションと送信者に基づいて CondaPackages パラメータの値を入力します。例えば、Blender を使用している場合、 CondaPackageパラメータは に設定されますblender=3.6.* blender-openjd=0.4.*。

キューの削除

> **Warning**
>
> キューを削除した場合、キュー内のジョブを復元することはできません。キューを削除すると、そのキュー内のジョブも削除されます。

1. Deadline Cloud ダッシュボードから、Farms およびその他のリソースを選択します。
2. ファームリストで、削除するキューを含むファームを選択します。
3. キューを選択し、削除を選択します。
4. 確認ウィンドウで、[Delete] を選択します。キューとキュー内のすべてのジョブが削除されます。

キューの編集

1. Deadline Cloud ダッシュボードから、Farms およびその他のリソースを選択します。
2. ファームリストで、編集するキューを含むファームを選択します。
3. キューを選択し、編集 を選択します。
4. 名前、説明、予算要件、ユーザーとして実行オプション、割り当てられたサービスロールを編集できます。既存のフリートをキューに関連付けることもできます。
5. [変更の保存] を選択します。

キューとフリートを関連付ける

1. フリートに関連付けるキューを選択します。
2. キューに関連付けるフリートを選択するには、フリートの関連付け を選択します。
3. フリートの選択ドロップダウンを選択します。使用可能なフリートのリストが表示されます。
4. 使用可能なフリートのリストから、キューに関連付けるフリートの横にあるチェックボックスをオンにします。
5. [関連付ける] を選択します。これで、フリートの関連付けステータスは の関連付けになります。

Deadline Cloud フリートの管理

このセクションでは、Deadline Cloud のサービスマネージドフリート (SMF) とカスタマーマネージドフリート (CMF) を管理する方法について説明します。

2 種類の Deadline Cloud フリートを設定できます。

- サービスマネージドフリートは、このサービスである Deadline Cloud によってデフォルト設定が提供されているワーカーのフリートです。これらのデフォルト設定は、効率的で費用対効果が高いように設計されています。
- カスタマーマネージドフリート (CMFsは、管理するワーカーのフリートです。CMF は、 AWS インフラストラクチャ内、オンプレミス、または同じ場所にあるデータセンター内に存在する場合があります。CMF はフリートの完全な制御と責任を提供します。これには、フリート内のワーカーのプロビジョニング、運用、管理、廃止が含まれます。

トピック

Deadline Cloud サービスマネージドフリートを管理する

サービスマネージドフリートは、Deadline Cloud によってデフォルト設定が提供されているワーカーのフリートです。これらのデフォルト設定は、効率的で費用対効果が高いように設計されています。

1. サービスマネージドフリート (SMF) を作成するには、フリートを作成するファームに移動します。
2. フリート タブを選択します。
3. [フリートの作成] を選択します。
4. フリートの名前を入力します。
5. [Description] を入力します。明確な説明は、フリートの目的をすばやく特定するのに役立ちます。
6. サービスマネージドフリートタイプを選択します。

7. フリートのスポットインスタンスまたはオンデマンドインスタンスのマーケットオプションを選択します。スポットインスタンスは、割引価格で使用できる予約されていない容量ですが、オンデマンドリクエストによって中断される可能性があります。オンデマンドインスタンスの料金は 2 秒目ですが、長期的なコミットメントはなく、中断されません。デフォルトでは、フリートはスポットインスタンスを使用します。
8. オプション フリートをスケーリングするインスタンスの最大数を設定して、キュー内のジョブで容量を使用可能にします。キューに入れられたジョブがないときにフリートがすべてのインスタンスを解放するように`0`、最小数のインスタンスを のままにしておくことをお勧めします。
9. フリートのサービスアクセスについては、既存のロールを選択するか、新しいロールを作成します。サービスロールは、フリート内のインスタンスに認証情報を提供し、ジョブを処理するアクセス許可とモニター内のユーザーに付与して、ログ情報を読み取ることができます。
10. [次へ] をクリックします。
11. フリートに必要な vCPU の最小値と最大値を入力します。
12. フリートに必要な最小メモリと最大メモリを入力します。
13. オプション 特定のインスタンスタイプをフリートに許可または除外して、それらのインスタンスタイプのみがこのフリートに使用されるようにすることができます。
14. オプション このフリートのワーカーにアタッチされる Amazon Elastic Block Store (Amazon EBS) gp3 ボリュームのサイズを指定できます。詳細については、「EBS ユーザーガイド」を参照してください。
15. [次へ] をクリックします。
16. オプション ジョブの送信時に指定されたカスタムホスト要件と組み合わせることができる、このフリートの機能を定義するカスタムワーカー要件を定義します。フリートを独自のライセンスサーバーに接続する場合は、特定のライセンスタイプがその一例です。
17. [次へ] をクリックします。
18. オプション フリートをキューに関連付けるには、ドロップダウンからキューを選択します。キューがデフォルトのCondaキュー環境で設定されている場合、フリートにはパートナー DCC アプリケーションとレンダラーをサポートするパッケージが自動的に提供されます。提供されているパッケージのリストについては、「」を参照してくださいデフォルトのCondaキュー環境。
19. [次へ] をクリックします。
20. オプション フリートにタグを追加するには、新しいタグを追加 を選択し、そのタグのキーと値を入力します。
21. [次へ] をクリックします。
22. フリート設定を確認し、フリートの作成 を選択します。作成後、フリートが表示されます。

VFX Reference Platform の互換性

VFX Reference Platform は VFX 業界共通のターゲットプラットフォームです。をサポートするソフトウェアで Amazon Linux 2023 を実行している標準サービスマネージドフリート Amazon EC2 インスタンスを使用するにはVFX Reference Platform、サービスマネージドフリートを使用するときに次の考慮事項に留意する必要があります。

VFX Reference Platform は毎年更新されます。Deadline Cloud サービスマネージドフリートを含む AL2023 を使用する際のこれらの考慮事項は、2022 年から 2024 年までの暦年 (CY) リファレンスプラットフォームに基づいています。詳細については、「VFX Reference Platform」を参照してください。

> **Note**
>
> カスタマーマネージドフリートのカスタム Amazon Machine Image (AMI) を作成する場合は、Amazon EC2 インスタンスを準備するときにこれらの要件を追加できます。

AL2023 Amazon EC2 インスタンスでVFX Reference Platformサポートされているソフトウェアを使用するには、次の点を考慮してください。

- AL2023 と共にインストールされる glibc バージョンは、ランタイムでの使用には互換性がありますが、CY2024 VFX Reference Platform 以前と互換性のあるソフトウェアの構築には互換性がありません。
- Python 3.9 および 3.11 にはサービスマネージドフリートが用意されており、CY2022 VFX Reference Platform および CY2024 と互換性があります。Python 3.7 および 3.10 は、サービスマネージドフリートでは提供されません。それらを必要とするソフトウェアは、キューまたはジョブ環境に Python インストールを提供する必要があります。
- サービスマネージドフリートで提供される一部の Boost ライブラリコンポーネントはバージョン 1.75 であり、 と互換性がありませんVFX Reference Platform。アプリケーションが Boost を使用している場合は、互換性のために独自のバージョンのライブラリを提供する必要があります。
- インテル TBB 更新 3 は、サービスマネージドフリートで提供されます。これは、VFX Reference PlatformCY2022, CY2023、および CY2024 と互換性があります。
- で指定されたバージョンを持つ他のライブラリVFX Reference Platformは、サービスマネージドフリートでは提供されません。サービスマネージドフリートで使用されるすべてのアプリケーションをライブラリに提供する必要があります。ライブラリのリストについては、「 リファレンスプラットフォーム」を参照してください。

Deadline Cloud の顧客管理フリートを管理

このセクションでは、Deadline Cloudの顧客管理型車両群 (CMF) を管理する方法について説明します。

CMF はユーザーが管理する作業員群です。CMF は、 AWS インフラストラクチャ内、オンプレミス、または同じ場所にあるデータセンターに配置されている場合があります。CMF は車両全体を統制し、責任を負います。これには、車両内の作業員のプロビジョニング、運用、管理、廃止措置が含まれます。

トピック

カスタマーマネージドフリートを作成する

カスタマーマネージドフリート (CMF) を作成するには、次のステップを実行します。

Deadline Cloud console

Deadline Cloud コンソールを使用してカスタマーマネージドフリートを作成するには

1. Deadline Cloud コンソール を開きます。
2. Farms を選択します。使用可能なファームのリストが表示されます。
3. 作業するファームの名前を選択します。
4. フリート タブを選択します。
5. [フリートの作成] を選択します。
6. フリートの名前を入力します。

7. （オプション) フリートの説明を入力します。
8. フリートタイプ のカスタマーマネージド を選択します。
9. Auto Scaling タイプを選択します。詳細については、「EventBridge を使用して Auto Scaling イベントを処理する」を参照してください。
   - スケーリングなし: オンプレミスフリートを作成し、Deadline Cloud Auto Scaling をオプトアウトしたいと考えています。
   - スケーリングレコメンデーション: Amazon Elastic Compute Cloud (Amazon EC2) フリートを作成しています。
10. フリートのサービスアクセスを選択します。
    a. よりきめ細かいアクセス許可を制御するには、各フリートに新しいサービスロールの作成と使用オプションを使用することをお勧めします。このオプションはデフォルト選択。
    b. サービスロールの選択 を選択して、既存のサービスロールを使用することもできます。
11. 選択内容を確認し、次へ を選択します。
12. フリートのオペレーティングシステムを選択します。フリートのすべてのワーカーには共通のオペレーティングシステムが必要です。
13. ホスト CPU アーキテクチャ を選択します。
14. このフリートのワーカーホストについて、次のハードウェア要件を選択します。
    a. フリートのワークロードの需要を満たすために、vCPU とメモリのハードウェアの最小要件と最大要件を選択します。
    b. （オプション) GPU 要件を選択し、最小 GPU と最大 GPUsを入力します。
15. 選択内容を確認し、次へ を選択します。
16. （オプション) カスタムワーカーの要件を定義します。
17. ドロップダウンを使用して、フリートに関連付けるキューを 1 つ以上選択します。

> **Note**
>
> フリートは、すべて同じ信頼境界にあるキューにのみ関連付けることをお勧めします。これにより、同じワーカーでジョブを実行する間の強力なセキュリティ境界が確保されます。

18. キューの関連付けを確認し、次へ を選択します。

19. (オプション) デフォルトの Conda キュー環境では、ジョブによってリクエストされた Conda パッケージをインストールするキューの環境を作成します。

> **Note**
>
> Conda キュー環境は、ジョブによってリクエストされた Conda パッケージをインストールするために使用されます。CMFs にはデフォルトで必要な Conda コマンドがインストールされないため、通常、CMFs に関連付けられたキューの Conda キュー環境のチェックを解除する必要があります。

20. (オプション) CMF にタグを追加します。詳細については、「AWS リソースのタグ付け」を参照してください。
21. フリート設定を確認し、変更を加えます。
22. [フリートの作成] を選択します。
23. フリート タブを選択し、フリート ID を書き留めます。

AWS CLI

を使用してカスタマーマネージドフリート AWS CLI を作成するには

1. を開きます AWS CLI。
2. 編集 `fleet-trust-policy.json`。
   a. 次の IAM ポリシーを追加し、*ITALICIZED* テキストをアカウント ID と Deadline Cloud ファーム ID に置き換えます AWS 。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "credentials.deadline.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "ACCOUNT_ID"
                },
```

```
                "ArnEquals": {
                    "aws:SourceArn":
 "arn:aws:deadline:*:ACCOUNT_ID:farm/FARM_ID"
                }
            }
        }
    ]
}
```

b. 変更を保存します。

3. 編集 `create-cmf-fleet.json`。

   a. 次の IAM ポリシーを追加します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "deadline:AssumeFleetRoleForWorker",
                "deadline:UpdateWorker",
                "deadline:DeleteWorker",
                "deadline:UpdateWorkerSchedule",
                "deadline:BatchGetJobEntity",
                "deadline:AssumeQueueRoleForWorker"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": "${aws:ResourceAccount}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream"
            ],
            "Resource": "arn:aws:logs:*:*:*:/aws/deadline/*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": "${aws:ResourceAccount}"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:GetLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:*:/aws/deadline/*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": "${aws:ResourceAccount}"
                }
            }
        }
    ]
}
```

b. 変更を保存します。

4. フリート内のワーカーが使用する IAM ロールを追加します。

```
aws iam create-role --role-name FleetWorkerRoleName --assume-role-policy-
document file://fleet-trust-policy.json
aws iam put-role-policy --role-name FleetWorkerRoleName --policy-name
 FleetWorkerPolicy --policy-document file://fleet-policy.json
```

5. 編集 `create-fleet-request.json`。

a. 次の IAM ポリシーを追加し、ITALICIZED テキストを CMF の値に置き換えます。

> **Note**
>
> *`ROLE_ARN`* は にあります`create-cmf-fleet.json`。 *`OS_FAMILY`* の場合は、`linux`、、`macos`または のいずれかを選択する必要があります`windows`。

```
{
    "farmId": "FARM_ID",
    "displayName": "FLEET_NAME",
```

```
    "description": "FLEET_DESCRIPTION",
    "roleArn": "ROLE_ARN",
    "minWorkerCount": 0,
    "maxWorkerCount": 10,
    "configuration": {
        "customerManaged": {
            "mode": "NO_SCALING",
            "workerCapabilities": {
                "vCpuCount": {
                    "min": 1,
                    "max": 4
                },
                "memoryMiB": {
                    "min": 1024,
                    "max": 4096
                },
                "osFamily": "OS_FAMILY",
                "cpuArchitectureType": "x86_64",
            },
        },
    }
}
```

b. 変更を保存します。

6. フリートを作成します。

```
aws deadline create-fleet --cli-input-json file://create-fleet-request.json
```

ワーカーホストのセットアップと設定

ワーカーホストとは、Deadline Cloud ワーカーを実行するホストマシンを指します。このセクションでは、ワーカーホストをセットアップし、特定のニーズに合わせて設定する方法について説明します。各ワーカーホストは、ワーカーエージェント と呼ばれるプログラムを実行します。ワーカーエージェントは、次の作業を行います。

- ワーカーのライフサイクルの管理。
- 割り当てられた作業、その進行状況、結果を同期します。
- 実行中の作業のモニタリング。
- 設定された送信先にログを転送する。

提供された Deadline Cloud ワーカーエージェントを使用することをお勧めします。ワーカーエージェントはオープンソースであり、機能リクエストをお勧めしますが、ニーズに合わせて開発およびカスタマイズすることもできます。

以下のセクションのタスクを完了するには、以下が必要です。

Linux

- Linuxベースの Amazon Elastic Compute Cloud (Amazon EC2) インスタンス。Amazon Linux 2023 をお勧めします。
- sudo 権限。
- Python 3.9 以降。

Windows

- Windowsベースの Amazon Elastic Compute Cloud (Amazon EC2) インスタンス。をお勧めしますWindows Server 2022。
- ワーカーホストへの管理者アクセス
- すべてのユーザーにインストールされた Python 3.9 以降

Python 仮想環境を作成して設定する

Python 3.9 以降をインストールして に配置しLinuxている場合は、 で Python 仮想環境を作成できますPATH。

Python 仮想環境を作成してアクティブ化するには

1. を開きます AWS CLI。
2. Python 仮想環境を作成してアクティブ化します。

```
python3 -m venv /opt/deadline/worker
source /opt/deadline/worker/bin/activate
pip install --upgrade pip
```

Deadline Cloud ワーカーエージェントをインストールする

Python をセットアップし、 で仮想環境を作成したらLinux、Deadline Cloud ワーカーエージェントの Python パッケージをインストールします。

ワーカーエージェントの Python パッケージをインストールするには

1. ターミナルを開きます。
   a. でLinux、`root`ユーザーとしてターミナルを開きます (または `sudo` / を使用します`su`)
   b. でWindows、管理者コマンドプロンプトまたは PowerShellターミナルを開きます。
2. PyPI から Deadline Cloud ワーカーエージェントパッケージをダウンロードしてインストールします。

   > **Note**
   > ではWindows、エージェントファイルは Python のグローバル site-packages ディレクトリにインストールする必要があります。Python 仮想環境は現在サポートされていません。

   ```
   python -m pip install deadline-cloud-worker-agent
   ```

Deadline Cloud ワーカーエージェントを設定する

Deadline Cloud ワーカーエージェントの設定は、3 つの方法で設定できます。を使用してセットアップされたオペレーティングシステムを使用することをお勧めします`install-deadline-worker`。

コマンドライン引数 — コマンドラインから Deadline Cloud ワーカーエージェントを実行するときに引数を指定できます。一部の設定は、コマンドライン引数では利用できません。使用可能なすべてのコマンドライン引数を表示するには、 と入力`deadline-worker-agent --help`して、使用可能なすべてのコマンドライン引数を表示します。

環境変数 — Deadline Cloud ワーカーエージェントを設定するには、 で始まる環境変数を設定します`DEADLINE_WORKER_`。例えば、 `export DEADLINE_WORKER_VERBOSE=true`を使用してワーカーエージェントの出力を詳細に設定できます。その他の例と情報については、Linux「」または`/etc/amazon/deadline/worker.toml.example``C:\ProgramData\Amazon\Deadline\Config\worker.toml.example`「」を参照してくださいWindows。

設定ファイル — ワーカーエージェントをインストールすると、 Linuxまたは `/etc/amazon/deadline/worker.toml`の `C:\ProgramData\Amazon\Deadline\Config\worker.toml` にある設定ファイルが作成されますWindows。ワーカーエージェントは、起動時にこの設定ファイルをロードします。サンプル設定ファイル (`/etc/amazon/deadline/worker.toml.example` Linux または `C:\ProgramData\Amazon\Deadline\Config\worker.toml.example` Windows) を使用して、特定のニーズに合わせてデフォルトのワーカーエージェント設定ファイルをカスタマイズできます。

最後に、ワーカーエージェントの自動シャットダウンを有効にすることをお勧めします。これにより、ワーカーフリートは必要に応じてスケールアップし、レンダリングジョブが終了したときにシャットダウンできます。Auto Scaling は、必要に応じてリソースのみを使用するようにするのに役立ちます。

自動シャットダウンを有効にするには

`root` ユーザーとして：

- パラメータ を使用してワーカーエージェントをインストールします**`--allow-shutdown`**。

  Linux

  次のように入力します。

  ```
  /opt/deadline/worker/bin/install-deadline-worker \
    --farm-id FARM_ID \
    --fleet-id FLEET_ID \
    --region REGION \
    --allow-shutdown
  ```

  Windows

  次のように入力します。

  ```
  install-deadline-worker ^
    --farm-id FARM_ID ^
    --fleet-id FLEET_ID ^
    --region REGION ^
    --allow-shutdown
  ```

ジョブのユーザーとグループを作成する

このセクションでは、エージェントユーザーとキューでjobRunAsUser定義されている との間の必要なユーザーとグループの関係について説明します。

Deadline Cloud ワーカーエージェントは、ホスト上のエージェント固有の専用ユーザーとして実行する必要があります。Deadline Cloud キューの jobRunAsUserプロパティを設定して、ワーカーがキュージョブを特定のオペレーティングシステムのユーザーおよびグループとして実行できるようにする必要があります。つまり、ジョブが持つ共有ファイルシステムのアクセス許可を制御できます。また、ジョブとワーカーエージェントユーザー間の重要なセキュリティ境界として も提供します。

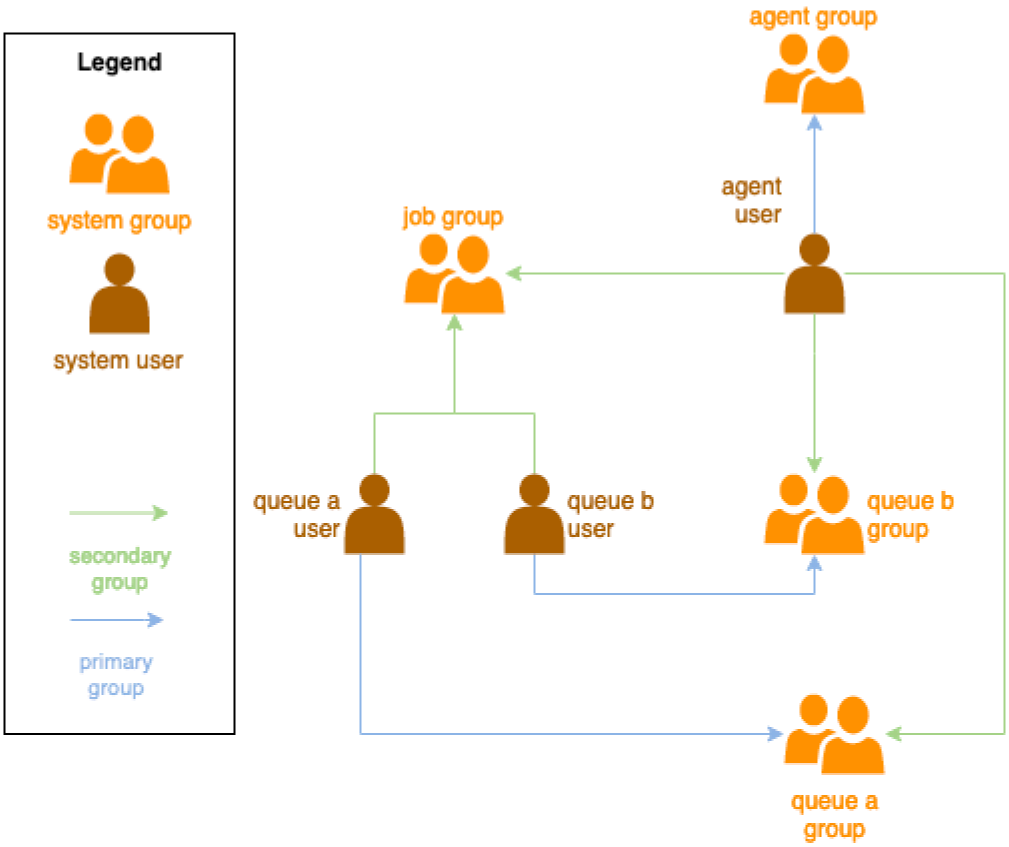
Linux ジョブのユーザーとグループ

エージェントユーザーと を設定するにはjobRunAsUser、次の要件を満たしていることを確認します。

- ごとに グループがありjobRunAsUser、対応する のプライマリグループですjobRunAsUser。
- エージェントユーザーは、ワーカーが作業を取得するキューjobRunAsUserの のプライマリグループに属します。セキュリティのベストプラクティスとして、エージェントユーザーのセカンダリグループとしてこれをお勧めします。この共有グループにより、ワーカーエージェントは実行中にジョブでファイルを使用できるようになります。
- jobRunAsUser は、エージェントユーザーのプライマリグループに属していません。セキュリティのベストプラクティス：
  - ワーカーエージェントによって書き込まれた機密ファイルは、エージェントのプライマリグループによって所有されます。
  - がこのグループにjobRunAsUser属し、ワーカーエージェントが書き込むファイルには、ワーカーで実行されているキューに送信されたジョブからアクセスできる場合があります。
- デフォルトの AWS リージョンは、ワーカーが属するファームのリージョンと一致する必要があります。詳細については、「設定と認証情報ファイルの設定」を参照してください。

  これは、以下に適用する必要があります。
  - エージェントユーザー
  - ワーカーのすべてのキューjobRunAsUserアカウント
- エージェントユーザーは としてsudoコマンドを実行できますjobRunAsUser。

次の図は、エージェントユーザーと、フリートに関連付けられたキューのjobRunAsUserユーザーおよびグループの関係を示しています。



Windows ユーザー

Windows ユーザーを として使用するには`jobRunAsUser`、次の要件を満たしている必要があります。

- すべてのキュー`jobRunAsUser`ユーザーが存在している必要があります。
- パスワードは、キューの `JobRunAsUser` フィールドで指定されたシークレットの値と一致する必要があります。手順については、「」のステップ 7 を参照してください[キューを作成する](#)。
- エージェントユーザーは、それらのユーザーとしてログオンできる必要があります。

Windows ジョブユーザーシークレットへのアクセスを管理する

Windows でキューを設定するときは`jobRunAsUser`、Secrets Manager シー AWS クレットを指定する必要があります。このシークレットの値は、JSON エンコードされた 形式のオブジェクトであることが想定されます。

```
{
    "password": "JOB_USER_PASSWORD"
```

```
}
```

ワーカーがキューの設定済み としてジョブを実行するにはjobRunAsUser、フリートの IAM ロールにシークレットの値を取得するためのアクセス許可が必要です。シークレットがカスタマーマネージド KMS キーを使用して暗号化されている場合、フリートの IAM ロールには KMS キーを使用して復号するアクセス許可も必要です。

これらのシークレットの最小特権の原則に従うことを強くお勧めします。つまり、キューの jobRunAsUser → windows → のシークレット値を取得するためのアクセスpasswordArnは次のようになります。

- フリートとキューの間にキューフリートの関連付けが作成されたときにフリートロールに付与される
- フリートとキューの間でキューとフリートの関連付けが削除されたときにフリートロールから取り消された

さらに、jobRunAsUserパスワードを含む AWS Secrets Manager シークレットは、使用されなくなったときに削除する必要があります。

パスワードシークレットへのアクセスを許可する

Deadline Cloud フリートは、キューとフリートが関連付けられているときに、キューのjobRunAsUserパスワードシークレットに保存されているパスワードにアクセスする必要があります。Secrets Manager リソースポリシーを使用して AWS 、フリートロールへのアクセスを許可することをお勧めします。このガイドラインに厳密に従うことで、シークレットにアクセスできるフリートロールを簡単に判断できます。

シークレットへのアクセスを許可するには

1. AWS Secret Manager コンソールを開いてシークレットを開きます。
2. 「リソースのアクセス許可」セクションに、 形式のポリシーステートメントを追加します。

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

```
        "AWS" : "FLEET_ROLE_ARN"
      },
      "Action" : "secretsmanager:GetSecretValue",
      "Resource" : "*"
    }
    // ...
  ]
}
```

パスワードシークレットへのアクセスを取り消す

フリートがキューへのアクセスが不要になった場合は、キュー のパスワードシークレットへのアクセスを削除しますjobRunAsUser。 AWS Secrets Manager リソースポリシーを使用して、フリートロールへのアクセスを許可することをお勧めします。このガイドラインに厳密に従うことで、シークレットにアクセスできるフリートロールを簡単に判断できます。

シークレットへのアクセスを取り消すには

1. AWS Secret Manager コンソールを開いてシークレットを開きます。
2. 「リソースのアクセス許可」セクションで、 フォームのポリシーステートメントを削除します。

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    // ...
    {
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "FLEET_ROLE_ARN"
      },
      "Action" : "secretsmanager:GetSecretValue",
      "Resource" : "*"
    }
    // ...
  ]
}
```

ジョブに必要なソフトウェアをインストールして設定する

Deadline Cloud ワーカーエージェントを設定したら、ジョブの実行に必要なソフトウェアを使用してワーカーホストを準備できます。

関連付けられた を持つキューにジョブを送信すると`jobRunAsUser`、ジョブはそのユーザーとして実行されます。すべてのコマンドは、そのユーザーの `PATH` で利用できる必要があります。

Linux では、次のいずれかで`PATH`ユーザーの を指定できます。

- `~/.bashrc` または `~/.bash_profile`
- `/etc/profile.d/*` や などのシステム設定ファイル `/etc/profile`
- シェル起動スクリプト: `/etc/bashrc`。

Windows では、次のいずれかで`PATH`ユーザーの を指定できます。

- ユーザー固有の環境変数
- システム全体の環境変数

デジタルコンテンツ作成ツールアダプターをインストールする

Deadline Cloud は、ファーストパーティー統合サポートを備えたデジタルコンテンツ作成 (DCC) アプリケーションを提供します。これらの統合をカスタマーマネージドフリートで使用するには、DCC ソフトウェアとアダプターをインストールする必要があります。

カスタマーマネージドフリートに DCC アダプターをインストールするには

1. ターミナルを開きます。
    a. Linux では、`root`ユーザーとしてターミナルを開きます (または sudo / を使用します`su`)
    b. Windows で、管理者コマンドプロンプトまたは PowerShell ターミナルを開きます。
2. Deadline Cloud アダプターパッケージをインストールします。

```
pip install deadline deadline-cloud-for-maya deadline-cloud-for-nuke deadline-cloud-for-blender
```

AWS 認証情報の設定

このセクションでは、 AWS 認証情報を設定する方法について説明します。

ワーカーライフサイクルの初期フェーズはブートストラップです。このフェーズでは、ワーカーエージェントソフトウェアがフリートにワーカーを作成し、フリートのロールから AWS 認証情報を取得してさらに操作します。

AWS credentials for Amazon EC2

Amazon EC2 の AWS 認証情報を設定するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインでロールを選択し、ロールを作成します。
3. AWS サービス を選択します。
4. EC2 をサービスまたはユースケース として選択し、次へ を選択します。
5. `AWSDeadlineCloud-WorkerHost` AWS 管理ポリシーをアタッチします。

On-premise AWS credentials

AWS オンプレミス認証情報を設定するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインでロールを選択し、ロールを作成します。
3. を選択しAWS アカウント、次へ を選択します。
4. `AWSDeadlineCloud-WorkerHost` AWS 管理ポリシーをアタッチします。
5. AWS IAM ユーザーの IAM アクセスキーとシークレットキーを生成します。
    a. IAM Role Anywhere については、「IAM Roles Anywhere」を参照してください。
    b. ホストで認証情報を設定する最も安全な方法については、AWS Identity and Access Management Roles Anywhere からの一時的なセキュリティ認証情報の取得」を参照してください。
    c. CLI を代替認証として使用することもできます。詳細については、「IAM ユーザー認証情報による認証」を参照してください。
6. これらのキーは、ワーカーホストファイルシステムの エージェントユーザーの AWS 認証情報ファイルに保存します。

a. Linux の場合、これは にあります。 `~/.aws/credentials`

b. Windows では、これは にあります。 `%USERPROFILE%\.aws\credentials`

> **Note**
>
> 認証情報には、ワーカーエージェントをインストールした OS ユーザー名 (`deadline-worker-agent`) でのみアクセスできるようにする必要があります。

```
# Replace keys below
[default]
aws_access_key_id=ACCESS_KEY_ID
aws_secret_access_key=SECRET_ACCESSS_KEY
```

7. `deadline-worker-agent` 所有者とアクセス許可を変更します。

> **Note**
>
> ワーカーエージェントのインストール時に OS ユーザー (`deadline-worker-agent`) 名を変更した場合は、代わりにその名前を使用します。

Amazon Machine Image の作成

Amazon Elastic Compute Cloud Amazon Machine Image (Amazon EC2AMI) カスタマーマネージドフリート (CMF) で使用する () を作成するには、このセクションのタスクを完了します。先に進む前に、Amazon EC2 インスタンスを作成する必要があります。詳細については、「Linux インスタンス用 Amazon EC2 ユーザーガイド」の「インスタンスの起動」を参照してください。 Amazon EC2

> **Important**
>
> を作成すると、Amazon EC2 インスタンスのアタッチされたボリュームのスナップショットAMIが作成されます。インスタンスにインストールされているソフトウェアは保持されるため、インスタンスは からインスタンスを起動するときに再利用されますAMI。フリートに適用する前に、パッチ適用戦略を採用し、更新されたソフトウェアAMIで新しい を定期的に更新することをお勧めします。

Amazon EC2 インスタンスを準備する

を構築する前にAMI、ワーカーの状態を削除する必要があります。ワーカー状態は、ワーカーエージェントの起動後も維持されます。この状態が に保持される場合AMI、そこから起動されたすべてのインスタンスが同じ状態を共有します。

また、既存のログファイルをすべて削除することをお勧めします。AMI を準備するときに、ログファイルを Amazon EC2 インスタンスに残すことができます。これらのファイルを削除すると、AMI を使用するワーカーフリートで発生する可能性のある問題を診断する際の混乱が最小限に抑えられます。

また、Amazon EC2 の起動時に Deadline Cloud ワーカーエージェントが起動するように、ワーカーエージェントシステムサービスを有効にする必要があります。

最後に、ワーカーエージェントの自動シャットダウンを有効にすることをお勧めします。これにより、ワーカーフリートは必要に応じてスケールアップし、レンダリングジョブが終了したときにシャットダウンできます。この自動スケーリングは、必要に応じてリソースのみを使用するようにするのに役立ちます。

Amazon EC2 インスタンスを準備するには

1. Amazon EC2 コンソールを開きます。
2. Amazon EC2 インスタンスの起動 詳細については、「インスタンスの起動」を参照してください。
3. ID プロバイダー (IdP) に接続するようにホストを設定し、必要な共有ファイルシステムをマウントします。
4. チュートリアルに従って、、Deadline Cloud ワーカーエージェントをインストールする、ワーカーエージェントを設定するおよび を実行しますジョブのユーザーとグループを作成する。
5. VFX リファレンスプラットフォームと互換性のあるソフトウェアを実行するために Amazon Linux 2023 AMIに基づく を準備する場合は、いくつかの要件を更新する必要があります。詳細については、「VFX Reference Platform の互換性」を参照してください。
6. ターミナルを開きます。

   a. Linux では、`root`ユーザーとしてターミナルを開きます (または `sudo` / を使用します`su`)

   b. Windows で、管理者コマンドプロンプトまたは PowerShellターミナルを開きます。
7. ワーカーサービスが実行されておらず、起動時に起動するように設定されていることを確認します。

a. Linux では、 を実行します。

```
systemctl stop deadline-worker
systemctl enable deadline-worker
```

b. Windows では、 を実行します。

```
sc.exe stop DeadlineWorker
sc.exe config DeadlineWorker start= auto
```

8. ワーカーの状態を削除します。

   a. Linux では、 を実行します。

```
rm -rf /var/lib/deadline/*
```

   b. Windows では、 を実行します。

```
del /Q /S %PROGRAMDATA%\Amazon\Deadline\Cache\*
```

9. ログファイルを削除します。

   a. Linux では、 を実行します。

```
rm -rf /var/log/amazon/deadline/*
```

   b. Windows では、 を実行します。

```
del /Q /S %PROGRAMDATA%\Amazon\Deadline\Logs\*
```

10. Windows では、スタートメニューにある Amazon EC2Launch Settings アプリケーションを実行して、インスタンスの最終ホスト準備とシャットダウンを完了することをお勧めします。

> **Note**
>
> Sysprep なしでシャットダウンを選択し、Sysprep でシャットダウンを選択しないでください。Sysprep でシャットダウンすると、すべてのローカルユーザーが使用できなくなります。詳細については、Windows インスタンス用ユーザーガイドの「カスタム AMI の作成」トピックの「開始する前に」セクションを参照してください。

を構築する AMI

を構築するには AMI

1. Amazon EC2 コンソールを開きます。
2. ナビゲーションペインでインスタンスを選択し、インスタンスを選択します。
3. インスタンス状態 を選択し、次にインスタンスを停止 を選択します。
4. インスタンスが停止されたら、アクション を選択します。
5. イメージとテンプレート を選択し、イメージ を作成します。
6. イメージ名 を入力します。
7. （オプション）イメージの説明を入力します。
8. [イメージを作成] を選択します。

Amazon EC2 Auto Scaling グループを使用してフリートインフラストラクチャを作成する

このセクションでは、Amazon EC2 Auto Scaling フリートを作成する方法について説明します。

以下の AWS CloudFormation YAML テンプレートを使用して、Amazon EC2 Auto Scaling (Auto Scaling) グループ、2 つのサブネット、インスタンスプロファイル、およびインスタンスアクセスロールを持つ Amazon Virtual Private Cloud (Amazon VPC) を作成します。これらは、サブネットで Auto Scaling を使用してインスタンスを起動するために必要です。

レンダリングニーズに合わせてインスタンスタイプのリストを確認して更新する必要があります。

Amazon EC2 Auto Scaling フリートを作成するには

1. https://console.aws.amazon.com/cloudformation で AWS CloudFormation コンソールを開きます。
2. パラメータ `Farm ID`、、`Fleet ID`および を使用して CloudFormation テンプレートを作成します`AMI ID`。

```
AWSTemplateFormatVersion: 2010-09-09
Description: Amazon Deadline Cloud customer-managed fleet
Parameters:
  FarmId:
    Type: String
```

```
    Description: Farm ID
  FleetId:
    Type: String
    Description: Fleet ID
  AMIId:
    Type: String
    Description: AMI ID for launcing Workers
Resources:
  deadlineVPC:
    Type: 'AWS::EC2::VPC'
    Properties:
      CidrBlock: 100.100.0.0/16
  deadlineWorkerSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: !Join
        - ' '
        - - Security Group created for deadline workers in fleet
          - !Ref FleetId
      GroupName: !Join
        - ''
        - - deadlineWorkerSecurityGroup-
          - !Ref FleetId
      SecurityGroupEgress:
        - CidrIp: 0.0.0.0/0
          IpProtocol: '-1'
      SecurityGroupIngress: []
      VpcId: !Ref deadlineVPC
  deadlineIGW:
    Type: 'AWS::EC2::InternetGateway'
    Properties: {}
  deadlineVPCGatewayAttachment:
    Type: 'AWS::EC2::VPCGatewayAttachment'
    Properties:
      VpcId: !Ref deadlineVPC
      InternetGatewayId: !Ref deadlineIGW
  deadlinePublicRouteTable:
    Type: 'AWS::EC2::RouteTable'
    Properties:
      VpcId: !Ref deadlineVPC
  deadlinePublicRoute:
    Type: 'AWS::EC2::Route'
    Properties:
      RouteTableId: !Ref deadlinePublicRouteTable
```

```
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref deadlineIGW
    DependsOn:
      - deadlineIGW
      - deadlineVPCGatewayAttachment
  deadlinePublicSubnet0:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref deadlineVPC
      CidrBlock: 100.100.16.0/22
      AvailabilityZone: !Join
        - ''
        - - !Ref 'AWS::Region'
          - a
  deadlineSubnetRouteTableAssociation0:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref deadlinePublicRouteTable
      SubnetId: !Ref deadlinePublicSubnet0
  deadlinePublicSubnet1:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref deadlineVPC
      CidrBlock: 100.100.20.0/22
      AvailabilityZone: !Join
        - ''
        - - !Ref 'AWS::Region'
          - c
  deadlineSubnetRouteTableAssociation1:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref deadlinePublicRouteTable
      SubnetId: !Ref deadlinePublicSubnet1
  deadlineInstanceAccessAccessRole:
    Type: 'AWS::IAM::Role'
    Properties:
      RoleName: !Join
        - '-'
        - - deadline
          - InstanceAccess
          - !Ref FleetId
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
```

```
            Principal:
              Service: ec2.amazonaws.com
            Action:
              - 'sts:AssumeRole'
      Path: /
      ManagedPolicyArns:
        - 'arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy'
        - 'arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore'
        - 'arn:aws:iam::aws:policy/AWSDeadlineCloud-WorkerHost'
  deadlineInstanceProfile:
    Type: 'AWS::IAM::InstanceProfile'
    Properties:
      Path: /
      Roles:
        - !Ref deadlineInstanceAccessAccessRole
  deadlineLaunchTemplate:
    Type: 'AWS::EC2::LaunchTemplate'
    Properties:
      LaunchTemplateName: !Join
        - ''
        - - deadline-LT-
          - !Ref FleetId
      LaunchTemplateData:
        NetworkInterfaces:
          - DeviceIndex: 0
            AssociatePublicIpAddress: true
            Groups:
              - !Ref deadlineWorkerSecurityGroup
            DeleteOnTermination: true
        ImageId: !Ref AMIId
        InstanceInitiatedShutdownBehavior: terminate
        IamInstanceProfile:
          Arn: !GetAtt
            - deadlineInstanceProfile
            - Arn
        MetadataOptions:
          HttpTokens: required
          HttpEndpoint: enabled

  deadlineAutoScalingGroup:
    Type: 'AWS::AutoScaling::AutoScalingGroup'
    Properties:
      AutoScalingGroupName: !Join
        - ''
```

```
          - - deadline-ASG-autoscalable-
            - !Ref FleetId
        MinSize: 0
        MaxSize: 10
        VPCZoneIdentifier:
          - !Ref deadlinePublicSubnet0
          - !Ref deadlinePublicSubnet1
        NewInstancesProtectedFromScaleIn: true
        MixedInstancesPolicy:
          InstancesDistribution:
            OnDemandBaseCapacity: 0
            OnDemandPercentageAboveBaseCapacity: 0
            SpotAllocationStrategy: capacity-optimized
            OnDemandAllocationStrategy: lowest-price
          LaunchTemplate:
            LaunchTemplateSpecification:
              LaunchTemplateId: !Ref deadlineLaunchTemplate
              Version: !GetAtt
                - deadlineLaunchTemplate
                - LatestVersionNumber
            Overrides:
              - InstanceType: m5.large
              - InstanceType: m5d.large
              - InstanceType: m5a.large
              - InstanceType: m5ad.large
              - InstanceType: m5n.large
              - InstanceType: m5dn.large
              - InstanceType: m4.large
              - InstanceType: m3.large
              - InstanceType: r5.large
              - InstanceType: r5d.large
              - InstanceType: r5a.large
              - InstanceType: r5ad.large
              - InstanceType: r5n.large
              - InstanceType: r5dn.large
              - InstanceType: r4.large
        MetricsCollection:
          - Granularity: 1Minute
            Metrics:
              - GroupMinSize
              - GroupMaxSize
              - GroupDesiredCapacity
              - GroupInServiceInstances
              - GroupTotalInstances
```

```
            - GroupInServiceCapacity
            - GroupTotalCapacity
```

3. IAM ロールを作成したら、以下を確認する必要があります。

   - ワーカーの Amazon EC2 インスタンスにアタッチされた IAM ロールからの認証情報は、ジョブを含む、そのワーカーで実行されているすべてのプロセスで使用できます。ワーカーには、操作する権限が最小限必要です。 `deadline:CreateWorker` `deadline:AssumeFleetRoleForWorker.`
   - ワーカーエージェントは、キューロールの認証情報を取得し、ジョブの実行で使用するように設定します。Amazon EC2 インスタンスプロファイルロールには、ジョブに必要なアクセス許可を含めないでください。

Deadline Cloud スケールレコメンデーション機能を使用して Amazon EC2 フリートを自動スケーリングする

Deadline Cloud は、Amazon EC2 Auto Scaling (Auto Scaling) グループを活用して、Amazon EC2 カスタマーマネージドフリート (CMF) を自動的にスケーリングします。フリートを自動スケーリングするには、フリートモードを設定し、必要なインフラストラクチャをアカウントにデプロイする必要があります。デプロイしたインフラストラクチャはすべてのフリートで動作するため、設定する必要があるのは 1 回だけです。

基本的なワークフローは、フリートモードを自動スケーリングするように設定すると、Deadline Cloud は、推奨フリートサイズが変更されるたびに (1 つの EventBridge イベントにフリート ID、推奨フリートサイズ、およびその他のメタデータが含まれる) 、そのフリートのイベントを送信します。関連するイベントをフィルタリングする EventBridge ルールがあり、それらを使用する Lambda があります。Lambda は Amazon EC2 Auto Scaling と統合され`AutoScalingGroup`、Amazon EC2 フリートを自動的にスケーリングします。

フリートモードを に設定する **`EVENT_BASED_AUTO_SCALING`**

フリートモードを に設定します`EVENT_BASED_AUTO_SCALING`。コンソールを使用してこれを行うか、 AWS CLI を使用して `CreateFleet`または `UpdateFleet` API を直接呼び出すことができます。モードが設定されると、Deadline Cloud は推奨フリートサイズが変更されるたびにイベントの送信 EventBridge を開始します。

- `UpdateFleet` コマンドの例：

```
aws deadline update-fleet \
```

```
  --farm-id FARM_ID \
  --fleet-id FLEET_ID \
  --configuration file://configuration.json
```

- CreateFleet コマンドの例：

```
aws deadline create-fleet \
  --farm-id FARM_ID \
  --display-name "Fleet name" \
  --max-worker-count 10 \
  --configuration file://configuration.json
```

以下は、上記の CLI コマンド () configuration.jsonで使用される の例です--configuration file://configuration.json。

- フリートで Auto Scaling を有効にするには、 モードを に設定する必要がありますEVENT_BASED_AUTO_SCALING。
- workerCapabilities は、作成時に CMF に割り当てられたデフォルト値です。CMF で使用できるリソースを増やす必要がある場合は、これらの値を変更できます。

フリートモードを設定すると、Deadline Cloud はそのフリートのフリートサイズのレコメンデーションイベントの発行を開始します。

```
{
    "customerManaged": {
        "mode": "EVENT_BASED_AUTO_SCALING",
        "workerCapabilities": {
            "vCpuCount": {
                "min": 1,
                "max": 4
            },
            "memoryMiB": {
                "min": 1024,
                "max": 4096
            },
            "osFamily": "linux",
            "cpuArchitectureType": "x86_64",
        }
    }
}
```

AWS CloudFormation テンプレートを使用して Auto Scaling スタックをデプロイする

イベントをフィルタリングする EventBridge ルール、イベントを消費して Auto Scaling を制御する Lambda、未処理のイベントを保存する SQS キューを設定できます。次の AWS CloudFormation テンプレートを使用して、すべてをスタックにデプロイします。リソースを正常にデプロイすると、ジョブを送信でき、フリートは自動的にスケールアップします。

```
Resources:
  AutoScalingLambda:
    Type: 'AWS::Lambda::Function'
    Properties:
      Code:
        ZipFile: |-
          """
          This lambda is configured to handle "Fleet Size Recommendation Change"
          messages. It will handle all such events, and requires
          that the ASG is named based on the fleet id. It will scale up/down the fleet
          based on the recommended fleet size in the message.

          Example EventBridge message:
          {
              "version": "0",
              "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
              "detail-type": "Fleet Size Recommendation Change",
              "source": "aws.deadline",
              "account": "111122223333",
              "time": "2017-12-22T18:43:48Z",
              "region": "us-west-1",
              "resources": [],
              "detail": {
                  "farmId": "farm-12345678900000000000000000000000",
                  "fleetId": "fleet-12345678900000000000000000000000",
                  "oldFleetSize": 1,
                  "newFleetSize": 5,
              }
          }
          """

          import json
          import boto3
          import logging

          logger = logging.getLogger()
```

```
          logger.setLevel(logging.INFO)

          auto_scaling_client = boto3.client("autoscaling")

          def lambda_handler(event, context):
              logger.info(event)
              event_detail = event["detail"]
              fleet_id = event_detail["fleetId"]
              desired_capacity = event_detail["newFleetSize"]

              asg_name = f"deadline-ASG-autoscalable-{fleet_id}"
              auto_scaling_client.set_desired_capacity(
                  AutoScalingGroupName=asg_name,
                  DesiredCapacity=desired_capacity,
                  HonorCooldown=False,
              )

              return {
                  'statusCode': 200,
                  'body': json.dumps(f'Successfully set desired_capacity for {asg_name}
 to {desired_capacity}')
              }
      Handler: index.lambda_handler
      Role: !GetAtt
        - AutoScalingLambdaServiceRole
        - Arn
      Runtime: python3.11
    DependsOn:
      - AutoScalingLambdaServiceRoleDefaultPolicy
      - AutoScalingLambdaServiceRole
  AutoScalingEventRule:
    Type: 'AWS::Events::Rule'
    Properties:
      EventPattern:
        source:
          - aws.deadline
        detail-type:
          - Fleet Size Recommendation Change
      State: ENABLED
      Targets:
        - Arn: !GetAtt
            - AutoScalingLambda
            - Arn
          DeadLetterConfig:
```

```
          Arn: !GetAtt
            - UnprocessedAutoScalingEventQueue
            - Arn
        Id: Target0
        RetryPolicy:
          MaximumRetryAttempts: 15
  AutoScalingEventRuleTargetPermission:
    Type: 'AWS::Lambda::Permission'
    Properties:
      Action: 'lambda:InvokeFunction'
      FunctionName: !GetAtt
        - AutoScalingLambda
        - Arn
      Principal: events.amazonaws.com
      SourceArn: !GetAtt
        - AutoScalingEventRule
        - Arn
  AutoScalingLambdaServiceRole:
    Type: 'AWS::IAM::Role'
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: 'sts:AssumeRole'
            Effect: Allow
            Principal:
              Service: lambda.amazonaws.com
        Version: 2012-10-17
      ManagedPolicyArns:
        - !Join
          - ''
          - - 'arn:'
            - !Ref 'AWS::Partition'
            - ':iam::aws:policy/service-role/AWSLambdaBasicExecutionRole'
  AutoScalingLambdaServiceRoleDefaultPolicy:
    Type: 'AWS::IAM::Policy'
    Properties:
      PolicyDocument:
        Statement:
          - Action: 'autoscaling:SetDesiredCapacity'
            Effect: Allow
            Resource: '*'
        Version: 2012-10-17
      PolicyName: AutoScalingLambdaServiceRoleDefaultPolicy
      Roles:
```

```
        - !Ref AutoScalingLambdaServiceRole
  UnprocessedAutoScalingEventQueue:
    Type: 'AWS::SQS::Queue'
    Properties:
      QueueName: deadline-unprocessed-autoscaling-events
    UpdateReplacePolicy: Delete
    DeletionPolicy: Delete
  UnprocessedAutoScalingEventQueuePolicy:
    Type: 'AWS::SQS::QueuePolicy'
    Properties:
      PolicyDocument:
        Statement:
          - Action: 'sqs:SendMessage'
            Condition:
              ArnEquals:
                'aws:SourceArn': !GetAtt
                  - AutoScalingEventRule
                  - Arn
            Effect: Allow
            Principal:
              Service: events.amazonaws.com
            Resource: !GetAtt
              - UnprocessedAutoScalingEventQueue
              - Arn
        Version: 2012-10-17
      Queues:
        - !Ref UnprocessedAutoScalingEventQueue
```

カスタマーマネージドフリートをライセンスエンドポイントに接続する

AWS Deadline Cloud (Deadline Cloud) の使用状況ベースのライセンスサーバーは、一部のサードパーティー製品のオンデマンドライセンスを提供します。これにより、支払いをそのまま行うことができます。変更されるのは、使用した時間のみです。

Deadline Cloud の使用状況ベースのライセンスサーバーは、Deadline Cloud ワーカーがライセンスサーバーと通信できる限り、任意のフリートタイプで使用できます。これは、サービスマネージドフリートで自動的に設定されます。この設定は、カスタマーマネージドフリートでのみ必要です。

ライセンスサーバーを作成するには、以下が必要です。

- サードパーティーライセンスのトラフィックを許可するファームの VPC のセキュリティグループ。

- Deadline Cloud ライセンスエンドポイントオペレーションへのアクセスを許可するポリシーがアタッチされた AWS Identity and Access Management (IAM) ロール。

トピック

ステップ 1: セキュリティグループを作成する

Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を使用して、ファームの VPC のセキュリティグループを作成します。次のインバウンドルールを許可するようにセキュリティグループを設定します。

- Autodesk Maya と Arnold – 2701 - 2702、TCP、IPv4
- Autodesk 3ds Max – 2704、TCP、IPv4
- Foundry Nuke – 6101、TCP、IPv4
- SideFX Houdini、Mantra、Karma – 1715 - 1717、TCP、IPv4

各インバウンドルールのソースは、フリートのワーカーセキュリティグループです。

セキュリティグループの作成の詳細については、Amazon Virtual Private Cloud ユーザーガイドの「セキュリティグループの作成」を参照してください。

ステップ 2: ライセンスエンドポイントを設定する

ライセンスエンドポイントは、サードパーティー製品のライセンスサーバーへのアクセスを提供します。ライセンスリクエストはライセンスエンドポイントに送信されます。エンドポイントは、それらを適切なライセンスサーバーにルーティングします。ライセンスサーバーは、使用制限と使用権限を追跡します。作成するライセンスエンドポイントごとに料金が発生します。詳細については、「Amazon VPC の料金」を参照してください。

適切なアクセス許可 AWS Command Line Interface を持つ からライセンスエンドポイントを作成できます。ライセンスエンドポイントの作成に必要なポリシーについては、「ライセンスエンドポイントの作成を許可するポリシー」を参照してください。

AWS CloudShell（https://console.aws.amazon.com/cloudshell/) またはその他の AWS CLI 環境を使用して、次の AWS Command Line Interface コマンドを使用してライセンスエンドポイントを設定できます。

1. ライセンスエンドポイントを作成します。セキュリティグループ ID、サブネット ID、および VPC ID を、前に作成した値に置き換えます。複数のサブネットを使用する場合は、スペースで区切ります。

```
aws deadline create-license-endpoint \
  --security-group-id SECURITY_GROUP_ID \
  --subnet-ids SUBNET_ID1 SUBNET_ID2 \
  --vpc-id VPC_ID
```

2. 次のコマンドを使用して、エンドポイントが正常に作成されたことを確認します。VPC エンドポイントの DNS 名を覚えておいてください。

```
aws deadline get-license-endpoint \
  --license-endpoint-id LICENSE_ENDPOINT_ID
```

3. 使用可能な計測済み製品のリストを表示します。

```
aws deadline list-available-metered-products
```

4. 次のコマンドを使用して、計測対象製品をライセンスエンドポイントに追加します。

```
aws deadline put-metered-product \
--license-endpoint-id LICENSE_ENDPOINT_ID \
--product-id PRODUCT_ID
```

`remove-metered-product` コマンドを使用して、ライセンスエンドポイントから製品を削除できます。

```
aws deadline remove-metered-product \
  --license-endpoint-id LICENSE_ENDPOINT_ID \
  --productId PRODUCT_ID
```

コマンドを使用してライセンスエンドポイントを削除できます`delete-license-endpoint`。

```
aws deadline delete-license-endpoint \
```

```
--license-endpoint-id LICENSE_ENDPOINT_ID
```

ステップ 3: レンダリングアプリケーションをエンドポイントに接続する

ライセンスエンドポイントを設定すると、アプリケーションはサードパーティーのライセンスサーバーを使用するのと同じ方法でそれを使用します。通常、アプリケーションのライセンスサーバーを設定するには、環境変数または Microsoft Windows レジストリキーなどの他のシステム設定をライセンスサーバーのポートとアドレスに設定します。

ライセンスエンドポイントの DNS 名を取得するには、次の AWS CLI コマンドを使用します。

```
aws deadline get-license-endpoint
```

または、Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を使用して、前のステップで Deadline Cloud API によって作成された VPC エンドポイントを識別することもできます。

設定例

Example – Autodesk Maya と Arnold

環境変数`ADSKFLEX_LICENSE_FILE`を次のように設定します。

```
2702@VPC_Endpoint_DNS_Name:2701@VPC_Endpoint_DNS_Name
```

> **Note**
>
> Windows ワーカーの場合、エンドポイントを区切るには、コロン (:) の代わりにセミコロン (;) を使用します。

Example – Autodesk 3ds Max

環境変数`ADSKFLEX_LICENSE_FILE`を次のように設定します。

```
2704@VPC_Endpoint_DNS_Name
```

Example – ファウンドリー・ ヌーク

環境変数を に設定する `foundry_LICENSE` `6101@`*`VPC_Endpoint_DNS_Name`* ライセンスが正しく機能していることをテストするには、ターミナルで Nuke を実行します。

```
~/nuke/Nuke14.0v5/Nuke14.0 -x
```

Example – SideFX Houdini、Mantra、および Karma

次のコマンドを実行します。

```
/opt/hfs19.5.640/bin/hserver -S
 "http://VPC_Endpoint_DNS_Name:1715;http://VPC_Endpoint_DNS_Name:1716;http://
VPC_Endpoint_DNS_Name:1717;"
```

ライセンスが正常に機能していることをテストするには、次のコマンドを使用して Houdini シーンをレンダリングします。

```
/opt/hfs19.5.640/bin/hython ~/forpentest.hip -c "hou.node('/out/mantra1').render()"
```

Deadline Cloud でのユーザーの管理

AWS Deadline Cloud は AWS IAM Identity Center を使用してユーザーとグループを管理します。IAM Identity Center は、エンタープライズシングルサインオン (SSO) プロバイダーと統合できるクラウドベースのシングルサインオンサービスです。統合により、ユーザーは会社アカウントでサインインできます。

Deadline Cloud はデフォルトで IAM Identity Center を有効にし、Deadline Cloud をセットアップして使用する必要があります。詳細については、「ID ソースの管理」を参照してください。

の組織所有者 AWS Organizations は、Deadline Cloud モニターにアクセスできるユーザーとグループを管理する責任があります。これらのユーザーとグループは、IAM Identity Center または Deadline Cloud コンソールを使用して作成および管理できます。詳細については、「AWS Organizationsとは?」を参照してください。

Deadline Cloud コンソールを使用して、モニターを使用してファーム、キュー、フリートを管理できるユーザーとグループを作成および削除します。Deadline Cloud にユーザーを追加する場合、ユーザーはアクセスする前に IAM Identity Center を使用してパスワードをリセットする必要があります。

トピック

- モニターのユーザーとグループの管理
- ファーム、キュー、フリートのユーザーとグループを管理する

モニターのユーザーとグループの管理

Organizations の所有者は、Deadline Cloud コンソールを使用して、Deadline Cloud モニターにアクセスできるユーザーとグループを管理できます。既存の IAM Identity Center ユーザーとグループから選択することも、コンソールから新しいユーザーとグループを追加することもできます。

1. にサインイン AWS Management Console し、Deadline Cloud コンソール を開きます。メインページの「開始方法」セクションで、「Deadline Cloud のセットアップ」または「ダッシュボードに移動」を選択します。
2. 左側のナビゲーションペインで、ユーザー管理 を選択します。デフォルトでは、グループタブが選択されています。

実行するアクションに応じて、グループタブまたはユーザータブを選択します。

Monitor groups

グループを作成するには

1. [Create group] (グループの作成) を選択します。
2. グループ名を入力します。名前は、IAM Identity Center 組織内のグループ間で一意である必要があります。

グループを削除するには

1. 削除するグループを選択します。
2. [削除] を選択します。
3. 確認ダイアログで、グループの削除 を選択します。

> **Note**
> IAM Identity Center からグループを削除しています。グループメンバーは Deadline Cloud にサインインしたり、ファームリソースにアクセスしたりできなくなります。

Monitor users

ユーザーを追加するには

1. [ユーザー] タブを選択します。
2. [ユーザーの追加] を選択します。
3. 新しいユーザーの名前、E メールアドレス、ユーザー名を入力します。
4. 必要に応じて、新しいユーザーを追加する IAM Identity Center グループを 1 つ以上選択します。
5. 招待を送信 を選択して、IAM Identity Center 組織に参加する手順が記載された E メールを新しいユーザーに送信します。

ユーザーを削除するには、次の手順を実行します

1. モニタから削除するユーザーを選択します。
2. [削除] を選択します。

3. 確認ダイアログで、ユーザーの削除 を選択します。

> **Note**
>
> IAM Identity Center からユーザーを削除します。ユーザーは Deadline Cloud モニターにサインインしたり、ファームリソースにアクセスしたりできなくなります。

ファーム、キュー、フリートのユーザーとグループを管理する

1. まだ にサインインしていない場合は、 にサインイン AWS Management Console し、Deadline Cloud [コンソール](#) を開きます。
2. 左側のナビゲーションペインで、ファームおよびその他のリソース を選択します。
3. 管理するファームを選択します。ファーム名を選択して詳細ページを開きます。検索バーを使用してファームを検索できます。
4. キューまたはフリートを管理するには、キューまたはフリート タブを選択し、管理するキューまたはフリートを選択します。
5. アクセス管理タブを選択します。デフォルトでは、グループタブが選択されています。ユーザーを管理するには、トグルをユーザー に移動します。

実行するアクションに応じて、グループタブまたはユーザータブを選択します。

アクセスレベルの定義については、「アクセス[許可](#)」を参照してください。

Groups

グループを追加するには

1. グループトグルを選択します。
2. [Add Group (グループの追加)] を選択します。
3. ドロップダウンから、追加するグループを選択します。
4. グループアクセスレベルには、次のいずれかのオプションを選択します。
   - 表示者
   - 寄稿者
   - マネージャー

- [所有者]

5. [追加] を選択します。

グループを削除するには

1. 削除するグループを選択します。
2. [削除] を選択します。
3. 確認ダイアログで、[Remove] を選択します。

Users

ユーザーを追加するには

1. ユーザーを追加するには、ユーザーの追加 を選択します。
2. ドロップダウンから、ファームに追加するユーザーを選択します。
3. ユーザーアクセスレベルには、次のいずれかのオプションを選択します。
   - 表示者
   - 寄稿者
   - マネージャー
   - [所有者]
4. [追加] を選択します。ユーザーはファームに追加されます。

ユーザーを削除するには

1. 削除するユーザーを選択します。
2. 確認の削除ダイアログで、 の削除を選択します。その後、ユーザーは選択したファームから削除されます。

https://console.aws.amazon.com/singlesignon/ の IAM Identity Center コンソールを使用して、ユーザーおよびグループのファーム許可を追加または削除することもできます。

Deadline Cloud ジョブ

ジョブは、Deadline Cloud AWS が利用可能なワーカーの作業をスケジュールして実行するために使用する一連の手順です。ジョブを作成するときは、ジョブの送信先のファームとキューを選択します。また、ワーカーが処理する手順を提供する JSON または YAML ファイルも指定します。Deadline Cloud は、ジョブを記述するための Open Job Description (OpenJD) 仕様に従うジョブテンプレートを受け入れます。詳細については、 GitHub ウェブサイトの「オープンジョブの説明ドキュメント」を参照してください。

ジョブは以下で構成されます。

- ステップ – ワーカーで実行するスクリプトを定義します。ステップには、ワーカーの最小メモリや、最初に完了する必要があるその他のステップなどの要件があります。各ステップには 1 つ以上のタスクがあります。
- タスク – ワーカーに送信された作業単位。タスクは、ステップのスクリプトと、スクリプトで使用されるフレーム番号などのパラメータの組み合わせです。ジョブは、すべてのステップのすべてのタスクが完了すると完了します。
- 環境 – 複数のステップまたはタスクで共有される指示を設定および削除します。

ジョブは、次のいずれかの方法で作成できます。

- Deadline Cloud 送信者を使用します。
- ジョブバンドルを作成し、Deadline Cloud コマンドラインインターフェイス (Deadline Cloud CLI) を使用します。
- AWS SDK を使用します。
- AWS Command Line Interface () を使用しますAWS CLI。

送信者は、デジタルコンテンツ作成 (DCC) ソフトウェアへのインターフェイスでジョブの作成を管理するためのプラグインです。ジョブを作成したら、送信者を使用して処理のために Deadline Cloud に送信します。裏では、送信者はジョブを記述する OpenJD ジョブテンプレートを作成します。同時に、アセットファイルを Amazon Simple Storage Service (Amazon S3) バケットにアップロードします。ファイルの送信にかかる時間を短縮するために、最後にファイルをアップロードしてから変更されたファイルのみが Amazon S3 に送信されます。

Deadline Cloud にジョブを送信する独自のスクリプトとパイプラインを作成するには、Deadline Cloud CLI、 AWS SDK、または を使用して オペレーションを AWS CLI 呼び出し、ジョブを作成、

取得、表示、一覧表示できます。以下のトピックでは、Deadline Cloud CLI の使用方法について説明します。

Deadline Cloud CLI は、Deadline Cloud 送信者と共にインストールされます。詳細については、「Deadline Cloud 送信者を設定する」を参照してください。

トピック

Deadline Cloud CLI を使用したジョブの送信

Deadline Cloud コマンドラインインターフェイス (Deadline Cloud CLI) を使用してジョブを送信するには、 `deadline bundle submit` コマンドを使用します。

ジョブはキューに送信されます。ファームとキューをまだ設定していない場合は、Deadline Cloud コンソール (https://console.aws.amazon.com/https://console.aws.amazon.com/deadlinecloud/home) を使用してファームとキューをセットアップし、ファームとキュー ID を確認します。詳細については、「ファームの詳細の定義」および「キューの詳細の定義」を参照してください。

Deadline Cloud CLI のデフォルトのファームとキューを設定するには、次のコマンドを使用します。デフォルトを設定するときは、ファームやキューを指定せずに Deadline Cloud CLI コマンドを使用できます。次の例では、 *farmId*と *queueId*を独自の情報に置き換えます。

```
deadline config set defaults.farm_id farmId
deadline config set defaults.queue_id queueId
```

ジョブのステップとタスクを指定するには、OpenJD ジョブテンプレートを作成します。詳細については、「Open Job Description specification GitHub 」リポジトリの「Template Schemas [Version: 2023-09]」を参照してください。

次の例は、YAML ジョブテンプレートです。これは、2 つのステップとステップごとに 5 つのタスクを持つジョブを定義します。

```
name: Sample Job
specificationVersion: jobtemplate-2023-09
steps:
- name: Sample Step 1
  parameterSpace:
    taskParameterDefinitions:
    - name: var
      range: 1-5
      type: INT
  script:
    actions:
      onRun:
        args:
        - '1'
        command: /usr/bin/sleep
- name: Sample Step 2
  parameterSpace:
    taskParameterDefinitions:
    - name: var
      range: 1-5
      type: INT
  script:
    actions:
      onRun:
        args:
        - '1'
        command: /usr/bin/sleep
```

ジョブを作成するには、 という名前の新しいフォルダを作成し`sample_job`、テンプレートファイルを新しいフォルダに として保存します`template.yaml`。次の Deadline Cloud CLI コマンドを使用してジョブを送信します。

```
deadline bundle submit path/to/sample_job
```

コマンドからのレスポンスには、ジョブの識別子が含まれています。ジョブのステータスを後で確認できるように、ID を覚えておいてください。

```
Submitting to Queue: test-queue
Waiting for Job to be created...
Submitted job bundle:
   sample_job
Job creation completed successfully
```

```
jobId
```

ジョブを送信するときに使用できる追加オプションがあります。詳細については、「[Deadline Cloud CLI でジョブを送信するためのその他のオプション]」を参照してください。

Deadline Cloud CLI でジョブを送信するためのその他のオプション

`deadline bundle submit` Deadline Cloud CLI コマンドには、ジョブの追加情報を指定するために使用できるオプションが用意されています。次の例は、その方法を示しています。

- ジョブテンプレートを処理するときに使用するパラメータを指定します。
- 共有環境内のファイルとフォルダをジョブにアタッチします。
- ジョブがキャンセルされる前のタスク失敗の最大数を設定します。
- タスクの最大再試行回数を設定します。

ジョブのパラメータ

`parameters` オプションは、ジョブの作成時にジョブパラメータの値を設定します。ジョブテンプレートは フィールドを定義し、 `parameters`オプションは 値を設定します。パラメータにはデフォルト値を含めることができます。パラメータに値が指定されている場合、指定された値はデフォルト値よりも優先されます。

次のジョブテンプレートは、 `TestParameter`フィールドを定義します。

```
name: Sample Job With Job Parameter
parameterDefinitions:
- default: test
  name: TestParameter
  type: STRING
specificationVersion: jobtemplate-2023-09
steps:
- description: step description
  name: MyStep
  parameterSpace:
    taskParameterDefinitions:
    - name: var
      range: 1-5
      type: INT
  script:
```

```
    actions:
      onRun:
        args:
        - '1'
        command: /usr/bin/sleep
```

次のコマンドは、 の値をTestParameter「Hello AWS」に設定します。

```
deadline bundle submit sample_job --parameter "TestParameter=Hello AWS"
```

ストレージプロファイル

ストレージプロファイルは、異なるオペレーティングシステムを持つワーカー間でファイルを共有するのに役立ちます。Deadline Cloud コンソールを使用してストレージプロファイルを作成します。次に、 storage-profile-idパラメータを使用してストレージプロファイルを使用します。詳細については、「Deadline Cloud の共有ストレージ」を参照してください。

Deadline Cloud CLI を使用してジョブ送信のストレージプロファイルを設定するには、次のコマンドを使用して storage-profile-id設定パラメータを設定します。

```
deadline config set settings.storage_profile_id storageProfileId
```

失敗したタスクの最大数

max-failed-tasks-count オプションは、ジョブ全体が失敗し、残りのすべてのタスクが とマークされるまでに失敗できるタスクの最大数を設定しますCANCELED。デフォルト値は 100 です。

```
deadline bundle submit sample_job --max-failed-tasks-count 10
```

失敗したタスクの最大再試行回数

max-retries-per-task オプションは、タスクが失敗する前に再試行される最大回数を設定します。タスクが再試行されると、 READY状態になります。デフォルト値は 5 です。

```
deadline bundle submit sample_job --max-retries-per-task 10
```

Deadline Cloud でのジョブのスケジュール設定

ジョブが作成されると、Deadline Cloud AWS はキューに関連付けられた 1 つ以上のフリートで処理されるようにスケジュールします。特定のタスクを処理するフリートは、フリートに設定された機能と特定のステップのホスト要件に基づいて選択されます。

ジョブは、ベストエフォートの優先順位で、最も高いものから低いものまでスケジュールされます。2 つのジョブの優先度が同じ場合、最も古いジョブが最初にスケジュールされます。

以下のセクションでは、ジョブをスケジュールするプロセスの詳細について説明します。

フリートの互換性を確認する

ジョブが作成されると、Deadline Cloud はジョブの各ステップのホスト要件を、ジョブが送信されたキューに関連付けられたフリートの機能と照合します。フリートがホスト要件を満たしている場合、ジョブは `READY`状態になります。

ジョブ内のいずれかのステップに、キューに関連付けられたフリートが満たすことができない要件がある場合、ステップのステータスは に設定されます`NOT_COMPATIBLE`。さらに、ジョブの残りのステップはキャンセルされます。

フリートの機能はフリートレベルで設定されます。フリート内のワーカーがジョブの要件を満たしていても、フリートがジョブの要件を満たしていない場合、ジョブからタスクは割り当てられません。

次のジョブテンプレートには、ステップのホスト要件を指定するステップがあります。

```
name: Sample Job With Host Requirements
specificationVersion: jobtemplate-2023-09
steps:
- name: Step 1
  script:
    actions:
      onRun:
        args:
        - '1'
        command: /usr/bin/sleep
  hostRequirements:
    amounts:
    # Capabilities starting with "amount." are amount capabilities. If they start with
 "amount.worker.",
```

```
    # they are defined by the OpenJD specification. Other names are free for custom
 usage.
    - name: amount.worker.vcpu
      min: 4
      max: 8
    attributes:
    - name: attr.worker.os.family
      anyOf:
      - linux
```

このジョブは、次の機能を備えたフリートにスケジュールできます。

```
{
    "vCpuCount": {"min": 4, "max": 8},
    "memoryMiB": {"min": 1024},
    "osFamily": "linux",
    "cpuArchitectureType": "x86_64"
}
```

このジョブは、次のいずれかの機能を持つフリートにスケジュールすることはできません。

```
{
    "vCpuCount": {"min": 4},
    "memoryMiB": {"min": 1024},
    "osFamily": "linux",
    "cpuArchitectureType": "x86_64"
}
    The vCpuCount has no maximum, so it exceeds the maximum vCPU host requirement.

{
    "vCpuCount": {"max": 8},
    "memoryMiB": {"min": 1024},
    "osFamily": "linux",
    "cpuArchitectureType": "x86_64"
}
    The vCpuCount has no minimum, so it doesn't satisfy the minimum vCPU host
 requirement.

{
    "vCpuCount": {"min": 4, "max": 8},
    "memoryMiB": {"min": 1024},
    "osFamily": "windows",
    "cpuArchitectureType": "x86_64"
```

```
}
    The osFamily doesn't match.
```

フリートスケーリング

ジョブが互換性のあるサービスマネージドフリートに割り当てられると、フリートは自動スケーリングされます。フリート内のワーカーの数は、フリートが実行できるタスクの数に応じて変動します。

ジョブがカスタマーマネージドフリートに割り当てられると、ワーカーがすでに存在しているか、イベントベースの自動スケーリングを使用して作成できます。詳細については、「Amazon EC2 Auto Scaling EventBridgeユーザーガイド」の「 を使用して Auto Scaling イベントを処理する」を参照してください。 Amazon EC2 Auto Scaling

セッション

ジョブ内のタスクは 1 つ以上のセッションに分割されます。ワーカーはセッションを実行して環境をセットアップし、タスクを実行し、環境を破棄します。各セッションは、ワーカーが実行する必要がある 1 つ以上のアクションで構成されます。

ワーカーがセクションアクションを完了すると、追加のセッションアクションをワーカーに送信できます。ワーカーは、セッション内の既存の環境とジョブアタッチメントを再利用して、タスクをより効率的に完了します。

ジョブアタッチメントは、Deadline Cloud CLI ジョブバンドルの一部として、使用する送信者によって作成されます。`create-job` AWS CLI コマンドの `--attachments`オプションを使用してジョブアタッチメントを作成することもできます。環境は、特定のキューにアタッチされたキュー環境と、ジョブテンプレートで定義されたジョブステップ環境の 2 つの場所で定義されます。

セッションアクションには 4 つのタイプがあります。

- `syncInputJobAttachments` – 入力ジョブの添付ファイルをワーカーにダウンロードします。
- `envEnter` – 環境の`onEnter`アクションを実行します。
- `taskRun` – タスクの`onRun`アクションを実行します。
- `envExit` – 環境の`onExit`アクションを実行します。

次のジョブテンプレートにはステップ環境があります。ステップ環境を設定する`onEnter`定義、実行するタスクを定義する`onRun`定義、ステップ環境を破棄する`onExit`定義があります。このジョ

ブ用に作成されたセッションには、 envEnterアクション、1 つ以上の taskRun アクション、envExitアクションが含まれます。

```
name: Sample Job with Maya Environment
specificationVersion: jobtemplate-2023-09
steps:
- name: Maya Step
  stepEnvironments:
  - name: Maya
    description: Runs Maya in the background.
    script:
      embeddedFiles:
      - name: initData
        filename: init-data.yaml
        type: TEXT
        data: |
          scene_file: MyAwesomeSceneFile
          renderer: arnold
          camera: persp
      actions:
        onEnter:
          command: MayaAdaptor
          args:
          - daemon
          - start
          - --init-data
          - file://{{Env.File.initData}}
        onExit:
          command: MayaAdaptor
          args:
          - daemon
          - stop
  parameterSpace:
    taskParameterDefinitions:
    - name: Frame
      range: 1-5
      type: INT
  script:
    embeddedFiles:
    - name: runData
      filename: run-data.yaml
      type: TEXT
      data: |
```

```
          frame: {{Task.Param.Frame}}
    actions:
      onRun:
        command: MayaAdaptor
        args:
        - daemon
        - run
        - --run-data
        - file://{{ Task.File.runData }}
```

ステップの依存関係

Deadline Cloud では、ステップ間の依存関係の定義がサポートされているため、あるステップは別のステップが完了するまで待機してから開始できます。ステップには複数の依存関係を定義できます。依存関係を持つステップは、その依存関係がすべて完了するまでスケジュールされません。

ジョブテンプレートが循環依存関係を定義している場合、ジョブは拒否され、ジョブのステータスはに設定されますCREATE_FAILED。

次のジョブテンプレートは、2 つのステップでジョブを作成します。 StepBは に依存しますStepA。 が正常にStepA完了した後にStepBのみ実行されます。

ジョブが作成されると、 StepAは READY状態になり、 StepBは PENDING状態になります。StepA が終了すると、 は READY状態StepBに移行します。がStepA失敗した場合、または StepA がキャンセルされた場合、 は CANCELED状態StepBに移行します。

複数のステップに依存関係を設定できます。例えば、 StepCが StepAと の両方に依存する場合StepB、 StepCは他の 2 つのステップが終了するまで開始されません。

```
name: Step-Step Dependency Test
specificationVersion: 'jobtemplate-2023-09'
steps:
- name: A
  script:
    actions:
      onRun:
        command: bash
        args: ['{{ Task.File.run }}']
    embeddedFiles:
      - name: run
        type: TEXT
        data: |
```

```
          #!/bin/env bash

          set -euo pipefail

          sleep 1
          echo Task A Done!
- name: B
  dependencies:
  - dependsOn: A # This means Step B depends on Step A
  script:
    actions:
      onRun:
        command: bash
        args: ['{{ Task.File.run }}']
    embeddedFiles:
      - name: run
        type: TEXT
        data: |
          #!/bin/env bash

          set -euo pipefail

          sleep 1
          echo Task B Done!
```

Deadline Cloud CLI のジョブ状態

このトピックでは、 AWS Deadline Cloud コマンドラインインターフェイス (Deadline Cloud CLI) を使用してジョブまたはステップのステータスを表示する方法について説明します。Deadline Cloud モニターを使用してジョブまたはステップのステータスを表示する場合は、「」を参照してくださいDeadline Cloud でのジョブ、ステップ、タスクの表示と管理。

`deadline job get --job-id` Deadline Cloud CLI コマンドを使用して、ジョブのステータスを確認できます。コマンドへのレスポンスには、ジョブまたはステップのステータスと、各処理ステータスのタスク数が含まれます。

ジョブを初めて送信すると、ステータスは になります`CREATE_IN_PROGRESS`。ジョブが検証チェックに合格すると、そのステータスは に変わります`CREATE_COMPLETE`。そうでない場合、ステータスは に変わります`CREATE_FAILED`。

ジョブが検証チェックに失敗する理由には、次のようなものがあります。

- ジョブテンプレートが OpenJD 仕様に従っていません。
- ジョブに含まれるステップが多すぎます。
- ジョブの合計タスクが多すぎます。

ジョブ内のステップとタスクの最大数のクォータを確認するには、Service Quotas コンソールを使用します。詳細については、「のクォータ Deadline Cloud」を参照してください。

また、ジョブの作成を妨げる内部サービスエラーが発生する場合もあります。この場合、ジョブのステータスコードは INTERNAL_ERRORになり、ステータスメッセージフィールドにはより詳細な説明が表示されます。

次の Deadline Cloud CLI コマンドを使用して、ジョブの詳細を表示します。次の例では、 をユーザー自身の情報*jobID*に置き換えます。

```
deadline job get --job-id jobId
```

deadline job get コマンドからのレスポンスは次のとおりです。

```
jobId: jobId
name: Sample Job
lifecycleStatus: CREATE_COMPLETE
lifecycleStatusMessage: Job creation completed successfully
priority: 50
createdAt: 2024-03-26 18:11:19.065000+00:00
createdBy: Test User
startedAt: 2024-03-26 18:12:50.710000+00:00
taskRunStatus: STARTING
taskRunStatusCounts:
  PENDING: 0
  READY: 5
  RUNNING: 0
  ASSIGNED: 0
  STARTING: 0
  SCHEDULED: 0
  INTERRUPTING: 0
  SUSPENDED: 0
  CANCELED: 0
  FAILED: 0
  SUCCEEDED: 0
  NOT_COMPATIBLE: 0
```

```
maxFailedTasksCount: 100
maxRetriesPerTask: 5
```

ジョブまたはステップの各タスクには ステータスがあります。タスクステータスは、ジョブとステップの全体的なステータスを示すために結合されます。各状態のタスク数は、レスポンスの`taskRunStatusCounts`フィールドで報告されます。

ジョブまたはステップのステータスは、タスクのステータスによって異なります。ステータスは、これらのステータスを持つタスクによって順番に決まります。ステップステータスは、ジョブステータスと同じように決定されます。

次のリストでは、ステータスについて説明します。

`NOT_COMPATIBLE`

ジョブ内のタスクの 1 つを完了できるフリートがないため、ジョブはファームと互換性がありません。

`RUNNING`

1 人以上のワーカーがジョブからタスクを実行しています。実行中のタスクが少なくとも 1 つある限り、ジョブは とマークされます`RUNNING`。

`ASSIGNED`

1 人以上のワーカーに、次のアクションとしてジョブ内のタスクが割り当てられます。環境がある場合は、セットアップされます。

`STARTING`

1 人以上のワーカーがタスクを実行するための環境を設定しています。

`SCHEDULED`

ジョブのタスクは、ワーカーの次のアクションとして 1 人以上のワーカーにスケジュールされます。

`READY`

ジョブの少なくとも 1 つのタスクを処理する準備ができています。

`INTERRUPTING`

ジョブ内の少なくとも 1 つのタスクが中断されています。ジョブのステータスを手動で更新すると、中断が発生する可能性があります。また、Amazon Elastic Compute Cloud (Amazon EC2) スポット料金の変更による中断に応じて発生することもあります。

FAILED

ジョブ内の 1 つ以上のタスクが正常に完了しませんでした。

CANCELED

ジョブ内の 1 つ以上のタスクがキャンセルされました。

SUSPENDED

ジョブ内の少なくとも 1 つのタスクが中断されています。

PENDING

ジョブ内のタスクは、別のリソースの可用性を待っています。

SUCCEEDED

ジョブ内のすべてのタスクが正常に処理されました。

Deadline Cloud でのジョブの変更

次の AWS Command Line Interface (AWS CLI) `update` コマンドを使用して、ジョブの設定を変更するか、ジョブ、ステップ、またはタスクのターゲットステータスを設定できます。

- `aws deadline update-job`
- `aws deadline update-step`
- `aws deadline update-task`

次の`update`コマンドの例では、それぞれをユーザー自身の情報に置き換え*`user input placeholder`*ます。

Deadline Cloud モニターを使用して、ジョブの設定を変更することもできます。詳細については、「Deadline Cloud でのジョブ、ステップ、タスクの表示と管理」を参照してください。

Example – ジョブを再キューに入れる

ステップ依存関係がない限り、ジョブ内のすべてのタスクは `READY`ステータスに切り替わります。依存関係を持つステップは、復元`PENDING`時に `READY`または のいずれかに切り替わります。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
```

```
--job-id jobID \
--target-task-run-status PENDING
```

Example – ジョブをキャンセルする

ジョブ内のすべてのタスクで、 ステータスがないか、 とマークSUCCEEDEDFAILEDされていますCANCELED。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--target-task-run-status CANCELED
```

Example – ジョブが失敗したマーク

ステータスを持つジョブ内のすべてのタスクSUCCEEDEDは変更されません。他のすべてのタスクはとマークされますFAILED。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--target-task-run-status FAILED
```

Example – ジョブを成功としてマークする

ジョブ内のすべてのタスクが SUCCEEDED状態に移行します。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--target-task-run-status SUCCEEDED
```

Example – ジョブを一時停止する

SUCCEEDED、、CANCELEDまたは FAILED状態のジョブのタスクは変更されません。他のすべてのタスクは とマークされますSUSPENDED。

```
aws deadline update-job \
--farm-id farmID \
```

```
--queue-id queueID \
--job-id jobID \
--target-task-run-status SUSPENDED
```

Example – ジョブの優先度を変更する

ジョブの優先度を更新して、スケジュールされている順序を変更します。優先度の高いジョブは通常、最初にスケジュールされます。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--priority 100
```

Example – 失敗したタスクの許容数を変更する

残りのタスクがキャンセルされるまでにジョブが保持できる失敗したタスクの最大数を更新します。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--max-failed-tasks-count 200
```

Example – 許可されるタスクの再試行回数を変更する

タスクが失敗する前に、タスクの最大再試行回数を更新します。再試行の最大数に達したタスクは、この値が増加するまで再キューに入れることはできません。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--max-retries-per-task 10
```

Example – ジョブをアーカイブする

ジョブのライフサイクルステータスを に更新しますARCHIVED。アーカイブされたジョブはスケジュールまたは変更できません。アーカイブできるのは、、FAILED、、CANCELEDSUCCEEDEDまたは SUSPENDED状態のジョブのみです。

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--lifecycle-status ARCHIVED
```

Example – ステップを再キューに入れる

ステップの依存関係がない限り、ステップ内のすべてのタスクは READY状態に切り替わります。依存関係を持つステップのタスクは READYまたは のいずれかに切り替わりPENDING、タスクは復元されます。

```
aws deadline update-step \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--step-id stepID \
--target-task-run-status PENDING
```

Example – ステップをキャンセルする

ステップ内のすべてのタスクで、 ステータスがないか、 とマークSUCCEEDEDFAILEDされていますCANCELED。

```
aws deadline update-step \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--step-id stepID \
--target-task-run-status CANCELED
```

Example – ステップのマークに失敗しました

ステータスを持つステップ内のすべてのタスクSUCCEEDEDは変更されません。他のすべてのタスクは とマークされますFAILED。

```
aws deadline update-step \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
```

```
--step-id stepID \
--target-task-run-status FAILED
```

Example – ステップを成功としてマークする

ステップのすべてのタスクは とマークされますSUCCEEDED。

```
aws deadline update-step \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--step-id stepID \
--target-task-run-status SUCCEEDED
```

Example – ステップを一時停止する

SUCCEEDED、、CANCELEDまたは FAILED状態の ステップのタスクは変更されません。他のすべてのタスクは とマークされますSUSPENDED。

```
aws deadline update-step \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--step-id stepID \
--target-task-run-status SUSPENDED
```

Example – タスクのステータスを変更する

`update-task` Deadline Cloud CLI コマンドを使用すると、タスクは指定されたステータスに切り替わります。

```
aws deadline update-task \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--step-id stepID \
--task-id taskID \
--target-task-run-status SUCCEEDED | SUSPENDED | CANCELED | FAILED | PENDING
```

Deadline Cloud がジョブを処理する方法

ジョブを処理するために、 AWS Deadline Cloud は Open Job Description (OpenJD) ジョブテンプレートを使用して必要なリソースを決定します。Deadline Cloud は、キューに関連付けられたフリートからステップに適したワーカーを選択します。選択したワーカーは、ステップに必要なすべての機能属性を満たしています。

次に、Deadline Cloud はステップのセッションを設定する手順をワーカーに送信します。ステップに必要なソフトウェアは、ジョブを実行するワーカーインスタンスで利用できる必要があります。フリートのスケーリング設定に容量がある場合、サービスは複数のワーカーでセッションを開くことができます。

Amazon Machine Image (AMI) でソフトウェアをセットアップすることも、ワーカーが実行時にリポジトリまたはパッケージマネージャーからソフトウェアをロードすることもできます。キュー、ジョブ、またはステップ環境を使用して、目的のソフトウェアをデプロイできます。

Deadline Cloud サービスは OpenJD テンプレートを使用して、ジョブに必要なステップと各ステップに必要なタスクを決定します。一部のステップは他のステップに依存するため、Deadline Cloud はステップを完了する順序を決定します。次に、Deadline Cloud は各ステップのタスクをワーカーに送信して処理します。タスクが完了すると、サービスは同じセッションで別のタスクを送信するか、ワーカーは新しいセッションを開始できます。

Deadline Cloud モニター、Deadline Cloud コマンドラインインターフェイス (Deadline Cloud CLI)、または でジョブの進行状況を追跡できます AWS CLI。モニタの使用の詳細については、「」を参照してください[Deadline Cloud モニターの使用]。Deadline Cloud CLI の使用の詳細については、「」を参照してください[Deadline Cloud CLI のジョブ状態]。

各ステップのすべてのタスクが完了すると、ジョブが完了し、出力をワークステーションにダウンロードする準備が整います。ジョブが完了していなくても、完了した各ステップとタスクからの出力はダウンロードできます。

Deadline Cloud は、ジョブが送信されてから 120 日後にジョブを削除します。ジョブが削除されると、ジョブに関連付けられているすべてのステップとタスクも削除されます。ジョブを再実行する必要がある場合は、ジョブの OpenJD テンプレートを再度送信します。

Deadline Cloud ジョブのトラブルシューティング

AWS Deadline Cloud のジョブに関する一般的な問題については、以下のトピックを参照してください。

トピック

ジョブの作成が失敗したのはなぜですか？

ジョブが検証チェックに失敗する理由には、次のようなものがあります。

- ジョブテンプレートが OpenJD 仕様に従っていません。
- ジョブに含まれるステップが多すぎます。
- ジョブの合計タスクが多すぎます。
- ジョブの作成を妨げる内部サービスエラーが発生しました。

ジョブ内のステップとタスクの最大数のクォータを確認するには、Service Quotas コンソールを使用します。詳細については、「のクォータ Deadline Cloud」を参照してください。

ジョブに互換性がないのはなぜですか？

ジョブがキューと互換性がない一般的な理由は次のとおりです。

- ジョブが送信されたキューに関連付けられたフリートはありません。Deadline Cloud モニターを開き、キューにフリートが関連付けられていることを確認します。キューを表示する方法の詳細については、「」を参照してくださいDeadline Cloud でキューとフリートの詳細を表示する。
- ジョブには、キューに関連付けられているフリートによって満たされないホスト要件があります。確認するには、ジョブテンプレートの`hostRequirements`エントリをファーム内のフリートの設定と比較します。いずれかのフリートがホスト要件を満たしていることを確認します。フリートの互換性の詳細については、「」を参照してくださいフリートの互換性を確認する。フリート設定を表示するには、「」を参照してくださいDeadline Cloud でキューとフリートの詳細を表示する。

ジョブの準備が整うのはなぜですか？

ジョブが `READY`状態でスタックしているように見える理由には、次のようなものがあります。

- キューに関連付けられているフリートの最大ワーカー数は 0 に設定されています。確認するには、「」を参照してくださいDeadline Cloud でキューとフリートの詳細を表示する。
- キューには優先度の高いジョブがあります。確認するには、「」を参照してくださいDeadline Cloud でキューとフリートの詳細を表示する。
- カスタマーマネージドフリートの場合は、自動スケーリング設定を確認します。詳細については、「Deadline Cloud スケールレコメンデーション機能を使用して Amazon EC2 フリートを自動スケーリングする」を参照してください。

ジョブが失敗したのはなぜですか？

ジョブは、さまざまな理由で失敗する可能性があります。問題を検索するには、Deadline Cloud モニターを開き、失敗したジョブを選択します。失敗したタスクを選択し、タスクのログを表示します。手順については、「Deadline Cloud でログを表示する」を参照してください。

- ライセンスエラーが表示される場合、またはソフトウェアに有効なライセンスがないためにウォーターマークが表示された場合は、ワーカーが必要なライセンスサーバーに接続できることを確認してください。詳細については、「カスタマーマネージドフリートをライセンスエンドポイントに接続する」を参照してください。

ステップが保留になっているのはなぜですか？

ステップは、1 つ以上の依存関係が完了しない場合、 `PENDING`状態のままになることがあります。Deadline Cloud モニターを使用して、依存関係の状態を確認できます。手順については、「Deadline Cloud でステップを表示する」を参照してください。

Deadline Cloud のファイルストレージ

ワーカーは、ジョブの処理に必要な入力ファイルを含むストレージロケーションと、出力を保存するロケーションにアクセスできる必要があります。Deadline Cloud AWS には、ストレージロケーションに 2 つのオプションがあります。

- ジョブアタッチメント を使用すると、Deadline Cloud はジョブの入力ファイルと出力ファイルをワークステーションと Deadline Cloud ワーカー間で送受信します。ファイル転送を有効にするために、Deadline Cloud は で Amazon Simple Storage Service (Amazon S3) バケットを使用します AWS アカウント。

  サービスマネージドフリートでジョブアタッチメントを使用すると、仮想プライベートネットワーク (VPN) に仮想ファイルシステム (VFS) を設定できます。これにより、ワーカーは必要な場合にのみファイルをロードできます。
- 共有ストレージ では、オペレーティングシステムとのファイル共有を使用してファイルへのアクセスを提供します。

  クロスプラットフォーム共有ストレージを使用する場合、ワーカーが 2 つの異なるオペレーティングシステム間のファイルにパスをマッピングできるように、ストレージプロファイルを作成できます。

トピック

Deadline Cloud のジョブアタッチメント

ジョブアタッチメントを使用すると、ワークステーションと Deadline Cloud AWS 間でファイルを送受信できます。ジョブアタッチメントを使用すると、ファイル用に Amazon S3 バケットを手動で設定する必要はありません。代わりに、Deadline Cloud コンソールでキューを作成するときに、ジョブアタッチメントのバケットを選択します。

Deadline Cloud にジョブを初めて送信すると、ジョブのすべてのファイルが Deadline Cloud に転送されます。それ以降の送信では、変更されたファイルのみが転送され、時間と帯域幅の両方が節約されます。

処理が完了したら、ジョブの詳細ページから、または Deadline Cloud CLI `deadline job download-output` コマンドを使用して結果をダウンロードできます。

複数のキューに同じ S3 バケットを使用できます。バケット内のアタッチメントを整理するには、キューごとに異なるルートプレフィックスを設定します。

コンソールでキューを作成するときは、既存の AWS Identity and Access Management (IAM) ロールを選択するか、コンソールに新しいロールを作成させることができます。コンソールがロールを作成すると、キューに指定されたバケットにアクセスするためのアクセス許可が設定されます。既存のロールを選択する場合は、S3 バケットにアクセスするためのアクセス許可をロールに付与する必要があります。

ジョブアタッチメント S3 バケットの暗号化

ジョブアタッチメントファイルは、デフォルトで S3 バケットで自動的に暗号化されます。このアプローチは、不正アクセスから情報を保護するのに役立ちます。Deadline Cloud が提供するキーでファイルを暗号化するために何もする必要はありません。詳細については、Amazon S3ユーザーガイド」の「Amazon S3 がすべての新しいオブジェクトを自動的に暗号化するようになりました」を参照してください。 Amazon S3

独自のカスタマーマネージド AWS Key Management Service キーを使用して、ジョブアタッチメントを含む S3 バケットを暗号化できます。そのためには、バケットに関連付けられたキューの IAM ロールを変更して、 へのアクセスを許可する必要があります AWS KMS key。

キューロールの IAM ポリシーエディタを開くには

1. にサインイン AWS Management Console し、Deadline Cloud コンソール を開きます。メインページの「開始方法」セクションで「ファームの表示」を選択します。
2. ファームのリストから、変更するキューを含むファームを選択します。
3. キューのリストから、変更するキューを選択します。
4. キューの詳細セクションで、サービスロールを選択して、サービスロールの IAM コンソールを開きます。

次に、次の手順を実行します。

のアクセス許可でロールポリシーを更新するには AWS KMS

1. アクセス許可ポリシー のリストから、ロールのポリシーを選択します。

2. このポリシーで定義されているアクセス許可セクションで、編集 を選択します。
3. [新しいステートメントを追加] を選択します。
4. 次のポリシーをコピーしてエディタに貼り付けます。*Region*、*accountID*、 を独自の値*keyID*に変更します。

```
{
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt",
        "kms:DescribeKey",
        "kms:GenerateDataKey"
    ],
    "Resource": [
        "arn:aws:kms:Region:accountID:key/keyID"
    ]
}
```

5. [次へ] をクリックします。
6. ポリシーの変更を確認し、問題がなければ変更を保存 を選択します。

S3 バケットでのジョブアタッチメントの管理

Deadline Cloud は、ジョブに必要なジョブアタッチメントファイルを S3 バケットに保存します。これらのファイルは時間の経過とともに蓄積され、Amazon S3 のコストが増加します。コストを削減するために、S3 バケットに S3 ライフサイクル設定を適用できます。この設定では、バケット内のファイルを自動的に削除できます。S3 バケットは アカウントにあるため、S3 ライフサイクル設定はいつでも変更または削除できます。詳細については、「Amazon S3 ユーザーガイド」の「S3 ライフサイクル設定の例」を参照してください。 Amazon S3

より詳細な S3 バケット管理ソリューションでは、S3 バケット内のオブジェクトに最後にアクセスされた時刻に基づいて AWS アカウント 有効期限が切れるように を設定できます。詳細については、「 アーキテクチャブログ」の「最終アクセス日に基づく Amazon S3 オブジェクトの有効期限切れ AWS 」を参照してください。

Deadline Cloud 仮想ファイルシステム

Deadline Cloud AWS でのジョブアタッチメントの仮想ファイルシステムサポートにより、ワーカーのクライアントソフトウェアが Amazon Simple Storage Service と直接通信できるようになります。

ワーカーは、処理前にすべてのファイルをダウンロードするのではなく、必要な場合にのみファイルをロードできます。ファイルはローカルに保存されます。このアプローチにより、複数回使用されるアセットのダウンロードを回避できます。ジョブが完了すると、すべてのファイルが削除されます。

- 仮想ファイルシステムは、特定のジョブプロファイルのパフォーマンスを大幅に向上させます。一般に、ワーカーのフリートが多いファイルの合計のサブセットが小さいほど、最もメリットがあります。ワーカー数が少ない少数のファイルでは、処理時間がほぼ同等です。
- 仮想ファイルシステムのサポートは、サービスマネージドフリートのLinuxワーカーのみが利用できます。
- Deadline Cloud 仮想ファイルシステムは、以下のオペレーションをサポートしていますが、POSIX に準拠していません。
  - ファイル `create`、`delete`、`open`、`close`、`read`、`write`、`append`、、`truncate`、`rename`、`move`、`copy`、および `falloc`
  - ディレクトリ `create`、`delete`、`rename`、`move`、`copy`、および `stat`
- 仮想ファイルシステムは、タスクが大規模なデータセットの一部にのみアクセスする場合に、データ転送を減らし、パフォーマンスを向上させるように設計されており、すべてのワークロードに最適化されているわけではありません。本番稼働用ジョブを実行する前に、ワークロードをテストする必要があります。

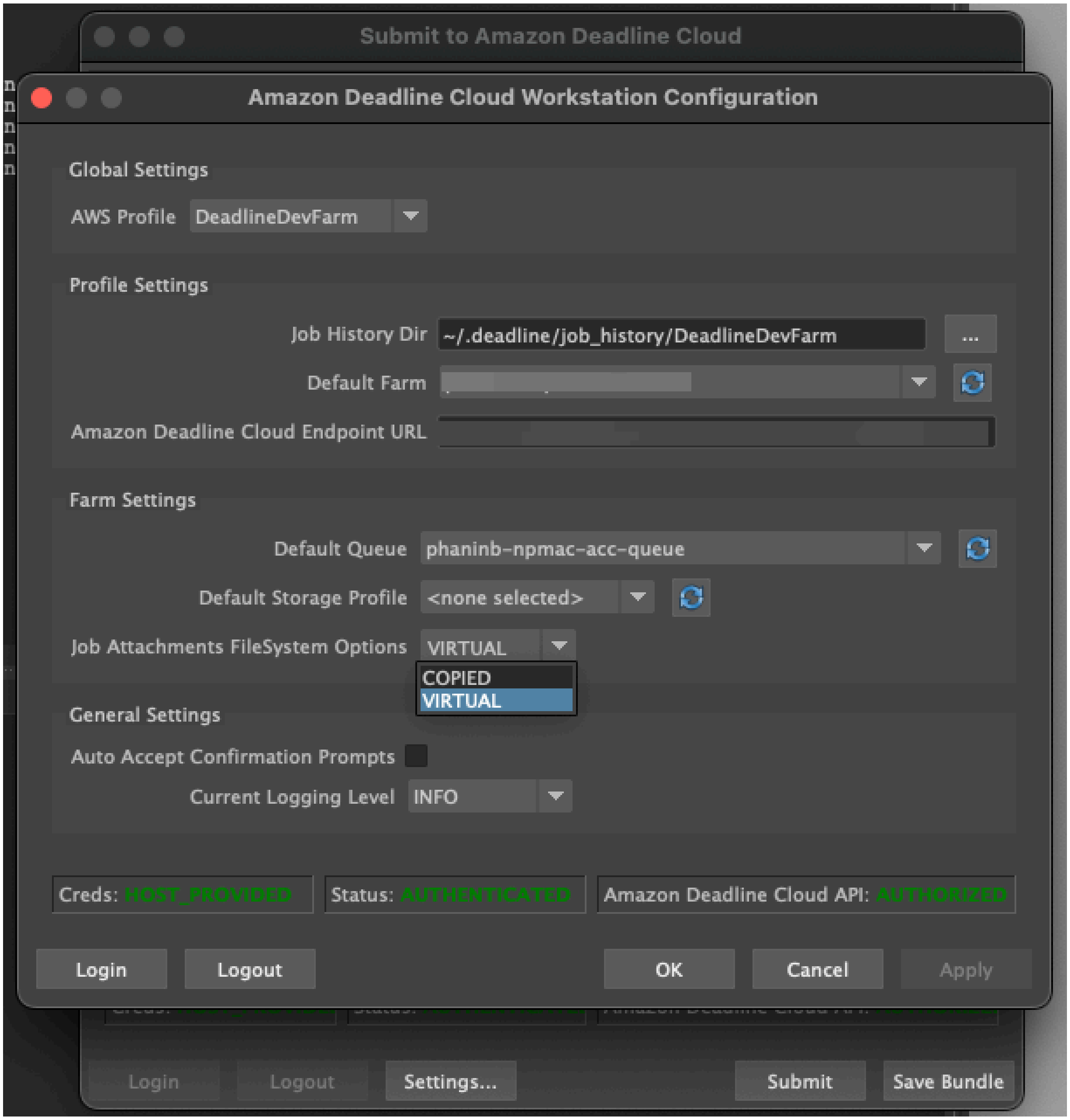
VFS サポートを有効にする

仮想ファイルシステムサポート (VFS) は、ジョブごとに有効になります。このような場合、ジョブはデフォルトのジョブアタッチメントフレームワークにフォールバックします。

- ワーカーインスタンスプロファイルは仮想ファイルシステムをサポートしていません。
- 問題により、仮想ファイルシステムプロセスの起動が妨げられます。
- 仮想ファイルシステムをマウントすることはできません。

送信者を使用して仮想ファイルシステムのサポートを有効にするには

1. ジョブを送信するときは、設定ボタンを選択して AWS Deadline Cloud ワークステーション設定パネル を開きます。
2. ジョブアタッチメントのファイルシステムオプションドロップダウンから、VIRTUAL を選択します。



3. 変更を保存するには、OK を選択します。

を使用して仮想ファイルシステムのサポートを有効にするには AWS CLI

- 保存したジョブを送信するときは、次のコマンドを使用します。

```
deadline bundle submit-job --job-attachments-file-system VIRTUAL
```

仮想ファイルシステムが特定のジョブに対して正常に起動されたことを確認するには、Amazon CloudWatch Logs のログを確認します。次のメッセージを探します。

```
Using mount_point mount_point
Launching vfs with command command
Launched vfs as pid PID number
```

ログに次のメッセージが含まれている場合、仮想ファイルシステムのサポートは無効になります。

```
Virtual File System not found, falling back to COPIED for JobAttachmentsFileSystem.
```

仮想ファイルシステムのサポートのトラブルシューティング

Deadline Cloud モニターを使用して、仮想ファイルシステムのログを表示できます。手順については、「Deadline Cloud でログを表示する」を参照してください。

仮想ファイルシステムログは、ワーカーエージェントの出力と共有されているキューに関連付けられている CloudWatch ロググループにも送信されます。

Deadline Cloud の共有ストレージ

共有ストレージ を使用するには、ワーカーはオペレーティングシステムのファイル共有システムを使用して、ジョブの入出力の共有ストレージスペースにアクセスします。

ファイルの共有に実際に使用する方法は、オペレーティングシステムと、ネットワークに共有ストレージを実装する方法によって異なります。ファイル共有の設定方法と、その設定がユーザーのニーズを満たすことを確認するのはユーザーの責任です。

クロスシステムファイル共有ソリューションを使用している場合は、ストレージプロファイルを使用して、 Linuxと ファイルシステム間でWindowsファイルの場所をマッピングできます。

Deadline Cloud のストレージプロファイル

ストレージプロファイルを使用すると、クロスプラットフォーム共有ストレージを使用してファームを設定できます。ストレージプロファイルは、送信されたワークステーションとは異なるオペレーティングシステムを持つワーカーで処理されたジョブのオペレーティングシステム間のパスをマッピングします。

ストレージプロファイルは、ワークステーションとワーカー間でオペレーティングシステムが混在するカスタマーマネージドフリートを使用する場合に必要です。ストレージプロファイルは、サービスマネージドフリートではサポートされていません。

ストレージプロファイルを作成したら、そのプロファイルを使用するキューとフリートへのアクセスを許可する必要があります。

ストレージプロファイルを作成するには

1. Deadline Cloud コンソール を開きます。
2. 開始方法 から、「Deadline Cloud ダッシュボードに移動」を選択します。
3. ファームを選択し、ストレージプロファイルタブを選択します。
4. ストレージプロファイルの作成 を選択します。
5. ドロップダウンからオペレーティングシステムを選択します。
6. プロファイルの名前を指定します。わかりやすい名前を付けると、ジョブの送信時に使用するストレージプロファイルを選択できます。
7. パス名 に、ジョブの送信元のワークステーション上のジョブデータのルートロケーションを入力します。
8. ストレージタイプ を選択します。
    - ローカルとは、ワーカーとワークステーション間で共有されないファイルの場所を指します。これらはジョブアタッチメントとしてアップロードされます。
    - 共有とは、ワーカーとワークステーション間で共有されるストレージを指します。共有ストレージ内のファイルはジョブアタッチメントとしてアップロードされません。
9. ファイルシステムの場所パス を指定します。これはジョブデータのルートディレクトリです。
10. [Create] (作成) を選択します。

ストレージプロファイルを作成したら、新しいプロファイルを使用するようにキューとカスタマーマネージドフリートを変更する必要があります。ストレージプロファイルへのアクセスを許可するには、前の手順を完了した後、次の手順を使用します。

キューとカスタマーマネージドフリートにストレージプロファイルの使用を許可するには

1. キューまたはフリート タブを選択します。
2. 変更するキューまたはフリートを選択します。
3. ストレージプロファイルの変更 を選択します。

4. 許可するストレージプロファイルとそのプロファイルからファイルシステムの場所を選択します。
5. [変更の保存] を選択します。

Deadline Cloud の予算と使用状況の管理

AWS Deadline Cloud 予算マネージャーと使用状況エクスプローラーは、コスト変数に関する利用可能な情報に基づいて Deadline Cloud を使用するおおよそのコストを提供するコスト管理ツールです。コスト管理ツールは、Deadline Cloud およびその他の AWS のサービスの実際の使用に対して支払うべき金額を保証するものではありません。

Deadline Cloud のコスト管理に役立つように、次の機能を使用できます。

- 予算マネージャー – Deadline Cloud 予算マネージャーを使用すると、予算を作成および編集してプロジェクトコストを管理できます。
- Usage Explorer – Deadline Cloud Usage Explorer を使用すると、使用されている AWS リソースの数と、それらのリソースの推定コストを確認できます。

コストの前提条件

Deadline Cloud コスト管理ツールで使用される基本的な計算は次のとおりです。

```
Cost per job =
    (CMF run time x CMF compute rate) +
    (SMF run time x SMF compute rate) +
    (License run time x license rate)
```

- ランタイムは、開始時刻から終了時刻までのジョブ内のすべてのタスクの合計です。
- コンピューティングレートは、サービスマネージドフリートの AWS Deadline Cloud の料金によって決まります。カスタマーマネージドフリートの場合、コンピューティングレートはワーカー 1 時間あたり 1 USD と推定されます。
- ライセンスレートは、Deadline Cloud の基本ライセンス料金によって決まります。追加の階層は含まれません。ライセンス料金の詳細については、「AWS Deadline Cloud の料金」を参照してください。

Deadline Cloud コスト管理ツールからのコスト見積もりは、さまざまな理由で実際のコストとは異なる場合があります。一般的な理由は次のとおりです。

- 顧客所有のリソースとその料金 オンプレミス AWS や他のクラウドプロバイダーから独自のリソースを持ち込むか、外部に持ち込むかを選択できます。これらのリソースの実際のコストは計算されません。
- アイドル状態のワーカーのコストは です。最小インスタンス数が 0 より大きいフリートの場合、アイドル状態のワーカーは計算に含まれません。
- プロモーションクレジット、割引、カスタム料金契約 。コスト管理ツールでは、プロモーションクレジット、プライベート料金契約、その他の割引は考慮されません。見積りに含まれない他の割引の対象となる場合があります。
- アセットストレージ 。アセットストレージは、コストと使用量の見積もりには含まれません。
- ほとんどの サービスの . AWS offers pay-as-you-go 料金の変更。料金は時間の経過とともに変更される場合があります。コスト管理ツールでは、公開されているほとんどの up-to-date 料金を使用しますが、変更後に遅延が発生する場合があります。
- 税金 コスト管理ツールには、サービスの購入に適用される税金は含まれません。
- の四捨五入。コスト管理ツールは、料金データの数学的四捨五入を実行します。
- 通貨 。コスト見積もりは米ドルで行われます。グローバル為替レートは、時間の経過とともに変化します。見積りを現在の交換に基づく別の通貨ベースに変換すると、為替レートの変更が見積りに影響します。
- 外部ライセンス 。事前に購入したライセンス (独自のライセンスを持つ) を使用することを選択した場合、Deadline Cloud コスト管理ツールはこのコストを考慮できません。

Deadline Cloud 予算マネージャーの使用

Deadline Cloud 予算マネージャーは、キュー、フリート、ファームなど、特定のリソースに対する支出を制御するのに役立ちます。予算の金額と制限を作成し、予算に対する追加支出を削減または停止するのに役立つ自動アクションを設定できます。

以下のセクションでは、Deadline Cloud 予算マネージャーを使用する手順について説明します。

トピック

- 前提条件
- 予算マネージャーにアクセスする
- 予算を作成する
- 予算を表示する

- 予算を編集する
- 予算を非アクティブ化する

前提条件

Deadline Cloud 予算マネージャーを使用するには、OWNERアクセスレベルが必要です。アクセスOWNER許可を付与するには、「」のステップに従いますDeadline Cloud でのユーザーの管理。

予算マネージャーにアクセスする

Deadline Cloud 予算マネージャーにアクセスするには、次の手順を使用します。

1. にサインイン AWS Management Console し、Deadline Cloud コンソール を開きます。
2. ファームの表示 を選択します。
3. 情報を取得するファームを見つけ、ジョブの管理 を選択します。Deadline Cloud モニターが新しいタブで開きます。
4. Deadline Cloud モニターの左側のナビゲーションペインで、 Budgets を選択します。

予算マネージャーの概要ページには、アクティブな予算と非アクティブな予算の両方のリストが表示されます。

- アクティブな予算は、選択したリソース (キュー) に対して追跡されます。
- 非アクティブな予算の有効期限が切れているか、ユーザーによってキャンセルされ、この予算の制限に対するコストを追跡しなくなりました。

予算を選択すると、予算概要ページには予算に関する基本情報が表示されます。提供される情報には、予算名、ステータス、リソース、残りの割合、残りの金額、合計予算、開始日、終了日が含まれます。

予算を作成する

予算を作成するには、次の手順を使用します。

1. まだ にサインインしていない場合は、 にサインインし AWS Management Console、Deadline Cloud コンソール を開き、ファームを選択し、ジョブの管理 を選択します。
2. Budget Manager ページから、予算の作成 を選択します。

3. 詳細セクションに、予算の予算名を入力します。
4. (オプション) 説明フィールドに、予算の明確で簡単な説明を入力します。
5. リソースからキュードロップダウンを選択して、予算を作成するキューを見つけて選択します。
6. 期間 では、次のステップを実行して予算の開始日と終了日を設定します。

    a. 開始日 には、予算追跡の最初の日付を YYYY/MM/DD 形式で入力するか、カレンダーアイコンを選択して日付 を選択します。

        デフォルトの開始日は、予算が作成された日付です。

    b. 終了日 には、予算追跡の最終日を YYYY/MM/DD 形式で入力するか、カレンダーアイコンを選択して日付 を選択します。

        デフォルトの終了日は、開始日から 120 日です。
7. 予算額 には、予算のドル額を入力します。
8. (オプション) 制限アラートを作成することをお勧めします。「アクションの制限」セクションでは、特定の金額が予算に残っているときに発生する自動アクションを実装できます。そのためには、以下のステップを完了します。

    a. 新しいアクションを追加 を選択します。
    b. 残額 には、アクションを開始するドル金額を入力します。
    c. アクションドロップダウンで、目的のアクションを選択します。アクションには以下が含まれます。

        - 現在の作業を終了した後で停止する – しきい値に達したときに現在実行中のすべての作業は、終了するまで実行が継続されます (コストが発生します)。
        - 作業をすぐに停止する – しきい値に達すると、すべての作業がすぐにキャンセルされます。
    d. 追加の制限アラートを作成するには、新しいアクションを追加を選択し、前の 2 つのステップを繰り返します。
9. [予算を作成] をクリックします。予算マネージャーページが表示されます。新しく作成された予算がアクティブ予算タブに表示されます。

予算を表示する

予算を作成したら、 Budget Manager ページで予算を表示できます。そこから、予算の合計金額と、特定の予算に割り当てられた全体的なコストを表示できます。

予算を表示するには、次の手順を使用します。

1. まだ にサインインしていない場合は、 にサインインし AWS Management Console、Deadline Cloud [コンソール](#) を開き、ファームを選択し、ジョブの管理 を選択します。
2. 左側のナビゲーションペインから Budgets を選択します。Budget Manager ページが表示されます。
3. アクティブな予算を表示するには、アクティブな予算 タブを選択し、表示する予算の名前を選択します。予算の詳細ページが表示されます。
4. 期限切れの予算の予算の詳細を表示するには、「非アクティブ予算」タブを選択します。次に、表示する予算の名前を選択します。予算の詳細ページが表示されます。

予算を編集する

アクティブな予算は編集できます。アクティブな予算を編集するには、次の手順を使用します。

1. まだ にサインインしていない場合は、 にサインインし AWS Management Console、Deadline Cloud [コンソール](#) を開き、ファームを選択し、ジョブの管理 を選択します。
2. Budget Manager ページの「アクティブ予算」タブで、編集する予算の横にあるボタンを選択します。
3. 右上隅のアクションドロップダウンメニューから、予算の編集 を選択します。
4. 必要な変更を行い、予算の更新 を選択します。

予算を非アクティブ化する

アクティブな予算は非アクティブ化できます。予算を非アクティブ化すると、そのステータスがアクティブ から非アクティブ に変更されます。予算が非アクティブ化されると、その予算の金額までリソースを追跡しなくなります。

予算を非アクティブ化するには、次の手順を使用します。

1. まだ にサインインしていない場合は、 にサインインし AWS Management Console、Deadline Cloud [コンソール](#) を開き、ファームを選択し、ジョブの管理 を選択します。
2. Budget Manager ページの Active Budgets タブで、非アクティブ化する予算の横にあるボタンを選択します。

3. 右上隅のアクションドロップダウンメニューから、予算の非アクティブ化 を選択します。しばらくすると、選択した予算がアクティブから非アクティブに変更され、アクティブ予算タブから非アクティブ予算タブに移動します。

Deadline Cloud 使用状況エクスプローラーの使用

Deadline Cloud 使用状況エクスプローラーを使用すると、各ファームで発生しているアクティビティに関するリアルタイムのメトリクスを確認できます。ファームのコストは、キュー、ジョブ、ライセンス製品、インスタンスタイプなど、さまざまな変数で確認できます。さまざまな時間枠を選択して、特定の期間における使用状況を確認し、その期間における使用状況の傾向を確認します。選択したデータポイントの詳細な内訳を表示して、メトリクスを詳しく調べることもできます。使用状況は、時間 (分と時間) またはコスト ($USD) で表示できます。

以下のセクションでは、Deadline Cloud 使用状況エクスプローラーにアクセスして使用する手順を示します。

トピック

- 前提条件
- 使用状況エクスプローラーを開く
- 使用状況エクスプローラーを使用する

前提条件

Deadline Cloud 使用状況エクスプローラーを使用するには、 `MANAGER`または `OWNER`ファームのアクセス許可が必要です。詳細については、「ファーム、キュー、フリートのユーザーとグループを管理する」を参照してください。

使用状況エクスプローラーを開く

Deadline Cloud 使用状況エクスプローラーを開くには、次の手順を使用します。

1. にサインイン AWS Management Console し、Deadline Cloud コンソール を開きます。
2. 使用可能なすべてのファームを表示するには、ファームの表示 を選択します。
3. 情報を取得するファームを見つけ、ジョブの管理 を選択します。Deadline Cloud モニターが新しいタブで開きます。
4. Deadline Cloud モニターで、左側のメニューから Usage Explorer を選択します。

使用状況エクスプローラーを使用する

使用状況エクスプローラーページから、データを表示できる特定のパラメータを選択できます。デフォルトでは、過去 7 日間の時間 (時と分) の合計使用量が表示されます。これらのパラメータは変更でき、表示される情報はパラメータ設定に従って動的に変わります。

結果は、キュー、ジョブ、コンピューティング使用量、インスタンスタイプ、ライセンス製品に基づいてグループ化できます。ライセンス製品を選択した場合、コストは特定のライセンスに対して計算されます。他のすべてのグループについては、各タスクの実行にかかる時間を合計して時間が計算されます。

使用量エクスプローラーは、設定したフィルター条件に基づいて 100 件の結果のみを返します。結果は、作成された日付のタイムスタンプの降順で一覧表示されます。結果が 100 件を超える場合は、エラーメッセージが表示されます。クエリを絞り込んで、結果の数を減らすことができます。

- より小さい時間範囲を選択する
- 選択するキューの数を減らす
- ジョブではなくキュー別にグループ化するなど、別のグループを選択する

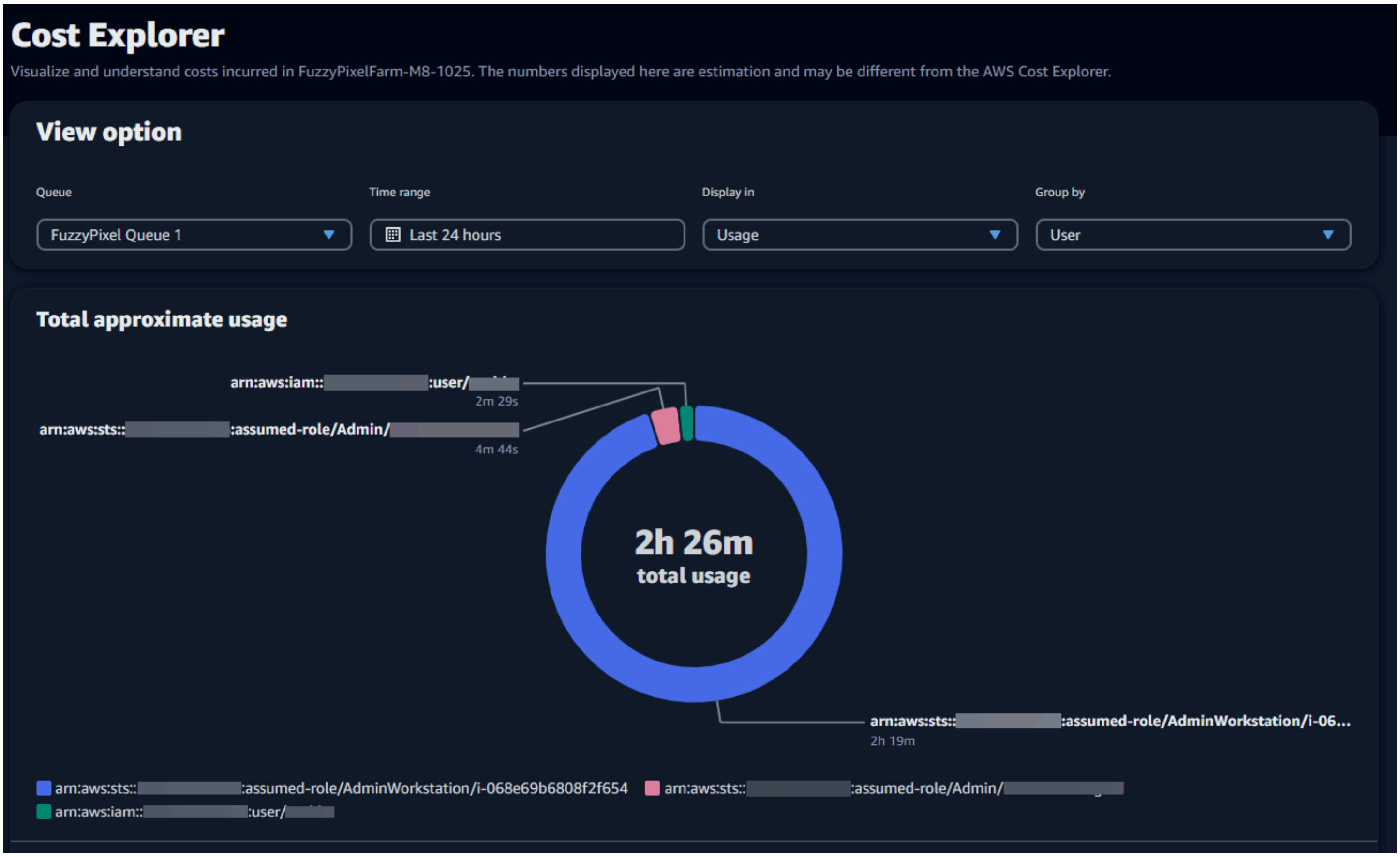
トピック

- ビジュアルグラフを使用してデータを確認する
- メトリクスの内訳を表示する
- キューのおおよそのランタイムを表示する

ビジュアルグラフを使用してデータを確認する

データを視覚的な形式で確認して、より多くの分析や注意が必要な傾向や潜在的な領域を特定できます。Usage Explorer には、全体的な使用量とコストを表示する円グラフと、合計を小さな小計にグループ化するオプションが用意されています。

> **Note**
>
> グラフには、上位 5 つの結果と他の結果が組み合わされた結果のみが「その他」セクションに表示されます。すべての結果は、グラフの下の内訳セクションで表示できます。



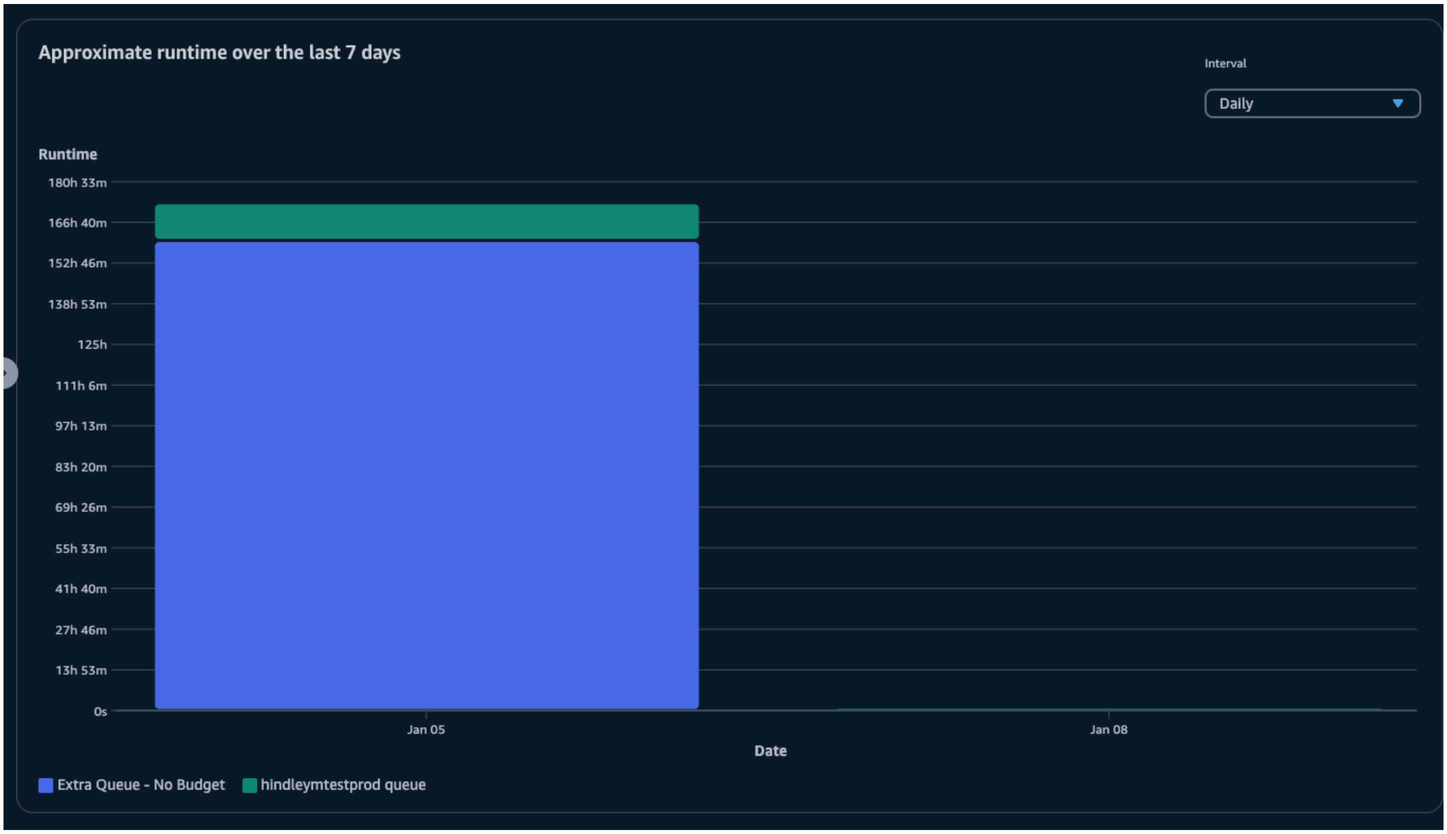
メトリクスの内訳を表示する

円グラフの下には、特定のメトリクスのより詳細な内訳が表示されます。これはパラメータの変更に応じて変化します。デフォルトでは、使用状況エクスプローラーに 5 つの結果が表示されます。内訳セクションのページ割り矢印を使用して結果をスクロールできます。

デフォルトでは、内訳は最小限に抑えられます。結果を展開して表示するには、すべての内訳を表示矢印を選択します。内訳をダウンロードするには、データのダウンロード を選択します。

キューのおおよそのランタイムを表示する

指定したさまざまな間隔に基づいて、キューのおおよそのランタイムを表示することもできます。間隔オプションは、時間単位、日単位、週単位、月単位です。間隔を選択すると、グラフにはキューのおおよそのランタイムが表示されます。



コスト管理

AWS Deadline Cloud は、ジョブのコストを制御および視覚化するのに役立つ予算と使用状況エクスプローラーを提供します。ただし、Deadline Cloud は Amazon S3 などの他の AWS サービスを使用します。これらのサービスのコストは、Deadline Cloud の予算や Usage Explorer には反映されず、使用量に基づいて個別に課金されます。Deadline Cloud の設定方法によっては、次の AWS サービスだけでなく、その他のサービスも使用できます。

サービス	料金ページ
Amazon CloudWatch Logs	Amazon CloudWatch Logs の料金
Amazon Elastic Compute Cloud	Amazon Elastic Compute Cloud の料金
AWS Key Management Service	AWS Key Management Service 料金表
AWS PrivateLink	AWS PrivateLink 料金表
Amazon Simple Storage Service	Amazon Simple Storage Service の料金表

サービス	料金ページ
Amazon Virtual Private Cloud	Amazon Virtual Private Cloud の料金

コスト管理のベストプラクティス

次のベストプラクティスを使用すると、Deadline Cloud を使用する際のコストと、コストと効率の間で得られるトレードオフを理解して制御できます。

> **Note**
>
> Deadline Cloud を使用する最終コストは、多数の AWS サービス、処理する作業量、ジョブを実行する AWS リージョン 間の相互作用によって異なります。以下のベストプラクティスはガイドラインであり、コストを大幅に削減できない場合があります。

CloudWatch ログのベストプラクティス

Deadline Cloud は、ワーカーログとタスクログを CloudWatch ログに送信します。これらのログの収集、保存、分析には料金が発生します。タスクのモニタリングに必要な最小限のデータのみをログに記録することで、コストを削減できます。

キューまたはフリートを作成すると、Deadline Cloud は次の名前の CloudWatch ロググループを作成します。

- `aws/deadline/<FARM_ID>/<FLEET_ID>`
- `aws/deadline/<FARM_ID>/<QUEUE_ID>`

デフォルトでは、これらのログには有効期限はありません。ロググループの保持ポリシーを調整して古いログを削除し、ストレージコストを削減できます。ログを Simple Storage Service (Amazon S3) にエクスポートすることもできます。Amazon S3 のストレージコストは、 のストレージコストよりも低くなります CloudWatch。詳細については、「Amazon S3 へのログデータのエクスポート」を参照してください。

Amazon EC2 のベストプラクティス

Amazon EC2 インスタンスは、サービスマネージドフリートとカスタマーマネージドフリートの両方に使用できます。次の 3 つの考慮事項があります。

- サービスマネージドフリートの場合、フリートの最小ワーカー数を設定することで、1 つ以上のインスタンスを常に使用可能にすることができます。最小ワーカー数を 0 に設定すると、フリートは常にこの数のワーカーを実行します。これにより、Deadline Cloud がジョブの処理を開始するのにかかる時間を短縮できますが、インスタンスのアイドル時間に対して課金されます。
- サービスマネージドフリートの場合は、フリートの最大サイズを設定します。これにより、フリートが自動スケーリングできるインスタンスの数が制限されます。処理を待っているジョブが他にもあっても、フリートはこのサイズを超えることはありません。
- サービスマネージドフリートとカスタマーマネージドフリートの両方で、フリートで Amazon EC2 インスタンスタイプを指定できます。小さいインスタンスを使用すると、1 分あたりのコストは削減されますが、ジョブの完了に時間がかかる場合があります。逆に、インスタンスが大きいほど 1 分あたりのコストは高くなりますが、ジョブを完了する時間を短縮できます。ジョブがインスタンスに配置する要求を理解することで、コストを削減できます。
- 可能な場合は、フリートの Amazon EC2 スポットインスタンスを選択します。スポットインスタンスは割引価格で利用できますが、オンデマンドリクエストによって中断される可能性があります。オンデマンドインスタンスは秒単位で課金され、中断されることはありません。

のベストプラクティス AWS KMS

デフォルトでは、Deadline Cloud は AWS 所有キーを使用してデータを暗号化します。このキーには課金されません。

カスタマーマネージドキーを使用してデータを暗号化することもできます。独自のキーを使用すると、キーの使用方法に基づいて課金されます。既存のキーを使用する場合、これは追加使用の増分コストになります。

のベストプラクティス AWS PrivateLink

を使用して AWS PrivateLink 、インターフェイスエンドポイントを使用して VPC と Deadline Cloud 間の接続を作成できます。接続を作成するときに、Deadline Cloud API アクションをすべて呼び出すことができます。作成したエンドポイントごとに 1 時間ごとに課金されます。を使用する場合は PrivateLink、少なくとも 3 つのエンドポイントを作成する必要があります。設定によっては、最大 5 つのエンドポイントが必要になる場合があります。

Amazon S3 のベストプラクティス

Deadline Cloud は Amazon S3 を使用して、処理、ジョブアタッチメント、出力、ログ用のアセットを保存します。Amazon S3 に関連するコストを削減するには、保存するデータの量を減らします。いくつかの提案：

- 現在使用中のアセット、または間もなく使用されるアセットのみを保存します。
- S3 ライフサイクル設定を使用して、S3 バケットから未使用のファイルを自動的に削除します。

Amazon VPC のベストプラクティス

カスタマーマネージドフリートに使用状況ベースのライセンスを使用する場合は、アカウントに作成された Amazon VPC エンドポイントである Deadline Cloud ライセンスエンドポイントを作成します。このエンドポイントは時間単位の料金で課金されます。コストを削減するには、使用量ベースのライセンスを使用していない場合はエンドポイントを削除します。

のセキュリティ Deadline Cloud

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、 AWS とユーザーの間で共有される責任です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ — AWS は、 AWS のサービス で実行されるインフラストラクチャを保護する責任を担います AWS クラウド。また、 は、安全に使用できるサービス AWS も提供します。コンプライアンスAWS プログラムコンプライアンスプログラム の一環として、サードパーティーの監査者は定期的にセキュリティの有効性をテストおよび検証。に適用されるコンプライアンスプログラムの詳細については AWS Deadline Cloud、「コンプライアンスプログラムAWS のサービス による対象範囲内の」、「コンプライアンスプログラム」を参照してください。
- クラウドのセキュリティ — お客様の責任は AWS のサービス 、使用する によって決まります。また、お客様は、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、 を使用する際の責任共有モデルの適用方法を理解するのに役立ちます Deadline Cloud。以下のトピックでは、セキュリティおよびコンプライアンスの目的 Deadline Cloud を達成するために を設定する方法を示します。また、 Deadline Cloud リソースのモニタリングや保護 AWS のサービス に役立つ他の の使用方法についても説明します。

トピック

でのデータ保護 Deadline Cloud

責任 AWS 共有モデル、 でのデータ保護に適用されます AWS Deadline Cloud。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する AWS のサービス のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された「AWS 責任共有モデルおよび GDPR」のブログ記事を参照してください。

データ保護の目的で、 認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須であり TLS 1.3 がお勧めです。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、API、Deadline Cloud または AWS のサービス SDK を使用して AWS CLIまたは他の を使用する場合も同様です。 AWS SDKs 名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

トピック

- 保管中の暗号化
- 転送中の暗号化

- キー管理
- ネットワーク間トラフィックのプライバシー
- オプトアウト

保管中の暗号化

AWS Deadline Cloud AWS Key Management Service（AWS KMS）に保存されている暗号化キーを使用して保管中のデータを暗号化することで、機密データを保護します。保管時の暗号化は、 AWS リージョン Deadline Cloud が利用可能なすべての で使用できます。

データの暗号化とは、ディスクに保存された機密データが、有効なキーがないユーザーまたはアプリケーションによって読み取れないことを意味します。データを復号できるのは、有効なマネージドキーを持つ当事者のみです。

Deadline Cloud が AWS KMS を使用して保管中のデータを暗号化する方法については、「」を参照してくださいキー管理。

転送中の暗号化

転送中のデータの場合、 AWS Deadline Cloud は Transport Layer Security (TLS) 1.2 または 1.3 を使用して、サービスとワーカーの間で送信されるデータを暗号化します。TLS 1.2 は必須であり TLS 1.3 がお勧めです。さらに、Virtual Private Cloud (VPC) を使用する場合は、 AWS PrivateLink を使用して VPC と の間にプライベート接続を確立できます Deadline Cloud。

キー管理

新しいファームを作成するときは、次のいずれかのキーを選択してファームデータを暗号化できます。

- AWS 所有の KMS キー – ファームの作成時にキーを指定しない場合のデフォルトの暗号化タイプ。KMS キーは によって所有されます AWS Deadline Cloud。 AWS 所有キーを表示、管理、または使用することはできません。ただし、データを暗号化するキーを保護するためにアクションを実行する必要はありません。詳細については、「デベロッパーガイド」の「AWS 所有キー」を参照してください。 AWS Key Management Service
- カスタマーマネージド KMS キー – ファームの作成時にカスタマーマネージドキーを指定します。ファーム内のすべてのコンテンツは KMS キーで暗号化されます。キーはアカウントに保存され、ユーザーが作成、所有、管理し、 AWS KMS 料金が適用されます。ユーザーは、KMS キーに関する完全なコントロール権を持ちます。次のようなタスクを実行できます。

- キーポリシーの確立と維持
- IAM ポリシーとグラントの策定と維持
- キーポリシーの有効化と無効化
- タグの追加
- キーエイリアスの作成

Deadline Cloud ファームで使用されるカスタマー所有のキーを手動でローテーションすることはできません。キーの自動ローテーションがサポートされています。

詳細については、「 AWS Key Management Service デベロッパーガイド」の「カスタマー所有キー」を参照してください。

カスタマーマネージドキーを作成するには、「 AWS Key Management Service デベロッパーガイド」の「対称カスタマーマネージドキーの作成」の手順に従います。

AWS KMS 許可 Deadline Cloud の使用方法

Deadline Cloud には、カスタマーマネージドキーを使用するための許可が必要です。カスタマーマネージドキーで暗号化されたファームを作成すると、 は、指定した KMS キーへのアクセスを取得する`CreateGrant`リクエスト AWS KMS を に送信することで、ユーザーに代わってグラント Deadline Cloud を作成します。

Deadline Cloud は複数の許可を使用します。各許可は、データの暗号化または復号 Deadline Cloud 化を必要とする の異なる部分によって使用されます。 Deadline Cloud また、 は、Amazon Simple Storage Service、Amazon Elastic Block Store、 など、ユーザーに代わってデータを保存するために使用される他の AWS サービスへのアクセスを許可するために許可も使用します OpenSearch。

がサービスマネージドフリート内のマシンを管理 Deadline Cloud できるようにする権限には、Deadline Cloud サービスプリンシパル`GranteePrincipal`の代わりに のアカウント番号とロールが含まれます。これは一般的ではありませんが、ファームに指定されたカスタマーマネージド KMS キーを使用して、サービスマネージドフリートのワーカーの Amazon EBS ボリュームを暗号化するために必要です。

お客様が管理する キーポリシー

キーポリシーは、カスタマーマネージドキーへのアクセスを制御します。各キーには、キーを使用できるユーザーとその使用方法を決定するステートメントを含むキーポリシーが 1 つだけ必要です。カスタマーマネージドキーを作成するときに、キーポリシーを指定できます。詳細について

は、AWS Key Management Service デベロッパーガイド の「カスタマーマネージドキーへのアクセスの管理」を参照してください。

の最小 IAM ポリシー CreateFarm

カスタマーマネージドキーを使用してコンソールまたは `CreateFarm` API オペレーションを使用してファームを作成するには、次の AWS KMS API オペレーションを許可する必要があります。

- `kms:CreateGrant` - カスタマーマネージドキーに許可を追加します。指定された AWS KMS キーへのコンソールアクセスを許可します。詳細については、「 デベロッパーガイド」の「許可の使用」を参照してください。 AWS Key Management Service
- `kms:Decrypt` – ファーム内のデータを復号 Deadline Cloud 化できるようにします。
- `kms:DescribeKey` – がキーを検証 Deadline Cloud できるように、カスタマーマネージドキーの詳細を提供します。
- `kms:GenerateDataKey` — 一意のデータキーを使用してデータを暗号化 Deadline Cloud できます。

次のポリシーステートメントは、 `CreateFarm`オペレーションに必要なアクセス許可を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DeadlineCreateGrants",
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey",
                "kms:CreateGrant",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws::kms:us-west-2:111122223333:key/1234567890abcdef0",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "deadline.us-west-2.amazonaws.com"
                }
            }
        }
    ]
```

```
}
```

読み取り専用オペレーションの最小 IAM ポリシー

ファーム、キュー、フリートに関する情報の取得など、読み取り専用 Deadline Cloud オペレーションにカスタマーマネージドキーを使用するには。次の AWS KMS API オペレーションを許可する必要があります。

- `kms:Decrypt` – ファーム内のデータを復号 Deadline Cloud 化できるようにします。
- `kms:DescribeKey` – がキーを検証 Deadline Cloud できるように、カスタマーマネージドキーの詳細を提供します。

次のポリシーステートメントは、読み取り専用オペレーションに必要なアクセス許可を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DeadlineReadOnly",
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "deadline.us-west-2.amazonaws.com"
                }
            }
        }
    ]
}
```

読み取り/書き込みオペレーションの最小 IAM ポリシー

ファーム、キュー、フリートの作成や更新などの読み取り/書き込み Deadline Cloud オペレーションにカスタマーマネージドキーを使用するには。次の AWS KMS API オペレーションを許可する必要があります。

- kms:Decrypt – ファーム内のデータを復号 Deadline Cloud 化できるようにします。
- kms:DescribeKey – がキーを検証 Deadline Cloud できるように、カスタマーマネージドキーの詳細を提供します。
- kms:GenerateDataKey — 一意のデータキーを使用してデータを暗号化 Deadline Cloud できます。

次のポリシーステートメントは、 CreateFarmオペレーションに必要なアクセス許可を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DeadlineReadWrite",
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey",
            ],
            "Resource": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "deadline.us-west-2.amazonaws.com"
                }
            }
        }
    ]
}
```

暗号化キーのモニタリング

Deadline Cloud ファームで AWS KMS カスタマーマネージドキーを使用する場合、 AWS CloudTrailまたは Amazon CloudWatch Logs を使用して、 Deadline Cloud が に送信するリクエストを追跡できます AWS KMS。

CloudTrail グラントの イベント

次の例の CloudTrail イベントは、許可が作成されたときに発生します。通常は`CreateFarm`、、`CreateMonitor`、または `CreateFleet`オペレーションを呼び出すときに発生します。


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:SampleUser01",
        "arn": "arn:aws::sts::111122223333:assumed-role/Admin/SampleUser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE",
                "arn": "arn:aws::iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-04-23T02:05:26Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "deadline.amazonaws.com"
    },
    "eventTime": "2024-04-23T02:05:35Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "deadline.amazonaws.com",
    "userAgent": "deadline.amazonaws.com",
    "requestParameters": {
        "operations": [
            "CreateGrant",
            "Decrypt",
            "DescribeKey",
            "Encrypt",
            "GenerateDataKey"
```

```
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:deadline:farmId": "farm-abcdef12345678900987654321fedcba",
                "aws:deadline:accountId": "111122223333"
            }
        },
        "granteePrincipal": "deadline.amazonaws.com",
        "keyId": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "retiringPrincipal": "deadline.amazonaws.com"
    },
    "responseElements": {
        "grantId": "6bbe819394822a400fe5e3a75d0e9ef16c1733143fff0c1fc00dc7ac282a18a0",
        "keyId": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE44444"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

CloudTrail 復号化の イベント

次の CloudTrail イベント例は、カスタマーマネージド KMS キーを使用して値を復号するときに発生します。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
```

```
        "principalId": "AROAIGDTESTANDEXAMPLE:SampleUser01",
        "arn": "arn:aws::sts::111122223333:assumed-role/SampleRole/SampleUser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE",
                "arn": "arn:aws::iam::111122223333:role/SampleRole",
                "accountId": "111122223333",
                "userName": "SampleRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-04-23T18:46:51Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "deadline.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:51:44Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "deadline.amazonaws.com",
    "userAgent": "deadline.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:deadline:farmId": "farm-abcdef12345678900987654321fedcba",
            "aws:deadline:accountId": "111122223333",
            "aws-crypto-public-key": "AotL+SAMPLEVALUEiOMEXAMPLEaaqNOTREALaGTESTONLY
+p/5H+EuKd4Q=="
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "responseElements": null,
    "requestID": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeffffff",
    "eventID": "ffffffff-eeee-dddd-cccc-bbbbbbaaaaaa",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
```

```
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

CloudTrail 暗号化の イベント

次の CloudTrail イベント例は、カスタマーマネージド KMS キーを使用して値を暗号化するときに発生します。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:SampleUser01",
        "arn": "arn:aws::sts::111122223333:assumed-role/SampleRole/SampleUser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE",
                "arn": "arn:aws::iam::111122223333:role/SampleRole",
                "accountId": "111122223333",
                "userName": "SampleRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-04-23T18:46:51Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "deadline.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:52:40Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
```

```
    "awsRegion": "us-west-2",
    "sourceIPAddress": "deadline.amazonaws.com",
    "userAgent": "deadline.amazonaws.com",
    "requestParameters": {
        "numberOfBytes": 32,
        "encryptionContext": {
            "aws:deadline:farmId": "farm-abcdef12345678900987654321fedcba",
            "aws:deadline:accountId": "111122223333",
            "aws-crypto-public-key": "AotL+SAMPLEVALUEiOMEXAMPLEaaqNOTREALaGTESTONLY
+p/5H+EuKd4Q=="
        },
        "keyId": "arn:aws::kms:us-
west-2:111122223333:key/abcdef12-3456-7890-0987-654321fedcba"
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws::kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE33333"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


カスタマーマネージド KMS キーの削除

AWS Key Management Service（AWS KMS) でカスタマー管理の KMS キーを削除すると、破壊的になり、潜在的に危険です。これにより、キーマテリアルとキーに関連付けられているすべてのメタデータが削除され、元に戻すことはできません。カスタマーマネージド KMS キーを削除すると、そのキーで暗号化されたデータを復号できなくなります。これは、データが回復不能になることを意味します。

そのため、 AWS KMS では、KMS キーを削除する前に、最大 30 日間の待機期間がお客様に与えられます。デフォルトの待機時間は、30 日です。

待機期間について

カスタマーマネージド KMS キーの削除は破壊的で潜在的に危険であるため、7～30 日間の待機期間を設定する必要があります。デフォルトの待機時間は、30 日です。

ただし、実際の待機期間は、スケジュールした期間よりも最大 24 時間長くなる場合があります。キーが削除される実際の日時を取得するには、 DescribeKeyオペレーションを使用します。また、[General configuration] (一般的な設定) セクションのキーの詳細ページにある AWS KMS コンソールでは、削除のためにスケジュールされた日付を確認することが可能です。タイムゾーンに注意してください。

削除の待機期間中は、カスタマーマネージドキーのステータスおよびキーの状態が削除保留中になります。

- 削除保留中のカスタマーマネージド KMS キーは、暗号化オペレーションに使用することはできません。
- AWS KMS は、削除保留中のカスタマーマネージド KMS キーのバッキングキーをローテーションしません。

カスタマーマネージド KMS キーの削除の詳細については、「デベロッパーガイド」の「カスタマーマスターキーの削除」を参照してください。 AWS Key Management Service

ネットワーク間トラフィックのプライバシー

AWS Deadline Cloud はAmazon Virtual Private Cloud (Amazon VPC) をサポートして接続を保護します。Amazon VPC は、Virtual Private Cloud (VPC) のセキュリティを強化、モニタリングするために使用できる機能を提供します。

VPC 内で実行される Amazon Elastic Compute Cloud (Amazon EC2) インスタンスを使用して、カスタマーマネージドフリート (CMF) を設定できます。を使用するように Amazon VPC エンドポイントをデプロイすることで AWS PrivateLink、CMF のワーカーと Deadline Cloud エンドポイント間のトラフィックは VPC 内に留まります。さらに、インスタンスへのインターネットアクセスを制限するように VPC を設定できます。

サービスマネージドフリートでは、ワーカーはインターネットからアクセスできませんが、インターネットアクセスがあり、インターネット経由で Deadline Cloud サービスに接続できます。

オプトアウト

AWS Deadline Cloud は、 の開発と改善に役立つ特定の運用情報を収集します Deadline Cloud。収集されたデータには、 AWS アカウント ID やユーザー ID などが含まれているため、 に問題がある場合、正しく識別できます Deadline Cloud。また、リソース IDs (該当する場合は FarmID または QueueID) JobAttachments、製品名 (例: など ） WorkerAgent、製品バージョンなどの Deadline Cloud 特定の情報を収集します。

アプリケーション設定を使用して、このデータ収集をオプトアウトできます。クライアントワークステーションとフリートワーカーの両方が Deadline Cloudとやり取りする各コンピュータは、個別にオプトアウトする必要があります。

Deadline Cloud モニター - デスクトップ

Deadline Cloud モニター - デスクトップは、クラッシュが発生したときやアプリケーションが開かれたときなどの運用情報を収集し、アプリケーションに問題が発生したときの把握に役立ちます。この運用情報の収集をオプトアウトするには、設定ページに移動し、データ収集をオンにして Deadline Cloud Monitor のパフォーマンスを測定します。

オプトアウトすると、デスクトップモニターは運用データを送信しなくなります。以前に収集されたデータは保持され、サービスの改善に引き続き使用できます。詳細については、 データプライバシーのよくある質問を参照してください。

AWS Deadline Cloud CLI とツール

AWS Deadline Cloud CLI、送信者、ワーカーエージェントはすべて、クラッシュが発生したときやジョブが送信されたときなどの運用情報を収集し、これらのアプリケーションで問題が発生したときの把握に役立ちます。この運用情報の収集をオプトアウトするには、次のいずれかの方法を使用します。

- ターミナルで、 と入力します**deadline config set telemetry.opt_out true**。

  これにより、現在のユーザーとして実行されているときに CLI、送信者、ワーカーエージェントがオプトアウトされます。
- Deadline Cloud ワーカーエージェントをインストールするときは、**--telemetry-opt-out**コマンドライン引数を追加します。例えば **./install.sh --farm-id $FARM_ID --fleet-id $FLEET_ID --telemetry-opt-out** です。
- ワーカーエージェント、CLI、または送信者を実行する前に、環境変数を設定します。**DEADLINE_CLOUD_TELEMETRY_OPT_OUT=true**

オプトアウトすると、 Deadline Cloud ツールは運用データを送信しなくなります。以前に収集されたデータは保持され、サービスの改善に引き続き使用できます。詳細については、 データプライバシーのよくある質問を参照してください。

Deadline Cloud での Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に Deadline Cloud リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで AWS のサービス 使用できる です。

トピック

対象者

AWS Identity and Access Management (IAM) の使用方法は、Deadline Cloud で行う作業によって異なります。

サービスユーザー – Deadline Cloud サービスを使用してジョブを実行する場合、管理者から必要な認証情報とアクセス許可が与えられます。Deadline Cloud の機能をさらに使用して作業を行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解しておくと、管理者に適切な許可をリクエストするうえで役立ちます。Deadline Cloud の機能にアクセスできない場合は、「」を参照してくださいAWS Deadline Cloud のアイデンティティとアクセスのトラブルシューティング。

サービス管理者 – 社内の Deadline Cloud リソースを担当している場合は、通常、Deadline Cloud へのフルアクセスがあります。サービスユーザーがどの Deadline Cloud 機能やリソースにアクセスす

るかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で Deadline Cloud で IAM を使用する方法の詳細については、「」を参照してくださいDeadline Cloud と IAM の連携方法。

IAM 管理者 – IAM 管理者は、Deadline Cloud へのアクセスを管理するポリシーの作成方法の詳細について確認する場合があります。IAM で使用できる Deadline Cloud アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいDeadline Cloud のアイデンティティベースのポリシーの例。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして AWS アカウントのルートユーザー、または IAM ロールを引き受けて認証（にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center（IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報は、フェデレーティッド ID の例です。フェデレーティッド ID としてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 ユーザーガイド」の「 へのサインイン AWS アカウント方法AWS サインイン 」を参照してください。

AWS プログラムで にアクセスする場合、 は Software Development Kit (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、IAM ユーザーガイドの API AWS リクエストの署名を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS では、アカウントのセキュリティを強化するために多要素認証 (MFA) を使用することをお勧めします。詳細については、「AWS IAM Identity Center ユーザーガイド」の「多要素認証」および「IAM ユーザーガイド」の「AWSでの多要素認証 (MFA) の使用」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての AWS のサービス およびリソースへの完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーティッドアイデンティティ

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに、一時的な認証情報を使用して にアクセスするための ID プロバイダーとのフェデレーションの使用を要求 AWS のサービス します。

フェデレーティッド ID は、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、AWS Directory Service、アイデンティティセンターディレクトリのユーザー、または ID ソースを通じて提供された認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレーティッド ID が にアクセスすると AWS アカウント、ロールを引き受け、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、 AWS IAM Identity Centerを使用することをお勧めします。IAM Identity Center でユーザーとグループを作成することも、独自の ID ソース内のユーザーとグループのセットに接続して同期して、すべての AWS アカウント とアプリケーションで使用することもできます。IAM Identity Center の詳細については、「AWS IAM Identity Center ユーザーガイド」の「IAM Identity Center とは」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、IAM ユーザーガイド の「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する許可を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時的な認証情報が提供されます。詳細については、「IAM ユーザーガイド」の「IAM ユーザー (ロールではなく) の作成が適している場合」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。ロール を切り替える AWS Management Console ことで、で IAM ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API AWS CLI オペレーションを呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションの詳細については、「IAM ユーザーガイド」の「Creating a role for a third-party Identity Provider」(サードパーティーアイデンティティプロバイダー向けロールの作成) を参照してください。IAM Identity Center を使用する場合は、許可セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。アクセス許可セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「アクセス許可セット」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、 (ロールをプロキシとして使用する代わりに) ポリシーをリソースに直接アタッチできます。

クロスアカウントアクセスのロールとリソースベースのポリシーの違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントリソースアクセス」を参照してください。

- クロスサービスアクセス — 一部の は、他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの許可、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行する場合 AWS、ユーザーはプリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。
  - サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。
  - サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスリンクロールの許可を表示できますが、編集することはできません。
- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。 AWS ロールを EC2 インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して許可を付与する」を参照してください。

IAM ロールと IAM ユーザーのどちらを使用するかについては、「IAM ユーザーガイド」の「(IAM ユーザーではなく) IAM ロールをいつ作成したら良いのか?」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、 AWS ID またはリソースにアタッチします。ポリシーは AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの許可を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 権限ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「マネージドポリシーとインラインポリシーの比較」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシー や Amazon S3 バケットポリシー があげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための許可を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon S3、 AWS WAF、および Amazon VPC は、ACLs。ACL の詳細については、『Amazon Simple Storage Service デベロッパーガイド』の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 - アクセス許可の境界は、アイデンティティベースのポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。アクセス許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティのアクセス許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPs は、 の組織または組織単位 (OU) に対する最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、 AWS

アカウント ビジネスが所有する複数の をグループ化して一元管理するサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、AWS Organizations ユーザーガイドの「SCP の仕組み」を参照してください。

- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

Deadline Cloud と IAM の連携方法

IAM を使用して Deadline Cloud へのアクセスを管理する前に、Deadline Cloud で使用できる IAM 機能について学びます。

Deadline Cloud で使用できる AWS IAM の機能

IAM 機能	Deadline Cloud のサポート
アイデンティティベースのポリシー	Yes
リソースベースのポリシー	No
ポリシーアクション	Yes
ポリシーリソース	はい
ポリシー条件キー (サービス固有)	はい

IAM 機能	Deadline Cloud のサポート
ACL	No
ABAC (ポリシー内のタグ)	はい
一時的な認証情報	はい
転送アクセスセッション (FAS)	はい
サービスロール	あり
サービスリンクロール	いいえ

Deadline Cloud およびその他の がほとんどの IAM 機能と AWS のサービス 連携する方法の概要を把握するには、「IAM ユーザーガイド」のAWS 「IAM と連携する のサービス」を参照してください。

Deadline Cloud のアイデンティティベースのポリシー

アイデンティティベースポリシーをサポートする	Yes

アイデンティティベースポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用できるすべての要素については、「IAM ユーザーガイド」の「IAM JSON ポリシーの要素のリファレンス」を参照してください。

Deadline Cloud のアイデンティティベースのポリシーの例

Deadline Cloud アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいDeadline Cloud のアイデンティティベースのポリシーの例。

Deadline Cloud 内のリソースベースのポリシー

リソースベースのポリシーのサポート	No

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシー や Amazon S3 バケットポリシー があげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、アカウント全体、または別のアカウントの IAM エンティティをリソースベースのポリシーのプリンシパルとして指定します。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる にある場合 AWS アカウント、信頼されたアカウントの IAM 管理者は、プリンシパルエンティティ (ユーザーまたはロール) にリソースへのアクセス許可も付与する必要があります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーをさらに付与する必要はありません。詳細については、「IAM ユーザーガイド」の「IAM でのクロスアカウントリソースアクセス」を参照してください。

Deadline Cloud のポリシーアクション

ポリシーアクションに対するサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

JSON ポリシーの`Action`要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。一致する API オペレーションのない許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数のアクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

このアクションは、関連付けられたオペレーションを実行するための権限を付与するポリシーで使用されます。

Deadline Cloud アクションのリストを確認するには、「サービス認証リファレンス」の「Deadline Cloud AWS で定義されるアクション」を参照してください。

Deadline Cloud のポリシーアクションは、アクションの前に次のプレフィックスを使用します。

```
deadline
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

```
"Action": [
      "deadline:action1",
      "deadline:action2"
         ]
```

Deadline Cloud アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいDeadline Cloud のアイデンティティベースのポリシーの例。

Deadline Cloud のポリシーリソース

ポリシーリソースに対するサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースにどのような条件でアクションを実行できるかということです。

Resource JSON ポリシー要素は、アクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

Deadline Cloud リソースタイプとその ARNs」の「Deadline Cloud AWS で定義されるリソース」を参照してください。 どのアクションで各リソースの ARN を指定できるかについては、「Deadline Cloud AWS で定義されるアクション」を参照してください。

Deadline Cloud アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいDeadline Cloud のアイデンティティベースのポリシーの例。

Deadline Cloud のポリシー条件キー

サービス固有のポリシー条件キーのサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

`Condition` 要素 (または `Condition` ブロック) を使用すると、ステートメントが有効な条件を指定できます。`Condition` 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成することで、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の `Condition` 要素を指定するか、1 つの `Condition` 要素に複数のキーを指定すると、 AWS は `AND` 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理`OR`オペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートします。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS 「 グローバル条件コンテキストキー」を参照してください。

Deadline Cloud の条件キーのリストを確認するには、「サービス認証リファレンス」の「Deadline Cloud AWS の条件キー」を参照してください。条件キーを使用できるアクションとリソースについては、「Deadline Cloud AWS で定義されるアクション」を参照してください。

Deadline Cloud アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいDeadline Cloud のアイデンティティベースのポリシーの例。

Deadline Cloud ACLs

ACL のサポート	No

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための許可を持つかをコントロールします。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Deadline Cloud での ABAC

ABAC のサポート (ポリシー内のタグ)	はい

属性ベースのアクセス制御 (ABAC) は、属性に基づいてアクセス許可を定義するアクセス許可戦略です。では AWS、これらの属性はタグ と呼ばれます。タグは、IAM エンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、ABAC の最初の手順です。その後、プリンシパルのタグがアクセスしようとしているリソースのタグと一致した場合にオペレーションを許可するように ABAC ポリシーをします。

ABAC は、急成長する環境やポリシー管理が煩雑になる状況で役立ちます。

タグに基づいてアクセスを管理するには、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値ははいです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

ABAC の詳細については、IAM ユーザーガイド の「ABAC とは?」を参照してください。ABAC をセットアップするステップを説明するチュートリアルについては、「IAM ユーザーガイド」の「属性ベースのアクセス制御 (ABAC) を使用する」を参照してください。

Deadline Cloud での一時的な認証情報の使用

一時的な認証情報のサポート	はい

一部の AWS のサービス は、一時的な認証情報を使用してサインインすると機能しません。一時的な認証情報 AWS のサービス を使用する などの詳細については、IAM ユーザーガイドのAWS のサービス 「IAM と連携する 」を参照してください。

ユーザー名とパスワード以外の AWS Management Console 方法で にサインインする場合、一時的な認証情報を使用します。例えば、会社の Single Sign-On (SSO) リンク AWS を使用して にアクセスすると、そのプロセスによって一時的な認証情報が自動的に作成されます。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えに関する詳細については、「IAM ユーザーガイド」の「ロールへの切り替え (コンソール)」を参照してください。

一時的な認証情報は、 AWS CLI または AWS API を使用して手動で作成できます。その後、これらの一時的な認証情報を使用して . AWS recommends にアクセスできます AWS。この際、長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成することをお勧めします。詳細については、「IAM の一時的セキュリティ認証情報」を参照してください。

Deadline Cloud の転送アクセスセッション

転送アクセスセッション (FAS) をサポート	はい

IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

Deadline Cloud のサービスロール

サービスロールに対するサポート	あり

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細につい

ては、「IAM ユーザーガイド」の「AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。

> **Warning**
>
> サービスロールのアクセス許可を変更すると、Deadline Cloud の機能が破損する可能性があります。Deadline Cloud が指示する場合以外は、サービスロールを編集しないでください。

Deadline Cloud のサービスにリンクされたロール

サービスにリンクされたロールのサポート	いいえ

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールの権限を表示できますが、編集することはできません。

サービスにリンクされたロールの作成または管理の詳細については、「IAM と提携するAWS のサービス」を参照してください。表の中から、Service-linked role (サービスにリンクされたロール) 列に Yes と記載されたサービスを見つけます。サービスリンクロールに関するドキュメントをサービスで表示するには、はい リンクを選択します。

Deadline Cloud のアイデンティティベースのポリシーの例

デフォルトでは、ユーザーとロールには Deadline Cloud リソースを作成または変更するアクセス許可はありません。また、、 AWS Command Line Interface （ AWS CLI ） AWS Management Console、または AWS API を使用してタスクを実行することはできません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き受けることができます。

これらサンプルの JSON ポリシードキュメントを使用して、IAM アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

各リソースタイプの ARNs」の「Deadline Cloud AWS のアクション、リソース、および条件キー」を参照してください。

トピック

- ポリシーのベストプラクティス
- Deadline Cloud コンソールの使用
- キューにジョブを送信するポリシー
- ライセンスエンドポイントの作成を許可するポリシー
- 特定のファームキューのモニタリングを許可するポリシー

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かが Deadline Cloud リソースを作成、アクセス、または削除できるどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、アクセス許可をさらに減らすことをお勧めします。詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」または「AWS ジョブ機能の管理ポリシー」を参照してください。
- 最小特権を適用する – IAM ポリシーで許可を設定する場合は、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用して許可を適用する方法の詳細については、「IAM ユーザーガイド」の「IAM でのポリシーとアクセス許可」を参照してください。
- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、 などの特定の を通じてサービスアクションが使用される場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、「IAM ユーザーガイド」の [IAM JSON policy elements: Condition] (IAM JSON ポリシー要素: 条件) を参照してください。
- IAM Access Analyzer を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM Access Analyzer は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサ

ポートします。詳細については、「IAM ユーザーガイド」の「IAM Access Analyzer ポリシーの検証」を参照してください。

- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化するために MFA を有効にします。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、「IAM ユーザーガイド」の「MFA 保護 API アクセスの設定」を参照してください。

IAM でのベストプラクティスの詳細については、「IAM ユーザーガイド」の「IAM でのセキュリティのベストプラクティス」を参照してください。

Deadline Cloud コンソールの使用

AWS Deadline Cloud コンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可により、 の Deadline Cloud リソースの詳細を一覧表示および表示できます AWS アカウント。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスが許可されます。

ユーザーとロールが引き続き Deadline Cloud コンソールを使用できるようにするには、エンティティに Deadline Cloud `ConsoleAccess`または `ReadOnly` AWS マネージドポリシーもアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

キューにジョブを送信するポリシー

この例では、特定のファーム内の特定のキューにジョブを送信するアクセス許可を付与するスコープダウンポリシーを作成します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SubmitJobsFarmAndQueue",
            "Effect": "Allow",
            "Action": "deadline:CreateJob",
```

```
            "Resource": "arn:aws:deadline:REGION:ACCOUNT_ID:farm/FARM_A/queue/QUEUE_B/
job/*"
        }
    ]
}
```

ライセンスエンドポイントの作成を許可するポリシー

この例では、ライセンスエンドポイントを作成および管理するために必要なアクセス許可を付与するスコープダウンポリシーを作成します。このポリシーを使用して、ファームに関連付けられた VPC のライセンスエンドポイントを作成します。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "SID": "CreateLicenseEndpoint",
        "Effect": "Allow",
        "Action": [
            "deadline:CreateLicenseEndpoint",
            "deadline:DeleteLicenseEndpoint",
            "deadline:GetLicenseEndpoint",
            "deadline:UpdateLicenseEndpoint",
            "deadline:ListLicenseEndpoints",
            "deadline:PutMeteredProduct",
            "deadline:DeleteMeteredProduct",
            "deadline:ListMeteredProducts",
            "deadline:ListAvailableMeteredProducts",
            "ec2:CreateVpcEndpoint",
            "ec2:DescribeVpcEndpoints",
            "ec2:DeleteVpcEndpoints"
        ],
        "Resource": "*"
    }]
}
```

特定のファームキューのモニタリングを許可するポリシー

この例では、特定のファームの特定のキュー内のジョブをモニタリングするアクセス許可を付与するスコープダウンポリシーを作成します。

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

```
    "Statement": [{
        "Sid": "MonitorJobsFarmAndQueue",
        "Effect": "Allow",
        "Action": [
            "deadline:SearchJobs",
            "deadline:ListJobs",
            "deadline:GetJob",
            "deadline:SearchSteps",
            "deadline:ListSteps",
            "deadline:ListStepConsumers",
            "deadline:ListStepDependencies",
            "deadline:GetStep",
            "deadline:SearchTasks",
            "deadline:ListTasks",
            "deadline:GetTask",
            "deadline:ListSessions",
            "deadline:GetSession",
            "deadline:ListSessionActions",
            "deadline:GetSessionAction"
        ],
        "Resource": [
            "arn:aws:deadline:REGION:123456789012:farm/FARM_A/queue/QUEUE_B",
            "arn:aws:deadline:REGION:123456789012:farm/FARM_A/queue/QUEUE_B/*"
        ]
    }]
}
```

AWS Deadline Cloud の マネージドポリシー

AWS 管理ポリシーは、 によって作成および管理されるスタンドアロンポリシーです AWS。 AWS 管理ポリシーは、多くの一般的なユースケースに対するアクセス許可を付与するように設計されているため、ユーザー、グループ、ロールへのアクセス許可の割り当てを開始できます。

AWS 管理ポリシーは、すべての AWS お客様が使用できるため、特定のユースケースに対して最小特権のアクセス許可を付与しない場合があることに注意してください。ユースケース別にカスタマーマネージドポリシーを定義して、マネージドポリシーを絞り込むことをお勧めします。

AWS 管理ポリシーで定義されているアクセス許可は変更できません。が AWS 管理ポリシーで定義されたアクセス許可 AWS を更新すると、ポリシーがアタッチされているすべてのプリンシパル ID (ユーザー、グループ、ロール) が更新されます。 は、新しい AWS のサービス が起動されたとき、

または既存のサービスで新しい API AWS オペレーションが使用可能になったときに、 AWS 管理ポリシーを更新する可能性が最も高くなります。

詳細については、「IAM ユーザーガイド」の「AWS 管理ポリシー」を参照してください。

AWS マネージドポリシー: AWSDeadlineCloud-FleetWorker

(IAM) ID に`AWSDeadlineCloud-FleetWorker`ポリシーをアタッチできます AWS Identity and Access Management 。

このポリシーは、このフリートのワーカーに、サービスに接続してタスクを受信するために必要なアクセス許可を付与します。

許可の詳細

このポリシーには、以下の許可が含まれています。

- `deadline` – プリンシパルがフリート内のワーカーを管理できるようにします。

ポリシーの詳細の JSON リストについては、AWS マネージドポリシーリファレンスガイドのAWSDeadlineCloud「-FleetWorker」を参照してください。

AWS マネージドポリシー: AWSDeadlineCloud-WorkerHost

`AWSDeadlineCloud-WorkerHost` ポリシーは IAM ID にアタッチできます。

このポリシーは、サービスに最初に接続するために必要なアクセス許可を付与します。Amazon Elastic Compute Cloud (Amazon EC2) インスタンスプロファイルとして使用できます。

許可の詳細

このポリシーには、以下の許可が含まれています。

- `deadline` — プリンシパルがワーカーを作成できるようにします。

ポリシーの詳細の JSON リストについては、AWS マネージドポリシーリファレンスガイドのAWSDeadlineCloud「-WorkerHost」を参照してください。

AWS 管理ポリシー: AWSDeadlineCloud-UserAccessFarms

`AWSDeadlineCloud-UserAccessFarms` ポリシーは IAM ID にアタッチできます。

このポリシーにより、ユーザーは自分がメンバーであるファームとメンバーシップレベルに基づいてファームデータにアクセスできます。

許可の詳細

このポリシーには、以下の許可が含まれています。

- `deadline` – ユーザーがファームデータにアクセスできるようにします。
- `ec2` – ユーザーが Amazon EC2 インスタンスタイプの詳細を表示できるようにします。
- `identitystore` – ユーザーがユーザー名とグループ名を表示できるようにします。

ポリシーの詳細の JSON リストについては、AWS マネージドポリシーリファレンスガイドのAWSDeadlineCloud「-UserAccessFarms」を参照してください。

AWS 管理ポリシー: AWSDeadlineCloud-UserAccessFleets

`AWSDeadlineCloud-UserAccessFleets` ポリシーは IAM ID にアタッチできます。

このポリシーにより、ユーザーは自分がメンバーであるファームとメンバーシップレベルに基づいてフリートデータにアクセスできます。

許可の詳細

このポリシーには、以下の許可が含まれています。

- `deadline` – ユーザーがファームデータにアクセスできるようにします。
- `ec2` – ユーザーが Amazon EC2 インスタンスタイプの詳細を表示できるようにします。
- `identitystore` – ユーザーがユーザー名とグループ名を表示できるようにします。

ポリシーの詳細の JSON リストについては、AWS マネージドポリシーリファレンスガイドのAWSDeadlineCloud「-UserAccessFleets」を参照してください。

AWS マネージドポリシー: AWSDeadlineCloud-UserAccessJobs

`AWSDeadlineCloud-UserAccessJobs` ポリシーは IAM ID にアタッチできます。

このポリシーにより、ユーザーは自分がメンバーであるファームとメンバーシップレベルに基づいてジョブデータにアクセスできます。

許可の詳細

このポリシーには、以下の許可が含まれています。

- `deadline` – ユーザーがファームデータにアクセスできるようにします。
- `ec2` – ユーザーが Amazon EC2 インスタンスタイプの詳細を表示できるようにします。
- `identitystore` – ユーザーがユーザー名とグループ名を表示できるようにします。

ポリシーの詳細の JSON リストについては、AWS マネージドポリシーリファレンスガイドのAWSDeadlineCloud「-UserAccessJobs」を参照してください。

AWS 管理ポリシー: AWSDeadlineCloud-UserAccessQueues

`AWSDeadlineCloud-UserAccessQueues` ポリシーは IAM ID にアタッチできます。

このポリシーにより、ユーザーは自分がメンバーであるファームとメンバーシップレベルに基づいてキューデータにアクセスできます。

許可の詳細

このポリシーには、以下の許可が含まれています。

- `deadline` – ユーザーがファームデータにアクセスできるようにします。
- `ec2` – ユーザーが Amazon EC2 インスタンスタイプの詳細を表示できるようにします。
- `identitystore` – ユーザーがユーザー名とグループ名を表示できるようにします。

ポリシーの詳細の JSON リストについては、AWS マネージドポリシーリファレンスガイドのAWSDeadlineCloud-UserAccessQueues」を参照してください。

AWS マネージドポリシーに対する Deadline Cloud の更新

Deadline Cloud の AWS マネージドポリシーの更新に関する詳細を、このサービスがこれらの変更の追跡を開始した以降の分について表示します。このページの変更に関する自動アラートについては、Deadline Cloud ドキュメント履歴ページの RSS フィードをサブスクライブしてください。

変更	説明	日付
Deadline Cloud が変更の追跡を開始しました	Deadline Cloud が AWS マネージドポリシーの変更の追跡を開始しました。	2024 年 4 月 2 日

AWS Deadline Cloud のアイデンティティとアクセスのトラブルシューティング

以下の情報は、Deadline Cloud と IAM の使用時に発生する可能性がある一般的な問題の診断と修正に役立ちます。

トピック

- Deadline Cloud でアクションを実行する権限がない
- iam を実行する権限がありません。PassRole
- 自分の 以外のユーザーに Deadline Cloud リソース AWS アカウント へのアクセスを許可したい

Deadline Cloud でアクションを実行する権限がない

「I am not authorized to perform an action in Amazon Bedrock」というエラーが表示された場合、そのアクションを実行できるようにポリシーを更新する必要があります。

次のエラー例は、`mateojackson` IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要な`deadline:`*GetWidget* アクセス許可を持っていない場合に発生するものです。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 deadline:GetWidget on resource: my-example-widget
```

この場合、`deadline:`*GetWidget* アクションを使用して *my-example-widget*リソースへのアクセスを許可するように、`mateojackson` ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

iam を実行する権限がありません。PassRole

`iam:PassRole` アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して Deadline Cloud にロールを渡すことができるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

次の例のエラーは、 という IAM `marymajor` ユーザーがコンソールを使用して Deadline Cloud でアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに `iam:PassRole` アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

自分の 以外のユーザーに Deadline Cloud リソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- Deadline Cloud がこれらの機能をサポートしているかどうかを確認するには、「」を参照してくださいDeadline Cloud と IAM の連携方法。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。

- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、IAM ユーザーガイドの外部認証されたユーザーへのアクセスの提供 (ID フェデレーション) を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いについては、IAM ユーザーガイドの「IAM でのクロスアカウントリソースアクセス」を参照してください。

のコンプライアンス検証 Deadline Cloud

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、コンプライアンスプログラムAWS のサービス による対象範囲内のコンプライアンスプログラムを参照し、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS 「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「 でのレポートのダウンロード AWS Artifact」の」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- アマゾン ウェブ サービスにおける HIPAA セキュリティとコンプライアンスのアーキテクチャ — このホワイトペーパーでは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法について説明します。

> **Note**
>
> すべて AWS のサービス HIPAA の対象となるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。

- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。

- AWS カスタマーコンプライアンスガイド — コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、ガイダンスを保護し AWS のサービス 、複数のフレームワーク (米国国立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化機構 (ISO) を含む) のセキュリティコントロールにマッピングするためのベストプラクティスをまとめています。
- 「デベロッパーガイド」の「ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に確認できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、「Security Hub のコントロールリファレンス」を参照してください。
- Amazon GuardDuty – これにより AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか環境を監視することで、、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。 GuardDuty は、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件への対応に役立ちます。
- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

の耐障害性 Deadline Cloud

AWS グローバルインフラストラクチャは、 AWS リージョン およびアベイラビリティーゾーンを中心に構築されています。 は、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離および分離された複数のアベイラビリティーゾーン AWS リージョン を提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョン およびアベイラビリティーゾーンの詳細については、AWS 「グローバルインフラストラクチャ」を参照してください。

AWS Deadline Cloud は、ジョブアタッチメント S3 バケットに保存されているデータをバックアップしません。ジョブアタッチメントデータのバックアップは、Amazon S3 S3 バックアップメカニズムを使用して有効にできますAWS Backup。

Deadline Cloud のインフラストラクチャセキュリティ

マネージドサービスである AWS Deadline Cloud は AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「 Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開した API コールを使用して、ネットワーク経由で Deadline Cloud にアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS)。TLS 1.2 は必須で TLS 1.3 がお勧めです。
- DHE (楕円ディフィー・ ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

Deadline Cloud は、 AWS PrivateLink Virtual Private Cloud (VPC) エンドポイントポリシーの使用をサポートしていません。エンドポイントへのフルアクセスを許可する AWS PrivateLink デフォルトのポリシーを使用します。詳細については、「 ユーザーガイド」の「デフォルトのエンドポイントポリシー」を参照してください。 AWS PrivateLink

Deadline Cloud での設定と脆弱性の分析

AWS は、ゲストオペレーティングシステム (OS) やデータベースのパッチ適用、ファイアウォール設定、ディザスタリカバリなどの基本的なセキュリティタスクを処理します。これらの手順は適切な第三者によって確認され、証明されています。詳細については、以下のリソース を参照してください。

- 責任共有モデル

- Amazon Web Services: セキュリティプロセスの概要 (ホワイトペーパー)

AWS Deadline Cloud は、サービスマネージドフリートまたはカスタマーマネージドフリートのタスクを管理します。

- サービスマネージドフリートの場合、Deadline Cloud はゲストオペレーティングシステムを管理します。
- カスタマーマネージドフリートの場合、オペレーティングシステムを管理する責任があります。

AWS Deadline Cloud の設定と脆弱性の分析の詳細については、「」を参照してください。

- Deadline Cloud のセキュリティのベストプラクティス

サービス間での不分別な代理処理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、1 つのサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスは、本来ならアクセスすることが許可されるべきではない方法でその許可を使用して、別のお客様のリソースに対する処理を実行するように操作される場合があります。これを防ぐために、 AWS には、アカウント内のリソースへのアクセス権が付与されたサービスプリンシパルですべてのサービスのデータを保護するために役立つツールが用意されています。

リソースポリシーで aws:SourceArnおよび aws:SourceAccount グローバル条件コンテキストキーを使用して、 が別のサービスに AWS Deadline Cloud 付与するアクセス許可をリソースに制限することをお勧めします。クロスサービスアクセスにリソースを 1 つだけ関連付けたい場合は、`aws:SourceArn` を使用します。そのアカウント内のリソースをクロスサービスの使用に関連付けることを許可する場合は、`aws:SourceAccount` を使用します。

「混乱した代理」問題から保護するための最も効果的な方法は、リソースの完全なAmazon リソースネーム (ARN)を指定しながら、グローバル条件コンテキストキー `aws:SourceArn` を使用することです。リソースの完全な ARN が不明な場合や、複数のリソースを指定する場合には、グローバルコンテキスト条件キー `aws:SourceArn` で、ARN の未知部分を示すためにワイルドカード文字 (*) を使用します。例えば、`arn:aws:deadline:*:123456789012:*` です。

`aws:SourceArn` の値に Amazon S3 バケット ARN などのアカウント ID が含まれていない場合は、両方のグローバル条件コンテキストキーを使用して、アクセス許可を制限する必要があります。

次の例は、 で `aws:SourceArn`および `aws:SourceAccount` グローバル条件コンテキストキーを使用して、混乱した代理問題 Deadline Cloud を回避する方法を示しています。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "deadline.amazonaws.com"
    },
    "Action": "deadline:ActionName",
    "Resource": [
      "*"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:deadline:*:123456789012:*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

インターフェイスエンドポイント (AWS PrivateLink) AWS Deadline Cloud を使用した へのアクセス

を使用して AWS PrivateLink 、VPC と の間にプライベート接続を作成できます AWS Deadline Cloud。インターネットゲートウェイ、NAT デバイス、VPN 接続、または AWS Direct Connect 接続を使用せずに、VPC 内にある Deadline Cloud かのように にアクセスできます。VPC のインスタンスは、パブリック IP アドレスがなくても Deadline Cloudにアクセスできます。

このプライベート接続を確立するには、 AWS PrivateLinkを利用したインターフェイスエンドポイントを作成します。インターフェイスエンドポイントに対して有効にする各サブネットにエンドポイン

トネットワークインターフェイスを作成します。これらは、 Deadline Cloud宛てのトラフィックのエントリポイントとして機能するリクエスタ管理型ネットワークインターフェイスです。

詳細については、『AWS PrivateLink ガイド』の「AWS のサービス でアクセスする」を参照してください。

に関する考慮事項 Deadline Cloud

のインターフェイスエンドポイントを設定する前に Deadline Cloud、「 AWS PrivateLink ガイド」の「インターフェイス VPC エンドポイントを使用して AWS サービスにアクセスする」を参照してください。

Deadline Cloud は、インターフェイスエンドポイントを介したすべての API アクションの呼び出しをサポートします。

デフォルトでは、インターフェイスエンドポイントを介して へのフルアクセス Deadline Cloud が許可されます。または、セキュリティグループをエンドポイントネットワークインターフェイスに関連付けて、インターフェイスエンドポイント Deadline Cloud を介した へのトラフィックを制御することもできます。

Deadline Cloud は VPC エンドポイントポリシーをサポートしていません。詳細については、『AWS PrivateLink ガイド』の「Control access to VPC endpoints using endpoint policies (エンドポイントポリシーを使用して VPC エンドポイントへのアクセスをコントロールする)」を参照してください。

Deadline Cloud エンドポイント

Deadline Cloud は、 を使用してサービスにアクセスするために 2 つのエンドポイントを使用します AWS PrivateLink。

ワーカーは`com.amazonaws.`*`region`*`.deadline.scheduling`エンドポイントを使用して、キューからタスクを取得し、進捗状況を に報告し Deadline Cloud、タスク出力を送り返します。カスタマーマネージドフリートを使用している場合は、管理オペレーションを使用しない限り、作成する必要があるのはスケジューリングエンドポイントだけです。例えば、ジョブがより多くのジョブを作成する場合は、管理エンドポイントが `CreateJob`オペレーションを呼び出すようにする必要があります。

Deadline Cloud モニターは を使用して、キューとフリートの作成と変更、ジョブ、ステップ、タスクのリストの取得など、ファーム内のリソース`com.amazonaws.`*`region`*`.deadline.management`を管理します。

Deadline Cloud には、次の AWS サービスエンドポイントのエンドポイントも必要です。

- Deadline Cloud は AWS STS を使用してワーカーを認証し、ワーカーがジョブアセットにアクセスできるようにします。の詳細については AWS STS、「 AWS Identity and Access Management ユーザーガイド」の「IAM での一時的なセキュリティ認証情報」を参照してください。
- インターネット接続のないサブネットにカスタマーマネージドフリートを設定する場合は、ワーカーがログを書き込めるように Amazon CloudWatch Logs の VPC エンドポイントを作成する必要があります。詳細については、「 によるモニタリング CloudWatch」を参照してください。
- ジョブアタッチメントを使用する場合は、ワーカーがアタッチメントにアクセスできるように、Amazon Simple Storage Service (Amazon S3) の VPC エンドポイントを作成する必要があります。詳細については、「 のジョブアタッチメント Deadline Cloud」を参照してください。

のエンドポイントを作成する Deadline Cloud

Amazon VPC コンソールまたは AWS Command Line Interface () Deadline Cloud を使用して、 のインターフェイスエンドポイントを作成できますAWS CLI。詳細については、「AWS PrivateLink ガイド」の「インターフェイスエンドポイントを作成」を参照してください。

次のサービス名 Deadline Cloud を使用して、 の管理エンドポイントとスケジューリングエンドポイントを作成します。*region* を、 AWS リージョン をデプロイした に置き換えます Deadline Cloud。

```
com.amazonaws.region.deadline.management
```

```
com.amazonaws.region.deadline.scheduling
```

インターフェイスエンドポイントのプライベート DNS を有効にすると、デフォルトのリージョン DNS 名 Deadline Cloud を使用して に API リクエストを実行できます。例えば、`worker.deadline.us-east-1.amazonaws.com`ワーカーオペレーションの場合は 、その他すべてのオペレーション`management.deadline.us-east-1.amazonaws.com`の場合は です。

また、次のサービス名 AWS STS を使用して のエンドポイントを作成する必要があります。

```
com.amazonaws.region.sts
```

カスタマーマネージドフリートがインターネット接続のないサブネット上にある場合は、次のサービス名を使用して CloudWatch Logs エンドポイントを作成する必要があります。

```
com.amazonaws.region.logs
```

ジョブアタッチメントを使用してファイルを転送する場合は、次のサービス名を使用して Amazon S3 エンドポイントを作成する必要があります。

```
com.amazonaws.region.s3
```

Deadline Cloud のセキュリティのベストプラクティス

AWS Deadline Cloud (Deadline Cloud) には、独自のセキュリティポリシーを開発および実装する際に考慮すべきいくつかのセキュリティ機能が用意されています。以下のベストプラクティスは一般的なガイドラインであり、完全なセキュリティソリューションを説明するものではありません。これらのベストプラクティスはお客様の環境に必ずしも適切または十分でない可能性があるので、処方箋ではなく、あくまで有用な考慮事項とお考えください。

> **Note**
>
> 多くのセキュリティトピックの重要性の詳細については、「 責任共有モデル」を参照してください。

データ保護

データ保護の目的で、 AWS アカウント 認証情報を保護し、 AWS Identity and Access Management (IAM) を使用して個々のアカウントを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須であり TLS 1.3 がお勧めです。
- を使用して API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon S3 (Amazon Simple Storage Service) に保存されている個人情報の発見と保護を支援する Amazon Macie などのアドバンストマネージドセキュリティサービスを使用します。
- コマンドラインインターフェイスまたは API を使用して AWS にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」 を参照してください。

顧客のアカウント番号などの機密の識別情報は、[Name] (名前)フィールドなどの自由形式のフィールドに配置しないことを強くお勧めします。これは、コンソール、API、または SDK AWS AWS のサービス を使用して Deadline Cloud AWS CLIまたは他の を使用する場合も同様です。 AWS SDKs Deadline Cloud または他のサービスに入力したデータは、診断ログに取り込まれる可能性があります。外部サーバーへの URL を指定するときは、そのサーバーへのリクエストを検証するための認証情報を URL に含めないでください。

AWS Identity and Access Management アクセス許可

ユーザー、 AWS Identity and Access Management (IAM) ロール、およびユーザーに最小権限を付与して、 AWS リソースへのアクセスを管理します。 AWS アクセス認証情報を作成、配布、ローテーション、および取り消すための認証情報管理ポリシーと手順を確立します。詳細については、「IAM ユーザーガイド」の「IAM のベストプラクティス」を参照してください。

ユーザーおよびグループとしてジョブを実行する

Deadline Cloud でキュー機能を使用する場合は、オペレーティングシステム (OS) ユーザーとそのプライマリグループを指定して、OS ユーザーがキューのジョブに対する最小特権のアクセス許可を持つようにするのがベストプラクティスです。

「ユーザーとして実行」 (およびグループ) を指定すると、キューに送信されたジョブのプロセスは、その OS ユーザーを使用して実行され、そのユーザーの関連する OS アクセス許可を継承します。

フリートとキューの設定を組み合わせて、セキュリティ体制を確立します。キュー側では、「ユーザーとして実行されるジョブ」と IAM ロールを指定して、キューのジョブに OS と AWS アクセス許可を使用できます。フリートは、特定のキューに関連付けられているときにキュー内でジョブを実行するインフラストラクチャ (ワーカーホスト、ネットワーク、マウントされた共有ストレージ) を定義します。ワーカーホストで使用可能なデータは、1 つ以上の関連付けられたキューのジョブによってアクセスされる必要があります。ユーザーまたはグループを指定すると、ジョブ内のデータを他のキュー、インストールされている他のソフトウェア、またはワーカーホストにアクセスできる他のユーザーから保護できます。キューにユーザーがない場合、キューはエージェントユーザーとして実行され、任意のキューユーザーを偽装 (sudo) できます。このようにして、ユーザーのないキューは別のキューに権限をエスカレートできます。

ネットワーク

トラフィックが傍受またはリダイレクトされないようにするには、ネットワークトラフィックがルーティングされる方法と場所を保護することが重要です。

ネットワーク環境は、次の方法で保護することをお勧めします。

- Amazon Virtual Private Cloud (Amazon VPC) サブネットルートテーブルを保護して、IP レイヤートラフィックのルーティング方法を制御します。
- ファームまたはワークステーションのセットアップで Amazon Route 53 (Route 53) を DNS プロバイダーとして使用している場合は、Route 53 API への安全なアクセスを確保します。
- オンプレミスのワークステーションやその他のデータセンターを使用する AWS など、 の外部で Deadline Cloud に接続する場合は、オンプレミスのネットワークインフラストラクチャを保護します。これには、ルーター、スイッチ、その他のネットワークデバイスの DNS サーバーとルートテーブルが含まれます。

ジョブとジョブデータ

Deadline Cloud ジョブは、ワーカーホストのセッション内で実行されます。各セッションはワーカーホストで 1 つ以上のプロセスを実行します。通常、出力を生成するにはデータを入力する必要があります。

このデータを保護するには、キューを使用してオペレーティングシステムユーザーを設定できます。ワーカーエージェントはキュー OS ユーザーを使用してセッションサブプロセスを実行します。これらのサブプロセスは、キュー OS ユーザーのアクセス許可を継承します。

これらのサブプロセスアクセスのデータへのアクセスを保護するためのベストプラクティスに従うことをお勧めします。詳細については、責任共有モデルを参照してください。

Farm 構造

Deadline Cloud フリートとキューは、さまざまな方法で配置できます。ただし、特定の手配にはセキュリティ上の影響があります。

ファームは、フリート、キュー、ストレージプロファイルなど、他のファームと Deadline Cloud リソースを共有できないため、最も安全な境界の 1 つです。ただし、ファーム内で外部 AWS リソースを共有することはでき、セキュリティ境界が侵害されます。

適切な設定を使用して、同じファーム内のキュー間にセキュリティ境界を確立することもできます。

次のベストプラクティスに従って、同じファームに安全なキューを作成します。

- フリートを同じセキュリティ境界内のキューにのみ関連付けます。次の点に注意してください。

- ワーカーホストでジョブが実行された後、一時ディレクトリやキューユーザーのホームディレクトリなどにデータが残される可能性があります。
- ジョブの送信先のキューに関係なく、同じ OS ユーザーがサービス所有のフリートワーカーホストですべてのジョブを実行します。
- ジョブがワーカーホストで実行されているプロセスを離れ、他のキューのジョブが実行中の他のプロセスを監視できるようになる場合があります。

- 同じセキュリティ境界内のキューのみが、ジョブアタッチメントの Amazon S3 バケットを共有していることを確認します。
- 同じセキュリティ境界内のキューのみが OS ユーザーを共有していることを確認します。
- ファームに統合されている他の AWS リソースを境界に保護します。

ジョブアタッチメントキュー

ジョブアタッチメントは、Amazon S3 バケットを使用するキューに関連付けられています。

- ジョブアタッチメントは、Amazon S3 バケットのルートプレフィックスに対して書き込みと読み取りを行います。`CreateQueue` API コールでこのルートプレフィックスを指定します。
- バケットには対応する がありQueue Role`、キューユーザーにバケットとルートプレフィックスへのアクセスを許可するロールを指定します。キューを作成するときは、ジョブアタッチメントバケットとルートプレフィックスとともに `Queue Role` Amazon リソースネーム (ARN) を指定します。
- `AssumeQueueRoleForRead`、、および `AssumeQueueRoleForWorker` API オペレーションへの許可された呼び出しは`AssumeQueueRoleForUser`、 の一時的なセキュリティ認証情報のセットを返します`Queue Role`。

キューを作成し、Amazon S3 バケットとルートプレフィックスを再利用すると、情報が不正な当事者に開示されるリスクがあります。例えば、QueueA と QueueB は同じバケットとルートプレフィックスを共有します。安全なワークフローでは、ArtistA は QueueA にアクセスできますが、QueueB にはアクセスできません。ただし、複数のキューがバケットを共有する場合、ArtistA は QueueB データ内のデータにアクセスできます。 QueueA

コンソールは、デフォルトで安全なキューを設定します。キューが共通のセキュリティ境界の一部でない限り、Amazon S3 バケットとルートプレフィックスの個別の組み合わせがあることを確認します。

キューを分離するには、バケットとルートプレフィックスへのキューアクセスQueue Roleのみを許可するように を設定する必要があります。次の例では、各*########*をリソース固有の情報に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:ListBucket",
        "s3:GetBucketLocation"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::JOB_ATTACHMENTS_BUCKET_NAME",
        "arn:aws:s3:::JOB_ATTACHMENTS_BUCKET_NAME/JOB_ATTACHMENTS_ROOT_PREFIX/*"
      ],
      "Condition": {
        "StringEquals": { "aws:ResourceAccount": "ACCOUNT_ID" }
      }
    },
    {
      "Action": ["logs:GetLogEvents"],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:REGION:ACCOUNT_ID:log-group:/aws/deadline/FARM_ID/*"
    }
  ]
}
```

また、ロールに信頼ポリシーを設定する必要があります。次の例では、*########*テキストをリソース固有の情報に置き換えます。

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

```
        "StringEquals": { "aws:SourceAccount": "ACCOUNT_ID" },
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:deadline:REGION:ACCOUNT_ID:farm/FARM_ID"
        }
      }
    },
    {
      "Action": ["sts:AssumeRole"],
      "Effect": "Allow",
      "Principal": { "Service": "credentials.deadline.amazonaws.com" },
      "Condition": {
        "StringEquals": { "aws:SourceAccount": "ACCOUNT_ID" },
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:deadline:REGION:ACCOUNT_ID:farm/FARM_ID"
        }
      }
    }
  ]
}
```

カスタムソフトウェア Amazon S3 バケット

に次のステートメントを追加して`Queue Role`、Amazon S3 バケット内のカスタムソフトウェアにアクセスできます。次の例では、*software_BUCKET_NAME* を S3 バケットの名前に置き換えます。

```
"Statement": [
    {
        "Action": [
            "s3:GetObject",
            "s3:ListBucket"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::SOFTWARE_BUCKET_NAME",
            "arn:aws:s3:::SOFTWARE_BUCKET_NAME/*"
        ]
    }
]
```

Amazon S3 セキュリティのベストプラクティスの詳細については、Amazon Simple Storage Service ユーザーガイドのAmazon S3 のセキュリティのベストプラクティス」を参照してください。

ワーカーホスト

ワーカーホストを保護して、各ユーザーが割り当てられたロールに対してのみオペレーションを実行できるようにします。

ワーカーホストを保護するために、次のベストプラクティスをお勧めします。

- これらのキューに送信されたジョブが同じセキュリティ境界内にある場合を除き、複数のキューで同じ`jobRunAsUser`値を使用しないでください。
- ワーカーエージェントが実行される OS ユーザーの名前`jobRunAsUser`にキューを設定しないでください。
- 目的のキューワークロードに必要な最小特権の OS アクセス許可をキューユーザーに付与します。エージェントプログラムファイルやその他の共有ソフトウェアを操作するためのファイルシステムの書き込み許可がないことを確認します。
- のルートユーザーLinuxと `Administrator` が所有するアカウントのみがWindows所有し、ワーカーエージェントプログラムファイルを変更できることを確認します。
- Linux ワーカーホストでは、ワーカーエージェントユーザーがキューユーザーとしてプロセスを起動`/etc/sudoers`できるようにする`umask`オーバーライドを で設定することを検討してください。この設定は、他のユーザーがキューに書き込まれたファイルにアクセスできないようにするのに役立ちます。
- 信頼できる個人に、ワーカーホストへの最小特権アクセスを付与します。
- ローカル DNS オーバーライド設定ファイル (`/etc/hosts` Linux`C:\Windows\system32\etc\hosts`と 、Windowsおよびワークステーションとワーカーホストオペレーティングシステムでテーブルをルーティングするアクセス許可を制限します。
- ワークステーションとワーカーホストオペレーティングシステムの DNS 設定へのアクセス許可を制限します。
- オペレーティングシステムとインストールされているすべてのソフトウェアに定期的にパッチを適用します。このアプローチには、送信者、アダプター、ワーカーエージェント、OpenJDパッケージなど、Deadline Cloud で特に使用されるソフトウェアが含まれます。
- Windows キュー には強力なパスワードを使用します`jobRunAsUser`。
- キュー のパスワードを定期的にローテーションします`jobRunAsUser`。
- Windows パスワードシークレットへの最小特権アクセスを確保し、未使用のシークレットを削除します。
- キューに、今後実行するスケジュールコマンドの`jobRunAsUser`アクセス許可を与えないでください。

- でLinux、これらのアカウントによる `cron`および へのアクセスを拒否します`at`。
- でWindows、これらのアカウントによるWindowsタスクスケジューラへのアクセスを拒否します。

> **Note**
>
> オペレーティングシステムとインストール済みソフトウェアに定期的にパッチを適用する重要性の詳細については、「責任共有モデル」を参照してください。

ワークステーション

Deadline Cloud にアクセスできるワークステーションを保護することが重要です。このアプローチは、Deadline Cloud に送信するジョブが、 に請求される任意のワークロードを実行できないようにするのに役立ちます AWS アカウント。

アーティストワークステーションを保護するには、次のベストプラクティスをお勧めします。詳細については、 責任共有モデルを参照してください。

- Deadline Cloud を含む AWS、 へのアクセスを提供する永続的な認証情報を保護します。詳細については、「 IAM ユーザーガイド」の「IAM ユーザーのアクセスキーの管理」を参照してください。
- 信頼できる安全なソフトウェアのみをインストールします。
- ID プロバイダーとフェデレーションして、一時的な認証情報 AWS を使用して にアクセスすることをユーザーに要求します。
- Deadline Cloud 送信者プログラムファイルに対する安全なアクセス許可を使用して、改ざんを防止します。
- 信頼できる個人にアーティストワークステーションへの最小特権アクセスを付与します。
- Deadline Cloud Monitor を通じて取得した送信者とアダプターのみを使用してください。
- ワークステーション`/etc/hosts`とワーカーホストオペレーティングシステムのテーブルへのアクセス許可を制限し、ルーティングします。
- ワークステーションとワーカーホストオペレーティングシステム`/etc/resolv.conf`のアクセス許可を に制限します。

- オペレーティングシステムとインストールされているすべてのソフトウェアに定期的にパッチを適用します。このアプローチには、送信者、アダプター、ワーカーエージェント、OpenJDパッケージなど、Deadline Cloud で特に使用されるソフトウェアが含まれます。

Deadline Cloud AWS のモニタリング

モニタリングは、Deadline Cloud (Deadline Cloud) AWS と AWS ソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。マルチポイント障害が発生した場合は、その障害をより簡単にデバッグできるように、 AWS ソリューションのすべての部分からモニタリングデータを収集します。Deadline Cloud のモニタリングを開始する前に、以下の質問に対する回答を含むモニタリング計画を作成する必要があります。

- どのような目的でモニタリングしますか?
- どのリソースをモニタリングしますか?
- どのくらいの頻度でこれらのリソースをモニタリングしますか?
- どのモニタリングツールを使用しますか?
- 誰がモニタリングタスクを実行しますか?
- 問題が発生したときに誰が通知を受け取りますか?

AWS および Deadline Cloud には、 リソースをモニタリングし、潜在的なインシデントに対応するために使用できるツールが用意されています。これらのツールの中にはモニタリングを行うものもあれば、手動による介入を必要とするものもあります。モニタリングタスクはできるだけ自動化する必要があります。

- Amazon CloudWatch は、 AWS リソースと で実行しているアプリケーションを AWS リアルタイムでモニタリングします。メトリクスを収集および追跡し、カスタマイズされたダッシュボードを作成し、指定されたメトリックが指定したしきい値に達したときに通知またはアクションを実行するアラームを設定できます。例えば、 で Amazon EC2 インスタンスの CPU 使用率やその他のメトリクス CloudWatch を追跡し、必要に応じて新しいインスタンスを自動的に起動できます。詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。

  Deadline Cloud には 3 つの CloudWatch メトリクスがあります。
- Amazon CloudWatch Logs を使用すると、Amazon EC2 インスタンスやその他のソースからログファイルをモニタリング、保存 CloudTrail、およびアクセスできます。 CloudWatch Logs はログファイル内の情報をモニタリングし、特定のしきい値に達したときに通知できます。高い耐久性を備えたストレージにログデータをアーカイブすることもできます。詳細については、「Amazon CloudWatch Logs ユーザーガイド」を参照してください。
- Amazon EventBridge を使用すると、 AWS サービスを自動化し、アプリケーションの可用性の問題やリソースの変更などのシステムイベントに自動的に対応できます。 AWS サービスからのイベ

ントは、ほぼリアルタイムで EventBridge に配信されます。簡単なルールを記述して、注目するイベントと、イベントがルールに一致した場合に自動的に実行するアクションを指定できます。詳細については、「Amazon ユーザーガイド EventBridge 」を参照してください。

- AWS CloudTrail は、 AWS アカウントによって、またはアカウントに代わって行われた API コールおよび関連イベントをキャプチャし、指定した Amazon S3 バケットにログファイルを配信します。を呼び出したユーザーとアカウント AWS、呼び出し元のソース IP アドレス、呼び出しが発生した日時を特定できます。詳細については、『AWS CloudTrail ユーザーガイド』を参照してください。

トピック

を使用した通話のログ記録 CloudTrail

AWS Deadline Cloud は、ユーザー AWS CloudTrail、ロール、または Deadline Cloud の によって実行されたアクションを記録するサービスである と統合 AWS のサービス されています。は、Deadline Cloud のすべての API コールをイベントとして CloudTrail キャプチャします。キャプチャされた呼び出しには、Deadline Cloud コンソールからの呼び出しと、Deadline Cloud API オペレーションへのコード呼び出しが含まれます。

証跡を作成する場合は、Deadline Cloud の CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、 CloudTrail コンソールのイベント履歴 で最新のイベントを表示できます。によって収集された情報を使用して CloudTrail、Deadline Cloud に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の詳細については CloudTrail、「 AWS CloudTrail ユーザーガイド」を参照してください。

の Deadline Cloud 情報 CloudTrail

CloudTrail アカウントを作成する AWS アカウント と、 で が有効になります。Deadline Cloud でアクティビティが発生すると、そのアクティビティは CloudTrail イベント履歴 の他の AWS のサービス イベントとともにイベントに記録されます。 で最近のイベントを表示、検索、ダウンロードできます AWS アカウント。詳細については、「イベント履歴を使用した CloudTrailイベントの表示」を参照してください。

CloudTrail は、ユーザーが Deadline Cloud モニターにサインインして認証情報を受け取る AWS と、イベントも記録します。ユーザーがサインインすると、ソース `signin.amazonaws.com` と名前 の CloudTrail イベントが発生します`UserAuthentication`。サインインしたユーザーにソース `sts.amazonaws.com` と名前 から AWS 認証情報が与えられると、2 つ目のイベントが発生します`AssumeRole`。ユーザーの ID は、ロールセッション名内の 2 番目のイベントに記録されます。

Deadline Cloud のイベントなど AWS アカウント、 のイベントの継続的な記録については、証跡を作成します。証跡により、 はログファイル CloudTrail を Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョンに適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、他の を設定 AWS のサービス して、 CloudTrail ログで収集されたイベントデータをより詳細に分析し、それに基づく対応を行うことができます。

詳細については、次を参照してください:

「証跡作成の概要」

CloudTrail がサポートするサービスと統合

の Amazon SNS 通知の設定 CloudTrail

複数のリージョンからの CloudTrail ログファイルの受信

複数のアカウントからの CloudTrail ログファイルの受信

Deadline Cloud は、以下のアクションをイベントとして CloudTrail ログファイルに記録することをサポートしています。

- associate-member-to-farm
- associate-member-to-fleet
- associate-member-to-job
- associate-member-to-queue
- assume-fleet-role-for読み取り
- assume-fleet-role-for-ワーカー
- assume-queue-role-for読み取り
- assume-queue-role-for- ユーザー
- assume-queue-role-for-ワーカー
- 予算の作成

- ファームの作成
- create-fleet
- create-license-endpoint
- モニターの作成
- キューの作成
- create-queue-environment
- create-queue-fleet-association
- create-storage-profile
- ワーカーの作成
- 予算を削除
- delete-farm
- delete-fleet
- delete-license-endpoint
- delete-metered-product
- モニターの削除
- delete-queue
- delete-queue-environment
- delete-queue-fleet-association
- delete-storage-profile
- ワーカーの削除
- disassociate-member-from-farm
- disassociate-member-from-fleet
- disassociate-member-from-job
- disassociate-member-from-queue
- get-application-version
- 予算の取得
- ファームの取得
- get-feature-map
- フリートの取得
- get-license-endpoint

- モニターの取得
- get-queue
- get-queue-environment
- get-queue-fleet-association
- get-sessions-statistics-aggregation
- get-storage-profile
- get-storage-profile-forキュー
- list-available-metered-products
- 予算のリスト
- list-farm-members
- ファームのリスト
- list-fleet-members
- フリートのリスト
- list-job-members
- list-license-endpoints
- list-metered-products
- モニターのリスト
- list-queue-environments
- list-queue-fleet-associations
- list-queue-members
- キューのリスト
- list-storage-profiles
- list-storage-profiles-forキュー
- list-tags-for-resource
- put-metered-product
- start-sessions-statistics-aggregation
- タグリソース
- タグなしリソース
- 予算の更新
- ファームの更新

- 更新フリート
- モニターの更新
- 更新キュー
- update-queue-environment
- update-queue-fleet-association
- update-storage-profile
- ワーカーの更新

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- リクエストがルートまたは AWS Identity and Access Management (IAM) ユーザー認証情報のどちらを使用して行われたか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の サービスによって行われたかどうか。

詳細については、CloudTrail「ユーザー ID 要素」を参照してください。

Deadline Cloud ログファイルエントリについて

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには 1 つ以上のログエントリが含まれます。イベントは任意のソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。 CloudTrail ログファイルはパブリック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

この JSON の例は、 **CreateFarm** API の呼び出しによって生成されたログを示しています。

```
{
    "eventVersion": "0",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE-PrincipalID:EXAMPLE-Session",
        "arn": "arn:aws:sts::111122223333:assumed-role/EXAMPLE-UserName/EXAMPLE-
Session",
        "accountId": "111122223333",
```

```
        "accessKeyId": "EXAMPLE-accessKeyId",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE-PrincipalID",
                "arn": "arn:aws:iam::111122223333:role/EXAMPLE-UserName",
                "accountId": "111122223333",
                "userName": "EXAMPLE-UserName"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-03-08T23:25:49Z"
            }
        }
    },
    "eventTime": "2021-03-08T23:25:49Z",
    "eventSource": "deadline.amazonaws.com",
    "eventName": "CreateFarm",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "EXAMPLE-userAgent",
    "requestParameters": {
        "displayName": "example-farm",
        "kmsKeyArn": "arn:aws:kms:us-west-2:111122223333:key/111122223333",
        "X-Amz-Client-Token": "12abc12a-1234-1abc-123a-1a11bc1111a",
        "description": "example-description",
        "tags": {
            "purpose_1": "e2e"
            "purpose_2": "tag_test"
        }
    },
    "responseElements": {
        "farmId": "EXAMPLE-farmID"
    },
    "requestID": "EXAMPLE-requestID",
    "eventID": "EXAMPLE-eventID",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
}
```

この例では、イベントを識別するのに役立つ AWS リージョン、IP アドレス、およびrequestParametersdisplayName「」や「」などのその他のkmsKeyArn「」を示しています。

によるモニタリング CloudWatch

Amazon CloudWatch（ CloudWatch) は raw データを収集し、読み取り可能なほぼリアルタイムのメトリクスに加工します。 CloudWatch コンソールを https://console.aws.amazon.com/cloudwatch/ で開き、Deadline Cloud メトリクスを表示およびフィルタリングできます。

- Deadline Cloud カスタマーマネージドフリートでは、 CloudWatch は と の 2 UnhealthyWorkerCount つのメトリクスを送信しますRecommendedFleetSize。
- これらのメトリクスの名前空間は です。 AWS/DeadlineCloud.
- ディメンションfarmIDと fleetIDを使用してメトリクスをフィルタリングできます。
- どちらのメトリクスも単位 を使用しますcount。

これらの統計は 15 か月間保持されるため、履歴情報にアクセスして、ウェブアプリケーションまたはサービスの動作をより的確に把握できます。また、特定のしきい値をモニタリングするアラームを設定し、しきい値に達したときに通知を送信したりアクションを実行したりできます。詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。

Deadline Cloud には、タスクログとワーカーログの 2 種類のログがあります。タスクログは、スクリプトまたは DCC の実行として実行ログを実行する場合です。タスクログには、アセットのロード、タイルのレンダリング、テクスチャが見つからないなどのイベントが表示される場合があります。

ワーカーログには、ワーカーエージェントのプロセスが表示されます。これには、ワーカーエージェントの起動時、自己登録時、進行状況の報告、設定のロード、タスクの完了などが含まれる場合があります。

Deadline Cloud の場合、ワーカーはこれらのログを Logs CloudWatch にアップロードします。デフォルトでは、ログは期限切れになりません。ジョブが大量のデータを出力する場合、追加のコストが発生する可能性があります。詳細については、「Amazon の CloudWatch 料金」を参照してください。

各ロググループの保持ポリシーを調整できます。保持期間を短くすると、古いログが削除され、ストレージコストの削減に役立ちます。ログを保持するには、ログを削除する前に Amazon Simple Storage Service にアーカイブします。詳細については、「Amazon ユーザーガイド」の「コンソールを使用してログデータを Amazon S3 にエクスポートする CloudWatch 」を参照してください。

> **Note**
>
> CloudWatch ログの読み取りは によって制限されます AWS。多くのアーティストをオンボーディングする場合は、 AWS カスタマーサポートに連絡して、 `GetLogEvents` のクォータの引き上げをリクエストすることをお勧めします CloudWatch。さらに、デバッグしない場合は、ログ末尾ポータルを閉じることをお勧めします。

詳細については、「Amazon CloudWatch ユーザーガイドCloudWatch 」の「ログクォータ」を参照してください。

EventBridge イベントへの対応

Deadline Cloud は Amazon にイベントを送信 EventBridge し、サービスの状態の変更を通知します。 EventBridge およびこれらのイベントを使用して、フリートに変更があったときに通知するなどのアクションを実行するルールを記述できます。詳細については、「Amazon とは EventBridge」を参照してください。

フリートサイズのレコメンデーションの変更

イベントベースの自動スケーリングを使用するようにフリートを設定すると、Deadline Cloud はフリートの管理に使用できるイベントを送信します。これらの各イベントには、フリートの現在のサイズとリクエストされたサイズに関する情報が含まれています。 EventBridge イベントの使用例と Lambda 関数の例を使用してイベントを処理する方法については、「」を参照してくださいDeadline Cloud スケールレコメンデーション機能を使用して Amazon EC2 フリートを自動スケーリングする。

フリートサイズのレコメンデーション変更イベントは、次の場合に送信されます。

- 推奨フリートサイズが変更され、 `oldFleetSize`と異なる場合`newFleetSize`。
- 実際のフリートサイズが推奨フリートサイズと一致しないことをサービスが検出した場合。`workerCount` GetFleet オペレーションレスポンスの から実際のフリートサイズを取得でき

ます。これは、アクティブな Amazon EC2 インスタンスが Deadline Cloud ワーカーとして登録できない場合に発生する可能性があります。

イベントには次の形式があります。

```
{
    "version": "0",
    "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
    "detail-type": "Fleet Size Recommendation Change",
    "source": "aws.deadline",
    "account": "111122223333",
    "time": "2017-12-22T18:43:48Z",
    "region": "us-west-1",
    "resources": [],
    "detail": {
        "farmId": "farm-12345678900000000000000000000000",
        "fleetId": "fleet-12345678900000000000000000000000",
        "oldFleetSize": 1,
        "newFleetSize": 5,
    }
}
```

以下のフィールドはイベントパターンを定義します。

`"source": "aws.deadline"`

このイベントのソースが Deadline Cloud であることを示します。

`"detail-type": "Fleet Size Recommendation Change"`

イベントタイプを特定します。

`"detail": { }`

フリートサイズの推奨変更に関する情報を提供します。

`"farmId": "farm-12345678900000000000000000000000"`

フリートを含むファームの識別子。

`"fleetId": "fleet-12345678900000000000000000000000"`

サイズ変更が必要なフリートの識別子。

`"oldFleetSize": 1`

フリートの現在のサイズ。

`"newFleetSize": 5`

フリートの推奨新しいサイズ。

のクォータ Deadline Cloud

AWS Deadline Cloud は、ジョブの処理に使用できるファーム、フリート、キューなどのリソースを提供します。を作成すると AWS アカウント、 ごとにこれらのリソースにデフォルトのクォータが設定されます AWS リージョン。

Service Quotas は、 のクォータを表示および管理できる中心的な場所です AWS のサービス。使用する多くのリソースのクォータ引き上げをリクエストすることもできます。

のクォータを表示するには Deadline Cloud、Service Quotas コンソール を開きます。ナビゲーションペインで、[AWS のサービス] を選択し、次に [Deadline Cloud] を選択します。

クォータの引き上げをリクエストするには、「Service Quotas ユーザーガイド」の「クォータ引き上げリクエスト」を参照してください。Service Quotas でService Quotas 引き上げフォーム を使用します。

を使用した Deadline Cloud AWS リソースの作成 AWS CloudFormation

AWS Deadline Cloud は AWS CloudFormation、 AWS リソースとインフラストラクチャの作成と管理に費やす時間を短縮できるように、リソースのモデル化とセットアップに役立つサービスであると統合されています。必要なすべての AWS リソース (ファーム、キュー、フリートなど) を記述するテンプレートを作成し、それらのリソースを AWS CloudFormation プロビジョニングして設定します。

を使用すると AWS CloudFormation、テンプレートを再利用して Deadline Cloud リソースを一貫して繰り返しセットアップできます。リソースを 1 回記述し、複数の AWS アカウント およびリージョンで同じリソースを何度もプロビジョニングします。

Deadline クラウドと AWS CloudFormation テンプレート

Deadline Cloud および関連サービスのリソースをプロビジョニングして設定するには、 AWS CloudFormation テンプレート を理解する必要があります。テンプレートは、JSON や YAML でフォーマットされたテキストファイルです。これらのテンプレートは、 AWS CloudFormation スタックでプロビジョニングするリソースを記述します。JSON または YAML に慣れていない場合は、デザイナー を使用して AWS CloudFormation AWS CloudFormation テンプレートの使用を開始できます。詳細については、「AWS CloudFormation ユーザーガイド」の「AWS CloudFormation Designer とは」を参照してください。

Deadline Cloud は、 でのファーム、キュー、フリートの作成をサポートしています AWS CloudFormation。ファーム、キュー、フリートの JSON テンプレートと YAML テンプレートの例を含む詳細については、「 AWS CloudFormation ユーザーガイド」のAWS 「Deadline Cloud」を参照してください。

の詳細 AWS CloudFormation

の詳細については AWS CloudFormation、以下のリソースを参照してください。

- AWS CloudFormation
- AWS CloudFormation ユーザーガイド
- AWS CloudFormation API リファレンス
- AWS CloudFormation コマンドラインインターフェイスユーザーガイド

Deadline クラウドユーザーガイドのドキュメント履歴

以下の表は、AWS Deadline Cloudユーザーガイドの各リリースにおける重要な変更点をまとめたものです。

変更	説明	日付
[初回リリース]	これはDeadline Cloudユーザーガイドの最初のリリースです。	2024 年 4 月 2 日

AWS 用語集

AWS 最新の用語については、『リファレンス』AWS の用語集を参照してください。AWS の用語集

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。