User Guide

AWS Deadline Cloud

Version latest

AWS Deadline Cloud: User Guide

Table of Contents

What is AWS Deadline Cloud?

Deadline Cloud is an AWS service you can use to create and manage rendering projects and jobs on Amazon Elastic Compute Cloud (Amazon EC2) instances directly from digital content creation pipelines and workstations.

Deadline Cloud provides console interfaces, local applications, command line tools, and an API. With Deadline Cloud, you can create, manage, and monitor farms, fleets, jobs, user groups, and storage. You can also specify hardware capabilities, create environments for specific workloads, and integrate the content creation tools that your production requires into your Deadline Cloud pipeline.

Deadline Cloud provides a unified interface to manage all of your rendering projects in one place. You can manage users, assign projects to them, and grant permissions for job roles.

Topics

Features of Deadline Cloud

Here are some of the key ways Deadline Cloud can help you run and manage visual compute workloads:

- Quickly create your farms, queues, and fleets. Monitor their status, and gain insights into the operation of your farm and jobs.
- Centrally manage Deadline Cloud users and groups, and assign permissions.
- Manage sign-in security for project users and external identity providers with AWS IAM Identity Center.
- Securely manage access to project resources with AWS Identity and Access Management (IAM) policies and roles.

- Use tags to organize and quickly find project resources.
- Manage project resource usage and estimated costs for your project.
- Provide a wide range of compute management options to support rendering in the cloud or in person.

Concepts and terminology for Deadline Cloud

To help you get started with AWS Deadline Cloud, this topic explains some of its key concepts and terminology.

Budget manager

Budget manager is part of the Deadline Cloud monitor. Use the budget manager to create and manage budgets. You can also use it to limit activities to stay within budget.

Deadline Cloud Client Library

The Client Library includes a command line interface and library for managing Deadline Cloud. Functionality includes submitting job bundles based on the Open Job Description specification to Deadline Cloud, downloading job attachment outputs, and monitoring your farm using the command line interface.

Digital content creation application (DCC)

Digital content creation applications (DCCs) are third-party products where you create digital content. Examples of DCCs are Maya, Nuke, and Houdini. Deadline Cloud provides job submitter integrated plugins for specific DCCs.

Farm

A farm is a where your project resources are located. It consists of queues and fleets.

Fleet

A fleet is a group of worker nodes that do the rendering. Worker nodes process jobs. A fleet can be associated to multiple queues, and a queue can be associated to multiple fleets.

Job

A job is a rendering request. Users submit jobs. Jobs contain specific job properties that are outlined as steps and tasks.

Job attachments

A job attachment is a Deadline Cloud feature that you can use to manage inputs and outputs for jobs. Job files are uploaded as job attachments during the rendering process. These files can be textures, 3D models, lighting rigs, and other similar items.

Job priority

Job priority is the approximate order that Deadline Cloud processes a job in a queue. You can set the job priority between 1 and 100, jobs with a higher number priority are generally processed first. Jobs with the same priority are processed in the order received.

Job properties

Job properties are settings that you define when submitting a render job. Some examples include frame range, output path, job attachments, renderable camera, and more. The properties vary based on the DCC that the render is submitted from.

Job template

A job template defines the runtime environment and all processes that run as part of a Deadline Cloud job.

Queue

A queue is where submitted jobs are located and scheduled to be rendered. A queue must be associated with a fleet to create a successful render. A queue can be associated with multiple fleets.

Queue-fleet association

When a queue is associated with a fleet, there is a queue-fleet association. Use an association to schedule workers from a fleet to jobs in that queue. You can start and stop associations to control scheduling of work.

Step

A step is one particular process to run in the job.

Deadline Cloud submitter

A Deadline Cloud submitter is a digital content creation (DCC) plugin. Artists use it to submit jobs from a third-party DCC interface that they are familiar with.

Tags

A tag is a label that you can assign to an AWS resource. Each tag consists of a key and an optional value that you define.

With tags, you can categorize your AWS resources in different ways. For example, you could define a set of tags for your account's Amazon EC2 instances that help you track each instance's owner and stack level.

You can also categorize your AWS resources by purpose, owner, or environment. This approach is useful when you have many resources of the same type. You can quickly identify a specific resources based on the tags that you've assigned to it.

Task

A task is a single component of a render step.

Usage-based licensing (UBL)

Usage-based licensing (UBL) is an on-demand licensing model that is available for select third-party products. This model is pay as your go, and you are charged for the number of hours and minutes that you use.

Usage explorer

Usage explorer is a feature of Deadline Cloud monitor. It provides an approximate estimate of your costs and usage.

Worker

Workers belong to fleets and run Deadline Cloud assigned tasks to complete steps and jobs. Workers store the logs from task operations in Amazon CloudWatch Logs. Workers can also use the job attachments feature to sync inputs and outputs to an Amazon Simple Storage Service (Amazon S3) bucket.

Getting started with Deadline Cloud

Use Deadline Cloud to quickly create a render farm with default settings and resources, such as Amazon EC2 instance configuration and Amazon Simple Storage Service (Amazon S3) buckets.

You can also define the settings and resources when you create a render farm. This method takes more time than using the default settings and resources but gives you more control.

After you're familiar with Deadline Cloud Concepts and terminology, see Getting started for step-by-step instructions for creating your farm, adding users, and links to helpful information.

Accessing Deadline Cloud

You can access Deadline Cloud in any of the following ways:

- **Deadline Cloud console**– Access the console in a browser to create a farm and its resources, and manage user access. For more information, see Getting started.
- **Deadline Cloud monitor**– Manage your render jobs, including updating priorities and job statuses. Monitor your farm and view logs and job status. For users with Owner permissions, the Deadline Cloud monitor also provides access to explore usage and create budgets. The Deadline Cloud monitor is available as both a web browser and a desktop application.
- **AWS SDK and AWS CLI**– Use the AWS Command Line Interface (AWS CLI) to call the Deadline Cloud API operations from the command line on your local system. For more information, see Set up a developer workstation.

Related services

Deadline Cloud works with the following AWS services:

- **Amazon CloudWatch**– With CloudWatch, you can monitor your projects and associated AWS resources. For more information, see Monitoring with CloudWatch in the *Deadline Cloud Developer Guide*.
- **Amazon EC2**–This AWS service provides virtual servers that run your applications in the cloud. You can configure your projects to use Amazon EC2 instances for your workloads. For more information, see Amazon EC2 instances.
- **Amazon EC2 Auto Scaling**– With Auto Scaling, you can automatically increase or decrease the number of instances as the demand on your instances changes. Auto Scaling helps to make sure that you're running your desired number of instances, even if an instance fails. If you enable Auto Scaling with Deadline Cloud, instances that are launched by Auto Scaling are automatically registered with the workload. Likewise, instances that are terminated by Auto Scaling are automatically de-registered from the workload. For more information, see the Amazon EC2 Auto Scaling User Guide.
- **AWS PrivateLink**– AWS PrivateLink provides private connectivity between virtual private clouds (VPCs), AWS services, and your on-premises networks, without exposing your traffic to the public

internet. AWS PrivateLink makes it easy to connect services across different accounts and VPCs. For more information, see AWS PrivateLink.

- **Amazon S3**– Amazon S3 is an object storage service. Deadline Cloud uses Amazon S3 buckets to store job attachments. For more information, see the Amazon S3 User Guide.
- **IAM Identity Center**– IAM Identity Center is an AWS service where you can provide users with single sign-on access to all their assigned accounts and applications from one place. You can also centrally manage multi-account access and user permissions to all of your accounts in AWS Organizations. For more information, see AWS IAM Identity Center FAQs.

How Deadline Cloud works

With Deadline Cloud, you can create and manage rendering projects and jobs directly from digital content creation (DCC) pipelines and workstations.

You submit jobs to Deadline Cloud using the AWS SDK, AWS Command Line Interface (AWS CLI), or Deadline Cloud job submitters. Deadline Cloud supports the Open Job Description (OpenJD) for job template specification. For more information, see Open Job Description on the GitHub website.

Deadline Cloud provides job submitters. A *job submitter* is a DCC plugin for submitting render jobs from a third-party DCC interface, such as Maya or Nuke. With a submitter, artists can submit rendering jobs from a third-party interface to Deadline Cloud where project resources are managed and jobs are monitored, all in one location.

With a Deadline Cloud farm, you can create queues and fleets, manage users, and manage project resource usage and costs. A *farm* consists of queues and fleets. A *queue* is where submitted jobs are located and scheduled to be rendered. A *fleet* is a group of worker nodes that run tasks to complete jobs. A queue must be associated with a fleet so that the jobs can render. A single fleet can support multiple queues and a queue can be supported by multiple fleets.

Jobs consist of steps, and each step consists of specific tasks. With the Deadline Cloud monitor, you can access statuses, logs, and other troubleshooting metrics for jobs, steps, and tasks.

Permissions in Deadline Cloud

Deadline Cloud supports the following:

- Managing access to its API operations using AWS Identity and Access Management (IAM)
- Managing access of workforce users using an integration with AWS IAM Identity Center

Before anyone can work on a project, they must have access to that project and the associated farm. Deadline Cloud is integrated with IAM Identity Center to manage workforce authentication and authorization. Users can be added directly to IAM Identity Center, or permission can be connected to your existing identity provider (IdP) such as Okta or Active Directory. IT administrators can grant access permissions to users and groups at different levels. Each subsequent level includes the permissions for the previous levels. The following list describes the four access levels from the lowest level to the highest level:

- **Viewer**– Permission to see resources in the farms, queues, fleets, and jobs they have access to. A viewer can't submit or make changes to jobs.
- **Contributor**– Same as a viewer, but with permission to submit jobs to a queue or farm.
- **Manager**– Same as contributor, but with permission to edit jobs in queues they have access to, and grant permissions on resources that they have access to.
- **Owner**– Same as manager, but can view and create budgets and see usage.

> **Note**
>
> These permissions don't give users access to the AWS Management Console or permission to modify Deadline Cloud infrastructure.

Users must have access to a farm before they can access the associated queues and fleets. User access is assigned to queues and fleets separately within a farm.

You can add users as individuals or as part of a group. Adding groups to a farm, fleet, or queue can make it easier to manage access permissions for large groups of people. For example, if you have a team that is working on a specific project, you can add each of the team members to a group. Then, you can grant access permissions to the entire group for the corresponding farm, fleet, or queue.

Software support with Deadline Cloud

Deadline Cloud works with any software application that can be run from a command line interface and controlled by using parameter values. Deadline Cloud supports the OpenJD specification for describing work as **jobs** with software script **steps** that are parameterized (such as across a frame range) into **tasks**. Assemble OpenJD job instructions into job bundles with Deadline Cloud tools and features to create, run, and license the steps from a third-party software application.

Jobs need licensing to render. Deadline Cloud offers usage-based-licensing (UBL) for a selection of software application licenses that is billed by the hour in minute increments based on usage. With Deadline Cloud, you can also use your own software licenses if you like. If a job can't access a license, it doesn't render and produces an error that displays in the task log in the Deadline Cloud monitor.

Getting started with Deadline Cloud

To create a farm in AWS Deadline Cloud, you can use either the Deadline Cloud console or the AWS Command Line Interface (AWS CLI). Use the console for a guided experience creating the farm, including queues and fleets. Use the AWS CLI to work directly with the service, or for developing your own tools that work with Deadline Cloud.

To create a farm and use the Deadline Cloud monitor, set up your account for Deadline Cloud. You only need to set up the Deadline Cloud monitor infrastructure once per account. From your farm, you can manage your project, including user access to your farm and its resources.

To create a farm without setting up the Deadline Cloud monitor infrastructure, set up a developer workstation for Deadline Cloud.

To create a farm with minimal resources to accept jobs, select **Quickstart** in the console home page. **Set up the Deadline Cloud monitor** walks you through those steps. These farms start with a queue and a fleet that are automatically associated. This approach is a convenient way to create sandbox style farms to experiment in.

Topics

Set up your AWS account

Set up your AWS account to use AWS Deadline Cloud.

If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account

1. Open https://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

When you sign up for an AWS account, an *AWS account root user* is created. The root user has access to all AWS services and resources in the account. As a security best practice, assign administrative access to a user, and use only the root user to perform tasks that require root user access.

When you first create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account.

Important

We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Set up the Deadline Cloud monitor

To get started, you'll need to create your Deadline Cloud monitor infrastructure and define your farm. You can also perform additional, optional steps including adding groups and users, choosing a service role, and adding tags to your resources.

Step 1: Create your monitor

The Deadline Cloud monitor uses AWS IAM Identity Center to authorize users. The IAM Identity Center instance that you use for Deadline Cloud must be in the same AWS Region as the monitor. If your console is using a different Region when you create the monitor, you'll get a reminder to change to the IAM Identity Center Region.

Your monitor's infrastructure consists of the following components:

- **Monitor display name**: The **Monitor display name** is how you can identify your monitor — for example *AnyCompany monitor*. Your monitor's name also determines your **monitor URL**.

> **Important**
>
> You can't change the **monitor display name** after you finish setting up.

- **Monitor URL**: You can access your monitor by using the **Monitor URL**. The URL is based on the **Monitor display name** — for example *https://anycompanymonitor.awsapps.com*.

> **Important**
>
> You can't change the **Monitor URL** after you finish setting up.

- **AWS Region**: The **AWS Region** is the physical location for a collection of AWS data centers. When you set up your monitor, the Region defaults to the closest location to you. We recommend changing the Region so it is located closest to your users. This reduces lag and improves data transfer speeds. AWS IAM Identity Center must be enabled in the same AWS Region as Deadline Cloud.

> **Important**
>
> You can't change your Region after you finish setting up Deadline Cloud.

Complete the tasks in this section to configure your monitor's infrastructure.

To configure your monitor's infrastructure

1. Sign in to the **AWS Management Console** to start the Welcome to Deadline Cloud setup, then choose **Next**.
2. Enter the **Monitor display name** — for example **`AnyCompany Monitor`**.
3. (Optional) To change the **Monitor name**, choose **Edit URL**.
4. (Optional) To change the **AWS Region** so it's closest to your users, choose **Change Region**.

    a. Select the Region closest to your users.

    b. Choose **Apply Region**.

    - (Optional) To add groups and users, select **(Optional) Add groups and users**.
    - (Optional) To further customize your monitor setup, select **Additional settings**.

5. If you are ready to **Step 2: Define farm details**, choose Next.

(Optional) Add groups and users

Before you complete Deadline Cloud monitor setup, you can add monitor users and add them to a group.

After setup is complete, you can create new users and groups, and manage users such as to assign them groups, permissions, and applications, or delete users from your monitor.

Additional settings

Deadline Cloud setup includes additional settings. With these settings, you can view all the changes Deadline Cloud setup makes to your AWS account, configure your monitor user role, and change your encryption key type.

AWS IAM Identity Center

AWS IAM Identity Center is a cloud-based single sign-on service for managing users and groups. IAM Identity Center can also be integrated with your enterprise single sign-on (SSO) provider so that users can sign in with their company account.

Deadline Cloud enables IAM Identity Center by default, and it is required to set up and use Deadline Cloud. The IAM Identity Center instance that you use for Deadline Cloud must be in the same AWS Region as the monitor. For more information, see What is AWS IAM Identity Center.

Configure service access role

An AWS service can assume a service role to perform actions on your behalf. Deadline Cloud requires a monitor user role for it to give users access to resources in your monitor.

You can attach AWS Identity and Access Management (IAM) managed policies to the monitor user role. The policies give users permissions to perform certain actions, such as creating jobs in a specific Deadline Cloud application. Because applications depend on specific conditions in the managed policy, if you don't use the managed policies, the application might not perform as expected.

You can change the monitor user role after you complete setup, at any time. For more information about user roles, see IAM Roles.

The following tabs contain instructions for two different use cases. To create and use a new service role, choose the **New service role** tab. To use an existing service role, choose the **Existing service role** tab.

New service role

To create and use a new service role

1. Select **Create and use a new service role**.
2. (Optional) Enter a **Service user role** name.
3. Choose **View permission details** for more information about the role.

Existing service role

To use an existing service role

1. Select **Use an existing service role**.
2. Open the dropdown list to choose an existing service role.
3. (Optional) Choose **View in IAM console** for more information about the role.

Step 2: Define farm details

Back on the Deadline Cloud console, complete the following steps to define the farm details.

1. In **Farm details**, add a **Name** for the farm.
2. For **Description**, enter the farm description. A clear description can help you quickly identify your farm's purpose.
3. (Optional) By default, your data is encrypted with a key that AWS owns and manages for your security. You can choose **Customize encryption settings (advanced)** to use an existing key or to create a new one that you manage.

   If you choose to customize encryption settings using the checkbox, enter a AWS KMS ARN, or create a new AWS KMS by choosing **Create new KMS key**.
4. (Optional) Choose **Add new tag** to add one or more tags to your farm.
5. Choose one of the following options:

   - Select **Skip to Review and Create** to review and create your farm.

- Select **Next** to proceed to additional, optional steps.

(Optional) Step 3: Define queue details

The queue is responsible for tracking progress and scheduling work for your jobs.

1. Starting in **Queue details,** provide a **Name** for the queue.
2. For **Description**, enter the queue description. A clear description can help you quickly identify your queue's purpose.
3. For **Job attachments**, you can either create a new Amazon S3 bucket or choose an existing Amazon S3 bucket. If you don't have an existing Amazon S3 bucket, you'll need to create one.

   a. To create a new Amazon S3 bucket, select **Create new job bucket**. You can define the name of the job bucket in the **Root prefix** field. We recommend calling the bucket **`deadlinecloud-job-attachments-[MONITORNAME]`**.

      You can only use lowercase letters and dashes. No spaces or special characters.

   b. To search for and select an existing Amazon S3 bucket, select **Choose from existing Amazon S3 bucket**. Then, search for an existing bucket by choosing **Browse S3**. When the list of your available Amazon S3 buckets display, select the Amazon S3 bucket you want to use for your queue.
4. If you are using customer-managed fleets, select **Enable association with customer-managed fleets**.

   - For customer-managed fleets, add a **Queue-configured user**, and then set the POSIX and/or Windows credentials. Alternatively, you can bypass the run-as functionality by selecting the checkbox.
5. Your queue requires permission to access Amazon S3 on your behalf. We recommend you create a new service role for every queue.

   a. For a new role, complete the following steps.

      i. Select **Create and use a new service role**.

      ii. Enter a **Role name** for your queue role or use the provided role name.

      iii. (Optional) Add a queue role **Description**.

iv. You can view the IAM permissions for the queue role by choosing **View permission details**.

b. Alternatively, you can select an existing service role.

6. (Optional) Add environment variables for the queue environment using name and value pairs.
7. (Optional) Add tags for the queue using key and value pairs.

After you enter all the queue details, choose **Next**.

(Optional) Step 4: Define fleet details

A fleet allocates workers to execute your rendering tasks. If you need a fleet for your rendering tasks, check the box for **Create fleet**.

1. **Fleet details**

   a. Provide both a **Name** and optional **Description** for your fleet.

   b. Select the way your compute resources should scale. The **Service-managed** option allows Deadline Cloud to auto scale your compute resources. The **Customer-managed** option leaves you in control of your own compute scaling.

2. In the **Instance option** section, choose either **Spot** or **On-demand**. Amazon EC2 On-demand instances provide faster availability and Amazon EC2 Spot instances are better for cost saving efforts.
3. For **Auto scaling** the number of instances in your fleet, choose both a **Minimum** number of instances and a **Maximum** number of instances.

   We strongly recommend to always set the minimum number of instances to **`0`** to avoid incurring extra costs.

4. Your fleet requires permission to write to CloudWatch on your behalf. We recommend you create a new service role for every fleet.

   a. For a new role, complete the following steps.

      i. Select **Create and use a new service role**.

      ii. Enter a **Role name** for your fleet role or use the provided role name.

      iii. (Optional) Add a fleet role **Description**.

      iv. To view the IAM permissions for the fleet role, choose **View permission details**.

b. Alternatively, you can use an existing service role.

5. (Optional) Add tags for the fleet using key and value pairs.

After you enter all the fleet details, choose **Next**.

(Optional) Step 5: Configure worker capabilities

Define the capabilities for your worker instances.

1. Choose the operating system for workers in your fleet. For this tutorial, leave the default, **Linux**.
2. Review the CPU architecture setting for awareness.
3. Update the minimum and maximum number of vCPUs for your hardware capabilities.
4. Update the minimum and maximum number of memory (GiB) for your hardware capabilities.
5. You can filter instance types by either allowing or excluding types of worker instances. In both filtering options, you can filter up to 10 Amazon EC2 instance types.
6. Under **Additional capabilities (Optional)**, you can define the root EBS volume by **Size (GiB)**, **IOPS**, and **Throughput (MiB/s)**.
7. After all worker capabilities are set, choose **Next** to define the access level of your groups.

(Optional) Step 6: Define access levels

If you have groups connected to your monitor, you can define their access level. Permission to use Deadline Cloud features is managed by access levels. You can assign different access levels to groups of users.

1. Use the **Deadline Cloud farm access level** menu to select the level of permission for the group.
2. Choose **Next** to continue and review all farm details entered.

Step 7: Review and create

Review all of the information entered to create your farm. When you're ready, choose **Create farm**.

The progress of your farm's creation is displayed on the **Farms** page. A success message displays when your farm is ready for use.

Set up Deadline Cloud submitters

This process is for administrators and artists who want to install, set up, and launch the AWS Deadline Cloud submitter. A Deadline Cloud *submitter* is a digital content creation (DCC) plugin. Artists use it to submit jobs from a third-party DCC interface that they're familiar with.

> **Note**
>
> This process must be completed on all workstations that artists will use for submitting renders.

Each workstation must have the DCC installed before installing the corresponding submitter. For example, if you want to download the Deadline Cloud submitter for Blender, you need to have Blender already installed on your workstation.

Topics

Step 1: Install the Deadline Cloud submitter

The following sections guide you through the steps to install the Deadline Cloud submitter.

Download the submitter installer

Before you can install the Deadline Cloud submitter, you must download the submitter installer. Currently, the Deadline Cloud submitter installer only supports Windows and Linux.

1. Sign in to the AWS Management Console and open the Deadline Cloud console.
2. From the side navigation pane, choose **Downloads**.
3. Locate the **Deadline Cloud submitter installer** section.
4. Select the **installer** for your computer's operating system, and then choose **Download**.

(Optional) Verify the authenticity of the downloaded software

To verify that the software you downloaded is authentic, use the following procedure for either Windows or Linux. You might want to do this to ensure no one has tampered with the files during or after the download process.

You can use these instructions to first verify the installer, and then verify the Deadline Cloud monitor after you download it in Step 2: Install and set up Deadline Cloud monitor.

Windows

To verify the authenticity of your downloaded files, complete the following steps.

1. In the following command, replace *file* with the file that you want to verify. For example, ***C:\PATH\TO\MY*****DeadlineCloudSubmitter-windows-x64-installer.exe** . Also, replace *signtool-sdk-version* with the version of the SignTool SDK installed. For example, **10.0.22000.0**.

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

2. For example, you can verify the Deadline Cloud submitter installer file by running the following command:

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

Linux

To verify the authenticity of your downloaded files, use the gpg command line tool.

1. Import the OpenPGP key by running the following command:

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

   mQINBGX6GQsBEADduUtJgqSXI+q76O6fsFwEYKmbnlyL0xKvlq32EZuyv0otZo5L
   le4m5Gg52AzrvPvDiUTLooAlvYeozaYyirIGsK08Ydz0Ftdjroiuh/mw9JSJDJRI
   rnRn5yKet1JFezkjopA3pjsTBP6lW/mb1bDBDEwwwtH0x9lV7A03FJ9T7Uzu/qSh
   qO/UYdkafro3cPASvkqgDt2tCvURfBcUCAjZVFcLZcVD5iwXacxvKsxxS/e7kuVV
   ```

```
I1+VGT8Hj8XzWYhjCZxOLZk/fvpYPMyEEujN0fYUp6RtMIXve0C9awwMCy5nBG2J
eE2Ol5DsCpTaBd4Fdr3LWcSs8JFA/YfP9auL3NczOozPoVJt+fw8CBlVIXO0J7l5
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. Determine whether to trust the OpenPGP key. Some factors to consider when deciding whether to trust the above key include the following:

   - The internet connection you've used to obtain the GPG key from this website is secure.
   - The device that you are accessing this website on is secure.
   - AWS has taken measures to secure the hosting of the OpenPGP public key on this website.

3. If you decide to trust the OpenPGP key, edit the key to trust with gpg similar to the following example:

```
$ 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. **Verify the Deadline Cloud submitter installer**

   To verify the Deadline Cloud submitter installer, complete the following steps:

   a. Return to the Deadline Cloud console **Downloads** page and download the signature file for the Deadline Cloud submitter installer.

   b. Verify the signature of the Deadline Cloud submitter installer by running:

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

5. **Verify the Deadline Cloud monitor**

> **Note**
>
> You can verify the Deadline Cloud monitor download using signature files or platform specific methods. For platform specific methods, see the Linux (Debian) tab, the Linux (RPM) tab, or the Linux (AppImage) tab based on your downloaded file type.

To verify the Deadline Cloud monitor desktop application with signature files, complete the following steps:

a. Return to the Deadline Cloud console **Downloads** page and download the corresponding .sig file, and then run

For .deb:

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

For .rpm:

```
gpg --verify ./deadline-cloud-monitor_<APP_VERSION>_x86_64.deb.sig ./
deadline-cloud-monitor_<APP_VERSION>_x86_64.rpm
```

For .AppImage:

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

b. Confirm that the output looks similar to the following:

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

gpg: using RSA key B840C08C29A90796A071FAA5F6CD3CE6B7
```

If the output contains the phrase `Good signature from "AWS Deadline Cloud"`, it means that the signature has successfully been verified and you can run the Deadline Cloud monitor installation script.

Linux (AppImage)

To verify packages that use a Linux .AppImage binary, first complete steps 1-3 in the Linux tab, then complete the following steps.

1. From the AppImageUpdate page in GitHub, download the **validate-x86_64.AppImage** file.
2. After downloading the file, to add execute permissions, run the following command.

```
chmod a+x ./validate-x86_64.AppImage
```

3. To add execute permissions, run the following command.

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

4. To verify the Deadline Cloud monitor signature, run the following command.

```
./validate-x86_64.AppImage ./deadline-cloud-monitor_<APP_VERSION>_amd64.AppImage
```

If the output contains the phrase `Validation successful`, it means that the signature has successfully been verified and you can safely run the Deadline Cloud monitor installation script.

Linux (Debian)

To verify packages that use a Linux .deb binary, first complete steps 1-3 in the Linux tab.

dpkg is the core package management tool in most debian based Linux distributions. You can verify the .deb file with the tool.

1. From the Deadline Cloud console **Downloads** page, download the Deadline Cloud monitor .deb file.
2. Replace *<APP_VERSION>* with the version of the .deb file you want to verify.

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

3. The output will be similar to:

```
ProcessingLinux deadline-cloud-monitor_<APP_VERSION>_amd64.deb...
GOODSIG _gpgbuilder B840C08C29A90796A071FAA5F6CD3C 171200
```

4. To verify the .deb file, confirm that `GOODSIG` is present in the output.

Linux (RPM)

To verify packages that use a Linux .rpm binary, first complete steps 1-3 in the Linux tab.

1. From the Deadline Cloud console **Downloads** page, download the Deadline Cloud monitor .rpm file.
2. Replace *<APP_VERSION>* with the version of the .rpm file to verify.

```
gpg --export --armor "Deadline Cloud" > key.pub
sudo rpm --import key.pub
rpm -K deadline-cloud-monitor-<APP_VERSION>-1.x86_64.rpm
```

3. The output will be similar to:

```
deadline-cloud-monitor-deadline-cloud-
monitor-<APP_VERSION>-1.x86_64.rpm-1.x86_64.rpm: digests signatures OK
```

4. To verify the .rpm file, confirm that `digests signatures OK` is in the output.

Install the Deadline Cloud submitter

You can install a Deadline Cloud submitter with Windows or Linux. With the installer, you can install the following submitters:

Software	Supported versions	Windows installer	Linux installer
Autodesk Arnold for Maya	7.1, 7.2	Included	Included

Software	Supported versions	Windows installer	Linux installer
Autodesk Maya	2023, 2024, 2025	Included	Included
Blender	3.6, 4.2	Included	Included
KeyShot Studio	2023, 2024	Included	Not included
Maxon Cinema 4D	2024, 2025	Included	Not included
Nuke	15	Included	Included
SideFX Houdini	19.5, 20, 20.5	Included	Included
Unreal Engine	5.2, 5.3, 5.4	Included	Not included

You can install other submitters not listed here. We use Deadline Cloud libraries to build submitters. Some of the submitters include After Effects, 3ds Max, and Rhino. You can find the source code for these libraries and submitters in the aws-deadline GitHub organization.

Windows

1. In a file browser, navigate to the folder where the installer downloaded, and then select `DeadlineCloudSubmitter-windows-x64-installer.exe`.
    a. If a **Windows protected your PC** pop-up displays, choose **More info**.
    b. Choose **Run anyway**.
2. After the AWS Deadline Cloud Submitter Setup Wizard opens, choose **Next**.
3. Choose the installation scope by completing one of the following steps:
    - To install for only the current user, choose **User**.
    - To install for all users, choose **System**.

    If you choose **System**, you must exit the installer and re-run it as an administrator by completing the following steps:

    a. Right-click on **`DeadlineCloudSubmitter-windows-x64-installer.exe`**, and then choose **Run as administrator**.
    b. Enter your administrator credentials, and then choose **Yes**.

c. Choose **System** for the installation scope.

4. After selecting the installation scope, choose **Next**.
5. Choose **Next** again to accept the installation directory.
6. Select **Integrated submitter for Nuke**, or whichever submitter you want to install.
7. Choose **Next**.
8. Review the installation, and choose **Next**.
9. Choose **Next** again, and then choose **Finish**.

Linux

> **Note**
>
> The Deadline Cloud integrated Nuke installer for Linux and Deadline Cloud monitor can only be installed on Linux distributions with at least GLIBC 2.31.

1. Open a terminal window.
2. To do a system install of the installer, enter the command **`sudo -i`** and press **Enter** to become root.
3. Navigate to the location where you downloaded the installer.

   For example, **`cd /home/USER/Downloads`**.
4. To make the installer executable, enter **`chmod +x DeadlineCloudSubmitter-linux-x64-installer.run`**.
5. To run the Deadline Cloud submitter installer, enter **`./DeadlineCloudSubmitter-linux-x64-installer.run`**.
6. When the installer opens, follow the prompts on your screen to complete the Setup Wizard.

Step 2: Install and set up Deadline Cloud monitor

You can install the Deadline Cloud monitor desktop application with Windows or Linux.

Windows

1. If you haven't already, sign in to the AWS Management Console and open the Deadline Cloud console.
2. From the left navigation pane, choose monitor **Downloads**.
3. In the **Deadline Cloud monitor** section, select the file for your computer's operating system.
4. To download the Deadline Cloud monitor, choose **Download**.

To perform a silent install, use the following command:

```
DeadlineCloudMonitor_VERSION_x64-setup.exe /S
```

By default the monitor is installed in `C:\Users{username}\AppData\Local\DeadlineCloudMonitor`. To change the installation directory, use this command instead:

```
DeadlineCloudMonitor_VERSION_x64-setup.exe /S /D={InstallDirectory}
```

Linux (AppImage)

To install Deadline Cloud monitor AppImage on Debian distros

1. Download the latest Deadline Cloud monitor AppImage.
2. **Note**

   This step is for Ubuntu 22 and up. For other versions of Ubuntu, skip this step.

   To install libfuse2, enter:

   ```
   sudo apt udpate
   sudo apt install libfuse2
   ```

3. To make the AppImage executable, enter:

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

Linux (Debian)

To install Deadline Cloud monitor Debian package on Debian distros

1. Download the latest Deadline Cloud monitor Debian package.
2. **Note**

   This step is for Ubuntu 22 and up. For other versions of Ubuntu, skip this step.

   To install libssl1.1, enter:

   ```
   wget http://archive.ubuntu.com/ubuntu/pool/main/o/openssl/
   libssl1.1_1.1.1f-1ubuntu2_amd64.deb
   sudo apt install ./libssl1.1_1.1.1f-1ubuntu2_amd64.deb
   ```
3. To install the Deadline Cloud monitor Debian package, enter:

   ```
   sudo apt update
   sudo apt install ./deadline-cloud-monitor_<APP_VERSION>_amd64.deb
   ```
4. If the install fails on packages that have unmet dependencies, fix the broken packages and then run the following commands.

   ```
   sudo apt --fix-missing update
   sudo apt update
   sudo apt install -f
   ```

Linux (RPM)

To install Deadline Cloud monitor RPM on Rocky Linux 9 or Alma Linux 9

1. Download the latest Deadline Cloud monitor RPM.
2. Add the extra packages for the Enterprise Linux 9 repository:

   ```
   sudo dnf install epel-release
   ```
3. Install compat-openssl11 for the libssl.so.1.1 dependency:

```
sudo dnf install compat-openssl11 deadline-cloud-monitor-<VERSION>-1.x86_64.rpm
```

To install Deadline Cloud monitor RPM on Red Hat Linux 9

1. Download the latest Deadline Cloud monitor RPM.
2. Enable the CodeReady Linux Builder repository:

```
subscription-manager repos --enable codeready-builder-for-rhel-9-x86_64-rpms
```

3. Install the extra packages for Enterprise RPM:

```
sudo dnf install https://dl.fedoraproject.org/pub/epel/epel-release-
latest-9.noarch.rpm
```

4. Install compat-openssl11 for the libssl.so.1.1 dependency:

```
sudo dnf install compat-openssl11 deadline-cloud-monitor-<VERSION>-1.x86_64.rpm
```

To install Deadline Cloud monitor RPM on Rocky Linux 8, Alma Linux 8, or Red Hat Linux 8

1. Download the latest Deadline Cloud monitor RPM.
2. Install the Deadline Cloud monitor:

```
sudo dnf install deadline-cloud-monitor-<VERSION>-1.x86_64.rpm
```

After you complete the download, you can verify the authenticity of the downloaded software. You might want to do this to ensure no one has tampered with the files during or after the download process. See Verify authenticity of downloaded software in Step 1.

After downloading Deadline Cloud monitor and verifying the authenticity, use the following procedure to set up the Deadline Cloud monitor.

To set up Deadline Cloud monitor

1. Open **Deadline Cloud monitor**.
2. When prompted to create a new profile, complete the following steps.

a. Enter your monitor URL into the URL input, which looks like **`https://MY-MONITOR.deadlinecloud.amazonaws.com/`**

b. Enter a **Profile** name.

c. Choose **Create Profile**.

Your profile is created and your credentials are now shared with any software that uses the profile name that you created.

3. After you create the Deadline Cloud monitor profile, you can't change the profile name or the studio URL. If you need to make changes, do the following instead:

a. Delete the profile. In the left navigation pane, choose **Deadline Cloud monitor** > **Settings** > **Delete**.

b. Create a new profile with the changes that you want.

4. From the left navigation pane, use the **>Deadline Cloud monitor** option to do the following:

- Change the Deadline Cloud monitor profile to log in to a different monitor.
- Enable **Autologin** so you don’t have to enter your monitor URL on subsequent opens of Deadline Cloud monitor.

5. Close the Deadline Cloud monitor window. It continues to run in the background and sync your credentials every 15 minutes.

6. For each digital content creation (DCC) application that you plan to use for your rendering projects, complete the following steps:

a. From your Deadline Cloud submitter, open the Deadline Cloud workstation configuration.

b. In the workstation configuration, select the profile that you created in the Deadline Cloud monitor. Your Deadline Cloud credentials are now shared with this DCC and your tools should work as expected.

Step 3: Launch the Deadline Cloud submitter

The following sections guide you through the steps to launch the Deadline Cloud submitter plugin in Blender, Cinema 4D, Nuke, Maya, Houdini, KeyShot, and Unreal Engine.

To launch the Deadline Cloud submitter in Blender

> **Note**
>
> Support for Blender is provided using the Conda environment for service-managed fleets. For more information, see Default Conda queue environment.

1. Open **Blender**.
2. Choose **Edit**, then **Preferences**. Under **File Paths** choose **Script Directories**, then choose **Add**. Add a script directory for the python folder where the Blender submitter was installed:

```
Windows:
   %USERPROFILE%\DeadlineCloudSubmitter\Submitters\Blender\python\
Linux:
   ~/DeadlineCloudSubmitter/Submitters/Blender/python/
```

3. Restart Blender.
4. Choose **Edit**, then **Preferences**. Next, choose **Add-ons**, then search for **Deadline Cloud for Blender**. Select the checkbox to enable the add-on.
5. Open a Blender scene with dependencies that exist within the asset root directory.
6. In the **Render** menu, select the Deadline Cloud dialog.
   a. If you are not already authenticated in the Deadline Cloud submitter, the **Credentials Status** shows as **NEEDS_LOGIN**.
   b. Choose **Login**.
   c. A login browser window displays. Log in with your user credentials.
   d. Choose **Allow**. You are now logged in and the **Credentials Status** shows as **AUTHENTICATED**.
7. Choose **Submit**.

To launch the Deadline Cloud submitter in Cinema 4D

> **Note**
>
> Support for Cinema 4D is provided using the Conda environment for service-managed fleets. For more information, see Default Conda queue environment.

1. Open Cinema 4D.
2. If prompted to install GUI components for AWS Deadline Cloud, complete the following steps:
   a. When the prompt displays, choose **Yes**, and wait for dependencies to install.
   b. Restart Cinema 4D to ensure the changes are applied.
3. Choose **Extensions** > **AWS Deadline Cloud Submitter**.

To launch the Deadline Cloud submitter in Foundry Nuke

> **Note**
>
> Support for Nuke is provided using the Conda environment for service-managed fleets. For more information, see Default Conda queue environment.

1. Open **Nuke**.
2. Open a Nuke script with dependencies that exist within the asset root directory.
3. Choose **AWS Deadline**, and then choose **Submit to Deadline Cloud** to launch the submitter.
   a. If you are not already authenticated in the Deadline Cloud submitter, the **Credentials Status** shows as **NEEDS_LOGIN**.
   b. Choose **Login**.
   c. In the login browser window, log in with your user credentials.
   d. Choose **Allow**. You are now logged in and the **Credentials Status** shows as **AUTHENTICATED**.
4. Choose **Submit**.

To launch the Deadline Cloud submitter in Maya

> **Note**
>
> Support for Maya and Arnold for Maya (MtoA) is provided using the Conda environment for service-managed fleets. For more information, see Default Conda queue environment.

1. Open **Maya**.
2. Set your project, and open a file that exists within the asset root directory.
3. Choose **Windows** → **Settings/Preferences** → **Plugin Manager**.
4. Search for **DeadlineCloudSubmitter**.
5. To load the Deadline Cloud submitter plugin, select **Loaded**.
   a. If you are not already authenticated in the Deadline Cloud submitter, the **Credentials Status** shows as **NEEDS_LOGIN**.
   b. Choose **Login**.
   c. A login browser window displays. Log in with your user credentials.
   d. Choose **Allow**. You are now logged in and the **Credentials Status** shows as **AUTHENTICATED**.
6. (Optional) To load the Deadline Cloud submitter plugin every time you open Maya, choose **Auto-load**.
7. Select the Deadline Cloud shelf, then select the green button to launch the submitter.

To launch the Deadline Cloud submitter in Houdini

> **Note**
>
> Support for Houdini is provided using the Conda environment for service-managed fleets. For more information, see Default Conda queue environment.

1. Open **Houdini**.
2. In the **Network Editor**, select the **/out** network.
3. Press **tab**, and enter **`deadline`**.

4. Select the Deadline Cloud option, and connect it to your existing network.
5. Double-click the **Deadline Cloud node**.

To launch the Deadline Cloud submitter in KeyShot

1. Open KeyShot.
2. Choose **Windows** > **Scripting console** > **Submit to AWS Deadline Cloud** and **Run**.

To launch the Deadline Cloud submitter in Unreal Engine

1. Create or open the folder you use for your Unreal Engine projects.
2. Open command line and run the following commands:
   - **`git clone https://github.com/aws-deadline/deadline-cloud-for-unreal-engine`**
   - **`cd deadline-cloud-for-unreal/test_projects`**
   - **`git lfs fetch –all`**
3. To download the plugin for Unreal Engine, open the Unreal Engine project folder, and launch **deadline-cloud-forunreal/test_projects/pull_ue_plugin.bat**.

   This puts the plugin files in **C:/LocalProjects/UnrealDeadlineCloudTest/Plugins/UnrealDeadlineCloudService**.
4. To download the submitter, open the UnrealDeadlineCloudService folder, and run **`deadline-cloud-forunreal/ test_projects/Plugins/UnrealDeadlineCloudService/install_unreal_submitter.bat`**.
5. To launch the submitter from Unreal Engine, complete the following steps:
   a. Choose **Edit** > **Project settings**.
   b. In the search bar, enter **`movie render pipeline`**.
   c. Adjust the following Movie Render Pipeline settings:
      i. For **Default Remote Executor**, enter **`MoviePipelineDeadlineCloudRemoteExecutor`**.
      ii. For **Default Executor Job**, enter **`MoviePipelineDeadlineCloudExecutorJob`**.

iii. For **Default Job Settings Classes**, choose the plus sign, and then enter **`DeadlineCloudRenderStepSetting`**.

With these settings, you can choose the Deadline Cloud plugin from Unreal Engine.

Using the Deadline Cloud monitor

The AWS Deadline Cloud monitor provides you with an overall view of your visual compute jobs. You can use it to monitor and manage jobs, view worker activity on fleets, track budgets and usage, and to download a job's results.

Each queue has a job monitor that shows you the status of jobs, steps, and tasks. The monitor provides ways to manage jobs directly from the monitor. You can make prioritization changes, cancel jobs, requeue jobs, and resubmit jobs.

The Deadline Cloud monitor has a table that shows summary status for a job, or you can select a job to see detailed task logs that help troubleshoot issues with a job.

You can use the Deadline Cloud monitor to download the results to the location on your workstation that was specified when the job was created.

The Deadline Cloud monitor also helps you monitor usage and manage costs. For more information, see Track spending and usage for Deadline Cloud farms.

Topics

Share the Deadline Cloud monitor URL

When you set up the Deadline Cloud service, by default you create a URL that opens the Deadline Cloud monitor for your account. Use this URL to open the monitor in your browser or on your desktop. Share the URL with other users so that they can access the Deadline Cloud monitor.

Before a user can open the Deadline Cloud monitor, you must grant the user access. To grant access, either add the user to the list of authorized users for the monitor or add them to a group with access to the monitor. For more information, see Managing users in Deadline Cloud.

To share the monitor URL

1. Open the Deadline Cloud console.
2. From **Get started**, choose **Go to Deadline Cloud dashboard**.
3. On the navigation pane, choose **Dashboard**.
4. In the **Account overview** section, choose **Account details**.
5. Copy and then securely send the **URL** to anyone who needs to access the Deadline Cloud monitor.

Open the Deadline Cloud monitor

You can open the Deadline Cloud monitor by any of the following ways:

- **Console** – Sign in to the AWS Management Console and open the Deadline Cloud console.
- **Web** – Go to the monitor URL that you created when you set up Deadline Cloud.
- **Monitor** – Use the desktop Deadline Cloud monitor.

When you use the console, you must be able to sign in to AWS using an AWS Identity and Access Management identity, and then sign in to the monitor with AWS IAM Identity Center credentials. If you only have IAM Identity Center credentials, you must sign in using the monitor URL or the desktop application.

To open the Deadline Cloud monitor (web)

1. Using a browser, open the monitor URL that you created when you set up Deadline Cloud.
2. Sign in with your user credentials.

To open the Deadline Cloud monitor (console)

1. Open the Deadline Cloud console.
2. In the navigation pane, select **Farms**.
3. Select a farm, then choose **Manage jobs** to open the **Deadline Cloud monitor** page.

4. Sign in with your user credentials.

To open the Deadline Cloud monitor (desktop)

1. Open the Deadline Cloud console.

   -or-

   Open the Deadline Cloud monitor - web from the monitor URL.

2. - On the Deadline Cloud console, do the following:
     1. In the monitor, choose **Go to Deadline Cloud dashboard**, and then choose **Downloads** from the left menu.
     2. From **Deadline Cloud monitor**, choose the monitor version for your desktop.
     3. Choose **Download**.
   - On the Deadline Cloud monitor - web, do the following:
     - From the left menu, choose **Workstation setup**. If the **Workstation setup** item isn't visible, use the arrow to open the left menu.
     - Choose **Download**.
     - From **Select an OS**, choose your operating system.
3. Download the Deadline Cloud monitor - desktop.
4. After you download and install the monitor, open it on your computer.

   - If this is your first time opening the Deadline Cloud monitor, you must provide the monitor URL and create a profile name. Next you sign in to the monitor with your Deadline Cloud credentials.
   - After you create a profile, you open the monitor by selecting a profile. You might need to enter your Deadline Cloud credentials.

View queue and fleet details in Deadline Cloud

You can use the Deadline Cloud monitor to view the configuration of the queues and fleets in your farm. You can also use the monitor to see a list of the jobs in a queue or the workers in a fleet.

You must have `VIEWING` permission to view queue and fleet details. If the details don't display, contact your administrator to get the correct permissions.

To view queue details

1. Open the Deadline Cloud monitor.
2. From the list of farms, choose the farm that contains the queue that you're interested in.
3. In the list of queues, choose a queue to display its details. To compare the configuration of two or more queues, select more than one check box.
4. To see a list of jobs in the queue, choose the queue name from the list of queues or from the details panel.

If the monitor is already open, you can select the queue from the **Queues** list in the left navigation pane.

To view fleet details

1. Open the Deadline Cloud monitor.
2. From the list of farms, choose the farm that contains the fleet that you're interested in.
3. In **Farm resources**, choose **Fleets**.
4. In the list of fleets, choose a fleet to display its details. To compare the configuration of two or more fleets, select more than one check box.
5. To see a list of workers in the fleet, choose the fleet name from the list of fleets or from the details panel.

If the monitor is already open, you can select the fleet from the **Fleets** list in the left navigation pane.

Manage jobs, steps, and tasks in Deadline Cloud

When you select a queue, the job monitor section of the Deadline Cloud monitor shows you the jobs in that queue, the steps in the job, and the tasks in each step. When you select a job, step, or task, you can use the **Actions** menu to manage each.

To open the job monitor, follow the steps to view a queue in View queue and fleet details in Deadline Cloud, then select the job, step, or task to work with.

For jobs, steps, and tasks, you can do the following:

- Change the status to **Requeued**, **Succeeded**, **Failed**, or **Canceled**.

- Download the processed output from the job, step, or task.
- Copy the ID of the job, step, or task.

For the selected job, you can:

- Archive the job.
- Modify the job properties, such as changing prioritization or viewing step to step dependencies.
- View additional details using the job's parameters.
- Resubmit the job.

For for more information, see View and manage job details in Deadline Cloud.

For each step, you can:

- View the dependencies for the step. The dependencies for a step must be completed before the step runs.

For details, see View a step in Deadline Cloud.

For each task, you can:

- View logs for the task.
- View task parameters.

For more information, see View a task in Deadline Cloud.

View and manage job details in Deadline Cloud

The **Job monitor** page in the Deadline Cloud monitor provides you with the following:

- An overall view of the progress of a job.
- A view of the steps and tasks that make up the job.

Choose a job from the list to view a list of steps for the job, and then choose a step from the list of steps to view the tasks for the job. After you choose an item, you can use the **Actions** menu for that item to view details.

To view job details

1. Follow the steps to view a queue in View queue and fleet details in Deadline Cloud.
2. In the navigation pane, select the queue where you submitted your job.
3. Select a job using one of the following methods:

    a. From the **Jobs** list, select a job to view its details.

    b. From the **search** field, enter any text associated with the job, such as the job name or user that created the job. From the results that display, select the job you want to view.

The details of a job include the steps in the job and the tasks in each step. You can use the **Actions** menu to do the following:

- Change the status of the job.
- View and modify the properties of a job.
    - You can view the dependencies between steps in the job.
    - You can change the priority of the job in a queue. Jobs with higher number priority are processed before jobs with lower number priority. Jobs can have a priority between 1 and 100. When two jobs have the same priority, the oldest job is scheduled first.
- View the parameters for the job that were set when the job was submitted.
- Download the output of a job. When you download the output of a job, it contains all of the output generated by the steps and tasks in the job.

Archive a job

To archive a job, it must be in a terminal state, `FAILED`, `SUCCEEDED`, `SUSPENDED`, or `CANCELED`. The `ARCHIVED` state is final. After a job is archived, it can't be requeued or modified.

The job's data is not affected by archiving the job. The data is deleted when the inactivity timeout is reached, or when the queue containing the job is deleted.

Other things that happen to archived jobs:

- Archived jobs are hidden in the Deadline Cloud monitor.
- Archived jobs are visible in a read-only state form the Deadline Cloud CLI for 120 days before deletion.

Requeue a job

When you requeue a job, all of the tasks without step dependencies switch to READY. The status of steps with dependencies switch to READY or PENDING as they are restored.

- All jobs, steps, and tasks switch to PENDING.
- If a step doesn't have a dependency, it switches to READY.

Resubmit a job

There might be times when you want to run a job again, but with different properties and settings. For example, you might submit a job to render a subset of testing frames, verify the output, then run the job again with the full frame range. To do this, resubmit the job.

When you resubmit a job, new tasks without dependencies become READY. New tasks with dependencies become PENDING.

- All new jobs, steps, and tasks become PENDING.
- If a new step doesn't have a dependency, it becomes READY.

When you resubmit a job, you can only change properties that were defined as configurable when the job was first created. For example, if the name of a job is not defined as a configurable property of the job when first submitted, then the name cannot be edited on resubmission.

View a step in Deadline Cloud

Use the AWS Deadline Cloud monitor to view the steps in your processing jobs. In the **Job monitor**, the **Steps** list shows the list of steps that make up the selected job. When you select a step, the **Tasks** list shows the tasks in the step.

To view a step

1. Follow the steps in View and manage job details in Deadline Cloud to view a list of jobs.
2. Select a job from the **Jobs** list.
3. Select a step from the **Steps** list.

You can use the **Actions** menu to do the following:

- Change the status of the step.
- Download the output of the step. When you download the output of a step, it contains all of the output generated by the tasks in the step.
- View the dependencies of a step. The dependencies table shows a list of steps that must be complete before the selected step starts, and a list of steps that are waiting for this step to complete.

View a task in Deadline Cloud

Use the AWS Deadline Cloud monitor to view the tasks in your processing jobs. In the **Job monitor**, the **Tasks** list shows the tasks that make up the step selected in the **Steps** list.

To view a task

1. Follow the steps in View and manage job details in Deadline Cloud to view a list of jobs.
2. Select a job from the **Jobs** list.
3. Select a step from the **Steps** list.
4. Select a task from the **Tasks** list.

You can use the **Actions** menu to do the following:

- Change the status of the task.
- View task logs. For more information, see View logs in Deadline Cloud.
- View that parameters that were set when the task was created.
- Download the output of the task. When you download the output of a task, it only contains the output generated by the selected task.

View logs in Deadline Cloud

Logs provide you with detailed information about the status and processing of tasks. In the AWS Deadline Cloud monitor, you can see the following two types of logs:

- *Session logs* detail the timeline of actions, including:
  - Setup actions, such as attachment syncing and loading the software environment
  - Running a task or set of tasks

- Closure actions, such as shutting down the environment on a worker

  A session includes processing of at least one task, and can include multiple tasks. Session logs also show information about Amazon Elastic Compute Cloud (Amazon EC2) instance type, vCPU, and memory. Session logs also include a link to the log for the worker used in the session.
- *Worker logs* provide details for the timeline of actions that a worker processes during its lifecycle. Worker logs can contain information about multiple sessions.

You can download session and worker logs so that you can examine them offline.

To view session logs

1. Follow the steps in View and manage job details in Deadline Cloud to view a list of jobs.
2. Select a job from the **Jobs** list.
3. Select a step from the **Steps** list.
4. Select a task from the **Tasks** list.
5. From the **Actions** menu, choose **View logs**.

The **Timelines** section shows a summary of the actions for the task. To see more tasks run in the session and to see the shutdown actions for the session, choose **View logs for all tasks**.

To view worker logs from a task

1. Follow the steps in View and manage job details in Deadline Cloud to view a list of jobs.
2. Select a job from the **Jobs** list.
3. Select a step from the **Steps** list.
4. Select a task from the **Tasks** list.
5. From the **Actions** menu, choose **View logs**.
6. Choose **Session info**.
7. Choose **View worker log**.

To view worker logs from fleet details

1. Follow the steps in View queue and fleet details in Deadline Cloud to view a fleet.
2. Select a **Worker ID** from the **Workers** list.

3. From the **Actions** menu, choose **View worker logs**.

Download finished output in Deadline Cloud

After a job is finished, you can use the AWS Deadline Cloud monitor to download the results to your workstation. The output file is stored with the name and location that you specified when you created the job.

Output files are stored indefinitely. To reduce storage costs, consider creating an S3 Lifecycle configuration for your queue's Amazon S3 bucket. For more information, see Managing your storage lifecycle in the *Amazon Simple Storage Service User Guide*.

To download the finished output of a job, step, or task

1. Follow the steps in View and manage job details in Deadline Cloud to view a list of jobs.
2. Select the job, step, or task that you want to download the output for.
   - If you select a job, you can download all of the output for all of the tasks in all of the steps for that job.
   - If you select a step, you can download all of the output for all of the tasks in that step.
   - If you select a task, you can download the output for that individual task.
3. From the **Actions** menu, choose **Download output**.
4. The output will be downloaded to the location set when the job was submitted.

> **Note**
>
> Downloading output using the menu is currently only supported for Windows and Linux. If you have a Mac and you choose the **Download output** menu item, a window shows the AWS CLI command that you can use to download the rendered output.

Deadline Cloud farms

With a Deadline Cloud farm, you can manage users and project resources. A *farm* is a where your project resources are located. Your farm consists of queues and fleets. A *queue* is where submitted jobs are located and scheduled to be rendered. A *fleet* is a group of worker nodes that run tasks to complete jobs. After you create a farm, you can create queues and fleets to meet your project's needs.

Create a farm

1. From the Deadline Cloud console, choose **Go to Dashboard**.
2. In the Farms section of the Deadline Cloud dashboard, choose **Actions** → **Create farm**.
   - Alternatively, in the left side panel choose **Farms and other resources**, then choose **Create Farm**.
3. Add a **Name** for your farm.
4. For **Description**, enter the farm description. A clear description can help you quickly identify your farm's purpose.
5. (Optional) By default, your data is encrypted with a key that AWS owns and manages for your security. You can choose **Customize encryption settings (advanced)** to use an existing key or to create a new one that you manage.

   If you choose to customize encryption settings using the checkbox, enter a AWS KMS ARN, or create a new AWS KMS by choosing **Create new KMS key**.
6. (Optional) Choose **Add new tag** to add one or more tags to your farm.
7. Choose **Create farm**. After creation, your farm displays.

Deadline Cloud queues

A queue is a farm resource that manages and processes jobs.

To work with queues, you should already have a monitor and farm set up.

Topics

Create a queue

1. From the Deadline Cloud console dashboard, select the farm that you want to create a queue for.
   - Alternatively, in the left side panel choose **Farms and other resources**, then select the farm you want to create a queue for.
2. In the **Queues** tab, choose **Create queue**.
3. Enter a name for your queue.
4. For **Description**, enter the queue description. A description helps you identify your queue's purpose.
5. For **Job attachments**, you can either create a new Amazon S3 bucket or choose an existing Amazon S3 bucket.
   a. To create a new Amazon S3 bucket
      i. Select **Create new job bucket**.
      ii. Enter a name for the bucket. We recommend naming the bucket `deadlinecloud-job-attachments-[MONITORNAME]`.
      iii. Enter a **Root prefix** to define or change your queue's root location.
   b. To choose an existing Amazon S3 bucket
      i. Select **Choose an existing S3 bucket** > **Browse S3**.
      ii. Select the S3 bucket for your queue from the list of available buckets.

6. (Optional) To associate your queue with a customer-managed fleet, select **Enable association with customer-managed fleets**.
7. If you enable association with customer-managed fleets, you must complete the following steps.

   > **Important**
   >
   > We strongly recommend specifying users and groups for run-as functionality. If you don't, it will degrade your farm's security posture because the jobs can then do everything the worker's agent can do. For more information about the potential security risks, see Run jobs as users and groups.

   a. For Run as user:

      To provide credentials for the queue's jobs, select **Queue-configured user**.

      Or, to opt out of setting your own credentials and run jobs as the worker agent user, select **Worker agent user**.

   b. (Optional) For Run as user credentials, enter a user name and group name to provide credentials for the queue's jobs.

      If you are using a Windows fleet, you must create an AWS Secrets Manager secret that contains the password for the Run as user. If you don't have an existing secret with the password, choose **Create secret** to open the Secrets Manager console to create a secret.
8. Requiring a budget helps manage costs for your queue. Select either **Don't require a budget** or **Require a budget**.
9. Your queue requires permission to access Amazon S3 on your behalf. You can create a new service role or use an existing service role. If you don't have an existing service role, create and use a new service role.

   a. To use an existing service role, select **Choose a service role**, and then select a role from the dropdown.

   b. To create a new service role, select **Create and use a new service role**, and then enter a role name and description.
10. (Optional) To add environment variables for the queue environment, choose **Add new environment variable**, and then enter a name and value for each variable you add.

11. (Optional) Choose **Add new tag** to add one or more tags to your queue.
12. To create a default Conda queue environment, keep the checkbox selected. To learn more about queue environments, see Create a queue environment. If you are creating a queue for a customer-managed fleet, clear the checkbox.
13. Choose **Create queue**.

Create a queue environment

A queue environment is a set of environment variables and commands that set up fleet workers. You can use queue environments to provide software applications, environment variables, and other resources to jobs in the queue.

When you create a queue, you have the option of creating a default Conda queue environment. This environment provides service-managed fleets access to packages for partner DCC applications and renderers. For more information, see Default Conda queue environment.

You can add queue environments using the console, or by editing the json or YAML template directly. This procedure describes how to create an environment with the console.

1. To add a queue environment to a queue, navigate to the queue and select the **Queue environments tab**.
2. Choose **Actions**, then **Create new with form**.
3. Enter a name and description for the queue environment.
4. Choose **Add new environment variable**, and then enter a name and value for each variable you add.
5. (Optional) Enter a priority for the queue environment. The priority indicates the order that this queue environment will run on the worker. Higher priority queue environments will run first.
6. Choose **Create queue environment**.

Default Conda queue environment

When you create a queue associated with a service-managed fleet, you have the option of adding a default queue environment that supports Conda to download and install packages in a virtual environment for your jobs.

Conda provides packages from *channels*. A channel is a location where packages are stored. Deadline Cloud provides a channel, `deadline-cloud`, that hosts Conda packages that support partner DCC applications and renderers. Select each tab below to view the available packages for Linux or Windows.

Linux

- Blender
  - `blender=3.6`
  - `blender=4.2`
  - `blender-openjd`
- Houdini
  - `houdini=19.5`
  - `houdini=20.0`
  - `houdini=20.5`
  - `houdini-openjd`
- Maya
  - `maya=2024`
  - `maya-mtoa=2024.5.3`

    We are investigating reports of stalled renders with MtoA version 2024.5.3. If you have tasks that have stalled without an error, contact support.
  - `maya-openjd`
- Nuke
  - `nuke=15`
  - `nuke-openjd`

Windows

- Cinema 4D
  - `cinema4d=2024`
  - `cinema4d=2025`
  - `cinema4d-openjd`

- KeyShot
  - `keyshot=2024`
  - `keyshot-openjd`

When you submit a job to a queue with the default Conda environment, the environment adds two parameters to the job. These parameters specify the Conda packages and channels to use to configure the job's environment before tasks are processed. The parameters are:

- `CondaPackages` – a space-separated list of package match specifications, such as `blender=3.6` or `numpy>1.22`. The default is empty to skip creating a virtual environment.
- `CondaChannels` – a space separated list of Conda channels such as `deadline-cloud`, `conda-forge`, or `s3://amzn-s3-demo-bucket/conda/channel`. The default is `deadline-cloud`, a channel available to service-managed fleets that provides partner DCC applications and renderers.

When you use an integrated submitter to send a job to Deadline Cloud from your DCC, the submitter populates the value of the `CondaPackages` parameter based on the DCC application and submitter. For example, if you are using Blender the `CondaPackage` parameter is set to `blender=3.6.* blender-openjd=0.4.*`.

Associate a queue and fleet

A queue must be associated with a fleet so that the jobs can render. A single fleet can support multiple queues and a queue can be supported by multiple fleets. To associate an existing queue with an existing fleet, complete the following procedure.

1. From your Deadline Cloud farm, select the **Queue** you want to associate with a fleet. The queue displays.
2. To select a fleet to associate with your queue, choose **Associate fleets**.
3. Choose the **Select fleets** dropdown. A list of available fleets displays.
4. From the list of available fleets, select the **checkbox** next to the fleet or fleets you want to associate with your queue.
5. Choose **Associate**. The fleet association status should now be **Associated**.

Deadline Cloud fleets

This section explains how to manage service-managed fleets and customer-managed fleets (CMF) for Deadline Cloud.

You can set up two types of Deadline Cloud fleets:

- Service-managed fleets are fleets of workers that have default settings provided by this service, Deadline Cloud. These default settings are designed to be efficient and cost effective.
- Customer-managed fleets (CMFs) provide you with full control over your processing pipeline. A CMF can reside within AWS infrastructure, on premise, or in a co-located data center. This includes provisioning, operations, management, and decommissioning workers in the fleet.

Topics

Service-managed fleets

A service-managed fleet (SMF) is a fleet of workers that have default settings provided by Deadline Cloud. These default settings are designed to be efficient and cost-effective.

Some of the default settings limit the amount of time that workers and tasks can run. A worker can only run for seven days and a task can only run for five days. When the limit is reached, the task or worker stops. If this happens, you might lose work that worker or task was running. To avoid this, monitor your workers and tasks to ensure they don't exceed the maximum duration limits. To learn more about monitoring your workers, see Using the Deadline Cloud monitor.

Create a service-managed fleet

1. From the Deadline Cloud console, navigate to the farm you want to create the fleet in.
2. Select the **Fleets** tab, and then choose **Create fleet**.
3. Enter a **Name** for your fleet.
4. (Optional) Enter a **Description**. A clear description can help you quickly identify your fleet's purpose.

5. Select **Service-managed** fleet type.
6. Choose either the **Spot** or **On-demand** instance market option for your fleet. Spot instances are unreserved capacity that you can used at a discounted price, but may be interrupted by On-demand requests. On-demand instances are priced by the second, but have no long-term commitment, and will not be interrupted. By default, fleets use Spot instances.
7. For service access for your fleet, select an existing role or create a new role. A service role provides credentials to instances in the fleet, granting them permission to process jobs, and to users in the monitor so that they can read log information.
8. Choose **Next**.
9. Choose between CPU only instances or GPU accelerated instances. GPU accelerated instances may be able to process your jobs faster, but can be more expensive.
10. Select the operating system for your workers. You can leave the default, **Linux** or choose **Windows**.
11. (Optional) If you selected GPU accelerated instances, set the maximum and minimum number of GPUs in each instance. For testing purposes you are limited to one GPU. To request more for your production workloads, see Requesting a quota increase in the *Service Quotas User Guide*.
12. Enter the minimum and maximum **vCPU's** that you require for you fleet.
13. Enter the minimum and maximum **memory** that you require for you fleet.
14. (Optional) You can choose to allow or exclude specific instance types from your fleet to ensure only those instance types are used for this fleet.
15. (Optional) Set the maximum number of instances to scale the fleet so that capacity is available for the jobs in the queue. We recommend that you leave the minimum number of instances at **0** to ensure the fleet releases all instances when no jobs are queued.
16. (Optional) You can specify the size of the Amazon Elastic Block Store (Amazon EBS) gp3 volume that will be attached to the workers in this fleet. For more information, see the EBS user guide.
17. Choose **Next**.
18. (Optional) Define custom worker capabilities that define features of this fleet that can be combined with custom host capabilities specified on job submissions. One example is a particular license type if you plan to connect your fleet to your own license server.
19. Choose **Next**.
20. (Optional) To associate your fleet with a queue, select a **queue** from the dropdown. If the queue is set up with the default Conda queue environment, your fleet is automatically

provided with packages that support partner DCC applications and renderers. For a list of provided packages, see Default Conda queue environment.

21. Choose **Next**.
22. (Optional) To add a tag to your fleet, choose **Add new tag**, and then enter the **key** and **value** for that tag.
23. Choose **Next**.
24. Review your fleet settings, and then choose **Create fleet**.

Use a GPU accelerator

You can configure worker hosts in your service-managed fleets to use one or more GPUs to accelerate processing your jobs. Using an accelerator can reduce the time that it takes to process a job, but can increase the cost of each worker instance. You should test your workloads to understand the trade offs between a fleet using GPU accelerators and fleets that don't.

> **Note**
>
> For testing purposes you are limited to one GPU. To request more for your production workloads, see Requesting a quota increase in the *Service Quotas User Guide*.

You decide whether your fleet will use GPU accelerators when you specify the worker instance capabilities. If you decide to use GPUs, you can specify the minimum and maximum number of GPUs for each instance, the types of GPU chips to use, and the runtime driver for the GPUs.

The available GPU accelerators are:

- `T4` - NVIDIA T4 Tensor Core GPU
- `A10G` - NVIDIA A10G Tensor Core GPU
- `L4` - NVIDIA L4 Tensor Core GPU
- `L40s` - NVIDIA L40S Tensor Core GPU

You can choose from the following runtime drivers:

- `Latest` - Use the latest runtime available for the chip. If you specify `latest` and a new version of the runtime is released, the new version of the runtime is used.

- `GRID:R550` - NVIDIA vGPU software 17
- `GRID:R535` - NVIDIA vGPU software 16

If you don't specify a runtime, Deadline Cloud uses `latest` as the default. However, if you have multiple accelerators and specify `latest` for some and leave others blank, Deadline Cloud raises an exception.

Software licensing for service-managed fleets

Deadline Cloud provides usage-based licensing (UBL) for commonly used software packages. Supported software packages are automatically licensed when they run on a service-managed fleet. You don't need to configure or maintain a software license server. Licenses scale so you won't run out for larger jobs.

You can install software packages that support UBL using the built-in Deadline Cloud conda channel, or you can use your own packages. For more information about the conda channel, see Create a queue environment.

For a list of supported software packages and information about pricing for UBL, see AWS Deadline Cloud pricing.

Bring your own license with service-managed fleets

With Deadline Cloud usage-based licensing (UBL) you don't need to manage separate licence agreements with software vendors. However, if you have existing licenses or need to use software that isn't available through UBL, you can use your own software licenses with your Deadline Cloud service-managed fleets. You connect your SMF to the software license server via the internet to check out a license for each worker in the fleet.

For an example of connecting to a license server using a proxy, see Connect service-managed fleets to a custom license server in the *Deadline Cloud Developer Guide*.

VFX Reference Platform compatibility

The VFX Reference Platform is a common target platform for the VFX industry. To use the standard service-managed fleet Amazon EC2 instance running Amazon Linux 2023 with software that supports the VFX Reference Platform, you should keep in mind the following considerations when using a service-managed fleet.

The VFX Reference Platform is updated annually. These considerations for using an AL2023 including Deadline Cloud service-managed fleets are based on the calendar year (CY) 2022 through 2024 Reference Platforms. For more information, see VFX Reference Platform.

> **Note**
>
> If you are creating a custom Amazon Machine Image (AMI) for a customer-managed fleet, you can add these requirements when you prepare the Amazon EC2 instance.

To use VFX Reference Platform supported software on an AL2023 Amazon EC2 instance, consider the following:

- The glibc version installed with AL2023 is compatible for runtime use, but not for building software compatible with the VFX Reference Platform CY2024 or earlier.
- Python 3.9 and 3.11 are provided with the service-managed fleet making it compatible with VFX Reference Platform CY2022 and CY2024. Python 3.7 and 3.10 are not provided in the service-managed fleet. Software requiring them must provide the Python installation in the queue or job environment.
- Some Boost library components provided in the service-managed fleet are version 1.75, which is not compatible with the VFX Reference Platform. If your application uses Boost, you must provide your own version of the library for compatibility.
- Intel TBB update 3 is provided in the service-managed fleet. This is compatible with VFX Reference Platform CY2022, CY2023, and CY2024.
- Other libraries with versions specified by the VFX Reference Platform are not provided by the service-managed fleet. You must provide the library with any application used on a service-managed fleet. For a list of libraries, see the reference platform.

Customer-managed fleets

When you want to use a fleet of workers that you manage, you can create a customer-managed fleet (CMF) that Deadline Cloud uses to process your jobs. Use a CMF when:

- You have existing on-premise workers to integrate with Deadline Cloud.
- You have workers in a co-located data center.
- You want direct control of Amazon Elastic Compute Cloud (Amazon EC2) workers.

When you use a CMF, you have full control over and responsibility for the fleet. This includes provisioning, operations, management, and decommissioning workers in the fleet.

For more information, see Create and use Deadline Cloud customer-managed fleets in the *Deadline Cloud Developer Guide*.

Managing users in Deadline Cloud

AWS Deadline Cloud uses AWS IAM Identity Center to manage users and groups. IAM Identity Center is a cloud-based single sign-on service that can be integrated with your enterprise single-sign on (SSO) provider. With integration, users can sign in with their company account.

Deadline Cloud enables IAM Identity Center by default, and it is required to set up and use Deadline Cloud. For more information, see Manage your identity source.

An organization owner for your AWS Organizations is responsible for managing the users and groups that have access to your Deadline Cloud monitor. You can create and manage these users and groups using IAM Identity Center or the Deadline Cloud console. For more information, see What is AWS Organizations.

You create and remove users and groups that can manage farms, queues, and fleets using the Deadline Cloud console. When you add a user to Deadline Cloud, they must reset their password using IAM Identity Center before they get access.

Topics

Manage users and groups for the monitor

An Organizations owner can use the Deadline Cloud console to manage the users and groups that have access to the Deadline Cloud monitor. You can choose from existing IAM Identity Center users and groups, or you can add new users and groups from the console.

1. Sign in to the AWS Management Console and open the Deadline Cloud console. From the main page, in the **Get started** section, choose **Set up Deadline Cloud** or **Go to dashboard**.
2. In the left navigation pane, choose **User management**. By default, the **Groups** tab is selected.

Depending on the action to take, choose either the **Groups** tab or **Users** tab.

Groups

To create a group

1. Choose **Create group**.
2. Enter a group name. The name must be unique among groups in your IAM Identity Center organization.

To remove a group

1. Select the group to remove.
2. Choose **Remove**.
3. In the confirmation dialog, choose **Remove group**.

> **Note**
>
> You are removing the group from IAM Identity Center. Group members can no longer sign in to the Deadline Cloud or access farm resources.

Users

To add users

1. Choose the **Users** tab.
2. Choose **Add users**.
3. Enter the name, email address, and username for the new user.
4. (Optional) Choose one or more IAM Identity Center groups to add the new user to.
5. Choose **Send invite** to send the new user an email with instructions for joining your IAM Identity Center organization.

To remove a user

1. Select the user you to remove.
2. Choose **Remove**.
3. In the confirmation dialog, choose **Remove user**.

> **Note**
>
> You are removing the user from IAM Identity Center. The user can no longer sign in to the Deadline Cloud monitor or access farm resources.

Manage users and groups for farms, queues, and fleets

As part of managing users and groups, you can grant access permissions at different levels. Each subsequent level includes the permissions for the previous levels. The following list describes the four access levels from the lowest level to the highest level:

- **Viewer** – Permission to see resources in the farms, queues, fleets, and jobs they have access to. A viewer can't submit or make changes to jobs.
- **Contributor** – Same as a viewer, but with permission to submit jobs to a queue or farm.
- **Manager** – Same as contributor, but with permission to edit jobs in queues they have access to, and grant permissions on resources that they have access to.
- **Owner** – Same as manager, but can view and create budgets and see usage.

> **Note**
>
> Changes to access permissions can take up to 10 minutes to reflect in the system.

1. If you haven't already, sign in to the AWS Management Console and open the Deadline Cloud console.
2. In the left navigation pane, choose **Farms and other resources**.
3. Select the farm to manage. Choose the farm name to open the details page. You can search for the farm using the search bar.
4. To manage a queue or fleet, choose the **Queues** or **Fleets** tab, and then choose the queue or fleet to manage.
5. Choose the **Access management** tab. By default, the **Groups** tab is selected. To manage users, choose **Users**.

Depending on the action to take, choose either the **Groups** tab or **Users** tab.

Groups

To add groups

1. Select the **Groups** toggle.
2. Choose **Add group**.
3. From the dropdown, select the groups to add.
4. For the group access level, choose one of the following options:
   - **Viewer**
   - **Contributor**
   - **Manager**
   - **Owner**
5. Choose **Add**.

To remove groups

1. Select the groups to remove.
2. Choose **Remove**.
3. In the confirmation dialog, choose **Remove group**.

Users

To add users

1. To add a user, choose **Add user**.
2. From the dropdown, select the users to add.
3. For the user access level, choose one of the following options:
   - **Viewer**
   - **Contributor**
   - **Manager**
   - **Owner**

4. Choose **Add**.

To remove users

1. Select the user to remove.
2. Choose **Remove**.
3. In the confirmation dialog, choose **Remove user**.

Deadline Cloud jobs

A *job* is a set of instructions that AWS Deadline Cloud uses to schedule and run work on available workers. When you create a job, you choose the farm and queue to send the job to. You also provide a JSON or YAML file that provides the instructions for workers to process. Deadline Cloud accepts job templates that follow the Open Job Description (OpenJD) specification for describing jobs. For more information, see the Open Job Description Documentation on the GitHub website.

A job consists of:

- *Priority* – The approximate order that Deadline Cloud processes a job in a queue. You can set the job priority between 1 and 100, jobs with a higher number priority are generally processed first. Jobs with the same priority are processed in the order received.
- *Steps* – Defines the script to run on workers. Steps can have requirements such as minimum worker memory or other steps that need to complete first. Each step has one or more tasks.
- *Tasks* – A unit of work sent to a worker to perform. A task is a combination of a step's script and parameters, such as frame number, that are used in the script. The job is complete when all tasks are complete for all steps.
- *Environments* – Set up and tear down instructions shared by multiple steps or tasks.

You can create a job in any of the following ways:

- Use a Deadline Cloud submitter.
- Create a job bundle and use the Deadline Cloud command line interface (Deadline Cloud CLI).
- Use the AWS SDK.
- Use the AWS Command Line Interface (AWS CLI).

A *submitter* is a plugin for your digital content creation (DCC) software that manages creating a job in the interface to your DCC software. After you create the job, you use the submitter to send it to Deadline Cloud for processing. Behind the scenes, the submitter creates an OpenJD job template that describes the job. At the same time, it uploads your asset files to an Amazon Simple Storage Service (Amazon S3) bucket. To reduce the time it takes to send files, only files that have changed since the last time you uploaded files are sent to Amazon S3.

You can create *limits* to manage how jobs use constrained resources such as software licenses. Jobs that use limits only use the number of resources allowed under the limit. For more information, see Create resource limits for jobs.

To create your own scripts and pipelines to submit jobs to Deadline Cloud, you can use the Deadline Cloud CLI, the AWS SDK, or the AWS CLI to call operations to create, get, view, and list jobs. The following topics explain how to use the Deadline Cloud CLI.

The Deadline Cloud CLI is installed along with the Deadline Cloud submitter. For more information, see Set up Deadline Cloud submitters.

Topics

Submit jobs with the Deadline Cloud CLI

To submit a job using the Deadline Cloud command line interface (Deadline Cloud CLI), use the `deadline bundle submit` command.

Jobs are submitted to queues. If you haven't set up a farm and queue yet, use the Deadline Cloud console to set up a farm and queue and to see the farm and queue ID. For more information, see Define farm details and Define queue details.

To set the default farm and queue for the Deadline Cloud CLI, use the following command. When you set the defaults, you can use Deadline Cloud CLI commands without specifying a farm or queue. In the following example, replace *farmId* and *queueId* with your own information:

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

To specify the steps and tasks in a job, create an OpenJD job template. For more information, see Template Schemas [Version: 2023-09] in the *Open Job Description specification* GitHub repository.

The following example is a YAML job template. It defines a job with two steps and five tasks per step.

```
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
```

To create a job, create a new folder named `sample_job`, then save the template file in the new folder as `template.yaml`. You submit the job with the following Deadline Cloud CLI command:

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

The response from the command contains an identifier for the job. Remember the ID so that you can check the job's status later.

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

There are additional options that you can use when submitting a job. For more information, see More options for submitting jobs with the Deadline Cloud CLI.

More options for submitting jobs with the Deadline Cloud CLI

The `deadline bundle submit` Deadline Cloud CLI command provides options that you can use to specify additional information for a job. The following examples show you how to:

- Specify parameters used when processing the job template.
- Attach files and folders in a shared environment to a job.
- Set the maximum number of workers that can process a job.
- Set the maximum number of task failures before a job is canceled.
- Set the maximum number of retries for a task.

Job parameters

The `parameters` option sets the value of a job parameter when you create the job. The job template defines the field, and the `parameters` option sets the value. A parameter can have a default value. If a value is specified for the parameter, the specified value overrides the default value.

The following job template defines the `TestParameter` field:

```
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
```

The following command sets the value of the `TestParameter` to "Hello AWS":

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

Storage profiles

Storage profiles help with sharing files between workers with different operating systems. Create a storage profile using the Deadline Cloud console. Then, use the `storage-profile-id` parameter to use the storage profile. For more information, see Storage profiles and path mapping in the *Deadline Cloud Developer Guide*.

To set the storage profile for job submissions, using the Deadline Cloud CLI, use the following command to set the `storage-profile-id` configuration parameter:

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

Maximum workers for a job

The `max-worker-count` option sets the maximum number of workers that can be assigned to a job. When the maximum is reached, no more workers are assigned to the job even if more workers are available in the fleet.

```
deadline bundle submit sample_job --max-worker-count 10
```

Maximum failed tasks

The `max-failed-tasks-count` option sets the maximum number of tasks that can fail before the entire job fails and all remaining tasks are marked `CANCELED`. The default value is 100.

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

Maximum failed task retries

The `max-retries-per-task` option sets the maximum number of times that a task is retried before it fails. When a task is retried, it is put in the `READY` state. The default value is 5.

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

Schedule jobs in Deadline Cloud

After a job is created, AWS Deadline Cloud schedules it to be processed on one or more of the fleets associated with a queue. The fleet that processes a particular task is chosen based on the capabilities configured for the fleet and the host requirements of a specific step.

Jobs in a queue are scheduled in a best-effort priority order, highest to lowest. When two jobs have the same priority, the oldest job is scheduled first.

The following sections provide details of the process of scheduling a job.

Determine fleet compatibility

After a job is created, Deadline Cloud checks the host requirements for each step in the job against the capabilities of the fleets associated with the queue the job was submitted to. If a fleet meets the host requirements, the job is put into the `READY` state.

If any step in the job has requirements that can't be met by a fleet associated with the queue, the step's status is set to `NOT_COMPATIBLE`. In addition, the rest of the steps in the job are canceled.

Capabilities for a fleet are set at the fleet level. Even if a worker in a fleet meets the job's requirements, it won't be assigned tasks from the job if its fleet doesn't meet the job's requirements.

The following job template has a step that specifies host requirements for the step:

```
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
```

This job can be scheduled to a fleet with the following capabilities:

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

This job can't be scheduled to a fleet with any of the following capabilities:

```
{
    "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.
```

Fleet scaling

When a job is assigned to a compatible service-managed fleet, the fleet is auto scaled. The number of workers in the fleet changes based on the number of tasks available for the fleet to run.

When a job is assigned to a customer-managed fleet, workers might already exist or can be created using event-based auto scaling. For more information, see Use EventBridge to handle auto scaling events in the *Amazon EC2 Auto Scaling User Guide*.

Sessions

The tasks in a job are divided into one or more sessions. Workers run the sessions to set up the environment, run the tasks, and then tear down the environment. Each session is composed of one or more actions that a worker must take.

As a worker completes section actions, additional session actions can be sent to the worker. The worker reuses existing environments and job attachments in the session to complete tasks more efficiently.

Job attachments are created by the submitter that you use as part of your Deadline Cloud CLI job bundle. You can also create job attachments using the `--attachments` option for the `create-job` AWS CLI command. Environments are defined in two places: queue environments attached to a specific queue, and job and step environments defined in the job template.

There are four session action types:

- `syncInputJobAttachments` – Downloads the input job attachments to the worker.
- `envEnter` – Performs the on`Enter` actions for an environment.

- `taskRun` – Performs the onRun actions for a task.
- `envExit` – Performs the on`Exit` actions for an environment.

The following job template has a step environment. It has an on`Enter` definition to set up the step environment, an onRun definition that defines the task to run, and an on`Exit` definition to tear down the step environment. The sessions created for this job will include an `envEnter` action, one or more `taskRun` actions, and then an `envExit` action.

```
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 }}
```

Step dependencies

Deadline Cloud supports defining dependencies between steps so that one step waits until another step is complete before starting. You can define more than one dependency for a step. A step with a dependency isn't scheduled until all of its dependencies are complete.

If the job template defines a circular dependency, the job is rejected and the job status is set to `CREATE_FAILED`.

The following job template creates a job with two steps. `StepB` depends on `StepA`. `StepB` only runs after `StepA` completes successfully.

After the job is created, `StepA` is in the `READY` state and `StepB` is in the `PENDING` state. After `StepA` finishes, `StepB` moves to the `READY` state. If `StepA` fails, or if `StepA` is canceled, `StepB` moves to the `CANCELED` state.

You can set a dependency on multiple steps. For example, if `StepC` depends on both `StepA` and `StepB`, `StepC` won't start until the other two steps finish.

```
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!
```

Job states in Deadline Cloud

This topic describes how to use the AWS Deadline Cloud command line interface (Deadline Cloud CLI) to view the status of a job or step. To use the Deadline Cloud monitor to view the status of jobs or steps, see Manage jobs, steps, and tasks in Deadline Cloud.

You can also create rules for the default Amazon EventBridge event bus to send an event to a target, such as the Amazon Simple Notification Service to send SMS texts or email when a job, step, or task changes state. For more information, see Managing Deadline Cloud events using Amazon EventBridge in the *Deadline Cloud Developer Guide*>.

You can see the status of a job using the `deadline job get --job-id` Deadline Cloud CLI command. The response to the commands include the status of the job or step and the number of tasks in each processing status.

When you first submit a job, the status is `CREATE_IN_PROGRESS`. If the job passes the validation checks, its status changes to `CREATE_COMPLETE`. If not, the status changes to `CREATE_FAILED`.

Some possible reasons that a job can fail validation checks include the following:

- The job template doesn't follow the OpenJD specification.
- The job contains too many steps.
- The job contains too many total tasks.

To see the quotas for the maximum number of steps and tasks in a job, use the Service Quotas console. For more information, see Quotas for Deadline Cloud.

There may also be an internal service error that prevents a job from being created. If this happens, the job's status code is `INTERNAL_ERROR` and the status message field provides a more detailed explanation.

Use the following Deadline Cloud CLI command to view the details for a job. In the following example, replace *jobID* with your own information:

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

The response from the `deadline job get` command is as follows:

```
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
```

Each task in a job or step has a status. The task statuses are combined to give an overall status for jobs and steps. The number of tasks in each state is reported in the `taskRunStatusCounts` field of the response.

The status of a job or step depends on the status of its tasks. The status is determined by tasks that have these statuses, in order. Step statuses are determined the same as the job status.

The following list describes the statuses:

`NOT_COMPATIBLE`

The job is not compatible with the farm because there are no fleets that can complete one of the tasks in the job.

`RUNNING`

One or more workers are running tasks from the job. As long as there is at least one running task, the job is marked `RUNNING`.

`ASSIGNED`

One or more workers are assigned tasks in the job as their next action. The environment, if any, is set up.

`STARTING`

One or more workers is setting up the environment for running tasks.

`SCHEDULED`

Tasks for the job are scheduled on one or more workers as the worker's next action.

`READY`

At least one task for the job is ready to be processed.

`INTERRUPTING`

At least one task in the job is being interrupted. Interruptions can happen when you manually update the job's status. It can also happen in response to an interruption due to Amazon Elastic Compute Cloud (Amazon EC2) Spot price changes.

`FAILED`

One or more tasks in the job didn't complete successfully.

`CANCELED`

One or more tasks in the job have been canceled.

`SUSPENDED`

At least one task in the job has been suspended.

`PENDING`

A task in the job is waiting on the availability of another resource.

`SUCCEEDED`

All tasks in the job were successfully processed.

Modify a job in Deadline Cloud

You can use the following AWS Command Line Interface (AWS CLI) `update` commands to modify the configuration of a job, or to set the target status of a job, step, or task:

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

In the following examples of the `update` commands, replace each *`user input placeholder`* with your own information.

You can also use the Deadline Cloud monitor to modify the configuration of a job. For more information, see Manage jobs, steps, and tasks in Deadline Cloud.

Example – Requeue a job

All tasks in the job switch to the READY status, unless there are step dependencies. Steps with dependencies switch to either READY or PENDING as they are restored.

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

Example – Cancel a job

All tasks in the job that don't have the status SUCCEEDED or FAILED are marked CANCELED.

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

Example – Mark a job failed

All tasks in the job that have the status SUCCEEDED are left unchanged. All other tasks are marked FAILED.

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

Example – Mark a job successful

All tasks in the job move to the SUCCEEDED state.

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

Example – Suspend a job

Tasks in the job in the SUCCEEDED, CANCELED, or FAILED state don't change. All other tasks are marked SUSPENDED.

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

Example – Change the priority of a job

Updates the priority of a job in a queue to change the order that it is scheduled. Higher priority jobs are generally scheduled first.

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

Example – Change the number of failed tasks allowed

Updates the maximum number of failed tasks that the job can have before the remaining tasks are canceled.

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

Example – Change the number of task retries allowed

Updates the maximum number of retries for a task before the task fails. A task that has reached the maximum number of retries can't be requeued until this value is increased.

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

```
--max-retries-per-task 10
```

Example – Archive a job

Updates the job's lifecycle status to ARCHIVED. Archived jobs can't be scheduled or modified. You can only archive a job that is in the FAILED, CANCELED, SUCCEEDED, or SUSPENDED state.

```
aws deadline update-job \
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--lifecycle-status ARCHIVED
```

Example – Requeue a step

All tasks in the step switch to the READY state, unless there are step dependencies. Tasks in steps with dependencies switch to either READY or PENDING, and the task is restored.

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

Example – Cancel a step

All tasks in the step that don't have the status SUCCEEDED or FAILED are marked CANCELED.

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

Example – Mark a step failed

All tasks in the step that have the status SUCCEEDED are left unchanged. All other tasks are marked FAILED.

```
aws deadline update-step \
```

```
--farm-id farmID \
--queue-id queueID \
--job-id jobID \
--step-id stepID \
--target-task-run-status FAILED
```

Example – Mark a step successful

All tasks in the step are marked SUCCEEDED.

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

Example – Suspend a step

Tasks in the step in the SUCCEEDED, CANCELED, or FAILED state don't change. All other tasks are marked SUSPENDED.

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

Example – Change the status of a task

When you use the `update-task` Deadline Cloud CLI command, the task switches to the specified status.

```
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
```

How Deadline Cloud processes jobs

To process a job, AWS Deadline Cloud uses the Open Job Description (OpenJD) job template to determine the resources needed. Deadline Cloud selects a suitable worker for a step from the fleets associated with your queue. The selected worker meets all of the capability attributes required for the step.

Next, Deadline Cloud sends instructions to the workers to set up a session for the step. The software required for the step must be available on the worker instance for the job to run. The service can open sessions on multiple workers if the scaling settings for the fleet have capacity.

You can set up the software in an Amazon Machine Image (AMI), or your worker can load the software at runtime from a repository or package manager. You can use queue, job, or step environments to deploy the software that you prefer.

The Deadline Cloud service uses the OpenJD template to determine the steps required for the job, and the tasks required for each step. Some steps have dependencies on other steps, so Deadline Cloud determines the order to complete the steps. Then, Deadline Cloud sends the tasks for each step to workers to process. When a task is finished, the service sends another task in the same session, or the worker can start a new session.

You can track the progress of the job in the Deadline Cloud monitor, the Deadline Cloud command line interface (Deadline Cloud CLI) or the AWS CLI. For more information about using the monitor, see Using the Deadline Cloud monitor. For more information about using the Deadline Cloud CLI, see Job states in Deadline Cloud.

After all tasks in each step are finished, the job is complete and the output is ready to download to your workstation. Even if the job didn't finish, the output from each step and task that finished is available to download.

Deadline Cloud removes jobs 120 days after they were submitted. When a job is removed, all of the steps and tasks associated with the job are also removed. If you need to re-run the job, submit the OpenJD template for the job again.

Create resource limits for jobs

Jobs submitted to Deadline Cloud may depend on resources that are shared between multiple jobs. For example, a farm may have more workers than floating licences for a specific resource. Or a

shared file server may only be able to serve data to a limited number of workers at the same time. In some cases, one or more jobs can claim all of these resources, causing errors due to unavailable resources when new workers start.

To help solve this, you can use *limits* for these constrained resources. Deadline Cloud accounts for the availability of constrained resources and uses that information to ensure that resources are available as new workers start up so that jobs have a lower likelihood of failing due to unavailable resources.

Limits are created for the entire farm. Jobs submitted to a queue can only acquire limits associated with the queue. If you specify a limit for a job that is not associated with the queue, the job isn't compatible and won't run.

To use a limit, you

- Create a limit
- Associate a limit and a queue
- Submit a job requiring limits

> **Note**
>
> If you run a job that has constrained resources in a queue that is not associated with a limit, that job can consume all of the resources. If you have a constrained resource, make sure that all of the steps in jobs in queues that use the resource are associated with a limit.

For limits defined in a farm, associated with a queue, and specified in a job, one of four things can happen:

- If you create a limit, associate it with a queue, and specify the limit in a job's template, the job runs and uses only the resources defined in the limit.
- If you create a limit, specify it in a job template, but don't associate the limit with a queue, the job is marked incompatible and won't run.
- If you create a limit, don't associate it with a queue, and don't specify the limit in a job's template, the job runs but does not use the limit.
- If you don't use a limit at all, the job runs.

If you associate a limit to multiple queues, the queues share the resources constrained by the limit. For example, if you create a limit of 100, and one queue is using 60 resources, other queues can only use 40 resources. When a resource is released, it can be taken by a task from any queue.

Deadline Cloud provides two AWS CloudFormation metrics to help you monitor the resources provided by a limit. You can monitor the current number of resources in use and the maximum number of resources available in the limit. For more information, see Resource limit metrics in the *Deadline Cloud Developer Guide*.

You apply a limit to a job step in a job template. When you specify the amount requirement name of a limit in the `amounts` section of the `hostRequirements` of a step and a limit with the same `amountRequirementName` is associated with the job's queue, tasks scheduled for this step are constrained by the limit for the resource.

If a step requires a resource that is constrained by a limit that is reached, tasks in that step won't be picked up by additional workers.

You can apply more than one limit to a job step. For example, if the step uses two different software licenses, you can apply a separate limit for each license. If a step requires two limits and the limit for one of the resources is reached, tasks in that step won't be picked up by additional workers until the resources become available.

Stopping and deleting limits

When you stop or delete the association between a queue and a limit, a job using the limit stops scheduling tasks from steps that require this limit and blocks the creation of new sessions for a step.

Tasks that are in the READY state remain ready, and tasks automatically resume with the association between the queue and the limit becomes active again. You don't need to requeue any jobs.

When you stop or delete the association between a queue and a limit, you have two choices on how to stop running tasks:

- *Stop and cancel tasks* – Workers with sessions that acquired the limit cancel all tasks.
- *Stop and finish running tasks* – Workers with sessions that acquired the limit complete their tasks.

When you delete a limit using the console, workers first stop running tasks immediately or eventually when they complete. When the association is deleted, the following happens:

- Steps requiring the limit are marked not compatible.
- The entire job containing those steps is canceled, including steps that don't require the limit.
- The job is marked not compatible.

If the queue associated with the limit has an associated fleet with a fleet capability that matches the amount requirement name of the limit, that fleet will continue to process jobs with the specified limit.

Create a limit

You create a limit using the Deadline Cloud console or the CreateLimit operation in the Deadline Cloud API. Limits are defined for a farm, but associated with queues. After you create a limit, you can associate it with one or more queues.

To create a limit

1. From the Deadline Cloud console (https://console.aws.amazon.com/deadlinecloud/home) dashboard, select the farm that you want to create a queue for.
2. Choose the farm to add the limit to, choose the **Limits** tab, and then choose **Create limit**.
3. Provide the details for the limit. The **Amount requirement name** is the name used in the job template to identify the limit. It must begin with the prefix **`amount.`** followed by the amount name. The amount requirement name must be unique in queues associated with the limit.
4. If you choose **Set a maximum amount**, that is the total number of resources allowed by this limit. If you choose **No maximum amount**, resource usage isn't limited. Even when resource usage isn't limited, the `CurrentCount` Amazon CloudWatch metric is emitted so that you can track usage. For more information, see CloudWatch metrics in the *Deadline Cloud Developer Guide*.
5. If you already know the queues that should use the limit, you can choose them now. You don't need to associate a queue to create a limit.
6. Choose **Create limit**.

Associate a limit and a queue

After you create a limit, you can associate one or more queues with the limit. Only queues that are associated with a limit use the values specified in the limit.

You create an association with a queue using the Deadline Cloud console or the CreateQueueLimitAssociation operation in the Deadline Cloud API.

To associate a queue with a limit

1. From the Deadline Cloud console (https://console.aws.amazon.com/deadlinecloud/home) dashboard, select the farm where you want to associate a limit with a queue.
2. Choose the **Limits** tab, choose the limit to associate a queue with, and then choose **Edit limit**.
3. In the **Associate queues** section, choose the queues to associate with the limit.
4. Choose **Save changes**.

Submit a job requiring limits

You apply a limit by specifying it as a host requirement for the job or job step. If you don't specify a limit in a step and that step uses an associated resource, the step's usage isn't counted against the limit when jobs are scheduled..

Some Deadline Cloud submitters enable you to set a host requirement. You can specify the limit's amount requirement name in the submitter to apply the limit.

If your submitter doesn't support adding host requirements, you can also apply a limit by editing the job template for the job.

To apply a limit to a job step in the job bundle

1. Open the job template for the job using a text editor. The job template is located in the job bundle directory for the job. For more information, see Job bundles in the *Deadline Cloud Developer Guide*.
2. Find the step definition for the step to apply the limit to.
3. Add the following to the step definition. Replace *amount.name* with the amount requirement name of your limit. For typical use, you should set the `min` value to 1.

   YAML

   ```
   hostRequirements:
     amounts:
     - name: amount.name
       min: 1
   ```

JSON

```
"hostRequirements": {
    "amounts": [
        {
            "name": "amount.name",
            "min": "1"
        }
    ]
}
```

You can add multiple limits to a job step as follows. Replace *amount.name_1* and *amount.name_2* with the amount requirement names of your limits.

YAML

```
  hostRequirements:
    amounts:
    - name: amount.name_1
      min: 1
    - name: amount.name_2
      min: 1
```

JSON

```
"hostRequirements": {
    "amounts": [
        {
            "name": "amount.name_1",
            "min": "1"
        },
        {
            "name": "amount.name_2",
            "min": "1"
        }
    ]
}
```

4. Save the changes to the job template.

Troubleshooting Deadline Cloud jobs

For information about common problems with jobs in AWS Deadline Cloud, see the following topics.

Topics

Why did creating my job fail?

Some possible reasons that a job can fail validation checks include the following:

- The job template doesn't follow the OpenJD specification.
- The job contains too many steps.
- The job contains too many total tasks.
- There was an internal service error that prevents the job from being created.

To see the quotas for the maximum number of steps and tasks in a job, use the Service Quotas console. For more information, see Quotas for Deadline Cloud.

Why is my job not compatible?

Common reasons that jobs are not compatible with queues include the following:

- No fleets are associated with the queue that the job was submitted to. Open the Deadline Cloud monitor, and check that the queue has associated fleets. For more information about how to view queues, see View queue and fleet details in Deadline Cloud.
- The job has host requirements that are not satisfied by any of the fleets associated with the queue. To check, compare the `hostRequirements` entry in the job template with the configuration of the fleets in your farm. Make sure that one of the fleets satisfies the host requirements. For more information about fleet compatibility, see Determine fleet compatibility. To view fleet configuration, see View queue and fleet details in Deadline Cloud.

Why is my job stuck in ready?

Possible reasons for your job appearing to be stuck in the READY state include the following:

- The maximum worker count for fleets associated with the queue is set to zero. To check, see View queue and fleet details in Deadline Cloud.
- There is a higher priority job in the queue. To check, see View queue and fleet details in Deadline Cloud.
- For customer-managed fleets, check the auto scaling configuration. For more information, see Create fleet infrastructure with an Amazon EC2 Auto Scaling group in the *Deadline Cloud Developer Guide*.

Why did my job fail?

A job can fail for many reasons. To search for the issue, open the Deadline Cloud monitor and choose the failing job. Choose a task that failed and then view the logs for the task. For instructions, see View logs in Deadline Cloud.

- If you see license errors or if you get a watermark that occurs because the software doesn't have a valid license, make sure that the worker can connect to the required license server. For more information, see Connect customer-managed fleets to a license endpoint in the *Deadline Cloud Developer Guide*.

Why is my step pending?

Steps may stay in the PENDING state when one or more of their dependencies are not complete. You can check the state of dependencies using the Deadline Cloud monitor. For instructions, see View a step in Deadline Cloud.

File storage for Deadline Cloud

Workers must have access to the storage locations that contain the input files necessary to process a job, and to the locations that store the output. AWS Deadline Cloud provides two options for storage locations:

- With *job attachments*, Deadline Cloud transfers the input and output files for your jobs back and forth between a workstation and Deadline Cloud workers. To enable the file transfers, Deadline Cloud uses an Amazon Simple Storage Service (Amazon S3) bucket in your AWS account.

  When you use job attachments with a service-managed fleet, you can set up a virtual file system (VFS) in your virtual private network (VPN). Then workers can load files only when needed.
- With *shared storage*, you use file sharing with your operating system to provide access to files.

  When you use cross-platform shared storage, you can create a *storage profile* so that workers can map the path to files between two different operating systems.

Topics

Job attachments in Deadline Cloud

Job attachments enable you to transfer files back and forth between your workstation and AWS Deadline Cloud. With job attachments, you don't need to manually set up an Amazon S3 bucket for your files. Instead, when you create a queue with the Deadline Cloud console, you choose the bucket for your job attachments.

The first time that you submit a job to Deadline Cloud, all of the files for the job are transferred to Deadline Cloud. For subsequent submissions, only the files that have changed are transferred, saving both time and bandwidth.

After processing is complete, you can download the result from the job detail page, or by using the Deadline Cloud CLI `deadline job download-output` command.

You can use the same S3 bucket for multiple queues. Set a different root prefix for each queue to organize the attachments in the bucket.

When you create a queue with the console, you can either choose an existing AWS Identity and Access Management (IAM) role or you can have the console create a new role. If the console creates the role, it sets permissions to access the bucket that's specified for the queue. If you choose an existing role, you must grant the role permissions to access the S3 bucket.

Encryption for job attachment S3 buckets

Job attachment files are encrypted in your S3 bucket by default. This helps secure your information from unauthorized access. You don't need to do anything to have your files encrypted with keys provided by Deadline Cloud. For more information, see Amazon S3 now automatically encrypts all new objects in the *Amazon S3 User Guide*.

You can use your own customer managed AWS Key Management Service key to encrypt the S3 bucket that contains your job attachments. To do so, you must modify the IAM role for the queue associated with the bucket to allow access to the AWS KMS key.

To open the IAM policy editor for the queue role

1. Sign in to the AWS Management Console and open the Deadline Cloud console. From the main page, in the **Get started** section, choose **View farms**.
2. From the list of farms, choose the farm that contains the queue to modify.
3. From the list of queues, choose the queue to modify.
4. In the **Queue details** section, choose the **Service role** to open the IAM console for the service role.

Next, complete the following procedure.

To update the role policy with permission for AWS KMS

1. From the list of **Permissions policies**, choose the policy for the role.
2. In the **Permissions defined in this policy** section, choose **Edit**.
3. Choose **Add new statement**.
4. Copy and paste the following policy into the editor. Change the *Region*, *accountID*, and *keyID* to your own values.

```
{
   "Effect": "Allow",
```

```
    "Action": [
        "kms:Decrypt",
        "kms:DescribeKey",
        "kms:GenerateDataKey"
    ],
    "Resource": [
        "arn:aws:kms:Region:accountID:key/keyID"
    ]
}
```

5. Choose **Next**.
6. Review the changes to the policy, and then when you're satisfied, choose **Save changes**.

Managing job attachments in S3 buckets

Deadline Cloud stores the job attachment files required for your job in an S3 bucket. These files accumulate over time, leading to increased Amazon S3 costs. To reduce costs, you can apply an S3 Lifecycle configuration to your S3 bucket. This configuration can automatically delete files in the bucket. Because the S3 bucket is in your account, you can choose to modify or remove the S3 Lifecycle configuration at any time. For more information, see Examples of S3 Lifecycle configuration in the *Amazon S3 User Guide*.

For a more granular S3 bucket management solution, you can set up your AWS account to expire objects in an S3 bucket based on the last time that they were accessed. For more information, see Expiring Amazon S3 objects based on last accessed date to decrease costs on the AWS Architecture Blog.

Deadline Cloud virtual file system

Virtual file system support for job attachments in AWS Deadline Cloud enables client software on workers to communicate directly with Amazon Simple Storage Service. Workers can load files only when needed instead of downloading all files before processing. Files are stored locally. This approach avoids downloading assets used more than once multiple times. All files are removed after the job completes.

- The virtual file system provides a significant performance boost for specific job profiles. In general, smaller subsets of total files with larger fleets of workers show the most benefit. Small numbers of files with fewer workers have roughly equivalent processing times.
- Virtual file system support is only available for Linux workers in service-managed fleets.

- The Deadline Cloud virtual file system supports the following operations, but is not POSIX compliant:
  - File `create`, `delete`, open, `close`, `read`, `write`, append, `truncate`, `rename`, `move`, `copy`, `stat`, `fsync`, and `falloc`
  - Directory `create`, `delete`, `rename`, `move`, `copy`, and `stat`
- The virtual file system is designed to reduce data transfer and improve performance when your tasks access only part of a large data set, and is not optimized for all workloads. You should test your workload before running production jobs.

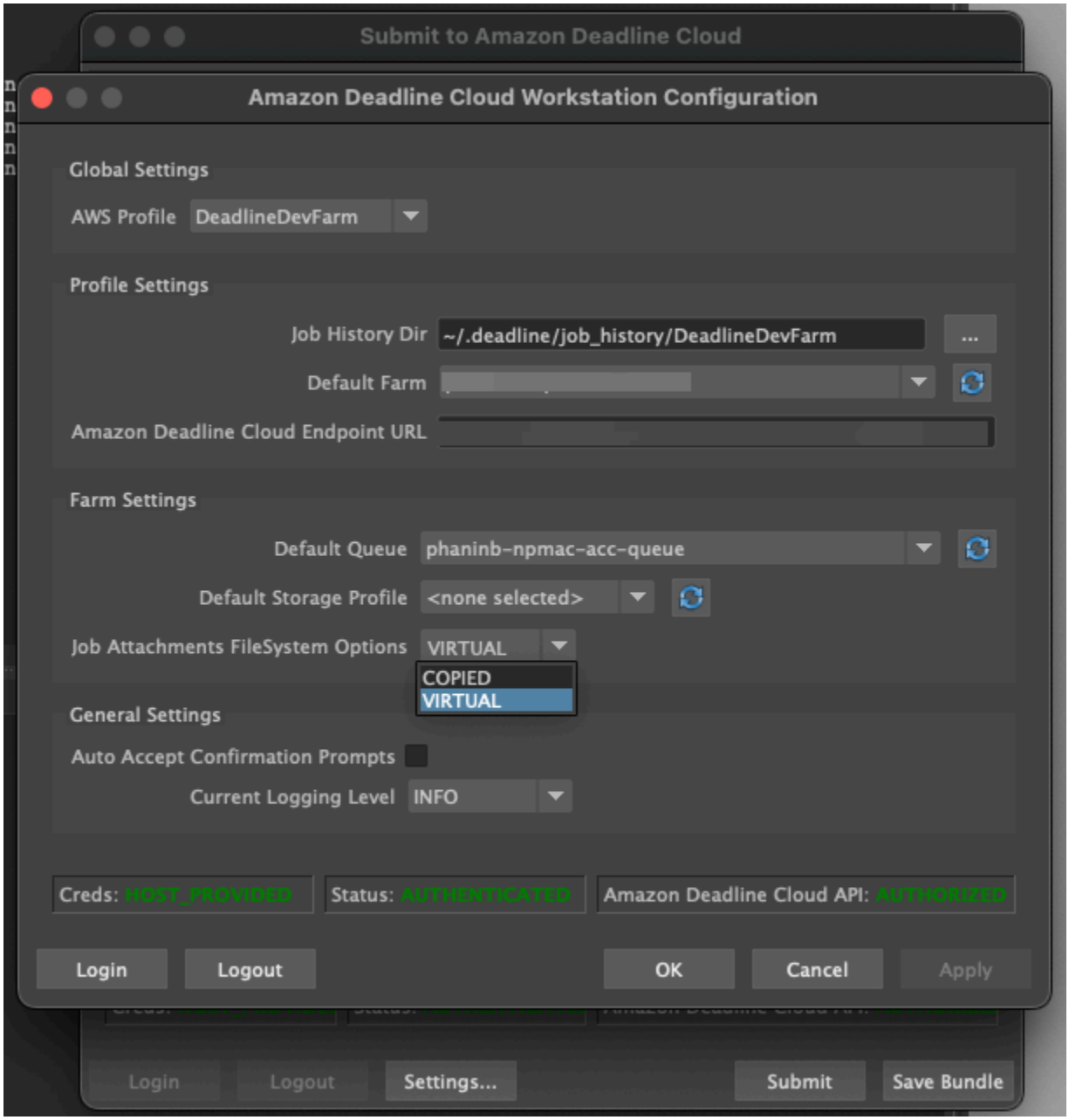
Enable VFS support

Virtual file system support (VFS) is enabled for each job. A job falls back to the default job attachments framework in these cases:

- A worker instance profile does not support a virtual file system.
- Problems prevent launching the virtual file system process.
- The virtual file system can't be mounted.

To enable virtual file system support using the submitter

1. When submitting a job, choose the **Settings** button to open the **AWS Deadline Cloud workstation configuration panel**.
2. From the **Job attachments filesystem options** dropdown, choose **VIRTUAL**.



3. To save your changes, choose **OK**.

To enable virtual file system support using the AWS CLI

- Use the following command when you submit a saved job:

```
deadline bundle submit-job --job-attachments-file-system VIRTUAL
```

To verify that the virtual file system launched successfully for a particular job, review your logs in Amazon CloudWatch Logs. Look for the following messages:

```
Using mount_point mount_point
Launching vfs with command command
Launched vfs as pid PID number
```

If the log contains the following message, virtual file system support is disabled:

```
Virtual File System not found, falling back to COPIED for JobAttachmentsFileSystem.
```

Troubleshooting virtual file system support

You can view logs for your virtual file system using the Deadline Cloud monitor. For instructions, see View logs in Deadline Cloud.

Virtual file system logs are also sent to the CloudWatch Logs group that's associated with the queue shared with the worker agent output.

Track spending and usage for Deadline Cloud farms

The AWS Deadline Cloud budget manager and usage explorer are cost management tools that provide the approximate cost of using Deadline Cloud based on available information about cost variables. The cost management tools don't guarantee the amount owed for your actual use of Deadline Cloud and other AWS services.

To help you manage costs for Deadline Cloud, you can use the following features:

- **Budget manager** – With the Deadline Cloud budget manager, you can create and edit budgets to help manage project costs.
- **Usage explorer** –With the Deadline Cloud usage explorer, you can view how many AWS resources are used and the estimated costs for those resources.

Cost assumptions

The basic calculation used by the Deadline Cloud cost management tools is:

```
Cost per job =
    (CMF run time x CMF compute rate) +
    (SMF run time x SMF compute rate) +
    (License run time x license rate)
```

- *Run time* is the sum of all tasks in a job, from start time to end time.
- *Compute rate* is determined by the AWS Deadline Cloud pricing for service-managed fleets. For customer-managed fleets, the compute rate is estimated to be $1 per worker hour.
- *License rate* is determined by the Deadline Cloud base license price and is only available for service-managed fleets. Additional tiers are not included. For more information about license pricing, see AWS Deadline Cloud pricing.

The cost estimate from the Deadline Cloud cost management tools may vary from your actual costs for a number of reasons. Common reasons include:

- *Customer owned resources and their pricing*. You can choose to bring your own resources, either from AWS or externally from on-premise or other cloud providers. Actual costs of these resources are not calculated.

- *Idle worker costs*. Idle worker costs are not included when the worker status is IDLE. This can happen for fleets with a minimum instance count greater than zero, or when workers transition between jobs. Idle worker cost are not included in calculations.
- *Worker stop and start time*. After workers complete a job, the cost for moving from IDLE to STOPPING and from STOPPING to STOPPED is not included in Deadline Cloud cost estimates.
- *Promotional credits, discounts, and custom pricing agreements*. The cost management tools don't account for promotional credits, private pricing agreements, or other discounts. You may be eligible for other discounts that are not part of the estimate.
- *Asset storage*. Asset storage is not included in the cost and usage estimates.
- *Changes in price*. AWS offers pay-as-you-go pricing for most services. Prices may change over time. The cost management tools use the most up-to-date prices publicly available, but there may be delays after changes.
- *Taxes*. The cost management tools don't include taxes applied to our purchase of the service.
- *Rounding*. The cost management tool perform mathematical rounding of pricing data.
- *Currency*. Cost estimates are made in U.S. dollars. Global exchange rates vary over time. If you translate estimates to a different currency base on the current exchange, changes in the exchange rate affect the estimate.
- *Outside licensing*. If you choose to use pre-purchased licences (Software licensing for service-managed fleets), Deadline Cloud cost management tools can't account for this cost.

Control costs with a budget

The Deadline Cloud budget manager helps you control spending on a given resource, such as a queue, fleet, or farm. You can create budget amounts and limits, and set automated actions to help reduce or stop additional spending against the budget.

The following sections provide you with the steps for using the Deadline Cloud budget manager.

Topics

- Deactivate a budget for a Deadline Cloud queue
- Monitor a budget with EventBridge events

Prerequisite

To use the Deadline Cloud budget manager, you must have `OWNER` access level. To grant `OWNER` permission, follow the steps in Managing users in Deadline Cloud.

Open the Deadline Cloud budget manager

To open the Deadline Cloud budget manager, use the following procedure.

1. Sign in to the AWS Management Console and open the Deadline Cloud console.
2. Choose **View farms**.
3. Locate the farm that you want to get information about, then choose **Manage jobs**.
4. In the Deadline Cloud monitor, in the left navigation pane, choose **Budgets**.

The budget manager summary page displays a list of both active and inactive budgets:

- **Active** budgets track against the selected resource (a queue).
- **Inactive** budgets have either expired or been canceled by a user, and are no longer tracking costs against this budget's limits.

After you choose a budget, the budget summary page contains basic information about the budget. Information provided includes the budget name, status, resources, remaining percentage, remaining amount, total budget, start date, and end date.

Create a budget for a Deadline Cloud queue

To create a budget, use the following procedure.

1. If you haven't already, sign in to the AWS Management Console, open the Deadline Cloud console, choose a farm, and then choose **Manage jobs**.
2. From the **Budget manager** page, choose **Create budget**.
3. In the details section, enter a **Budget name** for the budget.
4. (Optional) In the description field, enter a brief description of the budget.

5. From **Resource**, use the **Queue** dropdown to select the queue that you want to create a budget for.
6. For **Period**, set the start and end date for the budget by completing the following steps:

   a. For **Start date**, enter the first date of the budget tracking in YYYY/MM/DD format, or choose the **calendar** icon and select a **date**.

      The default start date is the date that the budget is created.

   b. For **End date**, enter the last date of the budget tracking in YYYY/MM/DD format or choose the **calendar** icon and select a **date**.

      The default end date is 120 days from the start date.
7. For **Budget amount**, enter the dollar amount of the budget.
8. (Optional) We recommend that you create limit alerts. In the **Limit actions** section, you can implement automated actions that occur when specific amounts remain in the budget. To do this, complete the following steps:

   a. Choose **Add new action**.

   b. For **Remaining amount**, enter the dollar amount that you want to start the action.

   c. In the **Action** dropdown, choose the action that you want. Actions include:

      - **Stop after finishing current work** – All work currently running when the threshold amount is met continue to run (and incur costs) until finished.
      - **Immediately stop work** – All work is canceled immediately when the threshold amount is met.

   d. To create additional limit alerts, choose **Add new action** and repeat the previous steps.
9. Choose **Create budget**.

View a Deadline Cloud queue budget

After you create a budget, you can view the budget on the **Budget manager** page. From there, you can view the budget's total amount and the overall cost allocated to the specific budget.

To view a budget, use the following procedure.

1. If you haven't already, sign in to the AWS Management Console, open the Deadline Cloud console, choose a farm, and then choose **Manage jobs**.

2. Choose **Budgets** from the left side navigation pane. The **Budget Manager** page appears.
3. To view an active budget, choose the **Active budgets** tab, and choose the name of the budget that you want to view. The budget details page appears.
4. To view the budget details for an expired budget, choose the **Inactive budgets** tab. Then, choose the name of the budget that you want to view. The budget details page appears.

Edit a budget for a Deadline Cloud queue

You can edit any active budget. To edit an active budget, use the following procedure.

1. If you haven't already, sign in to the AWS Management Console, open the Deadline Cloud console, choose a farm, and then choose **Manage jobs**.
2. From the **Budget Manager** page, in the **Active budgets** tab, choose the button next to the budget you want to edit.
3. From the **Actions** dropdown menu, select **Edit budget**.
4. Make the changes that you want, and then choose **Update budget**.

Deactivate a budget for a Deadline Cloud queue

You can deactivate any active budget. Deactivating a budget changes its status from **Active** to **Inactive**. When a budget is deactivated, it no longer tracks a resource to that budget's amount.

To deactivate a budget, use the following procedure.

1. If you haven't already, sign in to the AWS Management Console, open the Deadline Cloud console, choose a farm, and then choose **Manage jobs**.
2. From the **Budget manager** page, in the **Active Budgets** tab, choose the button next to the budget that you want to deactivate.
3. From the **Actions** dropdown menu, select **Deactivate budget**. In a few moments, the selected budget will change from **Active** to **Inactive** and will move from the **Active Budgets** tab to the **Inactive Budgets** tab.

Monitor a budget with EventBridge events

Deadline Cloud sends budget-related events, using Amazon EventBridge, to your default EventBridge event bus. You can create custom functions that receive the events and act on them to

send notifications to automatically notify users via email, Slack, or other channels when a budget reaches predefined levels. For example, you can send SMS messages when a budget reaches a certain threshold. This helps you stay on top of your spending and make informed decisions before your budget is exhausted.

Deadline Cloud periodically aggregates usage and cost data for each render farm. Then it checks to see if any of the budget thresholds has been crossed. If a threshold is crossed, Deadline Cloud triggers an event to alert you so that you can take the appropriate action. An event is triggered whenever a budget crosses one of these thresholds, specified in percent of the budget used:

- 10, 20, 30, 40, 50, 60, 70, 75, 80, 85, 90, 95, 96, 97, 98, 99, 100

The budget usage thresholds get closer together as a budget approaches 100 percent usage. This helps you closely monitor usage as the budget reaches its limit. You can also set your own budget thresholds. Deadline Cloud sends an event when usage passes your custom thresholds. After your budget reaches 100 percent, Deadline Cloud stops sending events. If you adjust your budget, Deadline Cloud sends events for your thresholds based on the new budget amount.

You can use the EventBridge console (https://console.aws.amazon.com/events/) to create rules to send the Deadline Cloud events to the appropriate target for the event. For example, you can send the event to an Amazon Simple Queue Service queue and from there to multiple targets, such as AWS End User Messaging SMS or a Amazon Relational Database Service database for logging.

For examples of an EventBridge rule, see the following topics:

- Send an email when events happen using Amazon EventBridge.
- Creating an Amazon EventBridge rule that sends notifications to AWS Chatbot.
- Getting started with Amazon EventBridge.

For more information about budget events, see the Budget Threshold Reached event in the *Deadline Cloud Developer Guide*.

Track usage and costs with the Deadline Cloud usage explorer

With the Deadline Cloud usage explorer, you can see real-time metrics on the activity happening on each farm. You can look at the farm's costs by different variables, such as queue, job, license product, or instance types. Select various time frames to see usage during a specific period of

time, and look at usage trends over the course of time. You can also see a detailed breakdown of selected data points, allowing for a closer look into metrics. Usage can be shown by time (minutes and hours) or by cost ($USD).

The following sections show you the steps for accessing and using the Deadline Cloud usage explorer.

Topics

Prerequisite

To use the Deadline Cloud usage explorer, you must have either `MANAGER` or `OWNER` farm permissions. For more information, see Manage users and groups for farms, queues, and fleets.

Open the usage explorer

To open the Deadline Cloud usage explorer, use the following procedure.

1. Sign in to the AWS Management Console and open the Deadline Cloud console.
2. To see all available farms, choose **View farms**.
3. Locate the farm that you want to get information about, then choose **Manage jobs**. The Deadline Cloud monitor opens in a new tab.
4. In the Deadline Cloud monitor, from the left menu, select **Usage explorer**.

Use the usage explorer

From the usage explorer page, you can select specific parameters in which the data can be displayed. By default, you see total usage in time (hours and minutes) within the last 7 days. You can change these parameters, and the information displayed changes dynamically in accordance to the parameter settings.

You can group the results based on the queue, job, compute usage, instance type, or license product. If you choose license product, costs are calculated for specific licenses. For all other groups the time is calculated by adding up the time taken for each task to run.

The usage explorer returns only 100 results based on the filter criteria that you set. The results are listed in descending order by the date created timestamp. If there are more than 100 results, you get an error message. You can refine your query to reduce the number of results:

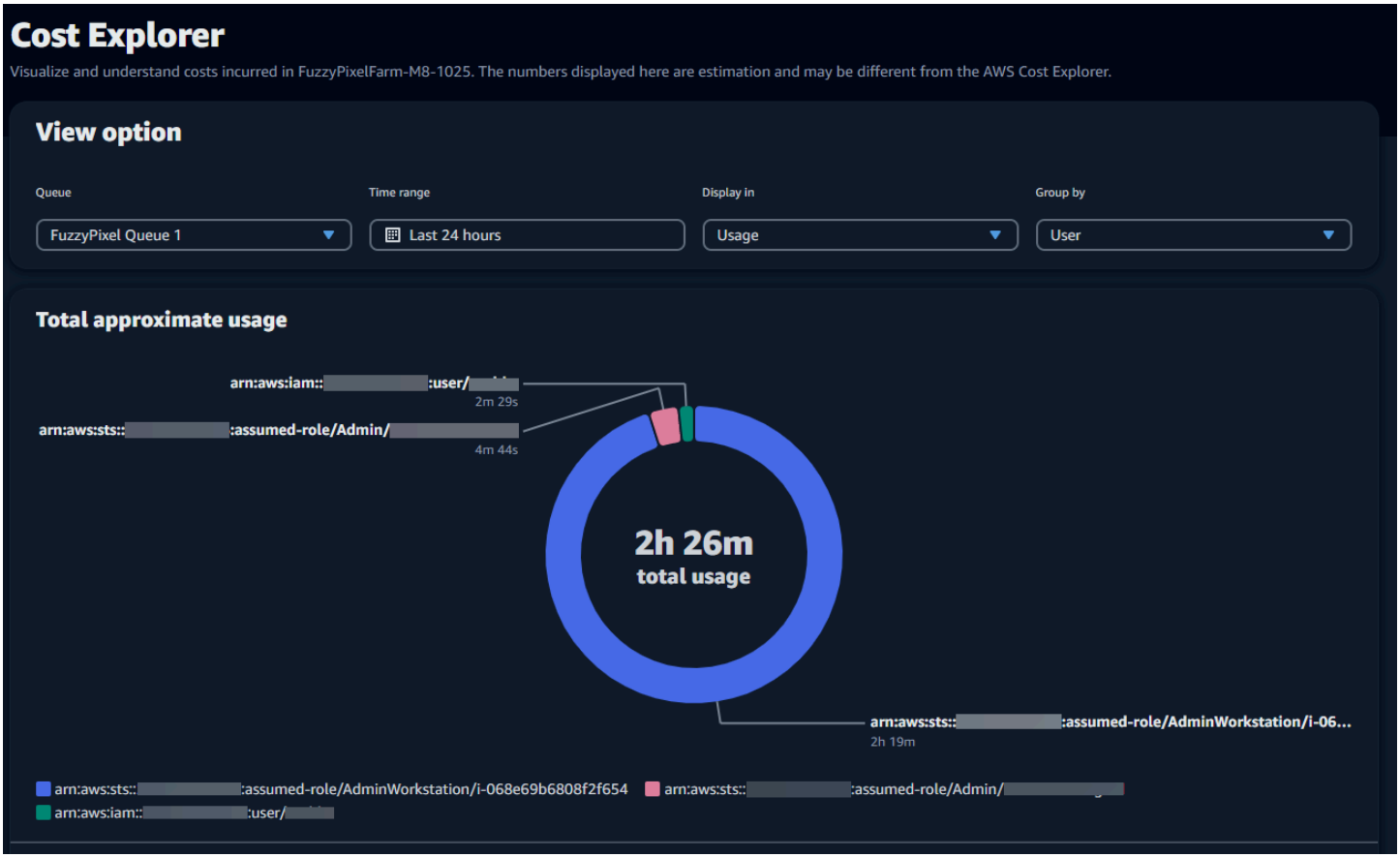
- Select a smaller time range
- Select fewer queues
- Select a different grouping, such as grouping by queue instead of job

Topics

Use visual graphs to review data

You can review data in a visual format to identify trends and potential areas that might need more analysis or attention. Usage explorer offers a pie chart that displays overall usage and cost with the option to group the totals into smaller subtotals.

> **Note**
>
> The chart *only* displays the top five results with other results combined in an "others" section. You can view all results in the breakdown section below the chart.



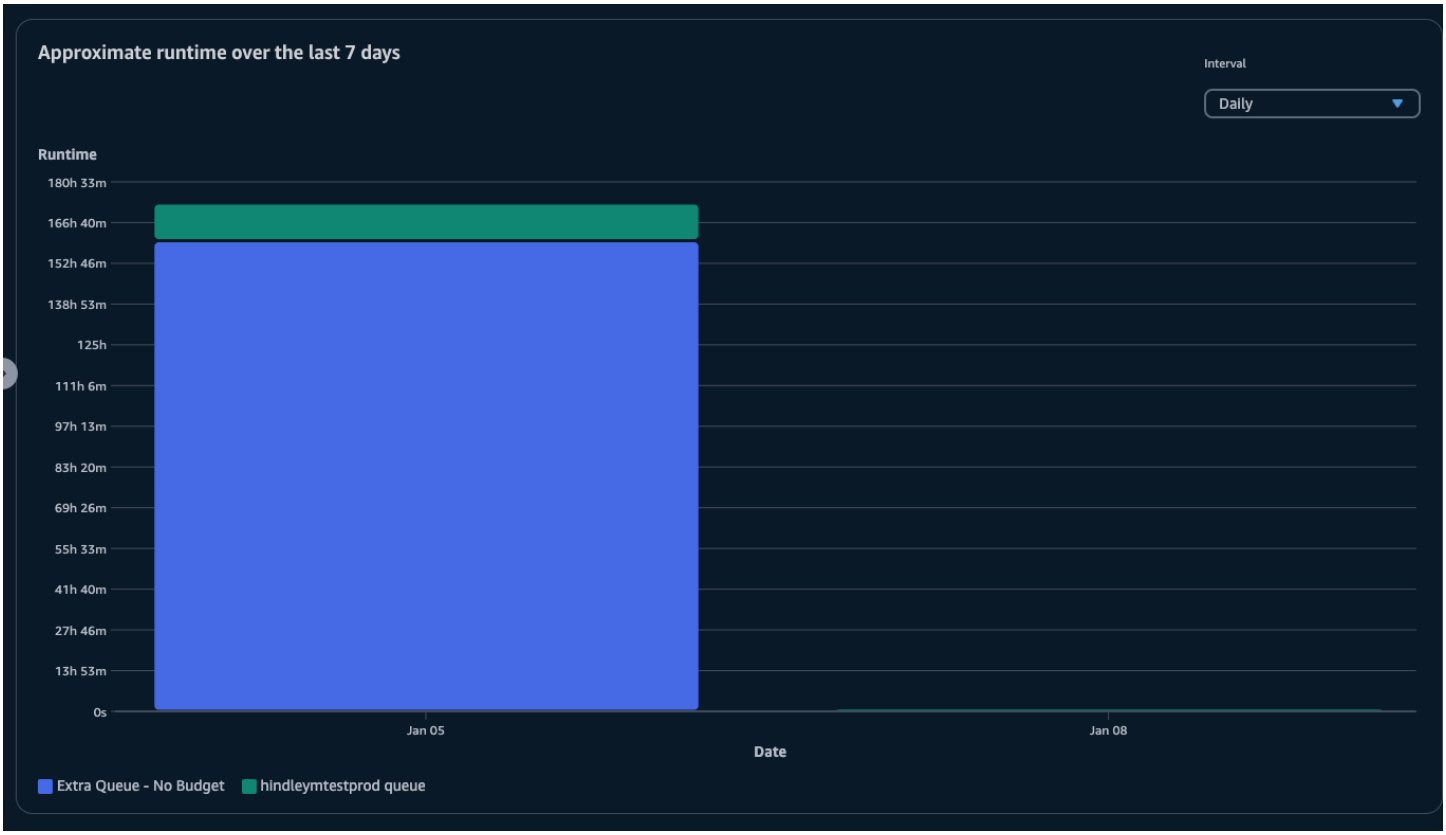
View a breakdown of metrics

Beneath the pie chart, usage explorer offers a more detailed breakdown of specific metrics, which will change as parameters change. By default, five results display in the usage explorer. You can scroll through results using the pagination arrows in the breakdown section.

Breakdown is minimized by default. To expand and display the results, select the **View all breakdown** arrow. To download the breakdown, choose **Download data**.

View approximate runtime of queues

You can also view the approximate runtime of your queues based on different intervals that you specify. The interval options are hourly, daily, weekly, and monthly. After you select an interval, the graph displays the approximate runtime of your queues.



Cost management

AWS Deadline Cloud provides budgets and the usage explorer to help you control and visualize costs for your jobs. However, Deadline Cloud uses other AWS services, such as Amazon S3. Costs for those services are not reflected in Deadline Cloud budgets or the usage explorer and are charged separately based on usage. Depending on how you configure Deadline Cloud, you may use the following AWS services, as well as others:

Service	Pricing page
Amazon CloudWatch Logs	Amazon CloudWatch Logs pricing
Amazon Elastic Compute Cloud	Amazon Elastic Compute Cloud pricing
AWS Key Management Service	AWS Key Management Service pricing
AWS PrivateLink	AWS PrivateLink pricing
Amazon Simple Storage Service	Amazon Simple Storage Service pricing

Service	Pricing page
Amazon Virtual Private Cloud	Amazon Virtual Private Cloud pricing

Cost management best practices

Using the following best practices can help you understand and control your costs when using Deadline Cloud and the tradeoffs you can make between cost and efficiency.

> **Note**
>
> The final cost of using Deadline Cloud depends on the interaction between a number of AWS services, the amount of work that you process, and the AWS Region where you run your jobs. The following best practices are guidelines and may not significantly reduce costs.

Best practices for CloudWatch Logs

Deadline Cloud sends worker and task logs to CloudWatch Logs. You are charged to collect, store, and analyze these logs. You can reduce costs by logging only the minimum amount of data required to monitor your tasks.

When you create a queue or fleet, Deadline Cloud creates a CloudWatch Logs log group with the following names:

- `/aws/deadline/<FARM_ID>/<FLEET_ID>`
- `/aws/deadline/<FARM_ID>/<QUEUE_ID>`

By default, these logs never expire. You can adjust the retention policy of log groups to remove old logs and help reduce storage costs. You can also export logs to Amazon S3. Amazon S3 storage costs are lower than those for CloudWatch. For more information, see Exporting log data to Amazon S3.

Best practices for Amazon EC2

You can use Amazon EC2 instances for both service-managed and customer-managed fleets. There are three considerations:

- For service-managed fleets, you can choose to have one or more instances available at all times by setting the minimum worker count for the fleet. When you set the minimum worker count above 0, the fleet always has this many workers running. This can reduce the amount of time that it takes for Deadline Cloud to start processing jobs, however you are charged for the instance's idle time.
- For service-managed fleets, set a maximum size for the fleet. This limits the number of instances that a fleet can auto scale to. Fleets won't grow past this size even if there are more jobs waiting to be processed.
- For both service-managed and customer-managed fleets, you can specify the Amazon EC2 instance types in your fleets. Using smaller instances costs less per minute, but may take longer to complete a job. Conversely, a larger instance costs more per minute, but can reduce the time to complete a job. Understanding the demands that your jobs place on an instance can help reduce your costs.
- When possible, choose Amazon EC2 Spot instances for your fleet. Spot instances are available for a reduced price, but may be interrupted by on-demand requests. On-demand instances are charged by the second and are not interrupted.

Best practices for AWS KMS

By default, Deadline Cloud encrypts you data with an AWS owned key. You are not charged for this key.

You may choose to use a customer managed key to encrypt your data. When you use your own key, you are charged based on how your key is used. If you use an existing key, this will be an incremental cost for the additional use.

Best practices for AWS PrivateLink

You can use AWS PrivateLink to create a connection between your VPC and Deadline Cloud using an interface endpoint. When you create a connection, you can call all of the Deadline Cloud API actions. You are charged per hour for each endpoint that you create. If you use PrivateLink, you must create at least three endpoints, and depending on your configuration, you may need as many as five.

Best practices for Amazon S3

Deadline Cloud uses Amazon S3 to store assets for processing, job attachments, output, and logs. To reduce the costs associated with Amazon S3, reduce the amount of data that you store. Some suggestions:

- Only store assets that are currently in use or that will be used shortly.
- Use an S3 Lifecycle configuration to automatically delete unused files from an S3 bucket.

Best practices for Amazon VPC

When you use usage-based licensing for your customer-managed fleet, you create a Deadline Cloud license endpoint, which is a Amazon VPC endpoint created in your account. This endpoint is charged at an hourly rate. To reduce costs, remove the endpoints when you are not using usage-based licenses.

Security in Deadline Cloud

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to AWS Deadline Cloud, see AWS services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Deadline Cloud. The following topics show you how to configure Deadline Cloud to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Deadline Cloud resources.

Topics

Data protection in Deadline Cloud

The AWS shared responsibility model applies to data protection in AWS Deadline Cloud. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see Working with CloudTrail trails in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with Deadline Cloud or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

The data entered into name fields in Deadline Cloud job templates may also be included in billing or diagnostic logs and should not contain confidential or sensitive information.

Topics

Encryption at rest

AWS Deadline Cloud protects sensitive data by encrypting it at rest using encryption keys stored in AWS Key Management Service (AWS KMS). Encryption at rest is available in all AWS Regions where Deadline Cloud is available.

Encrypting data means sensitive data saved on disks isn't readable by a user or application without a valid key. Only a party with a valid managed key can decrypt the data.

For information about how Deadline Cloud uses AWS KMS for encrypting data at rest, see Key management.

Encryption in transit

For data in transit, AWS Deadline Cloud uses Transport Layer Security (TLS) 1.2 or 1.3 to encrypt data sent between the service and workers. We require TLS 1.2 and recommend TLS 1.3. Additionally, if you use a virtual private cloud (VPC), you can use AWS PrivateLink to establish a private connection between your VPC and Deadline Cloud.

Key management

When creating a new farm, you can choose one of the following keys to encrypt your farm data:

- **AWS owned KMS key** – Default encryption type if you don't specify a key when you create the farm. The KMS key is owned by AWS Deadline Cloud. You can't view, manage, or use AWS owned keys. However, you don't need to take any action to protect the keys that encrypt your data. For more information, see AWS owned keys in the *AWS Key Management Service developer guide*.
- **Customer managed KMS key** – You specify a customer managed key when you create a farm. All of the content within the farm is encrypted with the KMS key. The key is stored in your account

and is created, owned, and managed by you and AWS KMS charges apply. You have full control over the KMS key. You can perform such tasks as:

- Establishing and maintaining key polices
- Establishing and maintaining IAM policies and grants
- Enabling and disabling key policies
- Adding tags
- Creating key aliases

You can't manually rotate a customer owned key used with a Deadline Cloud farm. Automatic rotation of the key is supported.

For more information, see Customer owned keys in the *AWS Key Management Service Developer Guide*.

To create a customer managed key, follow the steps for Creating symmetric customer managed keys in the *AWS Key Management Service Developer Guide*.

How Deadline Cloud use AWS KMS grants

Deadline Cloud requires a grant to use your customer managed key. When you create a farm encrypted with a customer managed key, Deadline Cloud creates a grant on your behalf by sending a `CreateGrant` request to AWS KMS to get access to the KMS key that you specified.

Deadline Cloud uses multiple grants. Each grant is used by a different part of Deadline Cloud that needs to encrypt or decrypt your data. Deadline Cloud also uses grants to allow access to other AWS services used to store data on your behalf, such as Amazon Simple Storage Service, Amazon Elastic Block Store, or OpenSearch.

Grants that enable Deadline Cloud to manage machines in a service-managed fleet include a Deadline Cloud account number and role in the `GranteePrincipal` instead of a service principal. While not typical, this is necessary to encrypt Amazon EBS volumes for workers in service-managed fleets using the customer managed KMS key specified for the farm.

Customer managed key policy

Key policies control access to your customer managed key. Each key must have exactly one key policy that contains statements that determine who can use the key and how they can use it. When

you create you customer managed key, you can specify a key policy. For more information, see Managing access to customer managed keys in the *AWS Key Management Service Developer Guide*.

Minimal IAM policy for CreateFarm

To use your customer managed key to create farms using the console or the `CreateFarm` API operation, the following AWS KMS API operations must be permitted:

- `kms:CreateGrant` – Adds a grant to a customer managed key. Grants console access to a specified AWS KMS key. For more informations, see Using grants in the *AWS Key Management Service developer guide*.
- `kms:Decrypt` – Allows Deadline Cloud to decrypt data in the farm.
- `kms:DescribeKey` – Provides the customer managed key details to allow Deadline Cloud to validate the key.
- `kms:GenerateDataKey` – Allows Deadline Cloud to encrypt data using a unique data key.

The following policy statement grants the necessary permissions for the `CreateFarm` operation.

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

Minimal IAM policy for read-only operations

To use your customer managed key for read-only Deadline Cloud operations, such getting information about farms, queues, and fleets. The following AWS KMS API operations must be permitted:

- kms:Decrypt – Allows Deadline Cloud to decrypt data in the farm.
- kms:DescribeKey – Provides the customer managed key details to allow Deadline Cloud to validate the key.

The following policy statement grants the necessary permissions for read-only operations.

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

Minimal IAM policy for read-write operations

To use your customer managed key for read-write Deadline Cloud operations, such as creating and updating farms, queues, and fleets. The following AWS KMS API operations must be permitted:

- kms:Decrypt – Allows Deadline Cloud to decrypt data in the farm.
- kms:DescribeKey – Provides the customer managed key details to allow Deadline Cloud to validate the key.

- kms:GenerateDataKey – Allows Deadline Cloud to encrypt data using a unique data key.

The following policy statement grants the necessary permissions for the CreateFarm operation.

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

Monitoring your encryption keys

When you use an AWS KMS customer managed key with your Deadline Cloud farms, you can use AWS CloudTrail or Amazon CloudWatch Logs to track requests that Deadline Cloud sends to AWS KMS.

CloudTrail event for grants

The following example CloudTrail event occurs when grants are created, typically when you call the CreateFarm, CreateMonitor, or CreateFleet operation.

```
{
    "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-abcdef1234567890098765432 1fedcba",
                "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 event for decryption

The following example CloudTrail event occurs when decrypting values using the customer managed KMS key.

```
{
    "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 event for encryption

The following example CloudTrail event occurs when encrypting values using the customer managed KMS key.


```
{
    "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"
}
```


Deleting a customer managed KMS key

Deleting a customer managed KMS key in AWS Key Management Service (AWS KMS) is destructive and potentially dangerous. It irreversibly deletes the key material and all metadata associated with the key. After a customer managed KMS key is deleted, you can no longer decrypt the data that was encrypted by that key. This means that the data becomes unrecoverable.

This is why AWS KMS gives customers a waiting period of up to 30 days before deleting the KMS key. The default waiting period is 30 days.

About the waiting period

Because it's destructive and potentially dangerous to delete a customer managed KMS key, we require that you set a waiting period of 7–30 days. The default waiting period is 30 days.

However, the actual waiting period might be up to 24 hours longer than the period you scheduled. To get the actual date and time when the key will be deleted, use the DescribeKey operation. You can also see the scheduled deletion date of a key in the AWS KMS console on the key's detail page, in the **General configuration** section. Notice the time zone.

During the waiting period, the customer managed key's status and key state is **Pending deletion**.

- A customer managed KMS key that is pending deletion can't be used in any cryptographic operations.
- AWS KMS doesn't rotate the backing keys of customer managed KMS keys that are pending deletion.

For more information about deleting a customer managed KMS key, see Deleting customer master keys in the *AWS Key Management Service Developer Guide*.

Inter-network traffic privacy

AWS Deadline Cloud supports Amazon Virtual Private Cloud (Amazon VPC) to secure connections. Amazon VPC provides features that you can use to increase and monitor the security for your virtual private cloud (VPC).

You can set up a customer-managed fleet (CMF) with Amazon Elastic Compute Cloud (Amazon EC2) instances that run inside a VPC. By deploying Amazon VPC endpoints to use AWS PrivateLink, traffic between workers in your CMF and the Deadline Cloud endpoint stays within your VPC. Furthermore, you can configure your VPC to restrict internet access to your instances.

In service-managed fleets, workers aren't reachable from the internet, but they do have internet access and connect to the Deadline Cloud service over the internet.

Opt out

AWS Deadline Cloud collects certain operational information to help us develop and improve Deadline Cloud. The collected data includes things such as your AWS account ID and user ID, so that we can correctly identify you if you have an issue with the Deadline Cloud. We also collect Deadline Cloud specific information, such as Resource IDs (a FarmID or QueueID when applicable), the product name (for example, JobAttachments, WorkerAgent, and more) and the product version.

You can choose to opt out from this data collection using application configuration. Each computer interacting with Deadline Cloud, both client workstations and fleet workers, needs to opt out separately.

Deadline Cloud monitor - desktop

Deadline Cloud monitor - desktop collects operational information, such as when crashes occur and when the application is opened, to help us know when you are having problems with the

application. To opt out from the collection of this operational information, go to the settings page and clear **Turn on data collection to measure Deadline Cloud Monitor's performance**.

After you opt out, the desktop monitor no longer sends the operational data. Any previously collected data is retained and may still be used to improve the service. For more information, see Data Privacy FAQ.

AWS Deadline Cloud CLI and Tools

The AWS Deadline Cloud CLI, submitters, and worker agent all collect operational information such as when crashes occur and when jobs are submitted to help us know when you are having problems with these applications. To opt out from the collection of this operational information, use any of the following methods:

- In the terminal, enter **`deadline config set telemetry.opt_out true`**.

  This will opt out the CLI, submitters, and worker agent when running as the current user.
- When installing the Deadline Cloud worker agent, add the **`--telemetry-opt-out`** command line argument. For example, **`./install.sh --farm-id $FARM_ID --fleet-id $FLEET_ID --telemetry-opt-out`**.
- Before running the worker agent, CLI, or submitter, set an environment variable: **`DEADLINE_CLOUD_TELEMETRY_OPT_OUT=true`**

After you opt out, the Deadline Cloud tools no longer send the operational data. Any previously collected data is retained and may still be used to improve the service. For more information, see Data Privacy FAQ.

Identity and Access Management in Deadline Cloud

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Deadline Cloud resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Authenticating with identities
- Managing access using policies
- How Deadline Cloud works with IAM
- Identity-based policy examples for Deadline Cloud
- AWS managed policies for Deadline Cloud
- Troubleshooting AWS Deadline Cloud identity and access

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in Deadline Cloud.

Service user – If you use the Deadline Cloud service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Deadline Cloud features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Deadline Cloud, see Troubleshooting AWS Deadline Cloud identity and access.

Service administrator – If you're in charge of Deadline Cloud resources at your company, you probably have full access to Deadline Cloud. It's your job to determine which Deadline Cloud features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Deadline Cloud, see How Deadline Cloud works with IAM.

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Deadline Cloud. To view example Deadline Cloud identity-based policies that you can use in IAM, see Identity-based policy examples for Deadline Cloud.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on

authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the *AWS IAM Identity Center User Guide* and AWS Multi-factor authentication in IAM in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A *federated identity* is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. To temporarily assume an IAM role in the AWS Management Console, you can switch from a user to an IAM role (console). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Methods to assume a role in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Create a role for a third-party identity provider

(federation) in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center User Guide*.

- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.
  - **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile

that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Use an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choose between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to

any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see Service control policies in the *AWS Organizations User Guide*.

- **Resource control policies (RCPs)** – RCPs are JSON policies that you can use to set the maximum available permissions for resources in your accounts without updating the IAM policies attached to each resource that you own. The RCP limits permissions for resources in member accounts and can impact the effective permissions for identities, including the AWS account root user, regardless of whether they belong to your organization. For more information about Organizations and RCPs, including a list of AWS services that support RCPs, see Resource control policies (RCPs) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How Deadline Cloud works with IAM

Before you use IAM to manage access to Deadline Cloud, learn what IAM features are available to use with Deadline Cloud.

IAM features you can use with AWS Deadline Cloud

IAM feature	Deadline Cloud support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes

IAM feature	Deadline Cloud support
Policy resources	Yes
Policy condition keys (service-specific)	Yes
ACLs	No
ABAC (tags in policies)	Yes
Temporary credentials	Yes
Forward access sessions (FAS)	Yes
Service roles	Yes
Service-linked roles	No

To get a high-level view of how Deadline Cloud and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for Deadline Cloud

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for Deadline Cloud

To view examples of Deadline Cloud identity-based policies, see Identity-based policy examples for Deadline Cloud.

Resource-based policies within Deadline Cloud

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Policy actions for Deadline Cloud

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

To see a list of Deadline Cloud actions, see Actions defined by AWS Deadline Cloud in the *Service Authorization Reference*.

Policy actions in Deadline Cloud use the following prefix before the action:

```
deadline
```

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [
      "deadline:action1",
      "deadline:action2"
         ]
```

To view examples of Deadline Cloud identity-based policies, see Identity-based policy examples for Deadline Cloud.

Policy resources for Deadline Cloud

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To see a list of Deadline Cloud resource types and their ARNs, see Resources defined by AWS Deadline Cloud in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions defined by AWS Deadline Cloud.

To view examples of Deadline Cloud identity-based policies, see Identity-based policy examples for Deadline Cloud.

Policy condition keys for Deadline Cloud

Supports service-specific policy condition keys: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To see a list of Deadline Cloud condition keys, see Condition keys for AWS Deadline Cloud in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see Actions defined by AWS Deadline Cloud.

To view examples of Deadline Cloud identity-based policies, see Identity-based policy examples for Deadline Cloud.

ACLs in Deadline Cloud

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with Deadline Cloud

Supports ABAC (tags in policies): Yes

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with Deadline Cloud

Supports temporary credentials: Yes

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see AWS services that work with IAM in the *IAM User Guide*.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then switch roles. For more information about switching roles, see Switch from a user to an IAM role (console) in the *IAM User Guide*.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM.

Forward access sessions for Deadline Cloud

Supports forward access sessions (FAS): Yes

When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.

Service roles for Deadline Cloud

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break Deadline Cloud functionality. Edit service roles only when Deadline Cloud provides guidance to do so.

Service-linked roles for Deadline Cloud

Supports service-linked roles: No

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a Yes in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for Deadline Cloud

By default, users and roles don't have permission to create or modify Deadline Cloud resources. They also can't perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Create IAM policies (console) in the *IAM User Guide*.

For details about actions and resource types defined by Deadline Cloud, including the format of the ARNs for each of the resource types, see Actions, resources, and condition keys for AWS Deadline Cloud in the *Service Authorization Reference*.

Topics

Policy best practices

Identity-based policies determine whether someone can create, access, or delete Deadline Cloud resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on

specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.

- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Using the Deadline Cloud console

To access the AWS Deadline Cloud console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the Deadline Cloud resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can still use the Deadline Cloud console, also attach the Deadline Cloud `ConsoleAccess` or `ReadOnly` AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*.

Policy to submit jobs to a queue

In this example, you create a scoped-down policy that grants permission to submit jobs to a specific queue in a specific farm.

```
{
    "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/*"
        }
    ]
}
```

Policy to allow creating a license endpoint

In this example, you create a scoped-down policy that grants the required permissions to create and manage license endpoints. Use this policy to create the license endpoint for the VPC associated with your farm.

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

Policy to allow monitoring a specific farm queue

In this example, you create a scoped-down policy that grants permission to monitor jobs in a specific queue for a specific farm.

```
{
    "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 managed policies for Deadline Cloud

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the *IAM User Guide*.

AWS managed policy: AWSDeadlineCloud-FleetWorker

You can attach the `AWSDeadlineCloud-FleetWorker` policy to your AWS Identity and Access Management (IAM) identities.

This policy grants workers in this fleet the permissions that are needed to connect to and receive tasks from the service.

Permissions details

This policy includes the following permissions:

- `deadline` – Allows principals to manage workers in a fleet.

For a JSON listing of the policy details, see AWSDeadlineCloud-FleetWorker in the *AWS Managed Policy reference guide*.

AWS managed policy: AWSDeadlineCloud-WorkerHost

You can attach the `AWSDeadlineCloud-WorkerHost` policy to your IAM identities.

This policy grants the permissions that are needed to initially connect to the service. It can be used as an Amazon Elastic Compute Cloud (Amazon EC2) instance profile.

Permissions details

This policy includes the following permissions:

- `deadline` – Allows principals to create workers.

For a JSON listing of the policy details, see AWSDeadlineCloud-WorkerHost in the *AWS Managed Policy reference guide*.

AWS managed policy: AWSDeadlineCloud-UserAccessFarms

You can attach the `AWSDeadlineCloud-UserAccessFarms` policy to your IAM identities.

This policy allows users to access farm data based on the farms that they are members of and their membership level.

Permissions details

This policy includes the following permissions:

- `deadline` – Allows the user to access farm data.
- `ec2` – Allows users to see details about Amazon EC2 instance types.
- `identitystore` – Allows users to see user and group names.

For a JSON listing of the policy details, see AWSDeadlineCloud-UserAccessFarms in the *AWS Managed Policy reference guide*.

AWS managed policy: AWSDeadlineCloud-UserAccessFleets

You can attach the `AWSDeadlineCloud-UserAccessFleets` policy to your IAM identities.

This policy allows users to access fleet data based on the farms that they are members of and their membership level.

Permissions details

This policy includes the following permissions:

- `deadline` – Allows the user to access farm data.
- `ec2` – Allows users to see details about Amazon EC2 instance types.
- `identitystore` – Allows users to see user and group names.

For a JSON listing of the policy details, see AWSDeadlineCloud-UserAccessFleets in the *AWS Managed Policy reference guide*.

AWS managed policy: AWSDeadlineCloud-UserAccessJobs

You can attach the `AWSDeadlineCloud-UserAccessJobs` policy to your IAM identities.

This policy allows users to access job data based on the farms that they are members of and their membership level.

Permissions details

This policy includes the following permissions:

- `deadline` – Allows the user to access farm data.
- `ec2` – Allows users to see details about Amazon EC2 instance types.
- `identitystore` – Allows users to see user and group names.

For a JSON listing of the policy details, see AWSDeadlineCloud-UserAccessJobs in the *AWS Managed Policy reference guide*.

AWS managed policy: AWSDeadlineCloud-UserAccessQueues

You can attach the `AWSDeadlineCloud-UserAccessQueues` policy to your IAM identities.

This policy allows users to access queue data based on the farms that they are members of and their membership level.

Permissions details

This policy includes the following permissions:

- `deadline` – Allows the user to access farm data.

- `ec2` – Allows users to see details about Amazon EC2 instance types.
- `identitystore` – Allows users to see user and group names.

For a JSON listing of the policy details, see AWSDeadlineCloud-UserAccessQueues in the *AWS Managed Policy reference guide*.

Deadline Cloud updates to AWS managed policies

View details about updates to AWS managed policies for Deadline Cloud since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Deadline Cloud Document history page.

Change	Description	Date
AWSDeadlineCloud-UserAccessFarms – Change AWSDeadlineCloud-UserAccessJobs – Change AWSDeadlineCloud-UserAccessQueues – Change	Deadline Cloud added new actions `deadline:GetJobTemplate` and `deadline:ListJobParameterDefinitions` to allow you to resubmit jobs.	October 7, 2024
Deadline Cloud started tracking changes	Deadline Cloud started tracking changes to its AWS managed policies.	April 2, 2024

Troubleshooting AWS Deadline Cloud identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with Deadline Cloud and IAM.

Topics

- I am not authorized to perform an action in Deadline Cloud

- I am not authorized to perform iam:PassRole
- I want to allow people outside of my AWS account to access my Deadline Cloud resources

I am not authorized to perform an action in Deadline Cloud

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `deadline:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 deadline:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `deadline:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to Deadline Cloud.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Deadline Cloud. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my Deadline Cloud resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Deadline Cloud supports these features, see How Deadline Cloud works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

Compliance validation for Deadline Cloud

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security Compliance & Governance – These solution implementation guides discuss architectural considerations and provide steps for deploying security and compliance features.
- HIPAA Eligible Services Reference – Lists HIPAA eligible services. Not all AWS services are HIPAA eligible.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Customer Compliance Guides – Understand the shared responsibility model through the lens of compliance. The guides summarize the best practices for securing AWS services and map the guidance to security controls across multiple frameworks (including National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), and International Organization for Standardization (ISO)).
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see Security Hub controls reference.
- Amazon GuardDuty – This AWS service detects potential threats to your AWS accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.
- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in Deadline Cloud

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

AWS Deadline Cloud does not back up data stored in your job attachments S3 bucket. You can enable backups of your job attachments data using any standard Amazon S3 backup mechanism, such as S3 Versioning or AWS Backup.

Infrastructure security in Deadline Cloud

As a managed service, AWS Deadline Cloud is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access Deadline Cloud through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Deadline Cloud doesn't support using AWS PrivateLink virtual private cloud (VPC) endpoint policies. It uses the AWS PrivateLink default policy, which grants full access to the endpoint. For more information, see Default endpoint policy in the *AWS PrivateLink user guide*.

Configuration and vulnerability analysis in Deadline Cloud

AWS handles basic security tasks like guest operating system (OS) and database patching, firewall configuration, and disaster recovery. These procedures have been reviewed and certified by the appropriate third parties. For more details, see the following resources:

- Shared Responsibility Model
- Amazon Web Services: Overview of Security Processes (whitepaper)

AWS Deadline Cloud manages tasks on service-managed or customer-managed fleets:

- For service-managed fleets, Deadline Cloud manages the guest operating system.
- For customer-managed fleets, you are responsible for managing the operating system.

For additional information about configuration and vulnerability analysis for AWS Deadline Cloud, see

- Security best practices for Deadline Cloud

Cross-service confused deputy prevention

The confused deputy problem is a security issue where an entity that doesn't have permission to perform an action can coerce a more-privileged entity to perform the action. In AWS, cross-service impersonation can result in the confused deputy problem. Cross-service impersonation can occur when one service (the *calling service*) calls another service (the *called service*). The calling service can be manipulated to use its permissions to act on another customer's resources in a way it should not otherwise have permission to access. To prevent this, AWS provides tools that help you protect your data for all services with service principals that have been given access to resources in your account.

We recommend using the aws:SourceArn and aws:SourceAccount global condition context keys in resource policies to limit the permissions that AWS Deadline Cloud gives another service to the resource. Use `aws:SourceArn` if you want only one resource to be associated with the cross-service access. Use `aws:SourceAccount` if you want to allow any resource in that account to be associated with the cross-service use.

The most effective way to protect against the confused deputy problem is to use the `aws:SourceArn` global condition context key with the full Amazon Resource Name (ARN) of the resource. If you don't know the full ARN of the resource or if you are specifying multiple resources, use the `aws:SourceArn` global context condition key with wildcard characters (*) for the unknown portions of the ARN. For example, `arn:aws:deadline:*:123456789012:*`.

If the `aws:SourceArn` value does not contain the account ID, such as an Amazon S3 bucket ARN, you must use both global condition context keys to limit permissions.

The following example shows how you can use the `aws:SourceArn` and `aws:SourceAccount` global condition context keys in Deadline Cloud to prevent the confused deputy problem.

```
{
```

```
  "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"
      }
    }
  }
}
```

Access AWS Deadline Cloud using an interface endpoint (AWS PrivateLink)

You can use AWS PrivateLink to create a private connection between your VPC and AWS Deadline Cloud. You can access Deadline Cloud as if it were in your VPC, without the use of an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to access Deadline Cloud.

You establish this private connection by creating an *interface endpoint*, powered by AWS PrivateLink. We create an endpoint network interface in each subnet that you enable for the interface endpoint. These are requester-managed network interfaces that serve as the entry point for traffic destined for Deadline Cloud.

For more information, see Access AWS services through AWS PrivateLink in the *AWS PrivateLink Guide*.

Considerations for Deadline Cloud

Before you set up an interface endpoint for Deadline Cloud, see Access an AWS service using an interface VPC endpoint in the *AWS PrivateLink Guide*.

Deadline Cloud supports making calls to all of its API actions through the interface endpoint.

By default, full access to Deadline Cloud is allowed through the interface endpoint. Alternatively, you can associate a security group with the endpoint network interfaces to control traffic to Deadline Cloud through the interface endpoint.

Deadline Cloud doesn't support VPC endpoint policies. For more information, see Control access to VPC endpoints using endpoint policies in the *AWS PrivateLink Guide*.

Deadline Cloud endpoints

Deadline Cloud uses two endpoints for access to the service using AWS PrivateLink.

Workers use the `com.amazonaws.region.deadline.scheduling` endpoint to get tasks from the queue, report progress to Deadline Cloud, and to send task output back. If you are using a customer-managed fleet, the scheduling endpoint is the only endpoint that you need to create unless you are using management operations. For example, if a job creates more jobs, you need to enable the management endpoint to call the `CreateJob` operation.

The Deadline Cloud monitor uses the `com.amazonaws.region.deadline.management` to manage the resources in your farm, such as creating and modifying queues and fleets or getting lists of jobs, steps, and tasks.

Deadline Cloud also requires endpoints for the following AWS service endpoints:

- Deadline Cloud uses AWS STS to authenticate workers so that they can access job assets. For more information about AWS STS, see Temporary security credentials in IAM in the *AWS Identity and Access Management User Guide*.
- If you set up your customer-managed fleet in a subnet with no internet connection you must create a VPC endpoint for Amazon CloudWatch Logs so that workers can write logs. For more information, see Monitoring with CloudWatch.
- If you use job attachments, you must create a VPC endpoint for Amazon Simple Storage Service (Amazon S3) so that workers can access the attachments. For more information, see Job attachments in Deadline Cloud.

Create endpoints for Deadline Cloud

You can create interface endpoints for Deadline Cloud using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Create an interface endpoint in the *AWS PrivateLink Guide*.

Create management and scheduling endpoints for Deadline Cloud using the following service names. Replace *region* with the AWS Region where you've deployed Deadline Cloud.

```
com.amazonaws.region.deadline.management
```

```
com.amazonaws.region.deadline.scheduling
```

If you enable private DNS for the interface endpoints, you can make API requests to Deadline Cloud using its default Regional DNS name. For example, `worker.deadline.us-east-1.amazonaws.com` for worker operations, or `management.deadline.us-east-1.amazonaws.com` for all other operations.

You must also create an endpoint for AWS STS using the following service name:

```
com.amazonaws.region.sts
```

If your customer-managed fleet is on a subnet without an internet connection, you must create a CloudWatch Logs endpoint using the following service name:

```
com.amazonaws.region.logs
```

If you use job attachments to transfer files, you must create an Amazon S3 endpoint using the following service name:

```
com.amazonaws.region.s3
```

Security best practices for Deadline Cloud

AWS Deadline Cloud (Deadline Cloud) provides a number of security features to consider as you develop and implement your own security policies. The following best practices are general guidelines and don't represent a complete security solution. Because these best practices might not

be appropriate or sufficient for your environment, treat them as helpful considerations rather than prescriptions.

> **Note**
>
> For more information about the importance of many security topics, see the Shared Responsibility Model.

Data protection

For data protection purposes, we recommend that you protect AWS account credentials and set up individual accounts with AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon Simple Storage Service (Amazon S3).
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a **Name** field. This includes when you work with AWS Deadline Cloud or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into Deadline Cloud or other services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

AWS Identity and Access Management permissions

Manage access to AWS resources using users, AWS Identity and Access Management (IAM) roles, and by granting the least privilege to users. Establish credential management policies and

procedures for creating, distributing, rotating, and revoking AWS access credentials. For more information, see IAM Best Practices in the *IAM User Guide*.

Run jobs as users and groups

When using queue functionality in Deadline Cloud, it's a best practice to specify an operating system (OS) user and its primary group so that the OS user has least-privilege permissions for the queue's jobs.

When you specify a "Run as user" (and group), any processes for jobs submitted to the queue will be run using that OS user and will inherit that user's associated OS permissions.

The fleet and queue configurations combine to establish a security posture. On the queue side, the "Job run as user" and IAM role can be specified to use the OS and AWS permissions for the queue's jobs. The fleet defines the infrastructure (worker hosts, networks, mounted shared storage) that, when associated to a particular queue, run jobs within the queue. The data available on the worker hosts needs to be accessed by jobs from one or more associated queues. Specifying a user or group helps protect the data in jobs from other queues, other installed software, or other users with access to the worker hosts. When a queue is without a user, it runs as the agent user which can impersonate (sudo) any queue user. In this way, a queue without a user can escalate privileges to another queue.

Networking

To prevent traffic from being intercepted or redirected, it's essential to secure how and where your network traffic is routed.

We recommend that you secure your networking environment in the following ways:

- Secure Amazon Virtual Private Cloud (Amazon VPC) subnet route tables to control how IP layer traffic is routed.
- If you are using Amazon Route 53 (Route 53) as a DNS provider in your farm or workstation setup, secure access to the Route 53 API.
- If you connect to Deadline Cloud outside of AWS such as by using on-premises workstations or other data centers, secure any on-premises networking infrastructure. This includes DNS servers and route tables on routers, switches, and other networking devices.

Jobs and job data

Deadline Cloud jobs run within sessions on worker hosts. Each session runs one or more processes on the worker host, which generally require that you input data to produce output.

To secure this data, you can configure operating system users with queues. The worker agent uses the queue OS user to run session sub-processes. These sub-processes inherit the queue OS user's permissions.

We recommend that you follow best practices to secure access to the data these sub-processes access. For more information, see Shared responsibility model.

Farm structure

You can arrange Deadline Cloud fleets and queues many ways. However, there are security implications with certain arrangements.

A farm has one of the most secure boundaries because it can't share Deadline Cloud resources with other farms, including fleets, queues, and storage profiles. However, you can share external AWS resources within a farm, which compromises the security boundary.

You can also establish security boundaries between queues within the same farm using the appropriate configuration.

Follow these best practices to create secure queues in the same farm:

- Associate a fleet only with queues within the same security boundary. Note the following:
  - After job runs on the worker host, data may remain behind, such as in a temporary directory or the queue user's home directory.
  - The same OS user runs all the jobs on a service-owned fleet worker host, regardless of which queue you submit the job to.
  - A job might leave processes running on a worker host, making it possible for jobs from other queues to observe other running processes.
- Ensure that only queues within the same security boundary share an Amazon S3 bucket for job attachments.
- Ensure that only queues within the same security boundary share an OS user.
- Secure any other AWS resources that are integrated into the farm to the boundary.

Job attachment queues

Job attachments are associated with a queue, which uses your Amazon S3 bucket.

- Job attachments write to and read from a root prefix in the Amazon S3 bucket. You specify this root prefix in the `CreateQueue` API call.
- The bucket has a corresponding `Queue Role`, which specifies the role that grants queue users access to the bucket and root prefix. When creating a queue, you specify the `Queue Role` Amazon Resource Name (ARN) alongside the job attachments bucket and root prefix.
- Authorized calls to the `AssumeQueueRoleForRead`, `AssumeQueueRoleForUser`, and `AssumeQueueRoleForWorker` API operations return a set of temporary security credentials for the `Queue Role`.

If you create a queue and reuse an Amazon S3 bucket and root prefix, there is a risk of information being disclosed to unauthorized parties. For example, QueueA and QueueB share the same bucket and root prefix. In a secure workflow, ArtistA has access to QueueA but not QueueB. However, when multiple queues share a bucket, ArtistA can access the data in QueueB data because it uses the same bucket and root prefix as QueueA.

The console sets up queues that are secure by default. Ensure that the queues have a distinct combination of Amazon S3 bucket and root prefix unless they're part of a common security boundary.

To isolate your queues, you must configure the `Queue Role` to only allow queue access to the bucket and root prefix. In the following example, replace each *`placeholder`* with your resource-specific information.

```
{
  "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/*"
    }
  ]
}
```

You must also set a trust policy on the role. In the following example, replace the *placeholder* text with your resource-specific information.

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

```
  ]
}
```

Custom software Amazon S3 buckets

You can add the following statement to your `Queue Role` to access custom software in your Amazon S3 bucket. In the following example, replace *SOFTWARE_BUCKET_NAME* with the name of your S3 bucket.

```
"Statement": [
    {
        "Action": [
            "s3:GetObject",
            "s3:ListBucket"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::SOFTWARE_BUCKET_NAME",
            "arn:aws:s3:::SOFTWARE_BUCKET_NAME/*"
        ]
    }
]
```

For more information about Amazon S3 security best practices, see Security best practices for Amazon S3 in the *Amazon Simple Storage Service User Guide*.

Worker hosts

Secure worker hosts to help ensure that each user can only perform operations for their assigned role.

We recommend the following best practices to secure worker hosts:

- Don't use the same `jobRunAsUser` value with multiple queues unless jobs submitted to those queues are within the same security boundary.
- Don't set the queue `jobRunAsUser` to the name of the OS user that the worker agent runs as.
- Grant queue users least-privileged OS permissions required for the intended queue workloads. Ensure that they don't have filesystem write permissions to work agent program files or other shared software.

- Ensure only the root user on Linux and the `Administrator` owns account on Windows owns and can modify the worker agent program files.
- On Linux worker hosts, consider configuring a `umask` override in `/etc/sudoers` that allows the worker agent user to launch processes as queue users. This configuration helps ensure other users can't access files written to the queue.
- Grant trusted individuals least-privileged access to worker hosts.
- Restrict permissions to local DNS override configuration files (`/etc/hosts` on Linux and `C:\Windows\system32\etc\hosts` on Windows), and to route tables on workstations and worker host operating systems.
- Restrict permissions to DNS configuration on workstations and worker host operating systems.
- Regularly patch the operating system and all installed software. This approach includes software specifically used with Deadline Cloud such as submitters, adaptors, worker agents, OpenJD packages, and others.
- Use strong passwords for the Windows queue `jobRunAsUser`.
- Regularly rotate the passwords for your queue `jobRunAsUser`.
- Ensure least privilege access to the Windows password secretes and delete unused secrets.
- Don't give the queue `jobRunAsUser` permission the schedule commands to run in the future:
  - On Linux, deny these accounts access to `cron` and `at`.
  - On Windows, deny these accounts access to the Windows task scheduler.

> **Note**
>
> For more information about the importance of regularly patching the operating system and installed software, see the Shared Responsibility Model.

Workstations

It's important to secure workstations with access to Deadline Cloud. This approach helps ensure that any jobs you submit to Deadline Cloud can't run arbitrary workloads billed to your AWS account.

We recommend the following best practice to secure artist workstations. For more information, see the Shared Responsibility Model.

- Secure any persisted credentials that provide access to AWS, including Deadline Cloud. For more information, see Managing access keys for IAM users in the *IAM User Guide*.
- Only install trusted, secure software.
- Require users federate with an identity provider to access AWS with temporary credentials.
- Use secure permissions on Deadline Cloud submitter program files to prevent tampering.
- Grant trusted individuals least-privileged access to artist workstations.
- Only use submitters and adaptors that you obtain through the Deadline Cloud Monitor.
- Restrict permissions to local DNS override configuration files (`/etc/hosts` on Linux and macOS, and `C:\Windows\system32\etc\hosts` on Windows), and to route tables on workstations and worker host operating systems.
- Restrict permissions to `/etc/resolve.conf` on workstations and worker host operating systems.
- Regularly patch the operating system and all installed software. This approach includes software specifically used with Deadline Cloud such as submitters, adaptors, worker agents, OpenJD packages, and others.

Monitoring AWS Deadline Cloud

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Deadline Cloud (Deadline Cloud) and your AWS solutions. Collect monitoring data from all of the parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. Before you start monitoring Deadline Cloud, you should create a monitoring plan that includes answers to the following questions:

- What are your monitoring goals?
- Which resources will you monitor?
- How often will you monitor these resources?
- Which monitoring tools will you use?
- Who will perform the monitoring tasks?
- Who should be notified when something goes wrong?

AWS and Deadline Cloud provide tools that you can use to monitor your resources and respond to potential incidents. Some of these tools do the monitoring for you, some of the tools require manual intervention. You should automate monitoring tasks as much as possible.

- *Amazon CloudWatch* monitors your AWS resources and the applications you run on AWS in real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For example, you can have CloudWatch track CPU usage or other metrics of your Amazon EC2 instances and automatically launch new instances when needed. For more information, see the Amazon CloudWatch User Guide.

  Deadline Cloud has three CloudWatch metrics.
- *Amazon CloudWatch Logs* enables you to monitor, store, and access your log files from Amazon EC2 instances, CloudTrail, and other sources. CloudWatch Logs can monitor information in the log files and notify you when certain thresholds are met. You can also archive your log data in highly durable storage. For more information, see the Amazon CloudWatch Logs User Guide.
- *Amazon EventBridge* can be used to automate your AWS services and respond automatically to system events, such as application availability issues or resource changes. Events from AWS services are delivered to EventBridge in near real time. You can write simple rules to indicate

which events are of interest to you and which automated actions to take when an event matches a rule. For more information, see Amazon EventBridge User Guide.

- *AWS CloudTrail* captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.

For more information, see the following topics in the *Deadline Cloud Developer Guide*:

- CloudTrail logs
- Managing events using EventBridge
- Monitoring with CloudWatch

Quotas for Deadline Cloud

AWS Deadline Cloud provides resources, such as farms, fleets, and queues, that you can use to process jobs. When you create your AWS account, we set default quotas on these resources for each AWS Region.

Service Quotas is a central location where you can view and manage your quotas for AWS services. You can also request a quota increase for many of the resources that you use.

To view the quotas for Deadline Cloud, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **Deadline Cloud**.

To request a quota increase, see Requesting a quota increase in the *Service Quotas User Guide*. If the quota is not yet available in Service Quotas, use the service quota increase form.

Creating AWS Deadline Cloud resources with AWS CloudFormation

AWS Deadline Cloud is integrated with AWS CloudFormation, a service that helps you to model and set up your AWS resources so that you can spend less time creating and managing your resources and infrastructure. You create a template that describes all the AWS resources that you want (such as farms, queues, and fleets), and AWS CloudFormation provisions and configures those resources for you.

When you use AWS CloudFormation, you can reuse your template to set up your Deadline Cloud resources consistently and repeatedly. Describe your resources once, and then provision the same resources over and over in multiple AWS accounts and Regions.

Deadline Cloud and AWS CloudFormation templates

To provision and configure resources for Deadline Cloud and related services, you must understand AWS CloudFormation templates. Templates are formatted text files in JSON or YAML. These templates describe the resources that you want to provision in your AWS CloudFormation stacks. If you're unfamiliar with JSON or YAML, you can use AWS CloudFormation Designer to help you get started with AWS CloudFormation templates. For more information, see What is AWS CloudFormation Designer? in the *AWS CloudFormation User Guide*.

Deadline Cloud supports creating farms, queues, and fleets in AWS CloudFormation. For more information, including examples of JSON and YAML templates for farms, queues, and fleets, see the AWS Deadline Cloud in the *AWS CloudFormation User Guide*.

Learn more about AWS CloudFormation

To learn more about AWS CloudFormation, see the following resources:

- AWS CloudFormation
- AWS CloudFormation User Guide
- AWS CloudFormation API Reference
- AWS CloudFormation Command Line Interface User Guide

Document history for the Deadline Cloud user guide

The following table describes important changes in each release of the *AWS Deadline Cloud user guide*.

Change	Description	Date
Job resource limits	Added documentation for new job resource limit and maximum number of worker hosts. For more information, see Create resource limits for jobs.	January 30, 2025
Reorganized content from the user guide	Moved developer focused content from the user guide to the developer guide: • Moved instructions for creating a customer-managed fleet to a new Customer-managed fleets chapter in the developer guide. • Moved information about using your own licenses to the new Using software licenses chapter in the developer guide. • Moved details about monitoring with CloudTrail, CloudWatch, and EventBrid ge to the Monitoring chapter in the developer guide.	January 6, 2025

Budget threshold event	Added new budget threshold EventBridge event. For more information, see Deadline Cloud events detail reference.	October 30, 2024
Job status events	Added new job and task status EventBridge events. For more information, see Deadline Cloud events detail reference.	October 24, 2024
Resubmit job	Added information about how to resubmit a job. For more information, see Resubmit a job.	October 7, 2024
AWS Managed policy updates	Updated existing AWS managed policies. For more information, see AWS managed policies for Deadline Cloud.	October 7, 2024
Bring your own license	Added information about how you can use your own license server or license proxy instance with Deadline Cloud. For more information, see Service-managed fleets.	July 26, 2024
Autodesk 3ds Max UBL	Added information about Autodesk 3ds Max usage-based licensing (UBL) for Deadline Cloud. For more information, see Connect to a license endpoint.	June 18, 2024

Monitoring and cost management features	You can use EventBridge to support monitoring in Deadline Cloud. For more information, see Acting on EventBridge events. Deadline Cloud provides budgets and the usage explorer to help you control and visualize costs for your jobs. Learn about some best practices to help manage those costs. For more information, see Cost management.	May 23, 2024
Initial release	This is the initial release of the Deadline Cloud user guide.	April 2, 2024

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.