Getting Started Guide

AWS Management Console

Version 1.0

AWS Management Console: Getting Started Guide

Table of Contents

What is the AWS Management Console?

The AWS Management Console is a web-based application that contains and provides centralized access to all individual AWS service consoles. You can use Unified Navigation in the AWS Management Console to search for services, view notifications, access AWS CloudShell, access account and billing information, and customize your general console settings. The home page of the AWS Management Console is called AWS Console Home. From AWS Console Home, you can manage your AWS applications and access all other individual service consoles. You can also customize AWS Console Home to show other helpful information about AWS and your resources by using widgets. You can add, remove, and rearrange widgets such as **Recently visited**, **AWS Health**, and more.

Topics

Features of AWS Management Console

Important features of the AWS Management Console include the following:

- **Navigate to AWS service consoles** – You can use Unified Navigation to access recently visited service consoles, view and add services to your **Favorites** list, access your console settings, and access AWS User Notifications.
- **Search for AWS services and other AWS information** – use Unified Search to search for AWS services and features, and AWS marketplace products.
- **Customize the console** – You can use Unified settings to customize various aspects of the AWS Management Console. This includes the language, default Region, and more.
- **Run CLI commands** – AWS CloudShell is accessible directly from the console. You can use CloudShell to run AWS CLI commands against your favorite services.
- **Access all AWS event notifications** – You can use the AWS Management Console to access notifications from AWS User Notifications and AWS Health.

- **Customize AWS Console Home** – You can completely customize your AWS Console Home experience by using widgets.
- **Create and manage AWS applications** – Manage and monitor the cost, health, security posture, and performance of your applications using myApplications in AWS Console Home.
- **Chat with Amazon Q** – You can get generative artificial intelligence (AI) assistant powered answers to your AWS service questions directly from the console. You can also get connected with a live agent for additional support.
- **Control AWS account access in your network** – You can use AWS Management Console Private Access to limit access to the AWS Management Console to a specified set of known AWS accounts when the traffic originates from within your network.

Individual AWS service consoles in the AWS Management Console

Each AWS service has its own individual service console that you can access within the AWS Management Console. Settings you choose in Unified Settings for the AWS Management Console, such as visual mode and default language, are applied to all individual AWS consoles. AWS service consoles offer a wide range of tools for cloud computing, as well as information about your account and about your billing. If you want to know more about a specific service and its console, for example Amazon Elastic Compute Cloud, navigate to its console using Unified Search in the AWS Management Console navigation bar and access the Amazon EC2 documentation from the AWS Documentation website.

When you navigate to an individual AWS service's console, you can still access features of the AWS Management Console using Unified Navigation at the top of console. You can leave feedback for an individual service's console by navigating to that console and choosing **Feedback** in the page's footer.

Accessing the AWS Management Console

You can access the AWS Management Console at https://console.aws.amazon.com/.

Accessing the AWS Management Console with mobile devices

The AWS Management Console is designed to work on tablets as well as other kinds of mobile devices:

- Horizontal and vertical space is maximized to show more on your screen.
- Buttons and selectors are larger for a better touch experience.

To access the AWS Management Console on a mobile device, you must use the AWS Console Mobile Application. This app is available for Android and iOS. The Console Mobile Application provides mobile-relevant tasks that are a good companion to the full web experience. For example, you can easily view and manage your existing Amazon EC2 instances and Amazon CloudWatch alarms from your phone. For more information, see What is the AWS Console Mobile Application? in the *AWS Console Mobile Application User Guide*.

You can download the Console Mobile Application from Amazon Appstore, Google Play, and the iOS App Store.

Getting started with a service in the AWS Management Console

The AWS Management Console provides multiple ways for navigating to individual service consoles.

To open a console for a service

Do one of the following:

- In the search box on the navigation bar, enter all or part of the name of the service. Under **Services,** choose the service that you want from the list of search results. For more information, see Searching for products, services, features, and more using Unified Search in the AWS Management Console.
- In the **Recently visited services** widget, choose a service name.
- In the **Recently visited services** widget, choose **View all AWS services**. Then, on the **All AWS services** page, choose a service name.
- On the navigation bar, choose **Services** to open a full list of services. Then choose a service under **Recently visited** or **All services**.

Using the AWS Management Console navigation bar via Unified Navigation

This topic describes how to use Unified Navigation. Unified Navigation refers to the navigation bar that acts as the header and footer of the console. You can use Unified Navigation to:

- Search for and access AWS services, features, products, and more.
- Launch AWS Cloudshell.
- Access AWS notifications and AWS Health events.
- Get support from a variety of AWS knowledge sources.
- Configure the AWS Management Console by choosing your default language, visual mode, Region, and more.
- Access account, organization, service quota, and billing information.

Topics

Accessing the Services menu in the AWS Management Console

You can use the Services menu, next to the search bar to access your recently visited services, view your Favorites list, and view all AWS services. You can also view services by type by choosing a service type, for example **Analytics** or **Application Integration**.

The following procedure describes how to access the **Services** menu.

To access the Services menu

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose **Services** (⁝⁝⁝).
3. (Optional) Choose **Recently visited** to view services and applications you recently interacted with.
4. (Optional) Choose **Favorites** to view your Favorites list.
5. (Optional) Choose **All applications** to view your myApplications applications.
6. (Optional) Choose **All services** to view an alphabetical list of all AWS services.
7. (Optional) Choose a service type to view AWS services by type.

Searching for products, services, features, and more using Unified Search in the AWS Management Console

The search box in the navigation bar provides a unified search tool for finding AWS services and features, service documentation, AWS Marketplace products, and more. Just enter a few characters or a question to start generating results from all available content types. Each word you enter further refines your results. The available content types include:

- Services
- Features
- Documents
- Blogs
- Knowledge Articles
- Events
- Tutorials
- Marketplace
- Resources

> **Note**
>
> You can filter your search results to show only resources by performing a focused search. To perform a focused search, enter `/Resources` at the beginning of your query in the search bar and choose **/Resources** from the dropdown menu. Then enter the rest of your query.

Topics

Searching for AWS products in the AWS Management Console

The following procedure details how to search for AWS products using the search tool.

To search for a service, feature, documentation, or AWS Marketplace product

1. In the search box on the navigation bar of the AWS Management Console, enter your query.
2. Choose any link to navigate to your intended destination.

> **Tip**
>
> You can also use your keyboard to quickly navigate to the top search result. First, press **Alt+s** (Windows) or **Option+s** (macOS) to access the search bar. Then start entering your search term. When the intended result appears at the top of the list, press **Enter**. For example, to quickly navigate to the Amazon EC2 console, enter **ec2** and press **Enter**.

Refining your search in the AWS Management Console

You can refine your search by content type and view additional information about search results.

To refine your search to a specific content type

1. In the search box on the navigation bar of the AWS Management Console, enter your query.
2. Choose one of the content types next to your search results.
3. (Optional) To see all results for a specific category:
   - Choose **Show more**. A new tab will open showing the results.
4. (Optional) To view additional information about your search results:
   a. In the search results, hover your cursor over a search result.
   b. View the available additional information.

Viewing features of a service in the AWS Management Console

You can view features of a service from within your search results.

To view features of a service

1. In the search box on the navigation bar of the AWS Management Console, enter your query.
2. In the search results, hover your cursor over a service in **Services**.
3. Choose one of the links in **Top features**.

Launching AWS CloudShell from the navigation bar in the AWS Management Console

AWS CloudShell is a browser-based, pre-authenticated shell that you can launch directly from the AWS Management Console navigation bar. You can run AWS CLI commands against services using your preferred shell (Bash, PowerShell, or Z shell).

You can launch CloudShell from the AWS Management Console using one of the following two methods:

- Choose the CloudShell icon in the footer of the console.
- Choose the CloudShell icon on the console navigation bar.

For more information about this service, see the AWS CloudShell User Guide.

For information about the AWS Regions where AWS CloudShell is available, see the AWS Regional Services List. The selection of the Console Region is in sync with the CloudShell Region. If CloudShell isn't available in a selected Region, then CloudShell will operate in the nearest Region.

Accessing AWS notifications and Health events

You can access some of your AWS notifications and view health events from the navigation bar. You can also access AWS User Notifications to view all of your AWS notifications and the AWS Health Dashboard from the navigation bar.

For more information see What is AWS User Notifications? in the *AWS User Notifications User Guide* and What is AWS Health? in the *AWS Health User Guide*

The following procedure describes how to access your AWS event information.

To access your AWS event information

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the bell icon.
3. View your notifications and health events.
4. (Optional) Choose **see all notifications** to navigate to the User Notifications console.
5. (Optional) Choose **see all Health events** to navigate to the AWS Health console.

Getting support

You can get support by choosing the question mark icon in the navigation bar. From the support menu, you can choose to:

- Navigate to the Support Center service console
- Get expert help from AWS IQ
- View curated knowledge from community articles and the knowledge center on AWS re:Post
- Go to AWS documentation
- Navigate to AWS trainings
- Navigate to the AWS getting started Resource Center
- Leave feedback for any service console you're currently accessing

> **Note**
>
> This can also be done by choosing **Feedback** in the console footer. The title of the modal that opens shows which console you're currently leaving feedback for

You can also get help anytime in the console, get connected with a live agent, and ask any question about AWS by chatting with AWS Q. For more information, see ???.

Configuring the AWS Management Console using Unified Settings

This topic describes how to configure your AWS Management Console using the Unified Settings page to set defaults that apply to all service consoles.

Topics

Configuring Unified Settings in the AWS Management Console

You can configure settings and defaults, such as display, language, and Region, from the AWS Management Console **Unified Settings** page. You can access Unified Settings via the navigation bar in Unified Navigation. The visual mode and default language can also be set directly from the navigation bar. These changes apply to all service consoles.

> **Important**
>
> To ensure that your settings, favorite services, and recently visited services persist globally, this data is stored in all AWS Regions, including Regions that are disabled by default. These

Regions are Africa (Cape Town), Asia Pacific (Hong Kong), Asia Pacific (Hyderabad), Asia Pacific (Jakarta), Europe (Milan), Europe (Spain), Europe (Zurich), Middle East (Bahrain), and Middle East (UAE). You still need to manually enable a Region to access it and to create and manage resources in that Region. If you don't want to store this data in all AWS Regions, choose **Reset all** to clear your settings, and then opt out of remembering recently visited services in Settings management.

Topics

Accessing Unified Settings in the AWS Management Console

The following procedure describes how to access Unified Settings.

To access Unified Settings

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. To open the **Unified Settings** page, choose **See all user settings**.

Resetting Unified Settings in the AWS Management Console

You can delete all Unified Settings configurations and restore the default settings by resetting Unified Settings.

Note

This affects multiple areas of AWS, including favorite services in navigation and the Services menu, recently visited services on Console Home widgets and in the AWS Console Mobile Application, and all settings that apply across services, such as default language, default Region, and visual mode.

To reset all Unified Settings

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. Open the **Unified Settings** page by choosing **See all user settings**.
4. Choose **Reset all**.

Editing Unified Settings in the AWS Management Console

The following procedure describes how to edit your preferred settings.

To edit Unified Settings

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. Open the **Unified Settings** page by choosing **See all user settings**.
4. Choose **Edit** next to your preferred settings:

   - **Localization and default Region:**
     - **Language** lets you select the default language for console text.
     - **Default Region** lets you select a default Region that applies each time you log in. You can select any of the available Regions for your account. You can also select the last used Region as your default.

       To learn more about Region routing in the AWS Management Console, see Choosing a Region.
   - **Display:**
     - **Visual mode** lets you set your console to light mode, dark mode, or the default display mode of your browser.

       Dark mode is a beta feature and might not apply across all AWS service consoles.
     - **Favorites bar display** toggles the **Favorites** bar display between the full service name with its icon or only the service's icon.
     - **Favorites bar icon size** toggles the size of the service icon on the **Favorites** bar display between small (16x16 pixels) and large (24x24 pixels).
   - **Settings management:**

- **Remember recently visited services** lets you choose if the AWS Management Console remembers your recently visited services. Turning this off also deletes your recently visited services history, so you will no longer see recently visited services in the Service menu, AWS Console Mobile Application, or on Console Home widgets.

5. Choose **Save changes**.

Changing the visual mode of the AWS Management Console

Your visual mode sets your console to light mode, dark mode, or the default display mode of your browser.

To change the visual mode from the navigation bar

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. For **Visual mode**, choose **Light** for light mode, **Dark** for dark mode, or **Browser default** for the default display mode of your browser.

Configuring visible Regions and services in the AWS Management Console

Account administrators can control which AWS Regions and AWS services are visible in the AWS Management Console navigation. These account-level settings are available on the **Account settings** tab of the Unified Settings page. When you hide a Region, it is removed from the Region selector for all users in the account. When you hide a service, it appears as unavailable in a separate section of the Services menu for all users in the account. Hidden services are also grayed out in Unified Search results and in the Recently Visited and Favorites widgets on Console Home.

If a user navigates directly to a hidden Region or service through a URL, they see an overlay that informs them the Region or service is hidden at the account level.

> **Note**
>
> Navigation to the Unified Settings page is always available, so administrators can't lock themselves out of these settings. If a user doesn't have the required permissions, or if the

AWS User Experience Customization service is unavailable, all Regions and services are visible by default.

Topics

Prerequisites for configuring visible Regions and services

To view and change visible Regions and services settings, you need specific IAM permissions.

- To view the settings, you need the `uxc:GetAccountCustomizations` permission.
- To change the settings, you need the `uxc:UpdateAccountCustomizations` permission.

The AWS managed policies `AWSManagementConsoleBasicUserAccess` and `AWSManagementConsoleAdministratorAccess` include these permissions.

For more information, see ???.

Configuring visible Regions in the AWS Management Console

You can choose which AWS Regions appear in the Region selector for all users in your account.

To configure visible Regions

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. Choose **See all user settings** to open the **Unified Settings** page.
4. Choose the **Account settings** tab.
5. For **Visible Regions**, select the check boxes for the Regions that you want to show, or clear the check boxes for the Regions that you want to hide.
6. Choose **Save changes**.

After you save, hidden Regions are removed from the Region selector for all users in the account.

Configuring visible services in the AWS Management Console

You can choose which AWS services appear in the Services menu for all users in your account.

To configure visible services

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. Choose **See all user settings** to open the **Unified Settings** page.
4. Choose the **Account settings** tab.
5. For **Visible services**, select the check boxes for the services that you want to show, or clear the check boxes for the services that you want to hide.
6. Choose **Save changes**.

After you save, hidden services appear as unavailable in a separate section of the Services menu for all users in the account. Hidden services are also grayed out in Unified Search results and in the Recently Visited and Favorites widgets on Console Home.

Choosing your Region

For many services, you can choose an AWS Region that specifies where your resources are managed. Regions are sets of AWS resources located in the same geographical area. You don't need to choose a Region for the AWS Management Console or for some services, such as AWS Identity and Access Management. To learn more about AWS Regions, see Managing AWS Regions in the *AWS General Reference*.

> **Note**
>
> If you have created AWS resources but you don't see those resources in the console, the console might be displaying resources from a different Region. Some resources (such as Amazon EC2 instances) are specific to the Region where they were created.

Topics

- Choosing a Region from the navigation bar in the AWS Management Console
- Setting the default Region in the AWS Management Console

Choosing a Region from the navigation bar in the AWS Management Console

The following procedure details how you can change your Region from the navigation bar.

To choose a Region from the navigation bar

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the name of the currently displayed Region.
3. Choose a Region to switch to.

Setting the default Region in the AWS Management Console

The following procedure details how you can change your default Region from the Unified Settings page.

To set your default Region

1. In the navigation bar, choose the gear icon (#).
2. Choose **See all user settings** to navigate to the **Unified Settings** page.
3. Choose **Edit** next to **Localization and default Region**.
4. In **Default Region**, choose a Region.

> **Note**
>
> If you do not select a default Region, the last Region you visited will be your default.

5. Choose **Save settings**.
6. (Optional) Choose **Go to new default Region** to immediately go to your new default Region.

Favorites in the AWS Management Console

To access your frequently used services and applications more quickly, you can save their service consoles to a list of **Favorites**. You can add and remove favorites using the AWS Management Console. When you add a service or application to your **Favorites**, it appears on the Favorites quickbar.

Topics

Adding favorites in the AWS Management Console

You can add services and applications to your favorites from the **Services** menu and the **Recently visited** menu. You can also add services to your favorites by using the search results page from the search box. Services and applications that you add to your favorites appear in the Favorites quickbar.

Topics

Favorites quickbar in the AWS Management Console

The favorites quickbar appears when you have at least one AWS service or application added to your favorites. The favorites quickbar is located following the navigation bar and is visible in all AWS service consoles, so you can quickly access your favorite services and applications. You can rearrange the order of the services and applications in the favorites quickbar by dragging a service or application to the left or right.

Adding services to your favorites in the AWS Management Console

You can add services to your favorites from the **Services** menu or the search results page from the search box.

Services menu

To add favorites from the Services menu

1. Open the AWS Management Console.
2. In the navigation bar, choose **Services** (⁝⁝⁝).
3. (Optional) Add a recently visited service to your favorites:

a. In **Recently visited**, hover your cursor over a service.
b. Select the star next to the service's name.

4. Choose **All services**.
5. Hover your cursor over your chosen service.
6. Select the star next to the service's name.

Search box

To add favorites from the search box

1. Open the AWS Management Console.
2. Enter the name of a service in the search box.
3. In the search results page, select the star next to the service's name.

> **Note**
>
> After you add a service to your favorites, it's added to the favorites quickbar following the navigation bar.

Adding applications to your favorites in the AWS Management Console

You can add applications to your favorites from the **Services** menu.

To add favorites from the Services menu

1. Open the AWS Management Console.
2. In the navigation bar, choose **Services** (⁝⁝⁝).
3. (Optional) Add a recently visited application to your favorites:
   a. In **Recently visited**, hover your cursor over an application.
   b. Select the star next to the application's name.
4. Choose **Applications**.
5. Hover your cursor over your chosen application.
6. Select the star next to the application's name.

> **Note**
>
> After you add an application to your favorites, it's added to the favorites quickbar following the navigation bar.

Accessing favorites in the AWS Management Console

You can access services and applications added to your favorites from the **Services** menu, the favorites quickbar, and the **Favorites** widget.

Services menu

To access your favorites from the Services menu

1. Open the AWS Management Console.
2. In the navigation bar, choose **Services** (⁝⁝⁝).
3. Choose **Favorites**.
4. View the services and applications you added to your favorites.
5. (Optional) View application resources:
    a. Select an application.
    b. (Optional) Select a view.
    c. View your resources.
    d. (Optional) Select a filter. You can filter your resources by **Properties** or by **Tags**. For more information, see Search query syntax reference for Resource Explorer in the *AWS Resource Explorer User Guide*.
    e. (Optional) Select a resource to view it in the relevant service console.

> **Tip**
>
> You can continue browsing resources where you left off by choosing **Services** (⁝⁝⁝). Your applied search filters will also persist.

Favorites quickbar

To access your favorites from the favorites quickbar

1. Open the AWS Management Console.
2. View the services and applications in the favorites quickbar.

Favorites widget

To access your favorites from the Favorites widget

1. Open the AWS Management Console.
2. (Optional) Add the **Favorites** widget if you don't have it:
   a. Choose the **+ Add widgets** button on the Console Home page.
   b. In the **Add widgets** menu, drag the **Favorites** widget by using the ⁝⁝ icon and place it on your Console Home page.
3. View the services and applications in the **Favorites** widget.

For more information about widgets, see the section called “Working with Widgets”.

Removing favorites in the AWS Management Console

You can remove services and applications from your favorites using the **Services** menu. You can also remove services by using the search results page from the search bar.

Services menu

To remove favorites from the Services menu

1. Open the AWS Management Console.
2. In the navigation bar, choose **Services**.
3. Choose **Favorites**.
4. Deselect the star next to the service or application.

Search box

> **Note**
>
> Currently, you can only remove services using the search results page from the search bar.

To remove favorites from the search box

1. Open the AWS Management Console.
2. Enter the name of a service in the search box.
3. In the search results page, deselect the star next to the service's name.

Changing your password in the AWS Management Console

You may be able to change your password from the AWS Management Console depending on your user type and your permissions. The following topic describes how to change your password for each user type.

Topics

- Root users in the AWS Management Console
- IAM users in the AWS Management Console
- IAM Identity Center users in the AWS Management Console
- Federated identities in the AWS Management Console

Root users in the AWS Management Console

Root users can change their passwords directly from the AWS Management Console. A Root user is the account owner with complete access to all AWS services and resources. You're the root user if you created the AWS account and you sign in using your root user email and password. For more information, see Root user in the *AWS IAM Identity Center User Guide*.

To change your password as a Root user

1. Sign in to the AWS Management Console.

2. In the navigation bar, choose your account name.
3. Choose **Security credentials**.
4. The options displayed will vary depending on your AWS account type. Follow the instructions shown on the console to change your password.
5. Enter your current password once and your new password twice.

   The new password must be at least eight characters long and must include the following:

   - At least one symbol
   - At least one number
   - At least one uppercase letter
   - At least one lowercase letter
6. Choose **Change Password** or **Save changes**.

IAM users in the AWS Management Console

IAM users may be able to change their password from the AWS Management Console depending on their permissions. Otherwise, they must use an AWS access portal. An IAM user is an identity within your AWS account that's granted specific custom permissions. You're an IAM user if you didn't create the AWS account and your administrator or help desk employee provided you your sign-in credentials that include an AWS account ID or account alias, an IAM user name, and password. For more information, see IAM user in the *AWS Sign-In User Guide*.

If you have permissions from the following policy: AWS: Allows IAM users to change their own console password on the Security credentials page, you can change your password from the console. For more information, see How an IAM user changes their own password in the *AWS Identity and Access Management User Guide*.

If you don't have the requisite permissions to change your password from the AWS Management Console see, Resetting your AWS IAM Identity Center user password in the *AWS IAM Identity Center User Guide*.

IAM Identity Center users in the AWS Management Console

AWS IAM Identity Center users must change their password from an AWS access portal. For more information, see Resetting your AWS IAM Identity Center user password in the *AWS IAM Identity Center User Guide*.

An IAM Identity Center user is a user whose AWS account is part of AWS Organizations who signs in through the AWS access portal with a unique URL. These users can be either created directly in the users in IAM Identity Center or in Active directory or another external identity provider. For more information, see AWS IAM Identity Center user in the *AWS Sign-In User Guide*.

Federated identities in the AWS Management Console

Federated identity users must change their password from an AWS access portal. For more information, see Resetting your AWS IAM Identity Center user password in the *AWS IAM Identity Center User Guide*.

Federated identity users sign in using an external identity provider (IdP). You're a federated identity if you either:

- Access your AWS account or resources with third party credentials like Login with Amazon, Facebook, or Google.
- Use the same credentials to sign in to corporate systems and AWS services and you use a custom company portal to sign-in to AWS.

For more information, see Federated identity in the *AWS Sign-In User Guide.*.

Changing the language of the AWS Management Console

The AWS Console Home experience includes the Unified Settings page where you can change the default language for AWS services in the AWS Management Console. You can also change the default language quickly from the settings menu from the navigation bar.

> **Note**
>
> The following procedures change the language for all AWS service consoles, but not for AWS documentation. To change the language used for documentation, use the language menu in the upper right of any documentation page.

Topics

- Changing the default language via Unified Settings in the AWS Management Console

Supported languages

The AWS Management Console currently supports the following languages:

- English (US)
- English (UK)
- Bahasa Indonesia
- German
- Spanish
- French
- Japanese
- Italian
- Portuguese
- Korean
- Chinese (Simplified)
- Chinese (Traditional)
- Turkish

Changing default language from the navigation bar in the AWS Management Console

The following procedure details how to change your default language directly from the navigation bar.

To change the default language from the navigation bar

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. For **Language**, choose either **Browser default** or the preferred language from the dropdown list.

Changing the default language via Unified Settings in the AWS Management Console

The following procedure details how to change your default language from the Unified Settings page.

To change the default language in Unified Settings

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the gear icon (#).
3. To open the **Unified Settings** page, choose **See all user settings**.
4. In **Unified Settings**, choose **Edit** next to **Localization and default Region**.
5. To select the language that you want for the console, choose one of the following options:

   - Choose the **Browser default** from the dropdown list, and then choose **Save settings**.

     The console text for all AWS services appears in your preferred language that you've set in your browser settings.

     > **Note**
     >
     > The browser default only supports languages supported by the AWS Management Console.

   - Choose the preferred language from the dropdown list, and then choose **Save settings**.

     The console text for all AWS services appears in your preferred language.

Accessing your AWS account, organization, service quota, and billing information in the AWS Management Console

If you have the necessary permissions, you can access information about your AWS account, service quotas, organization, and billing information from the console.

Note

The AWS Management Console only provides access to account, organization, service quota, and billing information. These services have their own separate consoles. For more information, see the following:

- Manage your AWS account in the *AWS Account Management Reference Guide*.
- What is AWS Organizations? in the *AWS Organizations User Guide*.
- What is Service Quotas? in the *Service Quotas User Guide*.
- Using the AWS Billing and Cost Management home page in the *AWS Billing User Guide*.

Tip

You can also get more information about any of these topics by asking Amazon Q. For more information, see Chat with Amazon Q Developer.

Topics

Accessing account information in the AWS Management Console

If you have the necessary permissions, you can access information about your AWS account from the console.

To access your account information

1. Sign in to the AWS Management Console.
2. On the navigation bar, choose your account name.
3. Choose **Account**.
4. View your account information.

Note

If you would like to close your AWS account, see Close an AWS account in the *AWS Account Management Reference Guide*.

Accessing organization information in the AWS Management Console

If you have the necessary permissions, you can access information about your AWS organizations from the console.

To access organization information

1. Sign in to the AWS Management Console.
2. On the navigation bar, choose your account name.
3. Choose **Organizations**.
4. View your organization information.

Accessing service quota information in the AWS Management Console

If you have the necessary permissions, you can access information about service quotas from the console.

To access service quota information

1. Sign in to the AWS Management Console.
2. On the navigation bar, choose your account name.
3. Choose **Service Quotas**.
4. View and manage your service quota information.

Accessing billing information in the AWS Management Console

If you have the necessary permissions, you can access information about your AWS charges from the console.

To access your billing information

1. Sign in to the AWS Management Console.
2. On the navigation bar, choose your account name.
3. Choose **Billing and Cost Management**.
4. Use the AWS Billing and Cost Management dashboard to find a summary and a breakdown of your monthly spending.

Signing in to multiple accounts

You can sign in to up to five different identities simultaneously in a single web browser in the AWS Management Console. These can be any combination of root, IAM, or federated roles in different accounts or in the same account. Each identity you sign in to opens its own instance of the AWS Management Console in a new tab.

When you enable multi-session support, the console URL contains a subdomain (for example, `https://000000000000-aaaaaaaa.us-east-1.console.aws.amazon.com/console/home?region=us-east-1`). Be sure to update your bookmarks and console links.

> **Note**
>
> You must opt-in to multi-session support by choosing **Turn on multi-session** in the account menu in the AWS Management Console, or by choosing **Enable multi-session** on https://console.aws.amazon.com/. You can opt-out of multi-sessions at any time by choosing **Disable multi-session** on https://console.aws.amazon.com/ or by clearing your browser cookies. Opt-in is browser specific.

To sign in to multiple identities

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose your account name.
3. Choose **Add session** and choose **Sign in**. A new tab will open for you to sign in.

> **Note**
>
> For more information about signing in as a root or IAM user, see Sign in to the AWS Management Console in the *AWS Sign-in* User Guide.

4. Enter your credentials.
5. Choose **Sign in**. The AWS Management Console loads in this tab as your chosen AWS identity.
6. **(Optional) To federate into additional roles**

    a. In the AWS IAM Identity Center access portal or your single-sign on (SSO) portal, sign in to the additional role.

    b. In the AWS Management Console choose your account name.

    c. View the additional sessions that you can choose.

AWS Recommended Actions in the AWS Management Console

AWS Recommended Actions helps you work more efficiently in the AWS Management Console by providing contextual suggestions for completing tasks and implementing best practices. When relevant recommendations are available, a dynamic button appears that you can use to quickly take action based on these suggestions.

> **Note**
>
> AWS Recommended Actions analyzes resource state to provide suggestions but doesn't process user data.

Topics

Features of AWS Recommended Actions

- **Action recommendations** — Get relevant suggestions based on resource state, best practices, and common usage patterns
- **One-click actions** — Complete recommended actions directly from success messages or resource views
- **Integrated right side panel** — Access an integrated side panel to implement suggestions without disrupting your workflow
- **Multi-service support** — Get recommendations across multiple AWS services

Using recommended actions

To use recommended actions

1. Sign in to the AWS Management Console
2. Look for the **# Recommended actions** button.

> **Note**
>
> The recommended actions button can appear anywhere in the AWS Management Console and is only accessible when recommended actions are available.

3. Choose the button to view available actions.
4. Run recommendations directly or through the side panel.

Logging AWS Recommended Actions API calls using AWS CloudTrail

AWS Recommended Actions is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service. CloudTrail captures all API calls for AWS Recommended Actions as events. The calls captured include calls from the AWS Management Console and code calls to the AWS Recommended Actions API operations. Using the information collected by CloudTrail, you can determine the request that was made to AWS Recommended Actions, the IP address from which the request was made, when it was made, and additional details.

CloudTrail is active in your AWS account when you create the account and you automatically have access to the CloudTrail **Event history**. The CloudTrail **Event history** provides a viewable, searchable, downloadable, and immutable record of the past 90 days of recorded management events in an AWS Region. For more information, see Working with CloudTrail Event history in the *AWS CloudTrail User Guide*. There are no CloudTrail charges for viewing the **Event history**.

For an ongoing record of events in your AWS account past 90 days, create a trail or a CloudTrail Lake event data store.

AWS Recommended Actions management events in CloudTrail

Management events provide information about management operations that are performed on resources in your AWS account. These are also known as control plane operations. By default, CloudTrail logs management events.

AWS Recommended Actions logs all AWS Recommended Actions control plane operations as management events.

AWS Recommended Actions event examples

An event represents a single request from any source and includes information about the requested API operation, the date and time of the operation, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so events don't appear in any specific order.

The following example shows a CloudTrail event that demonstrates the operation.


```
{
  "awsRegion": "us-east-2",
  "eventCategory": "Management",
  "eventID": "3510a29e-8070-4cbc-b6a0-9e11f18e26ec",
  "eventName": "ListRecommendedActions",
  "eventSource": "action-recommendations.amazonaws.com",
  "eventTime": "2025-09-03T03:52:02Z",
  "eventType": "AwsApiCall",
  "eventVersion": "1.09",
  "managementEvent": true,
  "readOnly": true,
  "recipientAccountId": "123456789098",
  "requestID": "ec431c91-0315-413d-bdb6-d282fd4f6d83",
  "requestParameters": {
```

```
    "context": "*",
    "uxChannel": "EXAMPLE"
  },
  "responseElements": null,
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "EXAMPLE",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROARZDBH75ZCUYWFSTUS:EXAMPLE",
    "arn": "arn:aws:sts::123456789098:assumed-role/EXAMPLE",
    "accountId": "12345678909",
    "accessKeyId": "ASIARZDBEXAMPLE",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "AROARZDBHEXAMPLE",
          "arn": "arn:aws:iam::12345678909:role/EXAMPLE",
          "accountId": "12345678909",
          "userName": "EXAMPLE"
        },
        "attributes": {
          "creationDate": "2025-09-03T03:52:00Z",
          "mfaAuthenticated": "false"
        }
      },
    "invokedBy": "action-recommendations.amazonaws.com"
  }
}
```


For information about CloudTrail record contents, see CloudTrail record contents in the *AWS CloudTrail User Guide*.

Using AWS Console Home in the AWS Management Console

This topic describes how to use AWS Console Home, including how to customize your Console Home page. Console Home is the home page of the AWS Management Console. When you first log in to the console, you land on the Console Home page. You can customize your Console Home page using widgets and applications. Widgets let you add custom components that track information about your AWS services and resources. Applications allow you to group your AWS resources and metadata. You can manage applications using myApplications. You can also use Console Home to view a list of all AWS services and chat with Amazon Q.

Topics

Viewing all AWS services in AWS Console Home

You can view a list of all AWS services and access their consoles from Console Home.

To access a complete list of AWS services

1. Sign in to the AWS Management Console.
2. Expand the Console Home menu by choosing the hamburger icon (☰).
3. Choose **All services**.
4. Select an AWS service to navigate to its console.

Working with widgets in AWS Console Home

The Console Home dashboard includes widgets that display important information about your AWS environment and provide shortcuts to your services. You can customize your experience by adding and removing widgets, rearranging them, or changing their size.

Managing widgets

You can manage widgets by adding, removing, rearraning, and resizing them. Default widgets can me removed and added again. You can also reset your Console Home to the default layout and request new widgets.

To add a widget

1. On the upper or lower right of the Console Home dashboard, choose the **+Add widgets** button.
2. Choose the **drag indicator**, represented by six vertical dots (⁝⁝) in the upper left of the widget title bar, and then drag it to your Console Home dashboard.

To remove a widget

1. Choose the **ellipsis**, represented by three vertical dots (⁝) in the upper right of the widget title bar.
2. Choose **Remove widget**.

To rearrange your widgets

- Choose the **drag indicator**, represented by six vertical dots (⁝⁝) in the upper left of the widget title bar, and then drag the widget to a new location on your Console Home dashboard.

To resize a widget

- Choose the **resize icon** at the bottom right of the widget, and then drag to resize the widget.

If you want to start over with organizing and setting up your widgets, you can reset the Console Home dashboard to the default layout. This will revert your changes to the Console Home dashboard layout, and restore all the widgets to their default location and size.

To reset the page to the default layout

1. On the upper right of the page, choose the **Reset to default layout** button.
2. To confirm, choose **Reset**.

> **Note**
>
> This will revert all your changes to the layout of the Console Home dashboard.

To request a new widget in the Console Home dashboard

1. On the lower left of the Console Home dashboard, choose **Want to see another widget? Tell us!**

   Describe the widget that you want to see added in the Console Home dashboard.
2. Choose **Submit**.

> **Note**
>
> Your suggestions are periodically reviewed and new widgets might be added in future updates to the AWS Management Console.

What is myApplications in AWS Console Home?

myApplications is an extension of Console Home that helps you manage and monitor the cost, health, security posture, and performance of your applications on AWS. Applications allow you to group resources and metadata. You can access all applications in your account, key metrics across all applications, and an overview of cost, security, and operations metrics and insights from multiple service consoles from one view in the AWS Management Console. myApplications includes the following:

- Applications widget on the Console Home page
- myApplications that you can use to view application resource costs and security findings
- myApplications dashboard that provides a view of key application metrics such as cost, performance, and security findings

Topics

- Accessing myApplications
- Pricing
- Supported Regions for myApplications
- Applications in myApplications
- Resources in myApplications
- myApplications dashboard in AWS Console Home

Features of myApplications

- **Create applications** – Create new applications and organize their resources. Your applications are automatically shown in the myApplications, so you can take action in the AWS Management Console, APIs, CLI, and SDKs. Infrastructure as code (IaC) is generated when you create an application and is accessible from the myApplication dashboard. IaC is useable in IaC tools including AWS CloudFormation and Terraform.
- **Access your applications** – You can quickly access any of your applications from the myApplications widget by selecting it.
- **Access your resources** – You can quickly view your application resources from the Services menu by selecting the application. When you select a resource, you go directly to the relevant service console. Your place in the resource table is saved, so you can continue browsing at any time from the Services menu.
- **Compare application metrics** – Use myApplications to compare key metrics for applications like cost of application resources and number of critical security findings for multiple applications.
- **Monitor and manage applications** – Assess application health and performance using alarms, canaries, and service level objectives from Amazon CloudWatch, findings from AWS Security Hub CSPM, and cost trends from AWS Cost Explorer Service. You can also find compute metrics summaries and optimizations and manage resource compliance and configuration status from AWS Systems Manager.

Related services

myApplications makes use of the following services:

- AppRegistry
- AppManager

- Amazon CloudWatch
- Amazon EC2
- AWS Lambda
- AWS Resource Explorer
- AWS Security Hub CSPM
- Systems Manager
- AWS Service Catalog
- Tagging

Accessing myApplications

You can access myApplications from the AWS Management Console by choosing **myApplications** in the left sidebar.

Pricing

myApplications on AWS is offered at no additional charge. There are no set-up fees or upfront commitments. Usage charges for the underlying resources and services that the myApplication dashboard summarizes still apply at published rates for those resources.

Supported Regions for myApplications

myApplications is available in the following AWS Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)

- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- South America (São Paulo)

Opt-in Regions

Opt-in Regions aren't enabled by default. You must manually enable these Regions to use them with myApplications. For more information about AWS Regions, see Managing AWS Regions. The following opt-in Regions are supported:

- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Hyderabad)
- Asia Pacific (Jakarta)
- Asia Pacific (Melbourne)
- Europe (Milan)
- Europe (Spain)
- Europe (Zurich)
- Middle East (Bahrain)
- Middle East (UAE)
- Israel (Tel Aviv)

Applications in myApplications

Applications allow you to group your resources and metadata. You can manage your applications by creating, onboarding, viewing, editing, or deleting them. You can also create code snippets to automatically add new resources to an application.

> **Note**
>
> You can also add applications to your **Favorites** so they're easier to access. For more information, see ???.

Topics

Creating applications in myApplications

You can create a new application or the section called “Onboarding applications” created before November 8, 2023 to get started with myApplications. When you create a new application, you can add resources by searching for them and selecting them or by using existing tags.

To create a new application

1. Sign in to the AWS Management Console.
2. Expand the left sidebar and choose **myApplications**.
3. Choose **Create application**.
4. Enter an application name.
5. (Optional) Enter an application description.
6. (Optional) Add tags. Tags are key-value pairs that are applied to resources to hold metadata about those resources.

> **Note**
>
> The AWS application tag is automatically applied to newly created applications. For more information, see The AWS application tag in the *AWS Service Catalog AppRegistry Administrator Guide*.

7. (Optional) Add attribute groups. You can use attribute groups to store application metadata.
8. Choose **Next**.
9. (Optional) Add resources:

   Search and select resources

   > **Note**
   >
   > To search and add resources, you must turn on AWS Resource Explorer. For more information, see Getting started with AWS Resource Explorer.
   > All added resources are tagged with the AWS application tag.

   To add resources using search

   1. Choose **Search and select resources**.
   2. Choose **Select resources**.
   3. (Optional) Choose a view.
   4. Search for your resources. You can search by keyword, name or type, or choose a resource type.

      > **Note**
      >
      > If you can't find the resource you're looking for, troubleshoot with AWS Resource Explorer. For more information, see Troubleshooting Resource Explorer search issues in the *Resource Explorer User Guide*.

   5. Select the checkbox next to the resources you want to add.
   6. Choose **Add**.
   7. Choose **Next**.
   8. Review your choices.

   Automatically add resources using tags

   When you create an application, you can bulk-onboard resources by specifying an existing tag key-value pair. With this method, AWS automatically applies the `awsApplication`

tag to all of the resources tagged with the specified key-value pair, and creates a tag-sync for the application's resources by default. With tag-sync enabled, any resources that are tagged with the specified tag key-value pair are automatically added to the application. For information about resolving tag-sync errors, see the section called "Resolving tag-sync errors in myApplications".

> **Note**
>
> Adding resources to an application using tags requires permissions to create an AppRegistry application, group and ungroup resources, and tag and untag resources. You can either add the Resource Groups `ResourceGroupsTaggingAPITagUntagSupportedResources` AWS managed policy, or you can create and maintain your own custom policy. The following permissions must be added to a user's policy statement in IAM:
>
> - `servicecatalog:CreateApplication`
> - `resource-groups:GroupResources`
> - `resource-groups:UngroupResources`
> - `tag:TagResources`
> - `tag:UntagResources`

To add resources using existing tags

1. Choose **Automatically add resources using tags**.
2. Select an existing tag key and value:
   a. Select the **Role** used to tag resources. For more information, see Tag-sync required permissions in the *AWS Service Catalog AppRegistry Administrator Guide*.
   b. Select a **Tag key**.
   c. Select a **Tag value**.
   d. (Optional) Choose **Preview resources** to preview which resources are tagged with the tag key-value pair.

e. Review and accept the **I acknowledge that Group Lifecycle Events will be enabled to create a tag sync** notice. GLE allows AWS to notice changes to the resources tagged with your key-value pair.

3. Choose **Next**.
4. Review your application details, the selected tag key-value pair, and the preview of the resources that will be added to the application.

> **Note**
>
> By default, creating an application using an existing tag key-value pair creates a tag-sync. After setup, tag-sync also continuously manages the application's resources, adding or removing resources as they are tagged or untagged with the specified key-value pair. You can manage tag-sync from the Manage resources page of the application.

10. If associating a CloudFormation stack, select the checkbox at the bottom of the page.

> **Note**
>
> Adding an CloudFormation stack to the application requires a stack update because all resources added to your application are tagged with the AWS application tag. Manual configurations performed after the stack was last updated may not be reflected after this update. This can cause downtime or other application issues. For more information, see Update behaviors of stack resources in the *CloudFormation User Guide*.

11. Choose **Create application**.

Onboard existing AppRegistry Applications in myApplications

You can onboard an existing AppRegistry application created before November 8, 2023 to get started with myApplications.

To onboard an existing AppRegistry application

1. Sign in to the AWS Management Console.
2. In the left sidebar, choose **myApplications**.
3. Use the searchbar to find your application.

4. Select your application.
5. Choose **Onboard *application name*.**
6. If associating a CloudFormation stack, select the checkbox in the alert box.
7. Choose **Onboard application**.

Viewing applications in myApplications

You can view your applications from myApplications or the Services menu. If viewing your applications from myApplications, you can view them across all AWS Regions or specific AWS Regions and their relevant information in a card or table view.

> **Note**
>
> You can also view applications added to your Favorites from the favorites menu. For more information, see Favorites in the AWS Management Console.

myApplications

To view applications in myApplications

1. Open the AWS Management Console.
2. In the left sidebar, choose **myApplications**.
3. In **Regions**, select **Current Region** or **Supported Regions**.
4. To find a specific application, enter its name, keywords, or description in the search bar.
5. (Optional) Your default view is the card view. To customize your application page:
    a. Select the gear icon.
    b. (Optional) Select your page size.
    c. (Optional) Choose card or table view.
    d. (Optional) Select your page size.
    e. (Optional) If using the table view, select the properties for your table view.
    f. (Optional) Toggle what application properties are visible and the order in which they appear.
    g. Choose **Confirm**.

Services menu

To view applications from the Services menu

1. Open the AWS Management Console.
2. In the navigation bar, choose **Services** (⁝⁝⁝).
3. Choose **All applications**.
4. Select an application.
5. (Optional) Select a view.
6. (Optional) Select a filter. You can filter your resources by **Properties** or by **Tags**. For more information, see Search query syntax reference for Resource Explorer in the *AWS Resource Explorer User Guide*.
7. (Optional) Select a resource to view it in the relevant service console.

> **Tip**
>
> You can continue browsing resources where you left off by choosing **Services** (⁝⁝⁝). Your applied search filters will also persist.

Editing applications in myApplications

Editing your application opens AppRegistry so you can update its description. You can also use AppRegistry to edit your application's tags and attribute groups.

To edit an application

1. Open the AWS Management Console.
2. In the left sidebar of the console, choose **myApplications**.
3. Select the application you want to edit.
4. On the myApplication dashboard, choose **Actions** and then choose **Edit application**.
5. In **Edit application**, make your desired changes to the description, tags, and attribute groups of your application.

> **Note**
>
> For more information about managing tags and attribute groups, see Managing tags and Editing attribute groups in the *AWS Service Catalog AppRegistry Administrator Guide*.

6. Choose **Update**.

Deleting applications in myApplications

You can delete applications if they are no longer needed. Before you delete an application, ensure you remove all associated resource shares and attribute groups that weren't created by an AWS service.

> **Note**
>
> Deleting an application doesn't impact your resources. Resources tagged with the AWS application tag will remain tagged.

To delete an application

1. Open the AWS Management Console.
2. In the left sidebar of the console, choose **myApplications**.
3. Select the application you want to delete.
4. On the myApplication dashboard, choose **Actions**.
5. Choose **Delete application**.
6. Confirm your deletion, and then choose **Delete**.

Creating code snippets in myApplications

myApplications creates code snippets for all of your applications. You can use code snippets to automatically add newly created resources to an application using Infrastructure as Code (IaC) tools. All added resources are tagged with the AWS application tag to associate it with your application.

To create a code snippet for your application

1. Open the AWS Management Console.
2. In the left sidebar of the console, choose **myApplications**.
3. Search for and select an application.
4. Choose **Actions**.
5. Choose **Get code snippet**.
6. Select a code snippet type.
7. Choose **Copy** to copy the code to your clipboard.
8. Paste your code into your IaC tool.

Resources in myApplications

In AWS, a resource is an entity you can work with. Examples include an Amazon EC2 instance, an AWS CloudFormation stack, or an Amazon S3 bucket. You can manage your resources in myApplications by adding and removing them from applications.

Topics

Adding resources in myApplications

Adding resources to your applications allows you to group them and manage their security, performance, and compliance. You can add resources to existing applications by searching for them and selecting them or by using existing tags and performing a tag-sync.

Search and select resources

To search and select resources

1. Open the AWS Management Console.
2. In the left sidebar of the console, choose **myApplications**.
3. Search for and select an application.

4. Choose **Manage resources**.
5. Choose **Add resources**.
6. (Optional) Choose a view.
7. Search for your resources. You can search by keyword, name or type, or choose a resource type.

> **Note**
>
> If you can't find the resource you're looking for, troubleshoot with AWS Resource Explorer. For more information, see Troubleshooting Resource Explorer search issues in the *Resource Explorer User Guide*.

8. Select the checkbox next to the resources you want to add.
9. Choose **Add**.

Automatically add resources using tags

When you create an application, you can bulk-onboard resources by specifying an existing tag key-value pair. With this method, AWS automatically applies the `awsApplication` tag to all of the resources, and creates a tag-sync for the application's resources by default. With tag-sync enabled, any resources that are tagged with the specified tag key-value pair are automatically added to the application.

To add resources using existing tags

1. Open the AWS Management Console.
2. In the left sidebar of the console, choose **myApplications**.
3. Choose **Manage resources**.
4. Choose **Create tag-sync**.
5. Select an existing tag key and value:
   a. Select the **Role** used to tag resources. For more information, see Tag-sync task required permissions in the *AWS Service Catalog AppRegistry Administrator Guide*.
   b. Select a **Tag key**.
   c. Select a **Tag value**.

d. Review and accept the **I acknowledge that Group Lifecycle Events will be enabled to create a tag sync** notice. GLE allows AWS to notice changes to the resources tagged with your key-value pair.

6. Choose **Create tag sync**.

Resolving tag-sync errors in myApplications

This section describes common tag-sync errors and how to resolve them. After attempting to resolve the error, you can retry the failed tag-sync task.

- **Insufficient permissions** — You do not have the required minimum permissions to start, update, or cancel the tag-sync. Review Tag-sync required permissions for more information. After ensuring the role you specify to perform the tag-sync has the minimum required permissions, retry the failed tag-sync task.
- **Already exists** — A task with this tag key-value pair already exists for this application. An application can support more than one tag-sync, but each tag-sync must have a different tag key-value pair. After you specify a different tag key-value pair, retry the failed tag-sync task.
- **Maximum limit reached** — You have reached the maximum of 100 tag-sync tasks per account, across all applications.

Removing resources in myApplications

You can remove resources to disassociate them from your application.

To remove resources

1. Open the AWS Management Console.
2. In the left sidebar of the console, choose **myApplications**.
3. Search for and select an application.
4. Choose **Manage resources**.
5. (Optional) Choose a view.
6. Search for your resources. You can search by keyword, name or type, or choose a resource type.

> **Note**
>
> If you can't find the resource you're looking for, troubleshoot with AWS Resource Explorer. For more information, see Troubleshooting Resource Explorer search issues in the *Resource Explorer User Guide*.

7. Choose **Remove**.
8. Confirm you want to remove the resource by choosing **Remove resources**.

Viewing resources in myApplications

You can view your application resources from myApplications and the **Services** menu.

myApplications

To view your resources in myApplications

1. Open the AWS Management Console.
2. Expand the left sidebar and choose **myApplications**.
3. Select an application.
4. In the **Resources** widget, view your resources.

Services menu

To view applications from the Services menu

1. Open the AWS Management Console.
2. In the navigation bar, choose **Services** (⋮⋮⋮).
3. Choose **All applications**.
4. Select an application.
5. (Optional) Select a view.
6. (Optional) Select a filter. You can filter your resources by **Properties** or by **Tags**. For more information, see Search query syntax reference for Resource Explorer in the *AWS Resource Explorer User Guide*.
7. (Optional) Select a resource to view it in the relevant service console.

> **Tip**
>
> You can continue browsing resources where you left off by choosing **Services** (⋮⋮⋮). Your applied search filters will also persist.

myApplications dashboard in AWS Console Home

Each application you create or onboard has its own myApplications dashboard. The myApplications dashboard contains cost, security, and operational widgets that surface insights from multiple AWS services. Each widget can also be favorited, reordered, removed, or resized. For more information, see Working with widgets in AWS Console Home.

Topics

Application dashboard setup widget

This widget contains a list of suggested getting started activities you can use to help you configure AWS services for managing application resources.

Application summary widget

This widget shows the name, description, and AWS application tag for your application. You can access and copy the application tag in Infrastructure as Code (IAC) to manually tag resources.

Compute widget

This widget displays information and metrics for compute resources, you add to your application. This includes total alarms and total compute resource types. The widget also shows resource performance metric trend charts from Amazon CloudWatch for Amazon EC2 instance CPU utilization and Lambda invocations.

Configuring the Compute widget

To populate data in the Compute widget, set up at least one Amazon EC2 instance or a Lambda function for your application. For more information, see the Amazon Elastic Compute Cloud Documentation and Getting started with Lambda in the *AWS Lambda Developer Guide*.

Cost and usage widget

This widget shows AWS cost and usage data for your application resources. You can use this data to compare monthly costs and view cost breakdowns by AWS service. This widget only summarizes costs for resources tagged with the AWS application tag, excluding taxes, fees, and other shared costs not directly associated with a resource. Costs shown are unblended and updated at least once every 24 hours. FOr more information, see Analyzing your costs with AWS Resource Explorer in the *AWS Cost Management User Guide*.

Configuring the Cost and usage widget

To configure the Cost and usage widget, enable AWS Cost Explorer Service for your application and account. This service is offered at no additional charge and there are no setup fees or upfront commitment. For more information, see Enabling Cost Explorer in the *AWS Cost Management User Guide*.

AWS Security widget

This widget displays security findings from AWS Security for your application. AWS Security provides a comprehensive view of security findings for your application in AWS. You can access recent priority findings by severity, monitor their security posture, access recent critical or high severity findings, and gain insight for next steps. For more information, see AWS Security Hub CSPM.

Configuring the AWS Security widget

To configure the AWS Security widget, set up AWS Security Hub CSPM for your application and account. For more information, see What is AWS Security Hub CSPM? in the *AWS Security Hub*

CSPM User Guide. For pricing information, see AWS Security Hub CSPM free trial, usage, and pricing in the *AWS Security Hub CSPM User Guide*.

AWS Security Hub CSPM requires you to configure AWS Config Recording. This service provides a detailed view of the resources associated with your AWS account. For more information, see AWS Systems Manager in the *AWS Systems Manager User Guide*.

AWS Resiliency widget

This widget displays resiliency details from AWS Resilience Hub for your applications. After initiating an assessment, AWS Resiliency Hub analyzes your applications' resiliency posture by evaluating their resources against a pre-defined resiliency policy. You can access metrics like resiliency score, policy breaches, policy drifts, resource drifts, and your resliency score history. Your applications are assessed daily for enhanced tracking, but you can disable this at any time. For more information, see AWS Resilience Hub. For pricing information, see AWS Resilience Hub pricing.

Configuring the AWS Resiliency widget

To configure the AWS Resiliency widget, add an application. For more information, see What is AWS Resilience Hub? in the *AWS Resilience Hub User Guide*.

Resources widget

This widget uses AWS Resource Explorer to show resources you have added to your application within a view. You can also use this widget to search or filter your resources using resource metadata like names, tags, and IDs. For more information, see AWS Resource Explorer.

Configuring the Resources widget

To configure the resources widget, onboard with Resource Explorer. For more information, see Getting started with Resource Explorer in the *AWS Resource Explorer User Guide*.

DevOps widget

This widget shows operational insights so you can assess compliance and take action for your application. These insights include:

- Fleet management
- State management

- Patch management
- Configuration and OpsItems management

Configuring the DevOps widget

To configure the DevOps widget, enable AWS Systems Manager OpsCenter for your application and account. For more information, see Getting started with Systems Manager Explorer and OpsCenter in the *AWS Systems Manager User Guide*. Enabling OpsCenter allows AWS Systems Manager Explorer to configure AWS Config and Amazon CloudWatch so that their events automatically create OpsItems based on commonly-used rules and events. For more information, see Set up OpsCenter in the *AWS Systems Manager User Guide*.

You can configure your instances for Systems Manager agents to run and apply permissions to enable patch scanning. For more information, see AWS Systems Manager Quick Setup in the *AWS Systems Manager User Guide*.

You can also set up automated patching of Amazon EC2 instances for your application by setting up AWS Systems Manager Patch Manager. For more information, see Using Quick Setup patch policies in the *AWS Systems Manager User Guide*.

For pricing information, see AWS Systems Manager pricing.

Monitoring and operations widget

This widget shows:

- Alarms and alerts for resources associated with your application
- Application service level objectives (SLOs) and metrics
- Available AWS Application Signals metrics

Configuring the Monitoring and operations widget

To configure the Monitoring and operations widget, create CloudWatch alarms and canaries in your AWS account. For more information, see Using Amazon CloudWatch alarms and Creating a canary in the *Amazon CloudWatch User Guide*. For CloudWatch alarm and synthetic canary pricing, see Amazon CloudWatch pricing and the AWS Cloud Operations and Migrations Blog respectively.

For more information about CloudWatch Application Signals, see Enable Amazon CloudWatch Application Signals in the *Amazon CloudWatch User Guide*.

Tags widget

This widget displays all tags associated with your application. You can use this widget to track and manage application metadata (criticality, environment, cost center). For more information, see What are tags? in the *Best practices for Tagging AWS Resources AWS Whitepaper*.

Chatting with Amazon Q Developer in AWS Console Home

Amazon Q Developer is a generative artificial intelligence (AI) powered conversational assistant that can help you understand, build, extend, and operate AWS applications. You can ask Amazon Q any questions about AWS, including questions about AWS architecture, your AWS resources, best practices, documentation, and more. You can also create support cases and receive assistance from a live agent. For more information, see What is Amazon Q? in the *Amazon Q Developer User Guide*.

Get started with Amazon Q

You can start chatting with Amazon Q in the AWS Management Console, AWS Documentation websites, AWS websites, or the AWS Console Mobile Application by choosing the hexagonal Amazon Q icon. For more information, see Get started with Amazon Q Developer in the *Amazon Q Developer User Guide*.

Example questions

Following are some example questions you can ask Amazon Q:

- `How do I get billing support?`
- `How do I create an EC2 instance?`
- `How do I troubleshoot a "Failed to load" error?`
- `How do I close an AWS account?`
- `Can you connect me with a person?`

AWS Management Console Private Access

AWS Management Console Private Access is an advanced security feature to control access to the AWS Management Console. Console Private Access is useful when you want to prevent users from signing in to unexpected AWS accounts from within your network. With this feature, you can limit access to the AWS Management Console only to a specified set of known AWS accounts when the traffic originates from within your network. Console Private Access is also useful when you want to ensure that all calls from the AWS Management Console to AWS services originate from within your network and from allowed accounts.

Topics

Supported AWS Regions, service consoles, and features for Private Access

AWS Management Console Private Access supports only a subset of Regions and AWS services. Unsupported service consoles will be inactive in the AWS Management Console. In addition, certain AWS Management Console features might be disabled when using AWS Management Console Private Access, for example, the Default Region selection in Unified Settings.

The following Regions and service consoles are supported.

Supported Regions

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)

- US West (Oregon)
- Asia Pacific (Hyderabad)
- Asia Pacific (Mumbai)
- Asia Pacific (Seoul)
- Asia Pacific (Osaka)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Malaysia)
- Asia Pacific (Thailand)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- South America (São Paulo)
- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Jakarta)
- Asia Pacific (Melbourne)
- Canada West (Calgary)
- Mexico (Central)
- Europe (Milan)
- Europe (Spain)
- Europe (Zurich)
- Middle East (Bahrain)
- Middle East (UAE)

- Israel (Tel Aviv)

Supported service consoles

- Amazon API Gateway
- AWS App Mesh
- AWS Application Migration Service
- AWS Artifact
- Amazon Athena
- AWS Audit Manager
- AWS Auto Scaling
- AWS Batch
- AWS Billing Conductor
- AWS Billing and Cost Management
- AWS Budgets
- AWS Certificate Manager
- AWS Cloud Map
- AWS CloudFormation
- Amazon CloudFront
- AWS CloudTrail
- Amazon CloudWatch
- AWS CodeArtifact
- AWS CodeBuild
- AWS CodeCommit
- AWS CodeDeploy
- Amazon CodeGuru
- AWS CodePipeline
- Amazon Comprehend
- Amazon Comprehend Medical
- AWS Compute Optimizer

- AWS Console Home
- AWS Control Tower
- Amazon DataZone
- AWS Database Migration Service
- AWS DataSync
- AWS DeepRacer
- AWS Direct Connect
- AWS Directory Service
- Amazon DocumentDB
- Amazon DynamoDB
- Amazon EC2
- Amazon EC2 Global View
- EC2 Image Builder
- Amazon EC2 Instance Connect
- Amazon Elastic Container Registry
- Amazon Elastic Container Service
- AWS Elastic Disaster Recovery
- Amazon Elastic File System
- Amazon Elastic Kubernetes Service
- Elastic Load Balancing
- Amazon ElastiCache
- Amazon EMR
- Amazon EventBridge
- AWS Firewall Manager
- Amazon GameLift Servers
- AWS Glue
- AWS Global Accelerator
- AWS Glue DataBrew
- AWS Ground Station

- Amazon GuardDuty
- AWS IAM Identity Center
- AWS Identity and Access Management
- AWS Identity and Access Management Access Analyzer
- Amazon Inspector
- Amazon Kendra
- AWS Key Management Service
- Amazon Kinesis
- Amazon Managed Service for Apache Flink
- Amazon Data Firehose
- Amazon Kinesis Data Streams
- Amazon Kinesis Video Streams
- AWS Lambda
- Amazon Lex
- AWS License Manager
- Amazon Managed Grafana
- Amazon Macie
- Amazon Managed Streaming for Apache Kafka
- Amazon Managed Workflows for Apache Airflow (MWAA)
- AWS Migration Hub Strategy Recommendations
- Amazon MQ
- Network Access Analyzer
- AWS Network Firewall
- AWS Network Manager
- Amazon OpenSearch Service
- AWS Organizations
- AWS Private Certificate Authority
- Public Health Dashboard
- Amazon Rekognition

- Amazon Relational Database Service
- AWS Resource Access Manager
- AWS Resource Groups and Tag Editor
- Amazon Route 53 Resolver
- Amazon Route 53 Resolver DNS Firewall
- Amazon S3 on Outposts
- Amazon SageMaker
- Amazon SageMaker Runtime
- Amazon SageMaker AI Synthetic Data
- AWS Secrets Manager
- AWS Service Catalog
- AWS Security Hub CSPM
- Service Quotas
- AWS Signer
- Amazon Simple Email Service
- Amazon SNS
- Amazon Simple Queue Service
- Amazon Simple Storage Service (Amazon S3)
- AWS SQL Workbench
- AWS Step Functions
- AWS Storage Gateway
- Support
- AWS Systems Manager
- Amazon Timestream
- AWS Transfer Family
- AWS Trusted Advisor
- Unified Settings
- Amazon VPC IP Address Manager
- Amazon Virtual Private Cloud

- Amazon WorkSpaces Thin client

Overview of AWS Management Console Private Access security controls

Account restrictions on the AWS Management Console from your network

AWS Management Console Private Access is useful in scenarios when you want to limit access to the AWS Management Console from your network only to a specified set of known AWS accounts in your organization. By doing so, you can prevent users from signing in to unexpected AWS accounts from within your network. You can implement these controls using the AWS Management Console VPC endpoint policy. For more information, see Implementing service control policies and VPC endpoint policies.

Connectivity from your network to the internet

Internet connectivity from your network is still required to access assets used by the AWS Management Console, such as static content (JavaScript, CSS, images), and all AWS services not enabled by AWS PrivateLink. For a list of the top-level domains used by the AWS Management Console, see Troubleshooting.

> **Note**
>
> Currently, AWS Management Console Private Access doesn't support endpoints such as `status.aws.amazon.com`, `health.aws.amazon.com`, and `docs.aws.amazon.com`. You will need to route these domains to the public internet.

Required VPC endpoints and DNS configuration

AWS Management Console Private Access requires the following two VPC endpoints per Region. Replace *region* with your own Region information.

1. com.amazonaws.*region*.console for AWS Management Console
2. com.amazonaws.*region*.signin for AWS Sign-In

> **Note**
>
> Always provision infrastructure and networking connectivity to the US East (N. Virginia) (us-east-1) Region, regardless of other Regions you use with the AWS Management Console. You can use AWS Transit Gateway to set up connectivity between the US East (N. Virginia) and every other Region. For more information, see Getting started with transit gateways in the *Amazon VPC Transit Gateways guide*. You can also use Amazon VPC peering. For more information, see What is VPC peering in the *Amazon VPC Peering Guide*. To compare these options, see Amazon VPC-to-Amazon VPC connectivity options in the *Amazon Virtual Private Cloud Connectivity Options whitepaper*.

Topics

DNS configuration for AWS Management Console and AWS Sign-In

To route your network traffic to respective VPC endpoints, configure DNS records in the network from which your users will be accessing the AWS Management Console. These DNS records will direct your users browser traffic toward the VPC endpoints you created.

You can create a single hosted zone. However, endpoints such as `health.aws.amazon.com` and `docs.aws.amazon.com` won't be accessible because they don't have VPC endpoints. You will need to route these domains to the public internet. We recommend that you create two private hosted zones per Region, one for `signin.aws.amazon.com` and one for `console.aws.amazon.com` with the following CNAME records:

- Sign-In
  - *`region`*.signin.aws.amazon.com pointing to the AWS Sign-In VPC endpoint in the signin DNS zone where *`region`* is the desired Region
  - signin.aws.amazon.com pointing to AWS Sign-In VPC endpoint in US East (N. Virginia) (us-east-1)
- Console
  - *`region`*.console.aws.amazon.com pointing to the AWS Management Console VPC endpoint in the console DNS zone where *`region`* is the desired Region

- *.*region*.console.aws.amazon.com pointing to the AWS Management Console VPC endpoint in the console DNS zone where *region* is the desired Region
- *.*region*.console.aws.amazon.com pointing to the AWS Management Console VPC endpoint in the console DNS zone
- Regionless CNAME records for the US East (N. Virginia) Region only. You always have to set up the US East (N. Virginia) Region.
  - signin.aws.amazon.com pointing to AWS Sign-In VPC endpoint in US East (N. Virginia) (us-east-1)
  - *.console.aws.amazon.com pointing to AWS Management Console VPC endpoint in US East (N. Virginia) (us-east-1)

For instructions on creating a CNAME record, see Working with records in the *Amazon Route 53 Developer Guide*.

Some AWS consoles, including Amazon S3, use different patterns for their DNS names. The following are two examples:

- support.console.aws.amazon.com
- s3.console.aws.amazon.com

To be able to direct this traffic to your AWS Management Console VPC endpoint, you need to add those names individually. We recommend that you configure routing for all endpoints for a fully private experience. However, this isn't required to use AWS Management Console Private Access.

The following `json` files contain the full list of AWS services and console endpoints to configure per Region. Use the `PrivateIpv4DnsNames` field under the `com.amazonaws.`*region*`.console` endpoint for the DNS names.

- https://configuration.private-access.console.amazonaws.com/us-east-1.config.json
- https://configuration.private-access.console.amazonaws.com/us-east-2.config.json
- https://configuration.private-access.console.amazonaws.com/us-west-2.config.json
- https://configuration.private-access.console.amazonaws.com/ap-northeast-1.config.json
- https://configuration.private-access.console.amazonaws.com/ap-northeast-2.config.json
- https://configuration.private-access.console.amazonaws.com/ap-southeast-1.config.json
- https://configuration.private-access.console.amazonaws.com/ap-southeast-2.config.json
- https://configuration.private-access.console.amazonaws.com/ap-south-1.config.json
- https://configuration.private-access.console.amazonaws.com/ap-south-2.config.json

- https://configuration.private-access.console.amazonaws.com/ca-central-1.config.json
- https://configuration.private-access.console.amazonaws.com/eu-central-1.config.json
- https://configuration.private-access.console.amazonaws.com/eu-west-1.config.json
- https://configuration.private-access.console.amazonaws.com/eu-west-2.config.json
- https://configuration.private-access.console.amazonaws.com/il-central-1.config.json

> **Note**
>
> This list is updated each month as we add additional endpoints to the scope of AWS Management Console Private Access. To keep your private hosted zones updated, periodically pull the preceding list of files.

If you use Route 53 to configure your DNS, go to https://console.aws.amazon.com/route53/v2/hostedzones# to verify the DNS setup. For each Private Hosted Zone in Route 53, verify that the following record sets are present.

- console.aws.amazon.com
- signin.aws.amazon.com
- *.*region*.console.aws.amazon.com
- *region*.console.aws.amazon.com
- *.*region*.console.aws.amazon.com
- signin.aws.amazon.com
- *region*.signin.aws.amazon.com
- Additional records present in the previously listed JSON files

VPC endpoints and DNS configuration for AWS services in the AWS Management Console

The AWS Management Console calls AWS services through a combination of direct browser requests and requests that are proxied by web servers. To direct this traffic to your AWS Management Console VPC endpoint, you must add the VPC endpoint and configure DNS for each dependent AWS service.

The following json files list the AWS PrivateLink supported AWS services that are available for you to use. If a service doesn't integrate with AWS PrivateLink, it isn't included in these files.

- https://configuration.private-access.console.amazonaws.com/us-east-1.config.json
- https://configuration.private-access.console.amazonaws.com/us-east-2.config.json
- https://configuration.private-access.console.amazonaws.com/us-west-2.config.json
- https://configuration.private-access.console.amazonaws.com/ap-northeast-1.config.json
- https://configuration.private-access.console.amazonaws.com/ap-northeast-2.config.json
- https://configuration.private-access.console.amazonaws.com/ap-southeast-1.config.json
- https://configuration.private-access.console.amazonaws.com/ap-southeast-2.config.json
- https://configuration.private-access.console.amazonaws.com/ap-south-1.config.json
- https://configuration.private-access.console.amazonaws.com/ap-south-2.config.json
- https://configuration.private-access.console.amazonaws.com/ca-central-1.config.json
- https://configuration.private-access.console.amazonaws.com/eu-central-1.config.json
- https://configuration.private-access.console.amazonaws.com/eu-west-1.config.json
- https://configuration.private-access.console.amazonaws.com/eu-west-2.config.json
- https://configuration.private-access.console.amazonaws.com/il-central-1.config.json

Use the `ServiceName` field for the corresponding service's VPC endpoint to add to your VPC.

> **Note**
>
> We update this list each month as we add support for AWS Management Console Private Access to more service consoles. To stay current, periodically pull the preceding list of files and update your VPC endpoints.

Implementing service control policies and VPC endpoint policies

You can use service control policies (SCPs) and VPC endpoint policies for AWS Management Console Private Access to limit the set of accounts that are allowed to use the AWS Management Console from within your VPC and its connected on-premises networks.

Topics

Using AWS Management Console Private Access with AWS Organizations service control policies

If your AWS organization is using a service control policy (SCP) that allows specific services, you must add `signin:*` to the allowed actions. This permission is needed because signing in to the AWS Management Console over a Private Access VPC endpoint performs an IAM authorization that the SCP blocks without the permission. As an example, the following service control policy allows the Amazon EC2 and CloudWatch services to be used in the organization, including when they are accessed using an AWS Management Console Private Access endpoint.

```
{
  "Effect": "Allow",
  "Action": [
    "signin:*",
    "ec2:*",
    "cloudwatch:*",
    ... Other services allowed
  },
  "Resource": "*"
}
```

For more information about SCPs, see Service control policies (SCPs) in the *AWS Organizations User Guide*.

Allow AWS Management Console use for expected accounts and organizations only (trusted identities)

AWS Management Console and AWS Sign-In support a VPC endpoint policy that specifically controls the identity of the signed-in account.

Unlike other VPC endpoint policies, the policy is evaluated before authentication. As a result, it specifically controls sign-in and use of the authenticated session only, and not any AWS service-

specific actions that the session takes. For example, as the session accesses an AWS service console, such as the Amazon EC2 console, these VPC endpoint policies will not be evaluated against the Amazon EC2 actions that are taken to display that page. Rather, you can use the IAM policies associated with the signed-in IAM Principal to control its permission to AWS service actions.

> **Note**
>
> VPC endpoint policies for AWS Management Console and SignIn VPC endpoints support only a limited subset of policy formulations. Every `Principal` and `Resource` should be set to `*` and the `Action` should be either `*` or `signin:*`. You control access to VPC endpoints using `aws:PrincipalOrgId` and `aws:PrincipalAccount` condition keys.

The following policies are recommended for both the Console and SignIn VPC endpoints.

This VPC endpoint policy allows sign-in to AWS accounts in the specified AWS organization and blocks sign-in to any other accounts.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgId": "o-xxxxxxxxxx"
        }
      }
    }
  ]
}
```

This VPC endpoint policy limits sign-in to a list of specific AWS accounts and blocks sign-in to any other accounts.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalAccount": [ "111122223333", "222233334444" ]
        }
      }
    }
  ]
}
```

Polices that limit AWS accounts or an organization on the AWS Management Console and Sign-In VPC endpoints are evaluated at the time of sign-in and are periodically re-evaluated for existing sessions.

Implementing identity-based policies and other policy types

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. This page describes how policies work when used together with AWS Management Console Private Access.

Supported AWS global condition context keys

AWS Management Console Private Access does not support `aws:SourceVpce` and `aws:VpcSourceIp` AWS global condition context keys. You can use the `aws:SourceVpc` IAM condition in your policies instead, when using AWS Management Console Private Access.

How AWS Management Console Private Access works with aws:SourceVpc

This section describes the various network paths that the requests generated by your AWS Management Console can take to AWS services. In general, AWS service consoles are implemented with a mix of direct browser requests and requests that are proxied by the AWS Management Console web servers to AWS services. These implementations are subject to change without notice. If your security requirements include access to AWS services using VPC endpoints, we recommend that you configure VPC endpoints for all of the services that you intend to use from VPC, whether directly or through AWS Management Console Private Access. Furthermore, you must use the `aws:SourceVpc` IAM condition in your policies rather than specific `aws:SourceVpce` values with the AWS Management Console Private Access feature. This section provides details about how the different network paths work.

After a user signs in to the AWS Management Console, they make requests to AWS services through a combination of direct browser requests and requests that are proxied by AWS Management Console web servers to AWS servers. For example, CloudWatch graph data requests are made directly from the browser. Whereas some AWS service console requests, such as Amazon S3, are proxied by the web server to Amazon S3.

For direct browser requests, using AWS Management Console Private Access does not change anything. As before, the request reaches the service through whatever network path the VPC has configured to reach monitoring.region.amazonaws.com. If the VPC is configured with a VPC endpoint for com.amazonaws.region.monitoring, the request will reach CloudWatch through that CloudWatch VPC endpoint. If there is no VPC endpoint for CloudWatch, the request will reach CloudWatch at its public endpoint, by way of an Internet Gateway on the VPC. Requests that arrive at CloudWatch by way of the CloudWatch VPC endpoint will have the IAM conditions `aws:SourceVpc` and `aws:SourceVpce` set to their respective values. Those that reach CloudWatch through its public endpoint will have `aws:SourceIp` set to the source IP address of the request. For more information about these IAM condition keys, see Global condition keys in the *IAM User Guide*.

For requests that are proxied by the AWS Management Console web server, such as the request that the Amazon S3 console makes to list your buckets when you visit the Amazon S3 console, the network path is different. These requests aren't initiated from your VPC and therefore don't use the VPC endpoint you may have configured on your VPC for that service. Even if you have a VPC endpoint for Amazon S3 in this case, your session's request to Amazon S3 to list the buckets doesn't use the Amazon S3 VPC endpoint. However, when you use AWS Management Console

Private Access with supported services, these requests (for example, to Amazon S3) will include the `aws:SourceVpc` condition key in their request context. The `aws:SourceVpc` condition key will be set to the VPC ID where your AWS Management Console Private Access endpoints for sign-in and console are deployed. So, if you are using `aws:SourceVpc` restrictions in your identity-based policies, you must add the VPC ID of this VPC that is hosting the AWS Management Console Private Access sign-in and console endpoints. The `aws:SourceVpce` condition will be set to the respective sign-in or console VPC endpoint IDs.

> **Note**
>
> If your users require access to service consoles that aren't supported by AWS Management Console Private Access, you must include a list of your expected public network addresses (such as your on-premises network range) using the `aws:SourceIP` condition key in the users' identity-based policies.

How different network paths are reflected in CloudTrail

Different network paths used by requests generated by your AWS Management Console are reflected in your CloudTrail event history.

For direct browser requests, using AWS Management Console Private Access doesn't change anything. CloudTrail events will include details about the connection, like the VPC endpoint ID that was used to make the service API call.

For requests that are proxied by the AWS Management Console web server, CloudTrail events will not include any VPC related details. However, initial requests to AWS Sign-In that are required to establish the browser session, such as the `AwsConsoleSignIn` event type, will include the AWS Sign-In VPC endpoint ID in the event details.

Try AWS Management Console Private Access

This section describes how to set up and test AWS Management Console Private Access in a new account.

AWS Management Console Private Access is an advanced security feature and requires prior knowledge about networking and setting up VPCs. This topic describes how you can try out AWS Management Console Private Access without a full scale infrastructure.

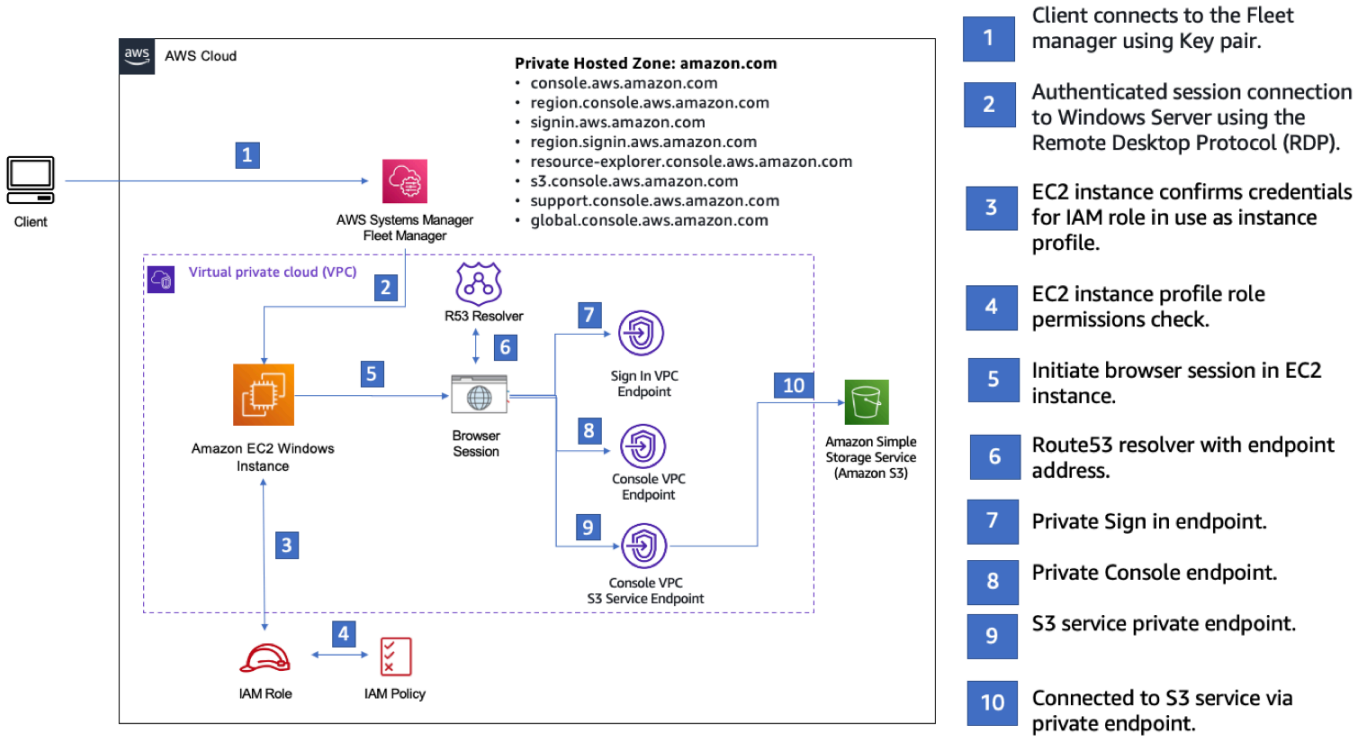
Topics

Test setup with Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2), provides scalable computing capacity in the Amazon Web Services cloud. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. In this setup, we use Fleet Manager, a capability of AWS Systems Manager, to connect to an Amazon EC2 Windows instance using the Remote Desktop Protocol (RDP).

This guide demonstrates a test environment to set up and experience an AWS Management Console Private Access connection to Amazon Simple Storage Service from an Amazon EC2 instance. This tutorial uses CloudFormation to create and configure the network setup to be used by Amazon EC2 to visualize this feature.

The following diagram describes the workflow for using Amazon EC2 to access an AWS Management Console Private Access setup. It shows how a user is connected to Amazon S3 using a private endpoint.



Copy the following CloudFormation template and save it to a file that you will use in step three of the *To set up a network* procedure.

> **Note**
>
> This CloudFormation template uses configurations that are currently not supported in the Israel (Tel Aviv) Region.

AWS Management Console Private Access environment Amazon EC2 CloudFormation template

```
Description: |
  AWS Management Console Private Access.
Parameters:
  VpcCIDR:
    Type: String
    Default: 172.16.0.0/16
    Description: CIDR range for VPC

  Ec2KeyPair:
    Type: AWS::EC2::KeyPair::KeyName
```

```
    Description: The EC2 KeyPair to use to connect to the Windows instance

  PublicSubnet1CIDR:
    Type: String
    Default: 172.16.1.0/24
    Description: CIDR range for Public Subnet A

  PublicSubnet2CIDR:
    Type: String
    Default: 172.16.0.0/24
    Description: CIDR range for Public Subnet B

  PublicSubnet3CIDR:
    Type: String
    Default: 172.16.2.0/24
    Description: CIDR range for Public Subnet C

  PrivateSubnet1CIDR:
    Type: String
    Default: 172.16.4.0/24
    Description: CIDR range for Private Subnet A

  PrivateSubnet2CIDR:
    Type: String
    Default: 172.16.5.0/24
    Description: CIDR range for Private Subnet B

  PrivateSubnet3CIDR:
    Type: String
    Default: 172.16.3.0/24
    Description: CIDR range for Private Subnet C

  LatestWindowsAmiId:
    Type: 'AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>'
    Default: '/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-Base'

  InstanceTypeParameter:
    Type: String
    Default: 't3.medium'


Resources:

#########################
```

```
# VPC AND SUBNETS
#########################

  AppVPC:
    Type: 'AWS::EC2::VPC'
    Properties:
      CidrBlock: !Ref VpcCIDR
      InstanceTenancy: default
      EnableDnsSupport: true
      EnableDnsHostnames: true

  PublicSubnetA:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      AvailabilityZone:
        Fn::Select:
          - 0
          - Fn::GetAZs: ""

  PublicSubnetB:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PublicSubnet2CIDR
      MapPublicIpOnLaunch: true
      AvailabilityZone:
        Fn::Select:
          - 1
          - Fn::GetAZs: ""

  PublicSubnetC:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PublicSubnet3CIDR
      MapPublicIpOnLaunch: true
      AvailabilityZone:
        Fn::Select:
          - 2
          - Fn::GetAZs: ""
```

```
  PrivateSubnetA:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PrivateSubnet1CIDR
      AvailabilityZone:
        Fn::Select:
          - 0
          - Fn::GetAZs: ""

  PrivateSubnetB:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PrivateSubnet2CIDR
      AvailabilityZone:
        Fn::Select:
          - 1
          - Fn::GetAZs: ""

  PrivateSubnetC:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PrivateSubnet3CIDR
      AvailabilityZone:
        Fn::Select:
          - 2
          - Fn::GetAZs: ""

  InternetGateway:
    Type: AWS::EC2::InternetGateway

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref AppVPC

  NatGatewayEIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment

  NatGateway:
```

```
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGatewayEIP.AllocationId
      SubnetId: !Ref PublicSubnetA

########################
# Route Tables
########################

  PrivateRouteTable:
    Type: 'AWS::EC2::RouteTable'
    Properties:
      VpcId: !Ref AppVPC

  DefaultPrivateRoute:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway

  PrivateSubnetRouteTableAssociation1:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref PrivateRouteTable
      SubnetId: !Ref PrivateSubnetA

  PrivateSubnetRouteTableAssociation2:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref PrivateRouteTable
      SubnetId: !Ref PrivateSubnetB

  PrivateSubnetRouteTableAssociation3:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref PrivateRouteTable
      SubnetId: !Ref PrivateSubnetC

  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref AppVPC
```

```
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnetARouteTableAssociation1:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnetA

  PublicSubnetBRouteTableAssociation2:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnetB

  PublicSubnetBRouteTableAssociation3:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnetC


##########################
# SECURITY GROUPS
##########################

  VPCEndpointSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow TLS for VPC Endpoint
      VpcId: !Ref AppVPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: !GetAtt AppVPC.CidrBlock

  EC2SecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
```

```
    Properties:
      GroupDescription: Default EC2 Instance SG
      VpcId: !Ref AppVPC

#########################
# VPC ENDPOINTS
#########################

  VPCEndpointGatewayS3:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.s3'
      VpcEndpointType: Gateway
      VpcId: !Ref AppVPC
      RouteTableIds:
        - !Ref PrivateRouteTable

  VPCEndpointInterfaceSSM:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      VpcEndpointType: Interface
      PrivateDnsEnabled: false
      SubnetIds:
        - !Ref PrivateSubnetA
        - !Ref PrivateSubnetB
      SecurityGroupIds:
        - !Ref VPCEndpointSecurityGroup
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.ssm'
      VpcId: !Ref AppVPC

  VPCEndpointInterfaceEc2messages:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      VpcEndpointType: Interface
      PrivateDnsEnabled: false
      SubnetIds:
        - !Ref PrivateSubnetA
        - !Ref PrivateSubnetB
        - !Ref PrivateSubnetC
      SecurityGroupIds:
        - !Ref VPCEndpointSecurityGroup
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.ec2messages'
      VpcId: !Ref AppVPC
```

```
  VPCEndpointInterfaceSsmmessages:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      VpcEndpointType: Interface
      PrivateDnsEnabled: false
      SubnetIds:
        - !Ref PrivateSubnetA
        - !Ref PrivateSubnetB
        - !Ref PrivateSubnetC
      SecurityGroupIds:
        - !Ref VPCEndpointSecurityGroup
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.ssmmessages'
      VpcId: !Ref AppVPC

  VPCEndpointInterfaceSignin:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      VpcEndpointType: Interface
      PrivateDnsEnabled: false
      SubnetIds:
        - !Ref PrivateSubnetA
        - !Ref PrivateSubnetB
        - !Ref PrivateSubnetC
      SecurityGroupIds:
        - !Ref VPCEndpointSecurityGroup
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.signin'
      VpcId: !Ref AppVPC

  VPCEndpointInterfaceConsole:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      VpcEndpointType: Interface
      PrivateDnsEnabled: false
      SubnetIds:
        - !Ref PrivateSubnetA
        - !Ref PrivateSubnetB
        - !Ref PrivateSubnetC
      SecurityGroupIds:
        - !Ref VPCEndpointSecurityGroup
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.console'
      VpcId: !Ref AppVPC

##########################
# ROUTE53 RESOURCES
```

```
#########################

  ConsoleHostedZone:
    Type: "AWS::Route53::HostedZone"
    Properties:
      HostedZoneConfig:
        Comment: 'Console VPC Endpoint Hosted Zone'
      Name: 'console.aws.amazon.com'
      VPCs:
        -
          VPCId: !Ref AppVPC
          VPCRegion: !Ref "AWS::Region"

  ConsoleRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: 'console.aws.amazon.com'
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  GlobalConsoleRecord:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: 'global.console.aws.amazon.com'
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  ConsoleS3ProxyRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: 's3.console.aws.amazon.com'
      AliasTarget:
```

```
       DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
       HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
     Type: A

 ConsoleSupportProxyRecordGlobal:
   Type: AWS::Route53::RecordSet
   Properties:
     HostedZoneId: !Ref 'ConsoleHostedZone'
     Name: "support.console.aws.amazon.com"
     AliasTarget:
       DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
       HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
     Type: A

 ExplorerProxyRecordGlobal:
   Type: AWS::Route53::RecordSet
   Properties:
     HostedZoneId: !Ref 'ConsoleHostedZone'
     Name: "resource-explorer.console.aws.amazon.com"
     AliasTarget:
       DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
       HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
     Type: A

 WidgetProxyRecord:
   Type: AWS::Route53::RecordSet
   Properties:
     HostedZoneId: !Ref 'ConsoleHostedZone'
     Name: "*.widget.console.aws.amazon.com"
     AliasTarget:
       DNSName: !Select ["1", !Split [":", !Select ["0", !GetAtt
VPCEndpointInterfaceConsole.DnsEntries],],]
       HostedZoneId: !Select ["0", !Split [":", !Select ["0", !GetAtt
VPCEndpointInterfaceConsole.DnsEntries],],]
     Type: A

 ConsoleRecordRegional:
   Type: AWS::Route53::RecordSet
```

```
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: !Sub "${AWS::Region}.console.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  ConsoleRecordRegionalMultiSession:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: !Sub "*.${AWS::Region}.console.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  SigninHostedZone:
    Type: "AWS::Route53::HostedZone"
    Properties:
      HostedZoneConfig:
        Comment: 'Signin VPC Endpoint Hosted Zone'
      Name: 'signin.aws.amazon.com'
      VPCs:
        -
          VPCId: !Ref AppVPC
          VPCRegion: !Ref "AWS::Region"

  SigninRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'SigninHostedZone'
      Name: 'signin.aws.amazon.com'
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceSignin.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceSignin.DnsEntries]]]
      Type: A
```

```
  SigninRecordRegional:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'SigninHostedZone'
      Name: !Sub "${AWS::Region}.signin.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
 VPCEndpointInterfaceSignin.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
 VPCEndpointInterfaceSignin.DnsEntries]]]
      Type: A

#########################
# EC2 INSTANCE
#########################

  Ec2InstanceRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore

  Ec2InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: /
      Roles:
       - !Ref Ec2InstanceRole

  EC2WinInstance:
    Type: 'AWS::EC2::Instance'
    Properties:
```

```
      ImageId: !Ref LatestWindowsAmiId
      IamInstanceProfile: !Ref Ec2InstanceProfile
      KeyName: !Ref Ec2KeyPair
      InstanceType:
        Ref: InstanceTypeParameter
      SubnetId: !Ref PrivateSubnetA
      SecurityGroupIds:
        - Ref: EC2SecurityGroup
      BlockDeviceMappings:
        - DeviceName: /dev/sda1
          Ebs:
            VolumeSize: 50
      Tags:
      - Key: "Name"
        Value: "Console VPCE test instance"
```

To set up a network

1. Sign in to the management account for your organization and open the CloudFormation console.
2. Choose **Create stack**.
3. Choose **With new resources (standard)**. Upload the CloudFormation template file that you previously created, and choose **Next**.
4. Enter a name for the stack, such as **`PrivateConsoleNetworkForS3`**, then choose **Next**.
5. For **VPC and subnets**, enter your preferred IP CIDR ranges, or use the provided default values. If you use the default values, verify that they don't overlap with existing VPC resources in your AWS account.
6. For the **Ec2KeyPair** parameter, select one from the existing Amazon EC2 key pairs in your account. If you don't have an existing Amazon EC2 key pair, you must create one before proceeding to the next step. For more information, see Create a key pair using Amazon EC2 in the *Amazon EC2 User Guide*.
7. Choose **Create stack**.
8. After the stack is created, choose the **Resources** tab to view the resources that have been created.

To connect to the Amazon EC2 instance

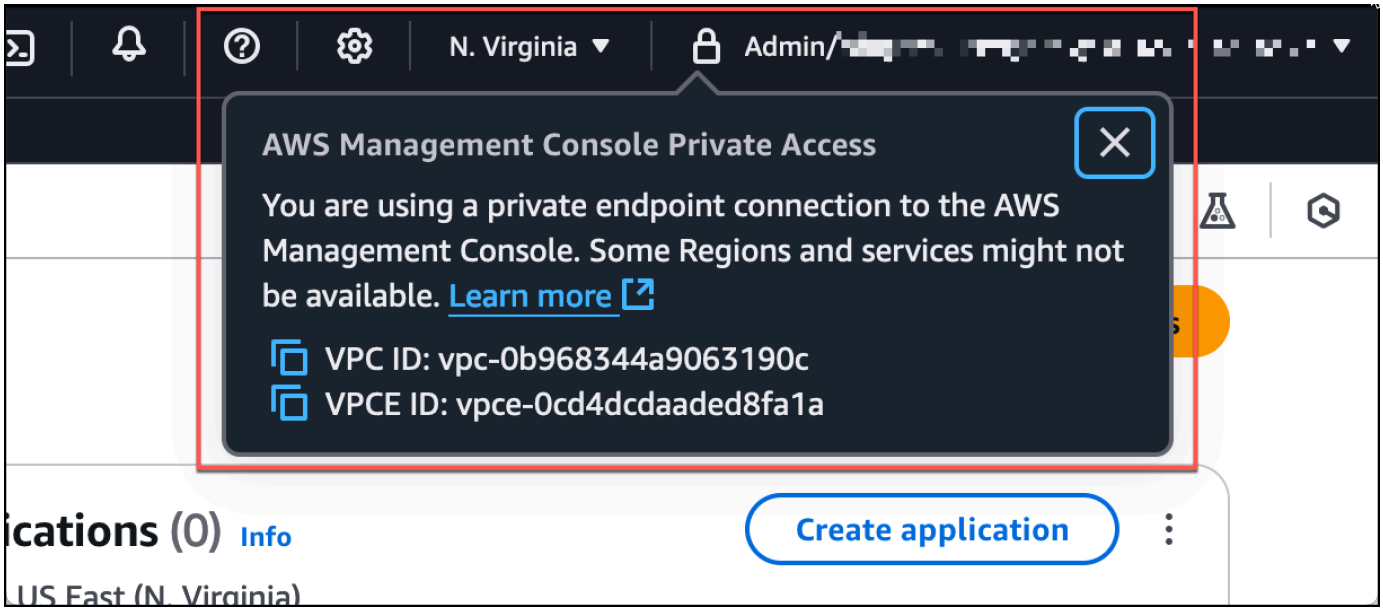
1. Sign in to the management account for your organization and open the Amazon EC2 console.
2. In the navigation pane, choose **Instances**.
3. On the **Instances** page, select **Console VPCE test instance** that was created by the CloudFormation template. Then choose **Connect**.

> **Note**
>
> This example uses Fleet Manager, a capability of AWS Systems Manager Explorer, to connect to your Windows Server. It might take a few minutes before the connection can be started.

4. On the **Connect to instance** page, choose **RDP Client**, then **Connect using Fleet Manager**.
5. Choose **Fleet Manager Remote Desktop**.
6. To get the administrative password for the Amazon EC2 instance and access the Windows Desktop using the web interface, use the private key associated with the Amazon EC2 key pair that you used when creating the CloudFormation template .
7. From the Amazon EC2 Windows instance, open the AWS Management Console in the browser.
8. After you sign in with your AWS credentials, open the Amazon S3 console and verify that you are connected using AWS Management Console Private Access.

To test AWS Management Console Private Access setup

1. Sign in to the management account for your organization and open the Amazon S3 console.
2. Choose the lock-private icon in the navigation bar to view the VPC endpoint in use. The following screenshot shows the location of the lock-private icon and the VPC information.



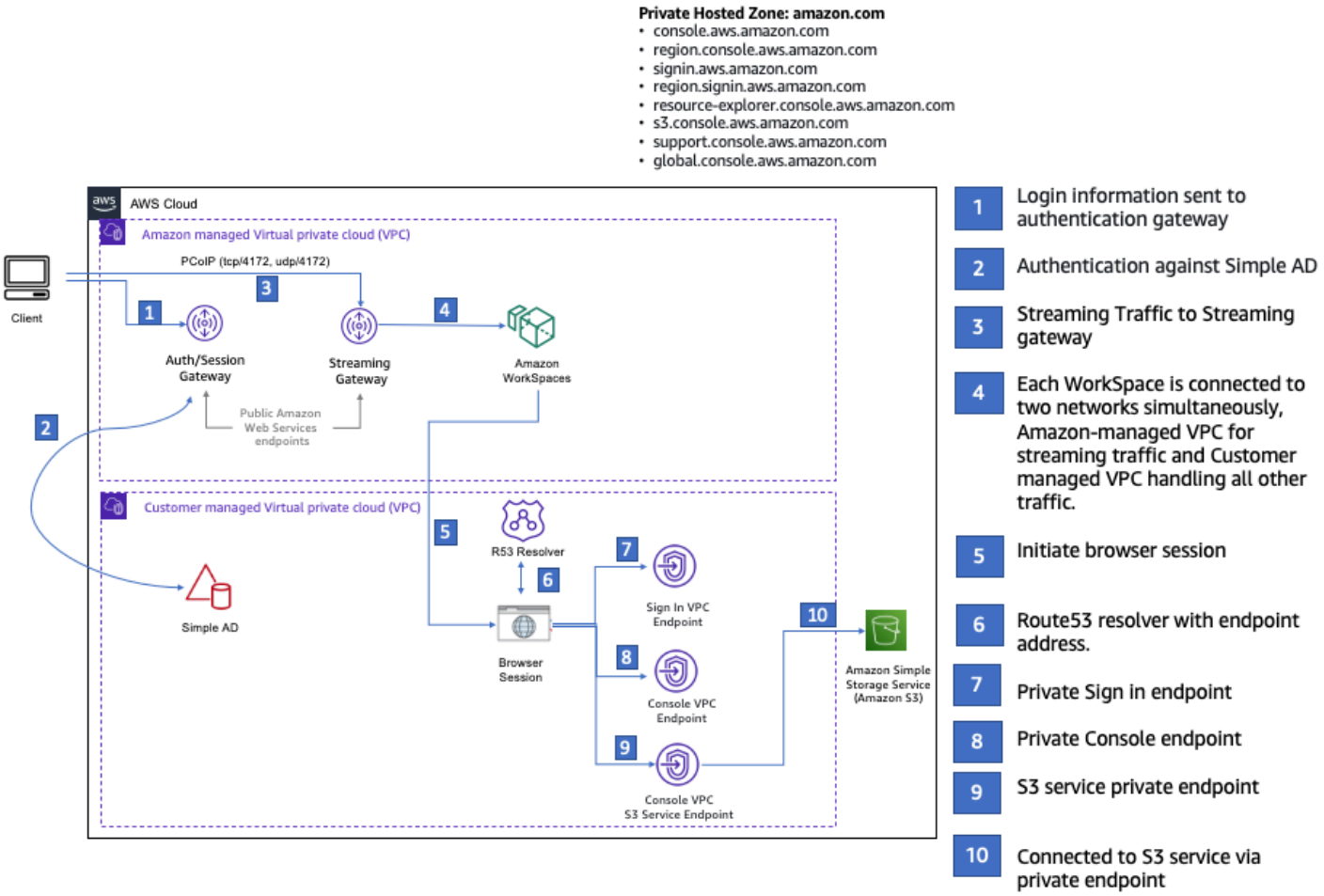
Test setup with Amazon WorkSpaces

Amazon WorkSpaces enables you to provision virtual, cloud-based Windows, Amazon Linux, or Ubuntu Linux desktops for your users, known as WorkSpaces. You can quickly add or remove users as your needs change. Users can access their virtual desktops from multiple devices or web browsers. To learn more about WorkSpaces, see the Amazon WorkSpaces Administration Guide.

The example in this section describes a test environment in which a user environment uses a web browser running on a WorkSpace to sign in to AWS Management Console Private Access. Then, the user visits the Amazon Simple Storage Service console. This WorkSpace is meant to simulate the experience of a corporate user with a laptop on a VPC-connected network, accessing the AWS Management Console from their browser.

This tutorial uses AWS CloudFormation to create and configure the network setup and a Simple Active Directory to be used by WorkSpaces along with step by step instructions to setup a WorkSpace using the AWS Management Console.

The following diagram describes the workflow for using a WorkSpace to test an AWS Management Console Private Access setup. It shows the relationship between a client WorkSpace, an Amazon managed VPC and a customer managed VPC.



Copy the following CloudFormation template and save it to a file that you will use in step 3 of the procedure to set up a network.

AWS Management Console Private Access environment CloudFormation template

```
Description: |
  AWS Management Console Private Access.
Parameters:

  VpcCIDR:
    Type: String
    Default: 172.16.0.0/16
    Description: CIDR range for VPC

  PublicSubnet1CIDR:
    Type: String
    Default: 172.16.1.0/24
```

```
    Description: CIDR range for Public Subnet A

  PublicSubnet2CIDR:
    Type: String
    Default: 172.16.0.0/24
    Description: CIDR range for Public Subnet B

  PrivateSubnet1CIDR:
    Type: String
    Default: 172.16.4.0/24
    Description: CIDR range for Private Subnet A

  PrivateSubnet2CIDR:
    Type: String
    Default: 172.16.5.0/24
    Description: CIDR range for Private Subnet B

  DSAdminPasswordResourceName:
    Type: String
    Default: ADAdminSecret
    Description: Password for directory services admin

# Amazon WorkSpaces is available in a subset of the Availability Zones for each
 supported Region.
# https://docs.aws.amazon.com/workspaces/latest/adminguide/azs-workspaces.html
Mappings:
  RegionMap:
    us-east-1:
      az1: use1-az2
      az2: use1-az4
      az3: use1-az6
    us-west-2:
      az1: usw2-az1
      az2: usw2-az2
      az3: usw2-az3
    ap-south-1:
      az1: aps1-az1
      az2: aps1-az2
      az3: aps1-az3
    ap-northeast-2:
      az1: apne2-az1
      az2: apne2-az3
    ap-southeast-1:
      az1: apse1-az1
```

```
      az2: apse1-az2
    ap-southeast-2:
      az1: apse2-az1
      az2: apse2-az3
    ap-northeast-1:
      az1: apne1-az1
      az2: apne1-az4
    ca-central-1:
      az1: cac1-az1
      az2: cac1-az2
    eu-central-1:
      az1: euc1-az2
      az2: euc1-az3
    eu-west-1:
      az1: euw1-az1
      az2: euw1-az2
    eu-west-2:
      az1: euw2-az2
      az2: euw2-az3
    sa-east-1:
      az1: sae1-az1
      az2: sae1-az3

Resources:

  iamLambdaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service:
              - lambda.amazonaws.com
            Action:
              - 'sts:AssumeRole'
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
      Policies:
        - PolicyName: describe-ec2-az
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
```

```
              - Effect: Allow
                Action:
                  - 'ec2:DescribeAvailabilityZones'
                Resource: '*'
      MaxSessionDuration: 3600
      Path: /service-role/

  fnZoneIdtoZoneName:
    Type: AWS::Lambda::Function
    Properties:
      Runtime: python3.8
      Handler: index.lambda_handler
      Code:
        ZipFile: |
          import boto3
          import cfnresponse

          def zoneId_to_zoneName(event, context):
              responseData = {}
              ec2 = boto3.client('ec2')
              describe_az = ec2.describe_availability_zones()
              for az in describe_az['AvailabilityZones']:
                  if event['ResourceProperties']['ZoneId'] == az['ZoneId']:
                      responseData['ZoneName'] = az['ZoneName']
                      cfnresponse.send(event, context, cfnresponse.SUCCESS,
responseData, str(az['ZoneId']))

          def no_op(event, context):
              print(event)
              responseData = {}
              cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
str(event['RequestId']))

          def lambda_handler(event, context):
              if event['RequestType'] == ('Create' or 'Update'):
                  zoneId_to_zoneName(event, context)
              else:
                  no_op(event,context)
      Role: !GetAtt iamLambdaExecutionRole.Arn

  getAZ1:
    Type: "Custom::zone-id-zone-name"
    Properties:
      ServiceToken: !GetAtt fnZoneIdtoZoneName.Arn
```

```
      ZoneId: !FindInMap [ RegionMap, !Ref 'AWS::Region', az1 ]
  getAZ2:
    Type: "Custom::zone-id-zone-name"
    Properties:
      ServiceToken: !GetAtt fnZoneIdtoZoneName.Arn
      ZoneId: !FindInMap [ RegionMap, !Ref 'AWS::Region', az2 ]

########################
# VPC AND SUBNETS
########################

  AppVPC:
    Type: 'AWS::EC2::VPC'
    Properties:
      CidrBlock: !Ref VpcCIDR
      InstanceTenancy: default
      EnableDnsSupport: true
      EnableDnsHostnames: true

  PublicSubnetA:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      AvailabilityZone: !GetAtt getAZ1.ZoneName

  PublicSubnetB:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PublicSubnet2CIDR
      MapPublicIpOnLaunch: true
      AvailabilityZone: !GetAtt getAZ2.ZoneName

  PrivateSubnetA:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PrivateSubnet1CIDR
      AvailabilityZone: !GetAtt getAZ1.ZoneName

  PrivateSubnetB:
    Type: 'AWS::EC2::Subnet'
```

```
    Properties:
      VpcId: !Ref AppVPC
      CidrBlock: !Ref PrivateSubnet2CIDR
      AvailabilityZone: !GetAtt getAZ2.ZoneName

  InternetGateway:
    Type: AWS::EC2::InternetGateway

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref AppVPC

  NatGatewayEIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment

  NatGateway:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGatewayEIP.AllocationId
      SubnetId: !Ref PublicSubnetA

#########################
# Route Tables
#########################

  PrivateRouteTable:
    Type: 'AWS::EC2::RouteTable'
    Properties:
      VpcId: !Ref AppVPC

  DefaultPrivateRoute:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway

  PrivateSubnetRouteTableAssociation1:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref PrivateRouteTable
```

```
      SubnetId: !Ref PrivateSubnetA

  PrivateSubnetRouteTableAssociation2:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref PrivateRouteTable
      SubnetId: !Ref PrivateSubnetB

  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref AppVPC

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnetARouteTableAssociation1:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnetA

  PublicSubnetBRouteTableAssociation2:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnetB


##########################
# SECURITY GROUPS
##########################

  VPCEndpointSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow TLS for VPC Endpoint
      VpcId: !Ref AppVPC
      SecurityGroupIngress:
```

```
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: !GetAtt AppVPC.CidrBlock

#########################
# VPC ENDPOINTS
#########################

  VPCEndpointGatewayS3:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.s3'
      VpcEndpointType: Gateway
      VpcId: !Ref AppVPC
      RouteTableIds:
        - !Ref PrivateRouteTable

  VPCEndpointInterfaceSignin:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      VpcEndpointType: Interface
      PrivateDnsEnabled: false
      SubnetIds:
        - !Ref PrivateSubnetA
        - !Ref PrivateSubnetB
      SecurityGroupIds:
        - !Ref VPCEndpointSecurityGroup
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.signin'
      VpcId: !Ref AppVPC

  VPCEndpointInterfaceConsole:
    Type: 'AWS::EC2::VPCEndpoint'
    Properties:
      VpcEndpointType: Interface
      PrivateDnsEnabled: false
      SubnetIds:
        - !Ref PrivateSubnetA
        - !Ref PrivateSubnetB
      SecurityGroupIds:
        - !Ref VPCEndpointSecurityGroup
      ServiceName: !Sub 'com.amazonaws.${AWS::Region}.console'
      VpcId: !Ref AppVPC
```

```
#########################
# ROUTE53 RESOURCES
#########################

  ConsoleHostedZone:
    Type: "AWS::Route53::HostedZone"
    Properties:
      HostedZoneConfig:
        Comment: 'Console VPC Endpoint Hosted Zone'
      Name: 'console.aws.amazon.com'
      VPCs:
        -
          VPCId: !Ref AppVPC
          VPCRegion: !Ref "AWS::Region"

  ConsoleRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: 'console.aws.amazon.com'
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  GlobalConsoleRecord:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: 'global.console.aws.amazon.com'
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  ConsoleS3ProxyRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: 's3.console.aws.amazon.com'
```

```
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  ConsoleSupportProxyRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: "support.console.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  ExplorerProxyRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: "resource-explorer.console.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  WidgetProxyRecord:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref "ConsoleHostedZone"
      Name: "*.widget.console.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ["1", !Split [":", !Select ["0", !GetAtt
VPCEndpointInterfaceConsole.DnsEntries],],]
        HostedZoneId: !Select ["0", !Split [":", !Select ["0", !GetAtt
VPCEndpointInterfaceConsole.DnsEntries],],]
      Type: A

  ConsoleRecordRegional:
```

```
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: !Sub "${AWS::Region}.console.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  ConsoleRecordRegionalMultiSession:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'ConsoleHostedZone'
      Name: !Sub "*.${AWS::Region}.console.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceConsole.DnsEntries]]]
      Type: A

  SigninHostedZone:
    Type: "AWS::Route53::HostedZone"
    Properties:
      HostedZoneConfig:
        Comment: 'Signin VPC Endpoint Hosted Zone'
      Name: 'signin.aws.amazon.com'
      VPCs:
        -
          VPCId: !Ref AppVPC
          VPCRegion: !Ref "AWS::Region"

  SigninRecordGlobal:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'SigninHostedZone'
      Name: 'signin.aws.amazon.com'
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceSignin.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
VPCEndpointInterfaceSignin.DnsEntries]]]
```

```
      Type: A

  SigninRecordRegional:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref 'SigninHostedZone'
      Name: !Sub "${AWS::Region}.signin.aws.amazon.com"
      AliasTarget:
        DNSName: !Select ['1', !Split [':', !Select ['0', !GetAtt
 VPCEndpointInterfaceSignin.DnsEntries]]]
        HostedZoneId: !Select ['0', !Split [':', !Select ['0', !GetAtt
 VPCEndpointInterfaceSignin.DnsEntries]]]
      Type: A

#########################
# WORKSPACE RESOURCES
#########################
  ADAdminSecret:
    Type: AWS::SecretsManager::Secret
    Properties:
      Name: !Ref DSAdminPasswordResourceName
      Description: "Password for directory services admin"
      GenerateSecretString:
        SecretStringTemplate: '{"username": "Admin"}'
        GenerateStringKey: password
        PasswordLength: 30
        ExcludeCharacters: '"@/\'

  WorkspaceSimpleDirectory:
    Type: AWS::DirectoryService::SimpleAD
    DependsOn: AppVPC
    Properties:
      Name: "corp.awsconsole.com"
      Password: '{{resolve:secretsmanager:ADAdminSecret:SecretString:password}}'
      Size: "Small"
      VpcSettings:
        SubnetIds:
          - Ref: PrivateSubnetA
          - Ref: PrivateSubnetB

        VpcId:
          Ref: AppVPC
```

```
Outputs:
  PrivateSubnetA:
    Description: Private Subnet A
    Value: !Ref PrivateSubnetA

  PrivateSubnetB:
    Description: Private Subnet B
    Value: !Ref PrivateSubnetB

  WorkspaceSimpleDirectory:
    Description: Directory to be used for Workspaces
    Value: !Ref WorkspaceSimpleDirectory

  WorkspacesAdminPassword:
    Description : "The ARN of the Workspaces admin's password.  Navigate to the Secrets
 Manager in the AWS Console to view the value."
    Value: !Ref ADAdminSecret
```

Note

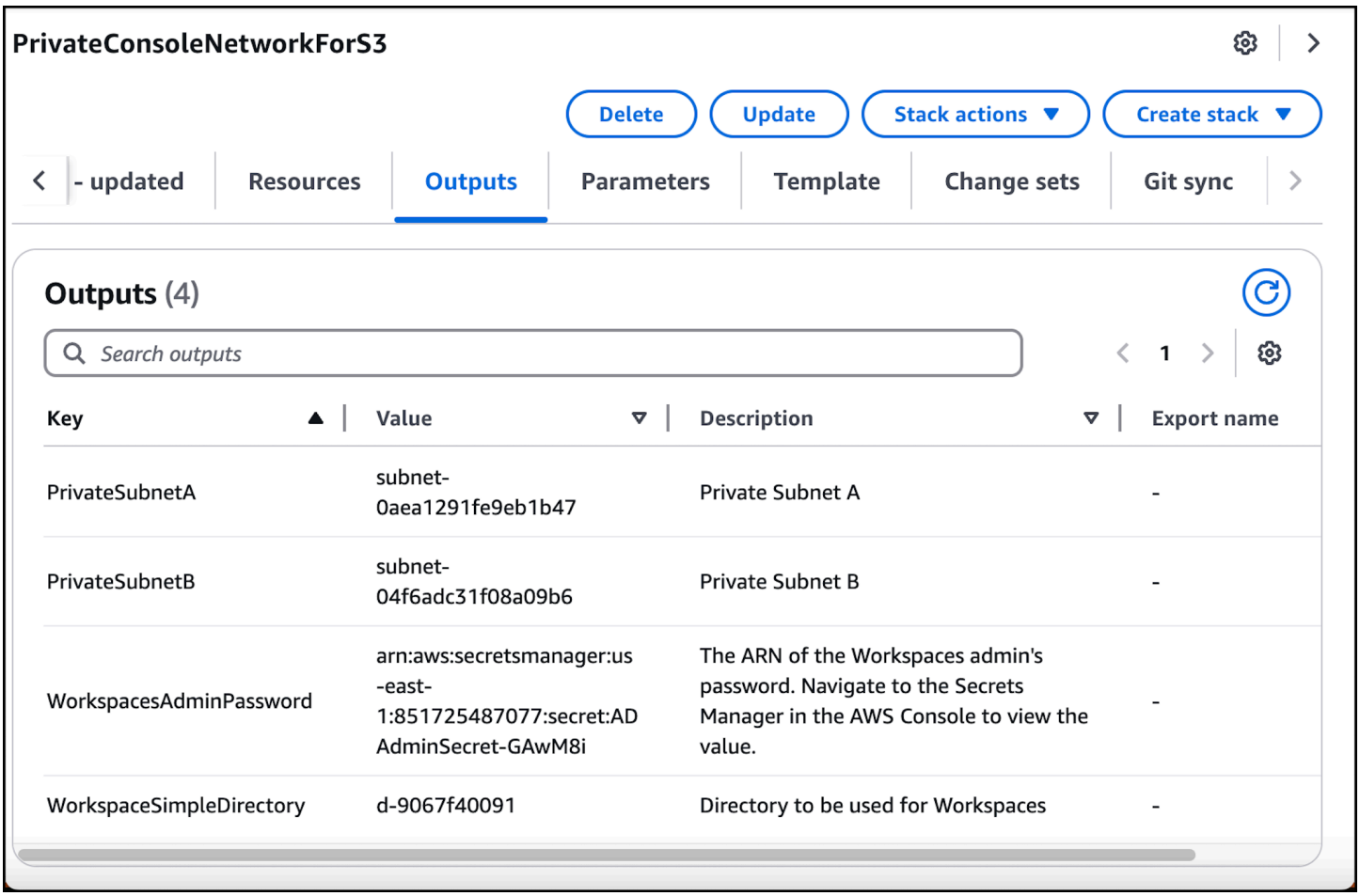
This test setup is designed to run in the US East (N. Virginia) (us-east-1) Region.

To set up a network

1. Sign in to the management account for your organization and open the CloudFormation console.
2. Choose **Create stack**.
3. Choose **With new resources (standard)**. Upload the CloudFormation template file that you previously created, and choose **Next**.
4. Enter a name for the stack, such as **`PrivateConsoleNetworkForS3`**, then choose **Next**.
5. For **VPC and subnets**, enter your preferred IP CIDR ranges, or use the provided default values. If you use the default values, verify that they don't overlap with existing VPC resources in your AWS account.
6. Choose **Create stack**.
7. After the stack is created, choose the **Resources** tab to view the resources that have been created.

8. Choose the **Outputs** tab, to view the values for private subnets and the Workspace Simple Directory. Take note of these values, as you will use them in step four of the next procedure for creating and configuring a WorkSpace.

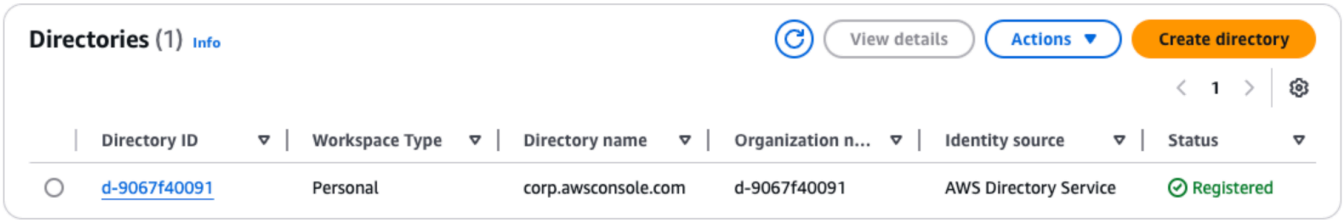
The following screenshot shows the view of the **Outputs** tab displaying the values for the private subnets and the Workspace Simple Directory.



Now that you have created your network, use the following procedures to create and access a WorkSpace.

To create a WorkSpace

1. Open the WorkSpaces console.
2. In the navigation pane, choose **Directories**.
3. On the **Directories** page, verify that the directory status is **Active**. The following screenshot shows a **Directories** page with an active directory.



4. To use a directory in WorkSpaces, you must register it. In the navigation pane, choose **WorkSpaces**, then choose **Create WorkSpaces**.
5. For **Select a directory**, choose the directory created by CloudFormation in the preceding procedure. On the **Actions** menu, choose **Register**.
6. For the subnet selection, select the two private subnets noted in step nine of the preceding procedure.
7. Select **Enable self-service permissions**, then choose **Register**.
8. After the directory is registered, continue creating the WorkSpace. Select the registered directory, then choose **Next**.
9. On the **Create users** page, choose **Create additional user**. Enter your name and email to enable you to use the WorkSpace. Verify that the email address is valid as the WorkSpace login information is sent to this email address.
10. Choose **Next**.
11. On the **Identify Users** page, select the user you created in step nine, then choose **Next**.
12. On the **Select Bundle** page, choose **Standard with Amazon Linux 2**, then choose **Next**.
13. Use the default settings for the running mode and user customization, and then choose **Create Workspace**. The WorkSpace starts out in `Pending` status and transitions to `Available` within about 20 minutes.
14. When the WorkSpace is available, you will receive an email with instructions for accessing it at the email address you provided in step nine.

After you sign in to your WorkSpace, you can test that you are accessing it using your AWS Management Console Private Access.

To access a WorkSpace

1. Open the email that you received in step 14 of the preceding procedure.

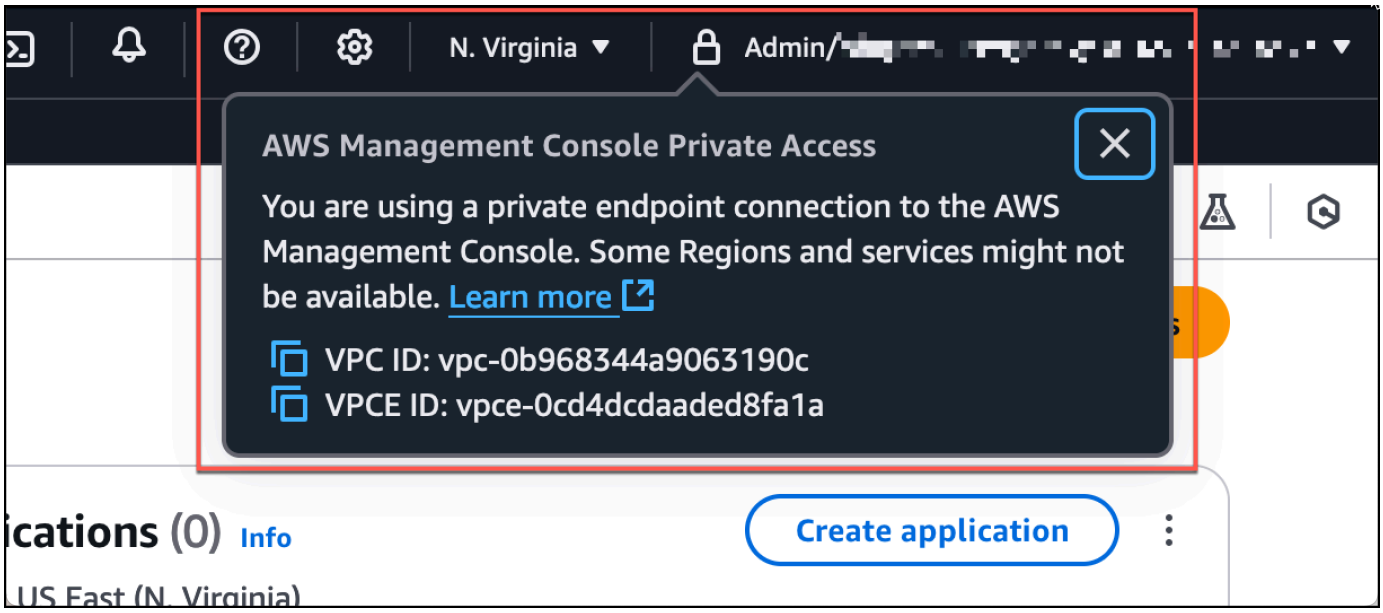
2. In the email, choose the unique link that is provided to set up your profile and download the WorkSpaces client.
3. Set your password.
4. Download the client of your choice.
5. Install and launch the client. Enter the registration code that was provided in your email, then choose **Register**.
6. Sign in to Amazon WorkSpaces using the credentials you created in step three.

To test AWS Management Console Private Access setup

1. From your WorkSpace, open your browser. Then, navigate to the AWS Management Console and sign in using your credentials.

> **Note**
>
> If you are using Firefox as your browser, verify that the **Enable DNS over HTTPS** option is turned off in your browser settings.

2. Open the Amazon S3 console where you can verify that you are connected using AWS Management Console Private Access.
3. Choose the lock-private icon on the navigation bar to view the VPC and VPC endpoint in use. The following screenshot shows the location of the lock-private icon and the VPC information.



Test VPC setup with IAM policies

You can further test your VPC that you have set up with Amazon EC2 or WorkSpaces by deploying IAM policies that restrict access.

The following policy denies access to Amazon S3 unless it is using your specified VPC.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "S3:*",
            "Resource": "*",
            "Condition": {
                "StringNotEqualsIfExists": {
                    "aws:SourceVpc": "vpc-12345678"
                },
                "Bool": {
                    "aws:ViaAwsService": "false"
                }
            }
        }
    ]
}
```

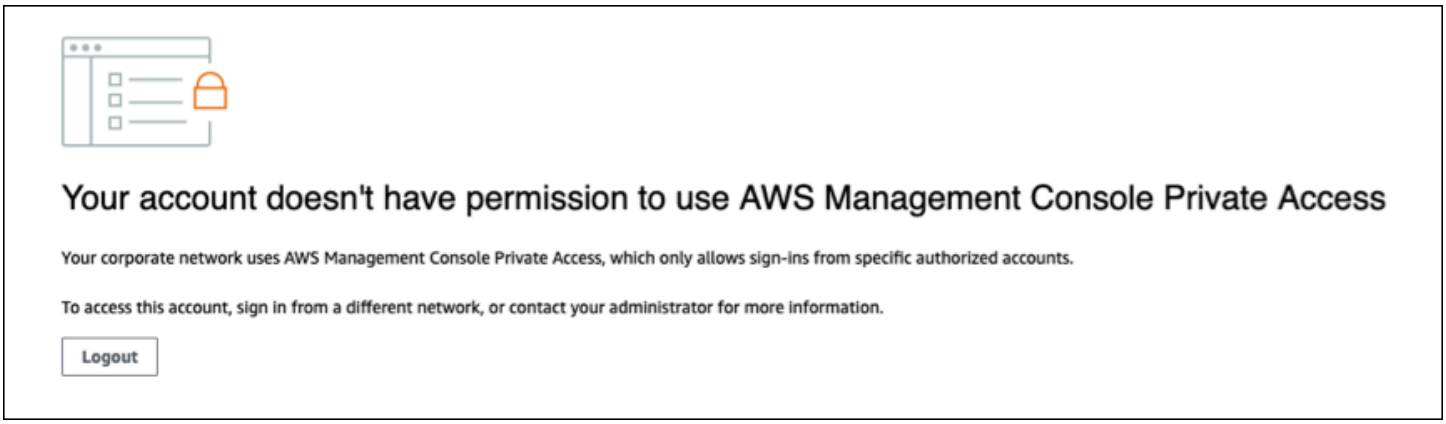
The following policy limits sign in to selected AWS account IDs by using a AWS Management Console Private Access policy for the sign-in endpoint.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": [
                        "AWSAccountID"
                    ]
                }
            }
        }
    ]
}
```

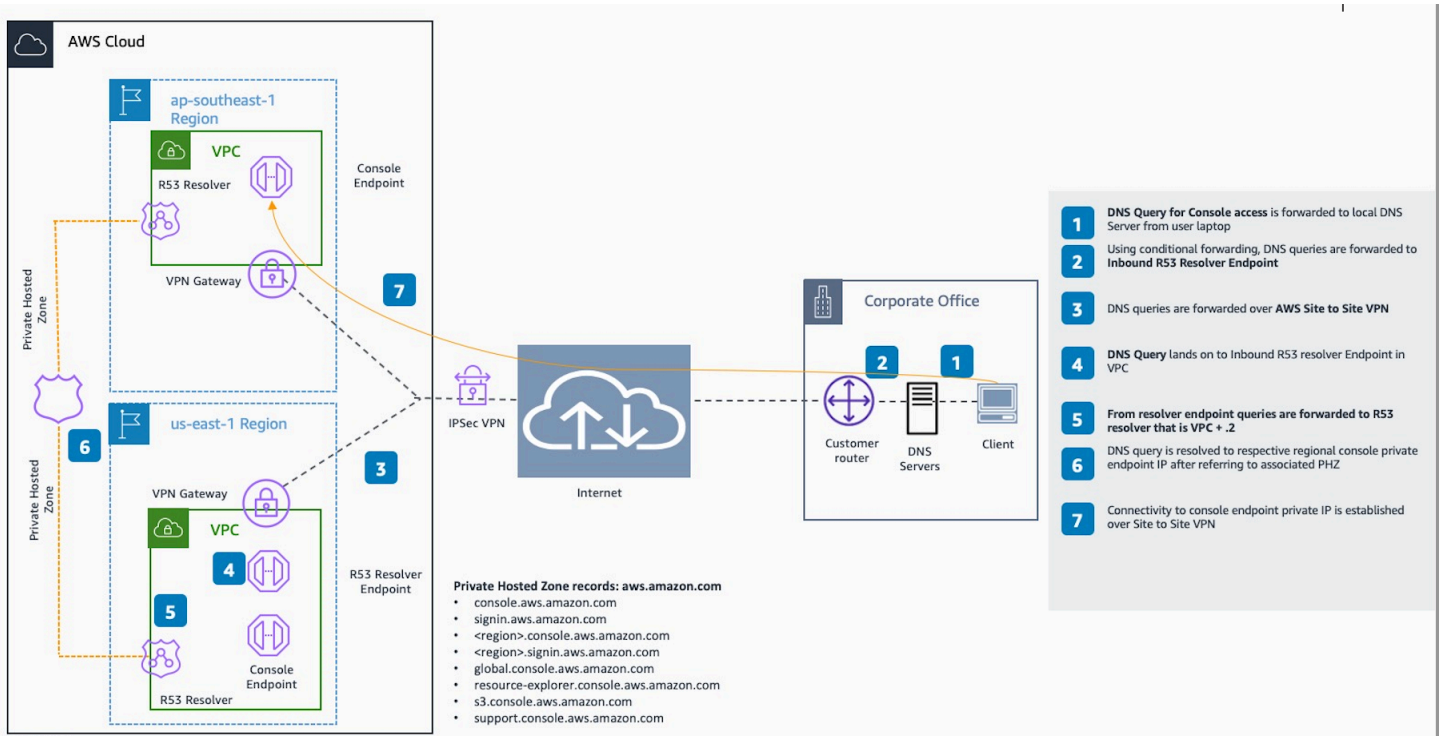
If you connect with an identity that does not belong to your account, the following error page is displayed.



Reference architecture

To connect privately to AWS Management Console Private Access from an on-premises network, you can leverage the AWS Site-to-Site VPN to AWS Virtual Private Gateway (VGW) connection option. AWS Site-to-Site VPN enables access to your remote network from your VPC by creating a connection, and configuring routing to pass traffic through the connection. For more information, see What is AWS Site-to-Site VPN in the *AWS Site-to-Site VPN User Guide*. AWS Virtual Private Gateway (VGW) is a highly available Regional service that acts as a gateway between a VPC and the on-premises network.

AWS Site-to-Site VPN to AWS Virtual Private Gateway (VGW)



An essential component in this reference architecture design is the Amazon Route 53 Resolver, specifically the inbound resolver. When you set it up in the VPC where the AWS Management Console Private Access endpoints are created, resolver endpoints (network interfaces) are created in the specified subnets. Their IP addresses can then be referred to in conditional forwarders on the on-premises DNS servers, to allow querying of records in a Private Hosted Zone. When on-premises clients connect to the AWS Management Console, they are routed to the AWS Management Console Private Access endpoints' private IPs.

Before setting up the connection to the AWS Management Console Private Access endpoint, complete the prerequisites steps of setting up the AWS Management Console Private Access endpoints in all the Regions where you want to access the AWS Management Console, as well as in US East (N. Virginia) Region, and configuring the Private Hosted Zone.

AWS User Experience Customization (UXC)

AWS User Experience Customization (UXC) is a utility that lets account administrators customize the visual appearance of the AWS Management Console and manage these settings at the account level.

With UXC, you can customize the following settings:

- **Account color** – You can set a color for your accounts to visually distinguish between them. For example, you can use green for development accounts, yellow for test accounts, and red for production accounts.
- **Service visibility** – You can control which AWS services appear in the console navigation. Service visibility simplifies the AWS Management Console to show only the AWS services that are relevant to your account.
- **Region visibility** – You can control which AWS Regions appear in the Region selector. Region visibility simplifies the AWS Management Console to show only the Regions that are relevant to your account.

If you have not configured a setting, then the default behavior applies: all services and Regions are visible, and no account color is set. You can reset account color to its default by setting the value to `"none"`. You can reset visible services and Regions to their defaults by setting their values to `null`.

> **Note**
>
> The `visibleServices` and `visibleRegions` settings control only the appearance of services and Regions in the AWS Management Console. They do not restrict access through the AWS Command Line Interface, SDKs, or other APIs.

Topics

Getting started with AWS User Experience Customization

With UXC, account administrators can configure account customizations for the AWS Management Console.

Prerequisites

Before you begin, you need the following:

- An AWS account
- Appropriate AWS Identity and Access Management (IAM) permissions for UXC. For more information, see How AWS User Experience Customization works with IAM and AWS managed policies for the AWS Management Console.

Accessing UXC settings in the AWS Management Console

To access account color in the AWS Management Console, see Accessing account information in the AWS Management Console. To access service visibility and Region visibility in the AWS Management Console, see Configuring the AWS Management Console using Unified Settings.

To set an account color in the console

1. Sign in to the AWS Management Console.
2. On the navigation bar, choose your account name.
3. Choose **Account**.
4. In **Account display settings**, choose a color.
5. Choose **Update**.

To set visible Regions in the console

1. Sign in to the AWS Management Console.
2. Open Unified Settings.
3. Choose **Edit** in the **Visible Regions** section.
4. Set your visible Regions to **All available Regions** or **Select Regions** and configure your list.
5. Choose **Save changes**.

To set visible services in the console

1. Sign in to the AWS Management Console.
2. Open Unified Settings.
3. Choose **Edit** in the **Visible services** section.
4. Set your visible services to **All services** or **Select services** and configure your list.
5. Choose **Save changes**.

Accessing UXC settings programmatically

You can also manage account customization settings programmatically or as infrastructure as code. For more information, see the AWS User Experience Customization API Reference and the AWS::UXC::AccountCustomization CloudFormation template reference.

Logging AWS User Experience Customization API calls using AWS CloudTrail

AWS User Experience Customization is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service. CloudTrail captures all API calls for UXC as events. The calls captured include calls from the UXC console and code calls to the UXC API operations. Using the information collected by CloudTrail, you can determine the request that was made to UXC, the IP address from which the request was made, when it was made, and additional details.

CloudTrail is active in your AWS account when you create the account and you automatically have access to the CloudTrail **Event history**. The CloudTrail **Event history** provides a viewable, searchable, downloadable, and immutable record of the past 90 days of recorded management events in an AWS Region. For more information, see Working with CloudTrail Event history in the *AWS CloudTrail User Guide*. There are no CloudTrail charges for viewing the **Event history**.

For an ongoing record of events in your AWS account past 90 days, create a trail or a CloudTrail Lake event data store.

UXC management events in CloudTrail

Management events provide information about management operations that are performed on resources in your AWS account. These are also known as control plane operations. By default, CloudTrail logs management events.

AWS User Experience Customization logs all UXC control plane operations as management events. For a list of the AWS User Experience Customization control plane operations that UXC logs to CloudTrail, see the AWS User Experience Customization API Reference.

UXC event examples

An event represents a single request from any source and includes information about the requested API operation, the date and time of the operation, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so events don't appear in any specific order.

The following example shows a CloudTrail event that demonstrates the operation.


```
{
  "eventVersion" : "1.09",
  "userIdentity" : {
    "type" : "AssumedRole",
    "principalId" : "AIDACKCEVSQ6C2EXAMPLE:jdoe",
    "arn" : "arn:aws:sts::111122223333:assumed-role/user/jdoe",
    "accountId" : "111122223333",
    "accessKeyId" : "AKIAIOSFODNN7EXAMPLE",
    "sessionContext" : {
      "sessionIssuer" : {
        "type" : "Role",
        "principalId" : "AIDACKCEVSQ6C2EXAMPLE",
        "arn" : "arn:aws:iam::111122223333:role/user",
        "accountId" : "111122223333",
        "userName" : "jdoe"
      },
      "webIdFederationData" : { },
      "attributes" : {
        "creationDate" : "2022-12-09T23:48:51Z",
        "mfaAuthenticated" : "false"
      }
    }
  },
```

```
  "eventTime" : "2022-12-09T23:50:03Z",
  "eventSource" : "uxc.amazonaws.com",
  "eventName" : "GetAccountColor",
  "awsRegion" : "us-east-2",
  "sourceIPAddress" : "10.24.34.3",
  "userAgent" : "PostmanRuntime/7.43.4",
  "requestParameters" : null,
  "responseElements" : null,
  "requestID" : "543db7ab-b4b2-11e9-8925-d139e92a1fe8",
  "eventID" : "5b2805a5-3e06-4437-a7a2-b5fdb5cbb4e2",
  "readOnly" : true,
  "eventType" : "AwsApiCall",
  "managementEvent" : true,
  "recipientAccountId" : "111122223333",
  "eventCategory" : "Management"
}
```


For information about CloudTrail record contents, see CloudTrail record contents in the *AWS CloudTrail User Guide*.

Security in AWS User Experience Customization

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 User Experience Customization, 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 UXC. The following topics show you how to configure UXC to meet your security and

compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your UXC resources.

Topics

Identity and Access Management for AWS User Experience Customization

AWS User Experience Customization (UXC) uses IAM policies to manage access to UXC API Operations.

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 User Experience Customization resources. IAM is an AWS service that you can use with no additional charge.

Topics

Audience

How you use AWS Identity and Access Management (IAM) differs based on your role:

- **Service user** - request permissions from your administrator if you cannot access features (see Troubleshooting AWS User Experience Customization identity and access)
- **Service administrator** - determine user access and submit permission requests (see How AWS User Experience Customization works with IAM)
- **IAM administrator** - write policies to manage access (see Identity-based policy examples for AWS User Experience Customization)

Authenticating with identities

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

You can sign in as a federated identity using credentials from an identity source like AWS IAM Identity Center (IAM Identity Center), single sign-on authentication, or Google/Facebook credentials. For more information about signing in, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

For programmatic access, AWS provides an SDK and CLI to cryptographically sign requests. For more information, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity called the AWS account *root user* that has complete access to all AWS services and resources. We strongly recommend that you don't use the root user for everyday tasks. For tasks that require root user credentials, see Tasks that require root user credentials in the *IAM User Guide*.

IAM users and groups

An *IAM user* is an identity with specific permissions for a single person or application. We recommend using temporary credentials instead of IAM users with long-term credentials. For more information, see Require human users to use federation with an identity provider to access AWS using temporary credentials in the *IAM User Guide*.

An *IAM group* specifies a collection of IAM users and makes permissions easier to manage for large sets of users. For more information, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity with specific permissions that provides temporary credentials. You can assume a role by switching from a user to an IAM role (console) or by calling an AWS CLI or AWS API operation. For more information, see Methods to assume a role in the *IAM User Guide*.

IAM roles are useful for federated user access, temporary IAM user permissions, cross-account access, cross-service access, and applications running on Amazon EC2. For more information, see Cross account resource access in IAM 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 defines permissions when associated with an identity or resource. AWS evaluates these policies when a principal makes a request. Most policies are stored in AWS as JSON documents. For more information about JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Using policies, administrators specify who has access to what by defining which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. An IAM administrator creates IAM policies and adds them to roles, which users can then assume. IAM policies define permissions regardless of the method used to perform the operation.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you attach to an identity (user, group, or role). These policies control what actions identities 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 *inline policies* (embedded directly into a single identity) or *managed policies* (standalone policies attached to multiple identities). To learn how to choose between managed and inline policies, 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 include 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. You must specify a principal in a resource-based policy.

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.

Other policy types

AWS supports additional policy types that can set the maximum permissions granted by more common policy types:

- **Permissions boundaries** – Set the maximum permissions that an identity-based policy can grant to an IAM entity. For more information, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – Specify the maximum permissions for an organization or organizational unit in AWS Organizations. For more information, see Service control policies in the *AWS Organizations User Guide*.
- **Resource control policies (RCPs)** – Set the maximum available permissions for resources in your accounts. For more information, see Resource control policies (RCPs) in the *AWS Organizations User Guide*.
- **Session policies** – Advanced policies passed as a parameter when creating a temporary session for a role or federated user. 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 AWS User Experience Customization works with IAM

AWS User Experience Customization (UXC) works with IAM policies to manage access to UXC API Operations.

Before you use IAM to manage access to AWS User Experience Customization (User Experience Customization), learn what IAM features are available to use with User Experience Customization. We recommend you integrate with User Experience Customization through an AWS managed policy, for more information, see AWS managed policies for the AWS Management Console.

Before you use IAM to manage access to User Experience Customization, learn what IAM features are available to use with User Experience Customization.

IAM feature	User Experience Customization support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes

IAM feature	User Experience Customization support
Policy resources	No
Policy condition keys	No
Temporary credentials	Yes
Cross-service principal permissions	No
Service-linked roles	No
Service roles	No

To get a high-level view of how User Experience Customization 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 User Experience Customization

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. 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*.

To view examples of User Experience Customization identity-based policies, see Identity-based policy examples for AWS User Experience Customization.

Policy actions for User Experience Customization

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

To see all User Experience Customization actions, refer to the API Reference.

Policy actions in User Experience Customization use the `uxc:` prefix before the action (for example `uxc:GetAccountCustomizations`).

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

```
"Action": [
      "uxc:GetAccountCustomizations",
      "uxc:ListServices"
         ]
```

To view examples of User Experience Customization identity-based policies, see Identity-based policy examples for AWS User Experience Customization.

Policy resources for User Experience Customization

User Experience Customization does not support policy resources.

Using temporary credentials with User Experience Customization

Temporary credentials provide short-term access to AWS resources and are automatically created when you use federation or switch roles. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM and AWS services that work with IAM in the *IAM User Guide*.

Troubleshooting AWS User Experience Customization identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with User Experience Customization and IAM.

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 `uxc:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 uxc:GetWidget on resource: my-example-widget because no identity-based policy allows
 the GetWidget action
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `uxc:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

> **Important**
>
> Do not provide your access keys to a third party, even to help find your canonical user ID. By doing this, you might give someone permanent access to your AWS account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see Managing access keys in the *IAM User Guide*.

To allow others to access User Experience Customization, you must grant permission to the people or applications that need access. If you are using AWS IAM Identity Center to manage people and applications, you assign permission sets to users or groups to define their level of access. Permission sets automatically create and assign IAM policies to IAM roles that are associated with the person or application. For more information, see Permission sets in the *AWS IAM Identity Center User Guide*.

If you are not using IAM Identity Center, you must create IAM entities (users or roles) for the people or applications that need access. You must then attach a policy to the entity that grants them the correct permissions in User Experience Customization. After the permissions are granted, provide the credentials to the user or application developer. They will use those credentials to access AWS. To learn more about creating IAM users, groups, policies, and permissions, see IAM Identities and Policies and permissions in IAM in the *IAM User Guide*.

Identity-based policy examples for AWS User Experience Customization

By default, users and roles don't have permission to get or modify UXC resources. To grant users permission to perform actions on resources, an IAM administrator can create IAM policies. 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*.

Topics

Policy best practices

Identity-based policies determine whether someone can create, access, or delete User Experience Customization 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 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*.

Read-only access to UXC Account Customizations

The following example shows how to create a policy that allows read-only access to UXC Account Customizations:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "uxc:GetAccountCustomizations",
                "uxc:ListServices"
            ],
            "Resource": "*"
        }
    ]
}
```

Full access to UXC Account Customizations

The following example shows how to create a policy that allows full access to UXC Account Customizations:

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

```
            "Effect": "Allow",
            "Action": [
                "uxc:*"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS managed policies for the AWS Management Console

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: AWSManagementConsoleBasicUserAccess

You can attach `AWSManagementConsoleBasicUserAccess` to your users, groups, and roles.

This policy grants the permissions necessary for non-administrative users of the AWS Management Console. This includes features such as resource discovery, notifications, browser-based shell access, and customized navigation.

Permissions details

This `AWSManagementConsoleBasicUserAccess` is grouped into the following sets of permissions:

- `cloudshell` – Allows principals full access to AWS CloudShell capabilities, including environment creation, session management, and command execution.
- `ec2` – Allows principals to describe Regions enabled for the account in Unified Navigation.
- `notifications` – Allows principals to obtain events from AWS User Notifications.
- `q` – Allows principals to chat with Amazon Q Developer.
- `resource-explorer-2` – Allows principals to search and discover AWS resources using Unified Search.
- `uxc` – Allows principals to read AWS User Experience Customization settings.
- `action-recommendations` – Allows principals to receive contextual action recommendations.
- `account` – Allows principals to retrieve information about the specified account including its account name, account ID, and account creation date and time.

To view the permissions for this policy, see AWSManagementConsoleBasicUserAccess in the *AWS Managed Policy Reference.*

AWS managed policy: AWSManagementConsoleAdministratorAccess

You can attach `AWSManagementConsoleAdministratorAccess` to your users, groups, and roles.

This policy grants full access to configure and customize the AWS Management Console. It allows administrators to set account colors, enable user notifications, and configure resource discovery. It also includes permissions from the `AWSManagementConsoleBasicUserAccess` managed policy, which are essential for non-administrative users of the AWS Management Console.

Permissions details

This `AWSManagementConsoleAdministratorAccess` is grouped into the following sets of permissions:

- `cloudshell` – Allows principals full access to AWS CloudShell capabilities, including environment creation, session management, and command execution.
- `ec2` – Allows principals to describe Regions enabled for the account in Unified Navigation.
- `notifications` – Allows principals to access and update notification configurations, events, and feature opt-in status.
- `q` – Allows principals to chat with Amazon Q Developer for AI-assisted support.
- `resource-explorer-2` – Allows principals to search and discover AWS resources using Unified Search.
- `uxc` – Allows principals full access to AWS User Experience Customization settings.
- `action-recommendations` – Allows principals to receive contextual action recommendations.
- `account` – Allows principals to retrieve information about the specified account including its account name, account ID, and account creation date and time.

To view the permissions for this policy, see AWSManagementConsoleAdministratorAccess in the *AWS Managed Policy Reference.*

AWS Management Console updates to AWS managed policies

View details about updates to AWS managed policies for the AWS Management Console since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the AWS Management Console Document history page.

Change	Description	Date
AWSManagementConsoleBasicUserAccess – Updated policy	Added `uxc:GetAccountCustomizations` and `uxc:ListServices` permissions.	March 26, 2026
AWSManagementConsoleAdministratorAccess – Updated policy	Added `uxc:GetAccountCustomizations` , `uxc:UpdateAccountCustomizations` , and	March 26, 2026

Change	Description	Date
	`uxc:ListServices` permissions.	
AWSManagementConsoleBasicUserAccess – Updated policy	Updated policy to add permissions to allow users to see account information and receive action recommendations while navigating the AWS Management Console.	December 9, 2025
AWSManagementConsoleAdministratorAccess – Updated policy	Updated policy to add permissions to allow users to see account information and receive action recommendations while navigating the AWS Management Console.	December 9, 2025
AWSManagementConsoleBasicUserAccess – New policy	Added a new AWS managed policy that grants permissions necessary for basic AWS Management Console navigation, account color viewing, and resource discovery.	August 14, 2025
AWSManagementConsoleAdministratorAccess – New policy	Added a new AWS managed policy that provides full access to configure and customize the AWS Management Console.	August 14, 2025
AWS Management Console started tracking changes	AWS Management Console started tracking changes for its AWS managed policies.	August 14, 2025

Using Markdown in the Console

Some services in the AWS Management Console, such as Amazon CloudWatch, support the use of Markdown in certain fields. This topic explains the types of Markdown formatting supported in the console.

Contents

Paragraphs, Line Spacing, and Horizontal Lines

Paragraphs are separated by a blank line. To make sure that the blank line between the paragraphs renders when it is converted to HTML, add a new line with a non-break space (` `) and then a blank line. Repeat this pair of lines to insert multiple blank lines one after the other, as in the following example:

```
 

 
```

To create a horizontal rule that separates the paragraphs, add a new line with three hyphens in a row: `---`

```
Previous paragraph.
---
Next paragraph.
```

To create a text block with monospace type, add a line with three backticks (`` ` ``). Enter the text to show in monospace type. Then, add another new line with three backticks. The following example shows text that will be formatted to monospace type when displayed:

```
```
This appears in a text box with a background shading.
The text is in monospace.
```
```

Headings

To create headings, use the pound sign (**#**). A single pound sign and a space indicate a top-level heading. Two pound signs create a second-level heading, and three pound signs create a third-level heading. The following examples show a top-level, second-level, and third-level heading:

```
# Top-level heading
```

```
## Second-level heading
```

```
### Third-level heading
```

Text Formatting

To format text as italic, surround it with a single underscore (**_**) or asterisk (*****) on each side.

```
*This text appears in italics.*
```

To format text as bold, surround it with double underscores or double asterisks on each side.

```
**This text appears in bold.**
```

To format text as strikethrough, surround it with two tildes (**~**) on each side.

```
~~This text appears in strikethrough.~~
```

Links

To add a text hyperlink, enter the link text surrounded by square brackets (**[ ]**), followed by the full URL in parentheses (**( )**), as in the following example:

```
Choose [link_text](http://my.example.com).
```

Lists

To format lines as part of a bulleted list, add them on separate lines that start with with a single asterisk (*) and then a space, as in the following example:

```
Here is a bulleted list:
* Ant
* Bug
* Caterpillar
```

To format lines as part of a numbered list, add them on separate lines that start with with a number, a period (.), and a space, as in the following example:

```
Here is a numbered list:
1. Do the first step
2. Do the next step
3. Do the final step
```

Tables and Buttons (CloudWatch Dashboards)

CloudWatch dashboards text widgets support Markdown tables and buttons.

To create a table, separate columns using vertical bars (|) and rows using new lines. To make the first row a header row, insert a line between the header row and the first row of values. Then, add at least three hyphens (-) for each column in the table. Separate columns using vertical bars. The following example shows Markdown for a table with two columns, a header row, and two rows of data:

```
Table | Header
----|-----
Amazon Web Services | AWS
1 | 2
```

The Markdown text in the previous example creates the following table:

Table	Header
Amazon Web Services	AWS
1	2

In a CloudWatch dashboard text widget, you can also format a hyperlink to appear as a button. To create a button, use `[button:`*`Button text`*`]`, followed by the full URL in parentheses(**( )**), as in the following example:

```
[button:Go to AWS](http://my.example.com)
[button:primary:This button stands out even more](http://my.example.com)
```

Troubleshooting

Consult this section to find solutions to common problems with the AWS Management Console.

You can also diagnose and troubleshoot common errors for some AWS services using Amazon Q Developer. For more information, see Diagnose common errors in the console with Amazon Q Developer in the *Amazon Q Developer User Guide*.

Topics

The page isn't loading properly

- If this problem only occurs occasionally, check your internet connection. Try to connect through a different network, or with or without a VPN, or try using a different web browser.
- If all impacted users are from the same team, it may be a privacy browser extension or security firewall issue. Privacy browser extensions and security firewalls can block access to the domains used by the AWS Management Console. Try turning off these extensions or adjusting firewall settings. To verify issues with your connection, open your browser developer tools (Chrome, Firefox) and inspect the errors in the **Console** tab. The AWS Management Console uses domains' suffixes including the following list. This list is not exhaustive and can change with time. These domains' suffixes aren't used exclusively by AWS.
  - .a2z.com
  - .amazon.com
  - .amazonaws.com
  - .aws
  - .aws.com
  - .aws.dev
  - .awscloud.com

- .awsplayer.com
- .awsstatic.com
- .cloudfront.net
- .live-video.net

> **Warning**
>
> Since July 31, 2022, AWS no longer supports Internet Explorer 11. We recommend that you use the AWS Management Console with other supported browsers. For more information, see AWS News Blog.

My browser displays an 'access denied' error when connecting to the AWS Management Console

Recent changes made to the console might affect your access if all of the following conditions are met:

- You access AWS Management Console from a network that is configured to reach AWS service endpoints through VPC endpoints.
- You restrict access to AWS services by either using `aws:SourceIp` or `aws:SourceVpc` global condition key in your IAM policies.

We recommend you review the IAM policies that contain the `aws:SourceIp` or `aws:SourceVpc` global condition key. Apply both `aws:SourceIp` and `aws:SourceVpc` where applicable.

Some AWS Management Console features use dual-stack domains that support both IPv4 and IPv6 connections. If your IAM policy restricts access using `aws:SourceIp` with only IPv4 CIDR blocks, requests might fail when your operating system prefers IPv6 connections (or vice versa). To avoid this, include both IPv4 and IPv6 CIDR blocks in your `aws:SourceIp` condition. For more information, see aws:SourceIp in the *AWS Identity and Access Management User Guide*.

You can also onboard to the AWS Management Console Private Access feature to access the AWS Management Console through a VPC endpoint and use `aws:SourceVpc` conditions in your policies. For more information, see the following:

- *AWS Management Console Private Access*
- the section called “How AWS Management Console Private Access works with aws:SourceVpc”
- the section called “Supported AWS global condition context keys”

My browser displays timeout errors when connecting to the AWS Management Console

If there's a service outage in your default AWS Region, your browser might display a 504 Gateway Timeout error when trying to connect to the AWS Management Console. To log in to the AWS Management Console from a different Region, specify an alternate Regional endpoint in the URL. For example, if there's an outage in the `us-west-1` (N. California) Region, to access the `us-west-2` (Oregon) Region use the following template:

```
https://region.console.aws.amazon.com
```

For more information, see AWS Management Console service endpoints in the *AWS General Reference*.

To view the status of all AWS services, including the AWS Management Console, see AWS Health Dashboard.

I want to change the language of the AWS Management Console but I can't find the language selection menu at the bottom of the page

The language selection menu has moved to the new Unified Settings page. To change the language of the AWS Management Console, navigate to the Unified Settings page, and then choose the language for the console.

For more information, see Changing the language of the AWS Management Console.

Document history

The following table describes important changes to the *AWS Management Console Getting Started Guide*, beginning in March 2021.

Change	Description	Date
Updated AWS managed policies	Updated the AWSManagementConsoleAdministratorAccess and AWSManagementConsoleBasicUserAccess policies with new UXC permissions. For more information, see ???.	March 26, 2026
Page added	New page added to explain recommended actions. For more information, see ???.	October 15, 2025
New AWS managed policies	Added two new policies to scope permissions for using, configuring, and customizing the AWS Management Console. • AWSManagementConsoleBasicUserAccess • AWSManagementConsoleAdministratorAccess	August 14, 2025
User Experience Customizations (UXC)	New service available.	August 14, 2025
Page updated	You can now view your applications in myApplications from the Services menu. For more information, see ???.	July 29, 2025

Change	Description	Date
Page added	New page added to explain multisession feature. For more information, see ???.	December 6, 2024
Page updated	Changing your password page updated. For more informati on, see ???.	June 18, 2024
New pages added	New pages added to describe how to access the Services menu and AWS event notifications. For more information, see ??? and ???.	June 18, 2024
Page updated	What is the AWS Managemen t Console? page updated. For more information, see ???.	June 18, 2024
Get support	A new page added to describe how to get support. For more information, see ???.	June 18, 2024
Unified Navigation and AWS Console Home	New pages added to describe how to work with the console. For more information, see ??? and ???.	June 18, 2024
Chat with Amazon Q	A new settings page detailing how users can ask AWS questions to Amazon Q Developer. For more informati on, see Chat with Amazon Q Developer.	May 29, 2024

Change	Description	Date
myApplications	A new page that introduces myApplications. For more information, see What is myApplications on AWS?.	November 29, 2023
Configuring Unified Settings	A new settings page for configuring settings and defaults that apply to the current user, including language and region. For more information, see Configuring Unified Settings.	April 6, 2022
New AWS Console Home UI	New AWS Console Home UI, which includes widgets for displaying important usage information and shortcuts to AWS services. For more information, see Working with widgets.	February 25, 2022
Changing the Console language	Choose a different language for the AWS Management Console. For more information, see Changing the language of the AWS Management Console.	April 1, 2021
Launching CloudShell	Open AWS CloudShell from the AWS Management Console and run AWS CLI commands. For more information, see Launching AWS CloudShell.	March 22, 2021