Getting Started Guide

AWS Management Console

Version 1.0

AWS Management Console: Getting Started Guide

Table of Contents

What is the AWS Management Console? 1
Using the device of your choice 1
Configuring the AWS Management Console 2
Working with widgets 2
.... 2
Configuring Unified Settings 4
Choosing a Region 6
Adding and removing favorites 7
Changing your password 8
Changing the language of the AWS Management Console 8
Getting started with a service 11
Searching for products, services, features, and more 12
myApplications on AWS 13
Features of myApplications 13
Related services 13
Accessing myApplications 14
Pricing 14
Supported Regions 14
Opt-in Regions 15
Getting started with myApplications 15
Step 1: Creating an application 15
Step 2: Viewing applications 17
Managing applications 18
Editing applications 18
Deleting applications 19
Creating code snippets 19
Managing resources 20
Adding resources 20
Removing resources 20
myApplications dashboard 21
Application dashboard setup widget 21
Application summary widget 21
Compute widget 22
Cost and usage widget 22

AWS Security widget 22
DevOps widget 23
Monitoring and operations widget 23
Tags widget 24
AWS Management Console Private Access 25
Supported AWS Regions, service consoles, and features 25
Overview of AWS Management Console Private Access security controls 29
Account restrictions on the AWS Management Console from your network 29
Connectivity from your network to the internet 29
Required VPC endpoints and DNS configuration 29
DNS configuration for AWS Management Console and AWS Sign-In 30
VPC endpoints and DNS configuration for AWS services 32
Implementing service control policies and VPC endpoint policies 33
Using AWS Management Console Private Access with AWS Organizations service control policies 33
Allow AWS Management Console use for expected accounts and organizations only (trusted identities) 34
Implementing identity-based policies and other policy types 36
Supported AWS global condition context keys 36
How AWS Management Console Private Access works with aws:SourceVpc 36
How different network paths are reflected in CloudTrail 37
Try AWS Management Console Private Access 38
Test setup with Amazon EC2 38
Test setup with Amazon WorkSpaces 52
Test VPC setup with IAM policies 69
Reference architecture 71
Launching AWS CloudShell on the Console Toolbar 73
Getting billing information 74
Markdown in AWS 75
Paragraphs, Line Spacing, and Horizontal Lines 75
Headings 76
Text Formatting 76
Links 76
Lists 77
Tables and Buttons (CloudWatch Dashboards) 77
Troubleshooting 79

The page isn't loading properly 79
My browser displays an 'access denied' error when connecting to the AWS Management Console 80
My browser displays timeout errors when connecting to the AWS Management Console 80
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 81
Document history 82
AWS Glossary 83

What is the AWS Management Console?

The AWS Management Console is a web application that comprises and refers to a broad collection of service consoles for managing AWS resources. When you first sign in, you see the console home page. The home page provides access to each service console and offers a single place to access the information you need to perform your AWS related tasks. It also lets you customize the Console Home experience by adding, removing, and rearranging widgets such as **Recently visited**, **AWS Health**, **Trusted Advisor**, and more.

Note

The language selection option has moved to the new Unified Settings page. For more information, see Changing the language of the AWS Management Console.

The individual service consoles, on the other hand, offer a wide range of tools for cloud computing, as well as information about your account and about your billing.

Using the device of your choice

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

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

The AWS Management Console is also available as an app for Android and iOS. This app 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.

You can download the AWS Console mobile app from Amazon Appstore, Google Play, or iTunes.

Configuring the AWS Management Console

This topic describes how to configure your AWS Management Console and how to use the Unified Settings page to set defaults that apply to all service consoles. It also explains widgets, a feature of the Console Home dashboard that lets you add custom components that track information about your AWS services and resources.

Topics

- Working with widgets
- Configuring Unified Settings
- Choosing a Region
- Adding and removing favorites
- Changing your password
- Changing the language of the AWS Management Console

Working with widgets

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.

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

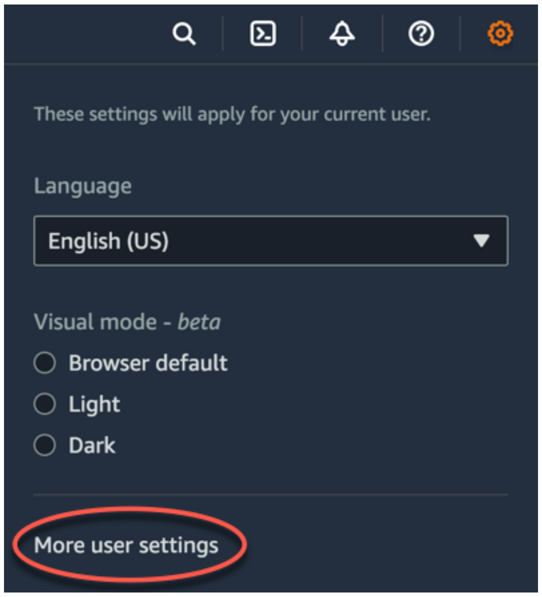
   Your suggestions are periodically reviewed and new widgets might be added in future updates to the AWS Management Console.

Configuring Unified Settings

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

To access Unified Settings

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the settings icon.
3. To open the **Unified Settings** page, choose **More user settings**.



These settings will apply for your current user.

Language

English (US)

Visual mode - *beta*

- Browser default
- Light
- Dark

More user settings

4. To delete all Unified Settings configurations and restore the default settings, choose **Reset all**.

   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.
5. 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.

6. Choose **Save changes**.

To change the visual mode from the navigation bar

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the settings 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.

To change the default language from the navigation bar

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

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.

Choosing a 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*.

To choose a Region

1. Sign in to the AWS Management Console.
2. Choose a service to go to that service's console.
3. In the navigation bar, choose the name of the currently displayed Region. Then choose the Region to which you want to switch.

To choose a default Region

1. In the navigation bar, choose the settings icon, and then choose **More user settings** to navigate to the **Unified Settings** page.
2. Choose **Edit** next to **Localization and default Region**.
3. Select your default Region, then choose **Save changes**. If you do not select a default Region, the last Region you visited will be your default.

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. To see them, use the Region selector to choose the Region that contains your resources.

Adding and removing favorites

To access your frequently used services more quickly, you can save their service consoles to a list of **Favorites**.

Note

Favorites are currently stored in browser cookies. If you delete your cookies between console sessions, your list of **Favorites** will be cleared.

To add a service to the list of Favorites

1. Sign in to the AWS Management Console.
2. Choose the **Add widgets** button on the upper or lower right side of the page.
3. In the **Add widgets** menu, select **Favorites** to add to the console, and then choose **Add**.

   The Favorites will be added to the bottom of your Console Home. You can drag and drop Favorites by selecting the title bar at the top of the widget, and then drag the widget to a new location on the page.
4. In the navigation bar, choose **Services**.
5. In either the **Recently visited** list or the **All services** list, hover over the name of the service that you want to add as a favorite.
6. Select the star to the left of the service name.
7. Repeat the previous two steps to add more services to your **Favorites** list.

To remove a service from the list of Favorites

1. In the navigation bar, choose **Services**.

2. Do one of the following:

   - In the **Favorites** list, hover over the name of a service. Then choose the **×** to the right of the service name.
   - In the **Recently visited** list or **All services** list, deselect the star by the name of a service that is in your **Favorites** list.

Changing your password

If you are an account owner, you can change your AWS account password from the AWS Management Console.

To change your password

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**.

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, which you can access from the navigation bar. You can make this change from anywhere in the console.

Note

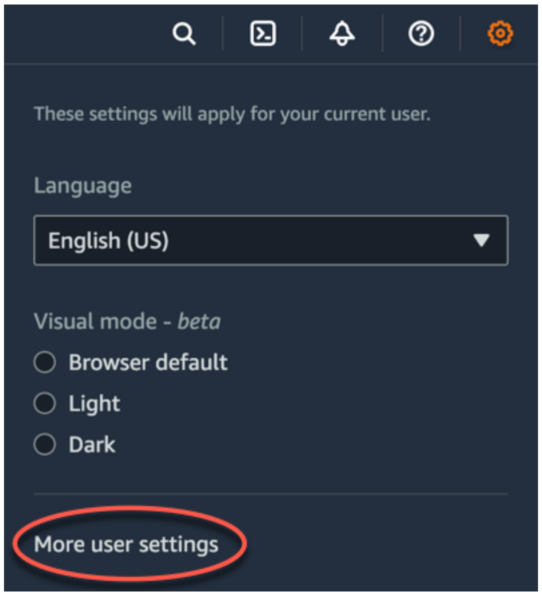
This procedure changes the language for all consoles, but not for AWS documentation. To change the language used for documentation, use the language menu in the upper right of any documentation page.

The AWS Management Console currently supports the following languages:

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

To change the default language in Unified Settings

1. Sign in to the AWS Management Console.
2. In the navigation bar, choose the settings icon.
3. To open the **Unified Settings** page, choose **More user settings**.



These settings will apply for your current user.

Language

English (US)

Visual mode - *beta*

Browser default

Light

Dark

More 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.

To change the default language from the navigation bar

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

Getting started with a service

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.
- 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**.

Searching for products, services, features, and more

The search box in the navigation bar provides a unified search tool for tracking down AWS services and features, service documentation, and AWS Marketplace. Just type in a few characters to see results from all these categories. The more characters you type, the more search refines your results.

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

1. In search box on the navigation bar of the AWS Management Console, enter all or part of your search terms.
2. Do any of the following to refine your search and get more detail:
    - To narrow the results to the type of content that you want, choose one of the categories on the left.
    - To see more results for a particular category, choose **See all *n* results** by each category heading. To return to the main results list, choose **Back** in the top left corner.
    - To quickly navigate to popular features of a service, pause on the service name in the results and choose a link.
    - To get more detail about a documentation or AWS Marketplace result, pause on the result title.
3. Choose any link to navigate to your intended service, topic, or AWS Marketplace page.

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**.

What is myApplications on AWS

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. 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

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.
- **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, 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
- 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

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:

- Asia Pacific (Jakarta)
- Middle East (Bahrain)

Getting started with myApplications

To get started using myApplications to create, monitor, and mange your applications, use the following steps.

Step 1: Creating an application

Create a new application or onboard an existing AppRegistry application created before November 8, 2023 to get started with myApplications.

Create an application

To create an application

1. Sign in to the AWS Management Console.
2. In the left sidebar, 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 and can be used to identify resources associated with your application. 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 existing 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.

   a. Choose **Select resources**.
   b. (Optional) Choose a view.
   c. 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*.

    d. Select the checkbox next to the resources you want to add.
    e. Choose **Add**.
    f. Choose **Next**.

10. Review your choices.
11. If associating a AWS CloudFormation stack, select the checkbox at the bottom of the page.

    Note

    Adding a AWS 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 *AWS CloudFormation User Guide*.

12. Choose **Create application**.

Onboard existing application

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**.

Step 2: Viewing applications

You can view your applications across all Regions or specific Regions and their relevant information in a card or table view.

To view applications

1. In the left sidebar, choose **myApplications**.
2. In **Regions**, select **Current Region** or **Supported Regions**.
3. To find a specific application, enter its name, keywords, or description in the search bar.
4. (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**.

Managing applications

This topic covers how you can manage your applications.

Editing applications

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 description**, update the description, and then choose **Save changes**.

To edit tags

- Follow the steps in Managing tags in the *AWS Service Catalog AppRegistry Administrator Guide*.

To edit attribute groups

- Follow the steps in Editing attribute groups in the *AWS Service Catalog AppRegistry Administrator Guide*.

Deleting applications

You can delete applications if they are no longer needed.

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. Choose **Delete**.
7. Confirm your deletion, and then choose **Delete application**.

Creating code snippets

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.

Managing resources

This topic covers how to manage your resources.

Adding resources

Adding resources to your applications allows you to group them and manage their security, performance, and compliance.

To add resources

1. Open the AWS Management Console.
2. In the left sidebar of the console, choose **myApplications**.
3. Seearch 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**.

Removing resources

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**.

myApplications dashboard

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.

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.

Configuring the AWS Security widget

To configure the AWS Security widget, set up AWS Security Hub for your application and account. For more information, see What is AWS Security Hub? in the *AWS Security Hub User Guide*. For pricing information, see AWS Security Hub free trial, usage, and pricing in the *AWS Security Hub User Guide*.

AWS Security Hub 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*.

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 insights 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*.

AWS Management Console Private Access

AWS Management Console Private Access is an advanced security feature to control access to the AWS Management Console. AWS Management 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.

Topics

- Supported AWS Regions, service consoles, and features
- Overview of AWS Management Console Private Access security controls
- Required VPC endpoints and DNS configuration
- Implementing service control policies and VPC endpoint policies
- Implementing identity-based policies and other policy types
- Try AWS Management Console Private Access
- Reference architecture

Supported AWS Regions, service consoles, and features

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 (Oregon)
- Asia Pacific (Hyderabad)
- Asia Pacific (Mumbai)

- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Ireland)
- Europe (London)
- Israel (Tel Aviv)

Supported service consoles

- Amazon API Gateway
- AWS App Mesh
- AWS Application Migration Service
- Amazon Athena
- AWS Auto Scaling
- AWS Billing Conductor
- AWS Certificate Manager
- AWS Cloud Map
- Amazon CloudFront
- Amazon CloudWatch
- AWS CodeArtifact
- Amazon CodeGuru
- Amazon Comprehend
- Amazon Comprehend Medical
- AWS Compute Optimizer
- AWS Console Home
- AWS Database Migration Service
- AWS DeepRacer

- 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
- Amazon ElastiCache
- Amazon EMR
- Amazon EventBridge
- Amazon GameLift
- AWS Global Accelerator
- AWS Glue DataBrew
- AWS Ground Station
- Amazon GuardDuty
- 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 Video Streams

- AWS Lambda
- Amazon Lex
- AWS License Manager
- Amazon Managed Grafana
- 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 Manager
- Amazon OpenSearch Service
- AWS Organizations
- Amazon S3 on Outposts
- Amazon SageMaker Runtime
- Amazon SageMaker Synthetic Data
- AWS Secrets Manager
- Service Quotas
- AWS Signer
- Amazon Simple Email Service
- Amazon Simple Queue Service
- Amazon Simple Storage Service (Amazon S3)
- AWS SQL Workbench
- AWS Step Functions
- AWS Support
- AWS Systems Manager
- AWS Transfer Family
- Unified Settings
- Amazon VPC IP Address Manager

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*.

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:

- Regional CNAME records (in all Regions)
- region.signin.aws.amazon.com pointing to the AWS Sign-In VPC endpoint in the signin DNS zone
- 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-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.signin.aws.amazon.com
- support.console.aws.amazon.com
- global.console.aws.amazon.com
- Additional records present in the previously listed JSON files

VPC endpoints and DNS configuration for AWS services

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-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.

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.

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

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

```
{
  "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

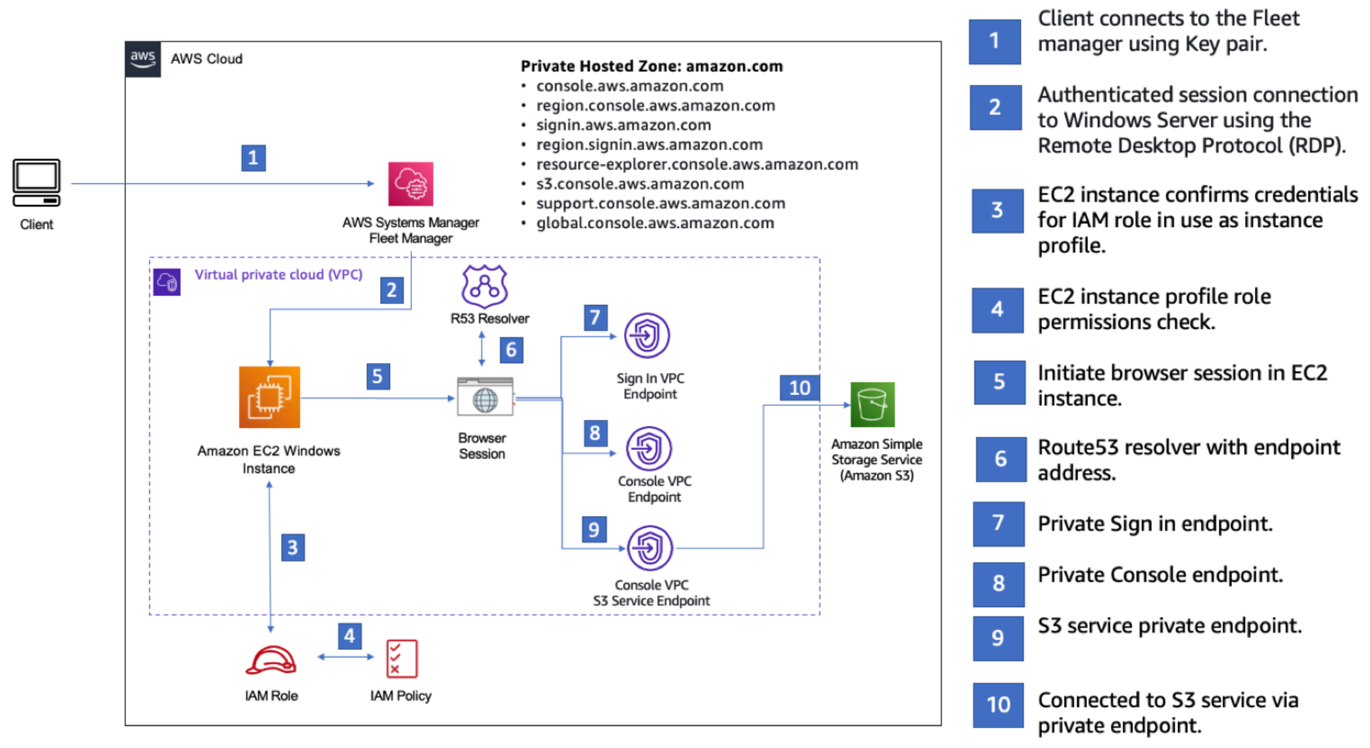
- Test setup with Amazon EC2
- Test setup with Amazon WorkSpaces
- Test VPC setup with IAM policies

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 AWS 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.


AWS Cloud
Private Hosted Zone: amazon.com
• console.aws.amazon.com
• region.console.aws.amazon.com
• signin.aws.amazon.com
• region.signin.aws.amazon.com
• resource-explorer.console.aws.amazon.com
• s3.console.aws.amazon.com
• support.console.aws.amazon.com
• global.console.aws.amazon.com
Client
AWS Systems Manager Fleet Manager
Virtual private cloud (VPC)
R53 Resolver
Amazon EC2 Windows Instance
Browser Session
Sign In VPC Endpoint
Console VPC Endpoint
Console VPC S3 Service Endpoint
Amazon Simple Storage Service (Amazon S3)
IAM Role
IAM Policy
1
2
3
4
5
6
7
8
9
10

1 Client connects to the Fleet manager using Key pair.

2 Authenticated session connection to Windows Server using the Remote Desktop Protocol (RDP).

3 EC2 instance confirms credentials for IAM role in use as instance profile.

4 EC2 instance profile role permissions check.

5 Initiate browser session in EC2 instance.

6 Route53 resolver with endpoint address.

7 Private Sign in endpoint.

8 Private Console endpoint.

9 S3 service private endpoint.

10 Connected to S3 service via private endpoint.

Copy the following AWS 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 AWS CloudFormation template uses configurations that are currently not supported in the Israel (Tel Aviv) Region.

AWS Management Console Private Access environment Amazon EC2 AWS 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: 't2.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

  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

  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 AWS CloudFormation console.
2. Choose **Create stack**.

3. Choose **With new resources (standard)**. Upload the AWS 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 for Linux Instances*.
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 AWS CloudFormation template. Then choose **Connect**.

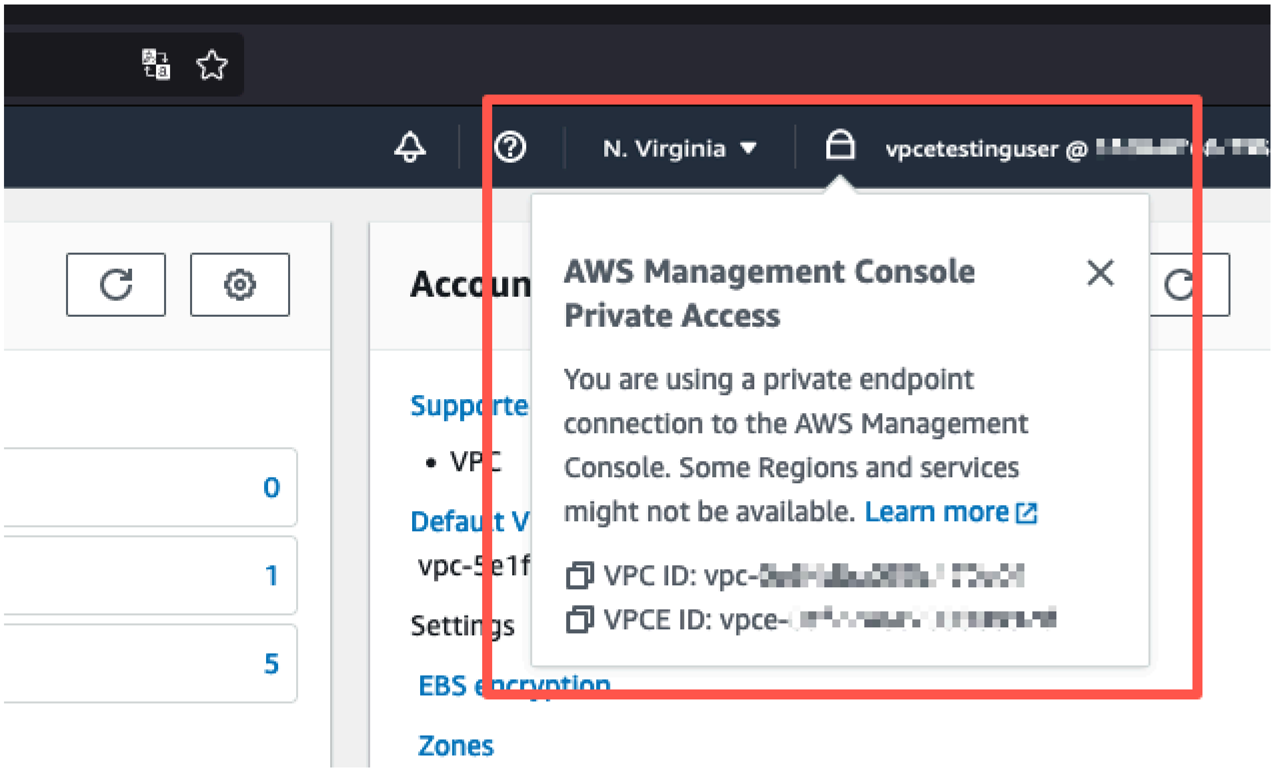
   Note

   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 AWS 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.



AWS Management Console Private Access

You are using a private endpoint connection to the AWS Management Console. Some Regions and services might not be available. Learn more

VPC ID: vpc-[redacted]

VPCE ID: vpce-[redacted]

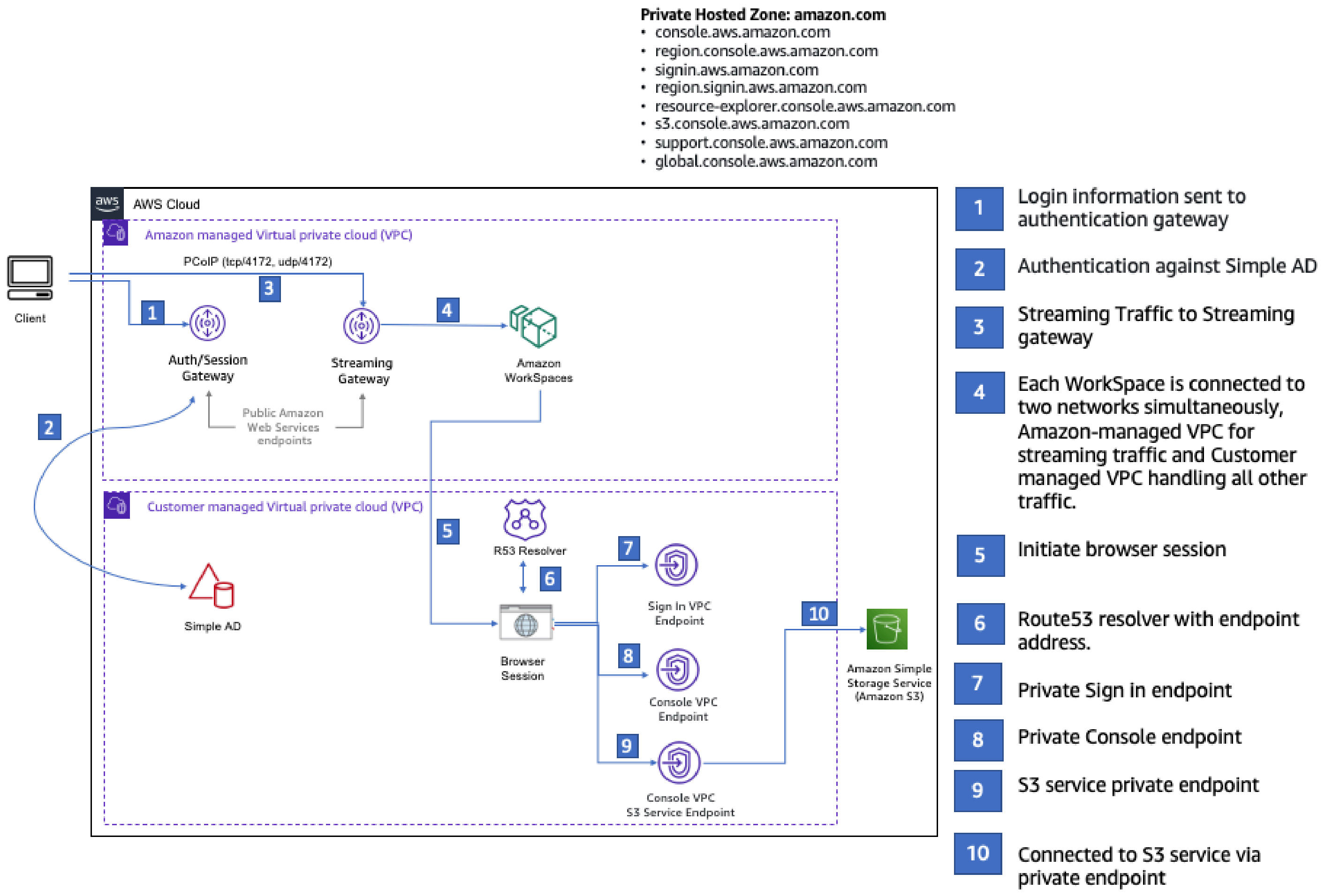
Test setup with Amazon WorkSpaces

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.


AWS Cloud
Amazon managed Virtual private cloud (VPC)
PCoIP (tcp/4172, udp/4172)
Client
1
2
3
4
Auth/Session Gateway
Streaming Gateway
Amazon WorkSpaces
Public Amazon Web Services endpoints
Customer managed Virtual private cloud (VPC)
5
R53 Resolver
6
Simple AD
Browser Session
7
Sign In VPC Endpoint
8
Console VPC Endpoint
9
Console VPC S3 Service Endpoint
10
Amazon Simple Storage Service (Amazon S3)

Private Hosted Zone: amazon.com

- console.aws.amazon.com
- region.console.aws.amazon.com
- signin.aws.amazon.com
- region.signin.aws.amazon.com
- resource-explorer.console.aws.amazon.com
- s3.console.aws.amazon.com
- support.console.aws.amazon.com
- global.console.aws.amazon.com

1	Login information sent to authentication gateway
2	Authentication against Simple AD
3	Streaming Traffic to Streaming gateway
4	Each WorkSpace is connected to two networks simultaneously, Amazon-managed VPC for streaming traffic and Customer managed VPC handling all other traffic.
5	Initiate browser session
6	Route53 resolver with endpoint address.
7	Private Sign in endpoint
8	Private Console endpoint
9	S3 service private endpoint
10	Connected to S3 service via private endpoint

Copy the following AWS 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 AWS 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

# 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

  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

  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: "ADAdminSecret"
      Description: "Password for directory services admin"
      GenerateSecretString:
        SecretStringTemplate: '{"username": "Admin"}'
        GenerateStringKey: password
        PasswordLength: 30
        ExcludeCharacters: '"@/\'

  WorkspaceSimpleDirectory:
    Type: AWS::DirectoryService::SimpleAD
    DependsOn: AppVPC
    DependsOn: PrivateSubnetA
    DependsOn: PrivateSubnetB
    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
```

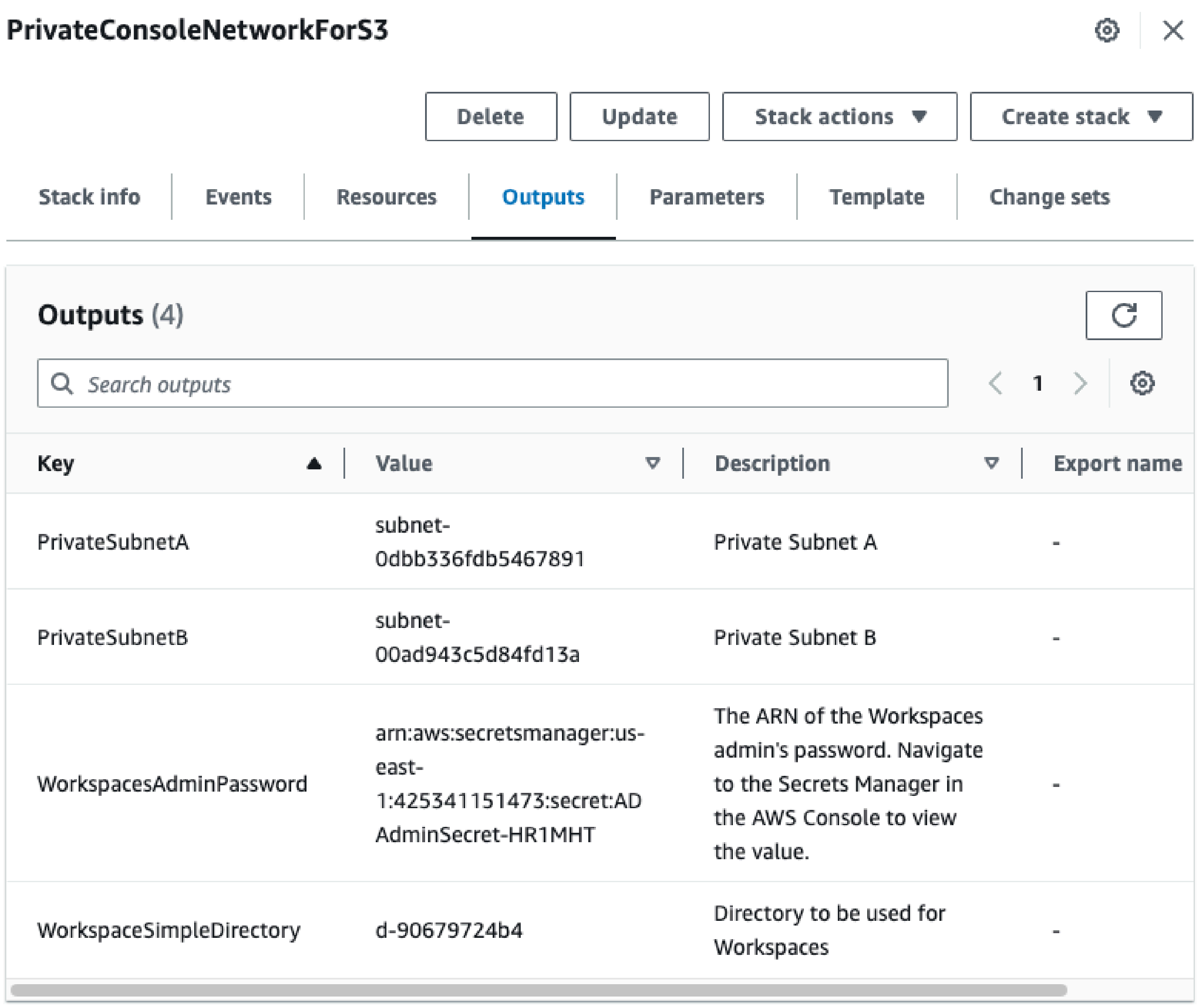
Note

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 AWS CloudFormation console.
2. Choose **Create stack**.
3. Choose **With new resources (standard)**. Upload the AWS 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.

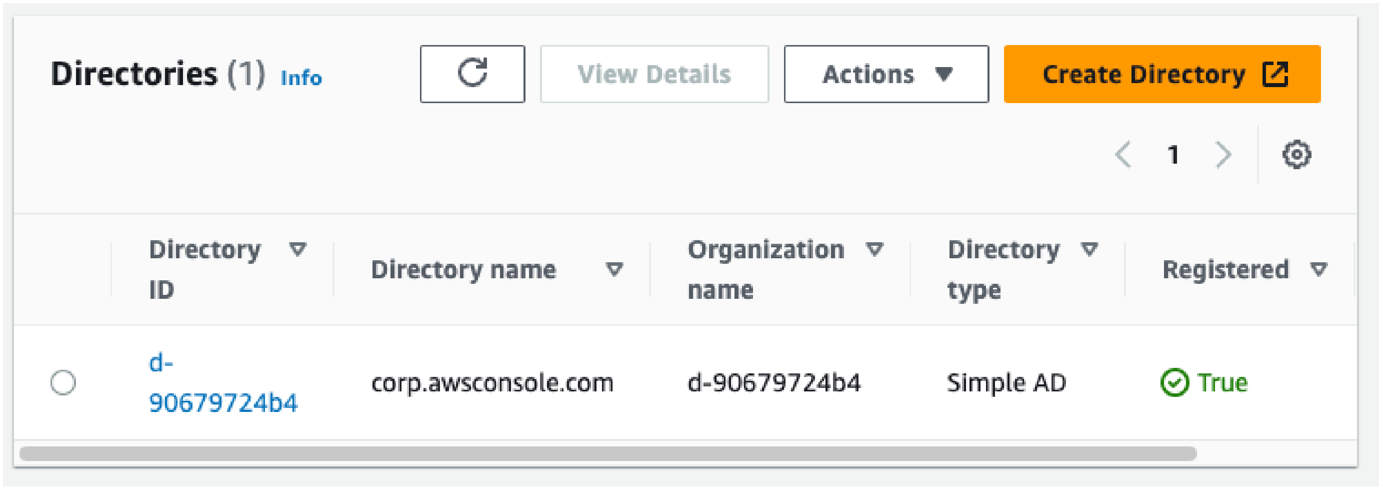


Key	Value	Description	Export name
PrivateSubnetA	subnet-0dbb336fdb5467891	Private Subnet A	-
PrivateSubnetB	subnet-00ad943c5d84fd13a	Private Subnet B	-
WorkspacesAdminPassword	arn:aws:secretsmanager:us-east-1:425341151473:secret:ADAdminSecret-HR1MHT	The ARN of the Workspaces admin's password. Navigate to the Secrets Manager in the AWS Console to view the value.	-
WorkspaceSimpleDirectory	d-90679724b4	Directory to be used for Workspaces	-

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.



	Directory ID	Directory name	Organization name	Directory type	Registered
○	d-90679724b4	corp.awsconsole.com	d-90679724b4	Simple AD	✓ True

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 AWS 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

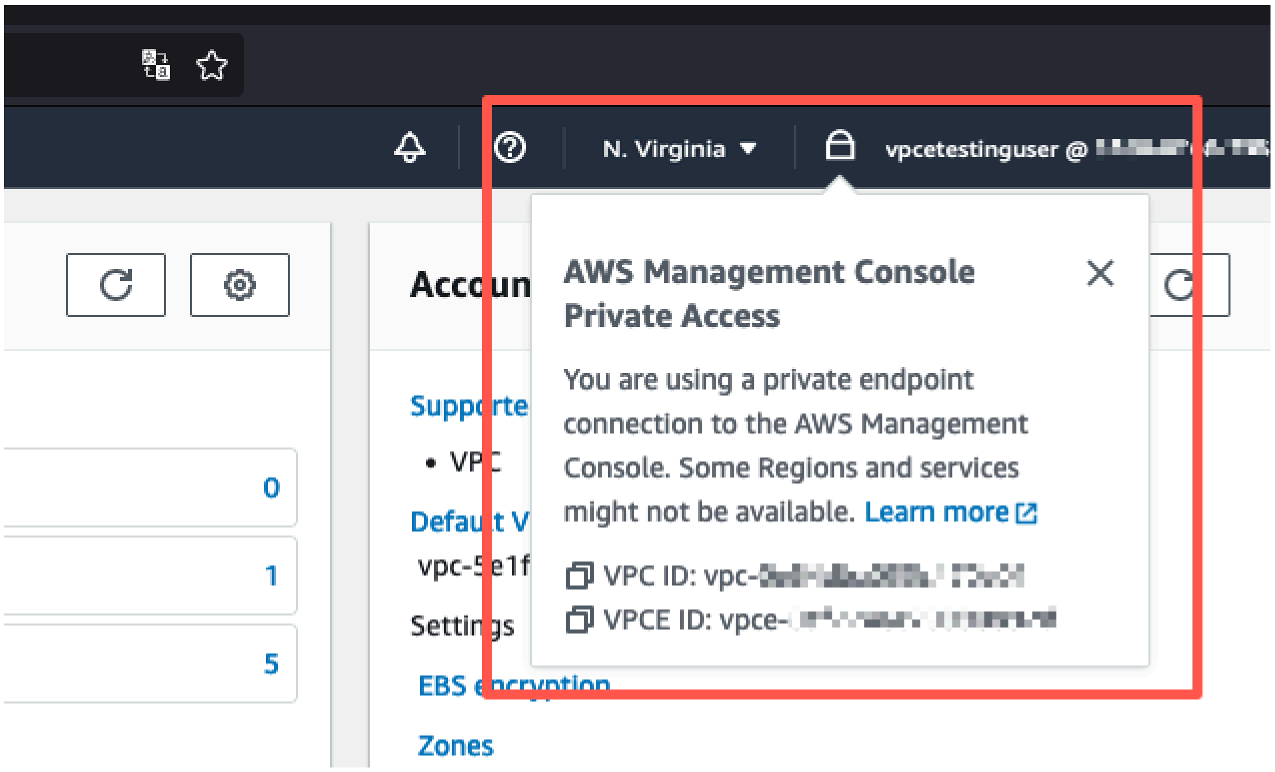
1. Open the email that you received in step 14 of the preceding procedure.
2. In the email, choose the unique link that is provided to set up your profile and download the WorkSpaces client.
3. Set your password.
4. Download the client of your choice.
5. Install and launch the client. Enter the registration 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.



AWS Management Console Private Access

You are using a private endpoint connection to the AWS Management Console. Some Regions and services might not be available. Learn more

VPC ID: vpc-[redacted]

VPCE ID: vpce-[redacted]

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.

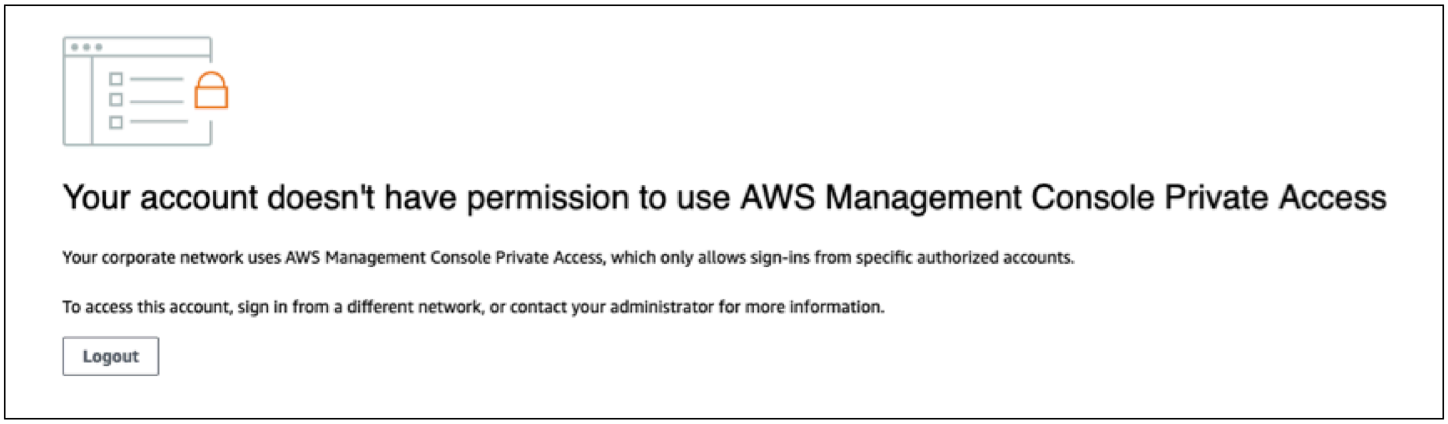
The following policy denies access to Amazon S3 unless it is using your specified VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "S3:*",
            "Resource": "*",
            "Condition": {
                "StringNotEqualsIfExists": {
                    "aws:SourceVpc": "sourceVPC"
                },
                "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.

```
{
    "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.



Your account doesn't have permission to use AWS Management Console Private Access

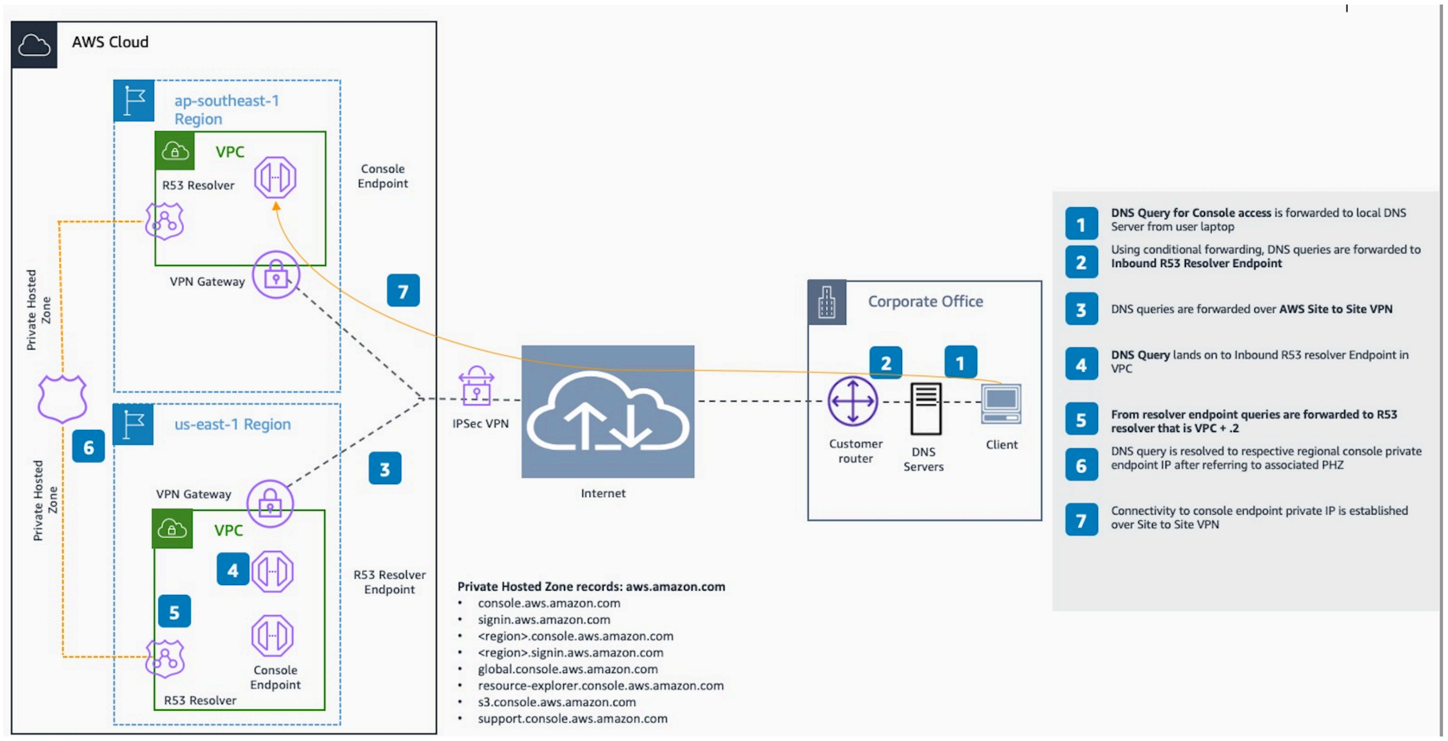
Your corporate network uses AWS Management Console Private Access, which only allows sign-ins from specific authorized accounts.

To access this account, sign in from a different network, or contact your administrator for more information.

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)



1. **DNS Query for Console access** is forwarded to local DNS Server from user laptop
2. Using conditional forwarding, DNS queries are forwarded to **Inbound R53 Resolver Endpoint**
3. DNS queries are forwarded over **AWS Site to Site VPN**
4. **DNS Query** lands on to Inbound R53 resolver Endpoint in VPC
5. **From resolver endpoint queries are forwarded to R53 resolver that is VPC + .2**
6. DNS query is resolved to respective regional console private endpoint IP after referring to associated PHZ
7. Connectivity to console endpoint private IP is established over Site to Site VPN

Private Hosted Zone records: aws.amazon.com

- console.aws.amazon.com
- signin.aws.amazon.com
- <region>.console.aws.amazon.com
- <region>.signin.aws.amazon.com
- global.console.aws.amazon.com
- resource-explorer.console.aws.amazon.com
- s3.console.aws.amazon.com
- support.console.aws.amazon.com

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.

Launching AWS CloudShell on the Console Toolbar

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

You can launch CloudShell on the Console Toolbar using one of the following two methods:

- Choose the CloudShell icon on the lower left 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.

Getting billing information

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

To get your billing information

1. On the navigation bar, choose your account name.
2. Choose **Billing Dashboard**.
3. Use the AWS Billing and Cost Management dashboard to find a summary and a breakdown of your monthly spending. To learn more, see the AWS Billing User Guide.

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
- Headings
- Text Formatting
- Links
- Lists
- Tables and Buttons (CloudWatch Dashboards)

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 (`&nbsp;`) and then a blank line. Repeat this pair of lines to insert multiple blank lines one after the other, as in the following example:

```
&nbsp;

&nbsp;
```

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.

Topics

- The page isn't loading properly
- My browser displays an 'access denied' error when connecting to the AWS Management Console
- My browser displays timeout errors when connecting to the AWS Management Console
- 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 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 you're using all of the following:

- A browser from within a VPC.
- VPC endpoints.
- IAM policies that contain an `aws:SourceIp` global condition key.

In the console, go to **IAM policies page**. We recommend you review the IAM policies that contain an `aws:SourceIp` global condition key and add `aws:SourceVpc` key.

Alternatively, you can consider onboarding 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 *AWS Management Console Private Access*.

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-code.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
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 informati on, see Changing the language of the AWS Management Console.	April 1, 2021
Launching CloudShell	Open AWS CloudShell from the AWS Managemen t Console and run AWS CLI commands. For more information, see Launching AWS CloudShell.	March 22, 2021

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.