
AWS Cloud Map

Developer Guide



AWS Cloud Map: Developer Guide

Copyright © Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What Is AWS Cloud Map?	1
Accessing AWS Cloud Map	1
AWS Identity and Access Management	2
AWS Cloud Map Pricing	2
AWS Cloud Map and AWS Cloud Compliance	2
Setting Up	3
Sign Up for AWS	3
Access Your Account	3
Access the AWS Management Console	4
Access the API, AWS CLI, AWS Tools for Windows PowerShell, or the AWS SDKs	4
Create an IAM User	4
Set Up the AWS Command Line Interface or AWS Tools for Windows PowerShell	6
Download an AWS SDK	6
Using AWS Cloud Map	7
Overview of How to Use AWS Cloud Map	7
Configuring AWS Cloud Map	9
Working with Namespaces	9
Working with Services	13
Working with Service Instances	20
AWS Cloud Map Features That Aren't Available on the AWS Cloud Map Console	25
Security	26
AWS Identity and Access Management	26
Authentication	27
Access Control	27
Overview of Managing Access	28
Using IAM Policies for AWS Cloud Map	31
AWS Cloud Map API Permissions Reference	34
Logging and Monitoring	38
Logging AWS Cloud Map API Calls with AWS CloudTrail	38
Compliance Validation	40
Resilience	40
Infrastructure Security	41
Tagging your resources	42
Tag basics	42
Tagging your resources	42
Tag restrictions	43
Working with tags using the CLI or API	43
AWS Cloud Map quotas	45
API request throttling quota	45
How throttling is applied	45
Adjusting API throttling quotas	46
Related Information	47
AWS resources	47
Third-Party Tools and Libraries	47
Document History	48
AWS glossary	49

What Is AWS Cloud Map?

AWS Cloud Map is a fully managed service that you can use to create and maintain a map of the backend services and resources that your applications depend on. Here's how AWS Cloud Map works:

1. You create a namespace that identifies the name that you want to use to locate your resources and also specifies how you want to locate resources: using AWS Cloud Map [DiscoverInstances](#) API calls, DNS queries in a VPC, or public DNS queries. In most cases, a namespace contains all the services for an application, such as a billing application.
2. You create an AWS Cloud Map service for each type of resource for which you want to use AWS Cloud Map to locate endpoints. For example, you might create services for web servers and database servers.

A service is a template that AWS Cloud Map uses when your application adds another resource, such as another web server. If you chose to locate resources using DNS when you created the namespace, a service contains information about the types of records that you want to use to locate the web server. A service also indicates whether you want to check the health of the resource and, if so, whether you want to use Amazon Route 53 health checks or a third-party health checker.

3. When your application adds a resource, it can call the AWS Cloud Map [RegisterInstance](#) API action, which creates a service instance. The service instance contains information about how your application can locate the resource, whether using DNS or using the AWS Cloud Map [DiscoverInstances](#) API action.
4. When your application needs to connect to a resource, it calls [DiscoverInstances](#) and specifies the namespace and service that are associated with the resource. AWS Cloud Map returns information about how to locate one or more resources. If you specified health checking when you created the service, AWS Cloud Map returns only healthy instances.

AWS Cloud Map is tightly integrated with Amazon Elastic Container Service (Amazon ECS). As new container tasks spin up or down, they automatically register with AWS Cloud Map. You can use the Kubernetes ExternalDNS connector to integrate Amazon Elastic Kubernetes Service with AWS Cloud Map. You can also use AWS Cloud Map to register and locate any cloud resources, such as Amazon EC2 instances, Amazon DynamoDB tables, Amazon S3 buckets, Amazon Simple Queue Service (Amazon SQS) queues, or APIs deployed on top of Amazon API Gateway, among others. You can specify attribute values for services instances, and clients can use these attributes to filter the resources that AWS Cloud Map returns. For example, an application can request resources in a particular deployment stage, like BETA or PROD.

Topics

- [Accessing AWS Cloud Map \(p. 1\)](#)
- [AWS Identity and Access Management \(p. 2\)](#)
- [AWS Cloud Map Pricing \(p. 2\)](#)
- [AWS Cloud Map and AWS Cloud Compliance \(p. 2\)](#)

Accessing AWS Cloud Map

You can access AWS Cloud Map in the following ways:

- **AWS Management Console** – The procedures throughout this guide explain how to use the AWS Management Console to perform tasks.
- **AWS SDKs** – If you're using a programming language that AWS provides an SDK for, you can use an SDK to access AWS Cloud Map. SDKs simplify authentication, integrate easily with your development

environment, and provide access to AWS Cloud Map commands. For more information, see [Tools for Amazon Web Services](#).

- **AWS Cloud Map API** – If you're using a programming language that an SDK isn't available for, see the [AWS Cloud Map API Reference](#) for information about API actions and about how to make API requests.
- **AWS Command Line Interface** – For more information, see [Getting Set Up with the AWS Command Line Interface](#) in the *AWS Command Line Interface User Guide*.
- **AWS Tools for Windows PowerShell** – For more information, see [Setting up the AWS Tools for Windows PowerShell](#) in the *AWS Tools for Windows PowerShell User Guide*.

AWS Identity and Access Management

AWS Cloud Map integrates with AWS Identity and Access Management (IAM), a service that your organization can use to do the following actions:

- Create users and groups under your organization's AWS account
- Share your AWS account resources among the users in the account in an efficient manner
- Assign unique security credentials to each user
- Granularly control user access to services and resources

For example, you can use IAM with AWS Cloud Map to control which users in your AWS account can create a new namespace or register instances.

For general information about IAM, see the following resources:

- [AWS Identity and Access Management in AWS Cloud Map \(p. 26\)](#)
- [AWS Identity and Access Management](#)
- [IAM User Guide](#)

AWS Cloud Map Pricing

AWS Cloud Map pricing is based on resources that you register in the service registry and API calls that you make to discover them. With AWS Cloud Map there are no upfront payments, and you only pay for what you use.

Optionally, you can enable DNS-based discovery for the resources with IP addresses. You can also enable health checking for your resources using Amazon Route 53 health checks, whether you're discovering instances using API calls or DNS queries. You will incur additional charges related to Route 53 DNS and health check usage.

For more information, see [AWS Cloud Map Pricing](#).

AWS Cloud Map and AWS Cloud Compliance

For information about AWS Cloud Map compliance with various security compliance regulations and audits standards, see the following pages:

- [AWS Cloud Compliance](#)
- [AWS Services in Scope by Compliance Program](#)

Setting Up AWS Cloud Map

The overview and procedures in this section are meant to help you get started with AWS.

Topics

- [Sign Up for AWS \(p. 3\)](#)
- [Access Your Account \(p. 3\)](#)
- [Create an IAM User \(p. 4\)](#)
- [Set Up the AWS Command Line Interface or AWS Tools for Windows PowerShell \(p. 6\)](#)
- [Download an AWS SDK \(p. 6\)](#)

Sign Up for AWS

When you sign up for AWS, your AWS account is automatically signed up for all services in AWS, including AWS Cloud Map. You're charged only for the services that you use.

If you have an AWS account already, skip to [Access Your Account \(p. 3\)](#). If you don't have an AWS account, use the following procedure to create one.

To create an AWS account

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

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

Note your AWS account number, because you'll need it later.

Access Your Account

You use AWS services by using any of the following options:

- AWS Management Console
- API for each service
- AWS Command Line Interface (AWS CLI)
- AWS Tools for Windows PowerShell
- AWS SDKs

For each of those options, you need to access your AWS account by providing credentials that verify that you have permissions to use the services.

Access the AWS Management Console

To access the AWS Management Console for the first time, you need to provide an email address and a password. The combination of your email address and password is referred to as your *root identity* or *root account credentials*. We strongly recommend that, after accessing your account for the first time, you don't use your root account credentials again for everyday use. Instead, you should create new credentials by using [AWS Identity and Access Management](#). To do that, you create a user account for yourself known as an *IAM user*, and then add the IAM user to an IAM group with administrative permissions. Or, you can alternatively grant the IAM user administrative permissions. You then can access AWS using a special URL and the credentials for the IAM user. You also can add other IAM users later, and restrict their access to specified resources in the account.

Note

Some ad-blocking plugins for web browsers interfere with AWS Cloud Map console operations, which can cause the console to behave unpredictably. If you installed an ad-blocking plugin for your browser, we recommend that you add the URL for the AWS Cloud Map console, <https://console.aws.amazon.com/cloudmap/home>, to the approved list for the plugin.

Access the API, AWS CLI, AWS Tools for Windows PowerShell, or the AWS SDKs

To use the API, the AWS CLI, AWS Tools for Windows PowerShell, or the AWS SDKs, you must create *access keys*. These keys consist of an access key ID and secret access key, which are used to sign programmatic requests that you make to AWS.

To create the keys, you sign in to the AWS Management Console. We strongly recommend that you sign in with your IAM user credentials instead of your root credentials. For more information, see [Managing Access Keys for IAM Users](#) in the *IAM User Guide*.

Create an IAM User

Perform the following procedures to create a group for administrators, create an IAM user, and then add the IAM user to the administrators group. If you signed up for AWS but have not created an IAM user for yourself, you can create one using the IAM console. If you aren't familiar with using the console, see [Working with the AWS Management Console](#) for an overview.

To create an administrator user for yourself and add the user to an administrators group (console)

1. Sign in to the [IAM console](#) as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

Note

We strongly recommend that you adhere to the best practice of using the **Administrator** IAM user that follows and securely lock away the root user credentials. Sign in as the root user only to perform a few [account and service management tasks](#).

2. In the navigation pane, choose **Users** and then choose **Add user**.
3. For **User name**, enter **Administrator**.
4. Select the check box next to **AWS Management Console access**. Then select **Custom password**, and then enter your new password in the text box.
5. (Optional) By default, AWS requires the new user to create a new password when first signing in. You can clear the check box next to **User must create a new password at next sign-in** to allow the new user to reset their password after they sign in.
6. Choose **Next: Permissions**.

7. Under **Set permissions**, choose **Add user to group**.
8. Choose **Create group**.
9. In the **Create group** dialog box, for **Group name** enter **Administrators**.
10. Choose **Filter policies**, and then select **AWS managed - job function** to filter the table contents.
11. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.

Note

You must activate IAM user and role access to Billing before you can use the `AdministratorAccess` permissions to access the AWS Billing and Cost Management console. To do this, follow the instructions in [step 1 of the tutorial about delegating access to the billing console](#).

12. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to see the group in the list.
13. Choose **Next: Tags**.
14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM entities](#) in the *IAM User Guide*.
15. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

You can use this same process to create more groups and users and to give your users access to your AWS account resources. To learn about using policies that restrict user permissions to specific AWS resources, see [Access management](#) and [Example policies](#).

To sign in as your new IAM user

1. Sign out of the AWS Management Console.
2. Sign in by using the following URL, where `your_aws_account_id` is your AWS account number without the hyphens. For example, if your AWS account number is 1234-5678-9012, your AWS account ID is 123456789012:

```
https://your_aws_account_id.signin.aws.amazon.com/console/
```

3. Enter the IAM user name (not your email address) and password that you just created. When you're signed in, the navigation bar displays "`your_user_name @ your_aws_account_id`".

If you don't want the URL for your sign-in page to contain your AWS account ID, you can create an account alias.

To create an account alias and conceal your account ID

1. On the IAM console, choose **Dashboard** in the navigation pane.
2. On the dashboard, choose **Customize** and enter an alias such as your company name.
3. Sign out of the AWS Management Console.
4. Sign in by using the following URL:

```
https://your_account_alias.signin.aws.amazon.com/console/
```

To verify the sign-in link for IAM users for your account, open the IAM console and check under **IAM users sign-in link** on the dashboard.

For more information about using IAM, see [AWS Identity and Access Management in AWS Cloud Map \(p. 26\)](#).

Set Up the AWS Command Line Interface or AWS Tools for Windows PowerShell

The AWS Command Line Interface (AWS CLI) is a unified tool for managing AWS services. For information about how to install and configure the AWS CLI, see [Getting Set Up with the AWS Command Line Interface](#) in the *AWS Command Line Interface User Guide*.

If you have experience with Windows PowerShell, you might prefer to use AWS Tools for Windows PowerShell. For more information, see [Setting up the AWS Tools for Windows PowerShell](#) in the *AWS Tools for Windows PowerShell User Guide*.

Download an AWS SDK

If you're using a programming language that AWS provides an SDK for, we recommend that you use an SDK instead of the AWS Cloud Map API. Using an SDK has several benefits. SDKs make authentication simpler, integrate easily with your development environment, and provide access to AWS Cloud Map commands. For more information, see [Tools for Amazon Web Services](#).

Using AWS Cloud Map

AWS Cloud Map is a managed solution that you can use to map logical names to the resources for an application. It also helps your applications discover resources using one of the AWS SDKs, RESTful API calls, or DNS queries. AWS Cloud Map serves only healthy resources, which can be Amazon DynamoDB (DynamoDB) tables, Amazon Simple Queue Service (Amazon SQS) queues, or any higher-level application services that are built using Amazon Elastic Compute Cloud (Amazon EC2) instances or Amazon Elastic Container Service (Amazon ECS) tasks.

Topics

- [Overview of How to Use AWS Cloud Map \(p. 7\)](#)
- [Configuring AWS Cloud Map \(p. 9\)](#)

Overview of How to Use AWS Cloud Map

The following is an overview of how you can use AWS Cloud Map:

1. Create a namespace, which is a logical grouping of services. When you create a namespace, you specify the name that you want your applications to use to discover instances. You also specify how you want to discover service instances that you register with AWS Cloud Map: using API calls or using DNS queries.

For more information, see the following topics:

- [Creating Namespaces \(p. 10\)](#)
- [CreatePublicDnsNamespace](#), [CreatePrivateDnsNamespace](#), and [CreateHttpNamespace](#) in the *AWS Cloud Map API Reference*

If you create a public or private DNS namespace, AWS Cloud Map automatically creates an Amazon Route 53 public or private hosted zone that has the same name as the namespace. Even with public and private DNS namespaces, you can still discover instances using AWS Cloud Map [DiscoverInstances](#) requests.

For a list of the endpoints that you can submit AWS Cloud Map API requests to, see [AWS Cloud Map](#) in the "AWS Regions and Endpoints" chapter in the *Amazon Web Services General Reference*.

2. If you created a public DNS namespace, perform the following steps to change the name servers for the domain registration to the name servers for the Route 53 hosted zone that AWS Cloud Map created when you created the namespace:
 - a. If you already registered a domain that has the same name as the public DNS namespace, skip to step 2b.
 - If you haven't registered a domain that has the same name as the namespace, register a domain. If you want to use Route 53 for domain registration, see [Registering a New Domain](#) in the *Amazon Route 53 Developer Guide*. Then skip to step 3.
 - b. Use the `OperationId` that was returned when you created the namespace to get the namespace ID. For more information, see [GetOperation](#).

Note

If you're using a programmatic method to perform these steps, you'll also use the namespace ID later in the process to create a service.

- c. Use the namespace ID that you got in step 2b to get the ID of the Route 53 hosted zone that AWS Cloud Map created. For more information, see [GetNamespace](#) in the *AWS Cloud Map API Reference*.
 - d. Using the hosted zone ID that you got in step 2c, get the names of the name servers that Route 53 assigned to your hosted zone. For more information, see [Getting the Name Servers for a Public Hosted Zone](#).
 - e. Change the name servers that are assigned to the domain. If the domain is registered with Route 53, see [Adding or Changing Name Servers and Glue Records for a Domain](#) for more information.
3. Create a service, which contains the service instances that identify how to contact the resources for an application, such as a web server, a DynamoDB table, or an Amazon S3 bucket.

If you created a public or private DNS namespace in step 1, the name that you specify for the service becomes part of the names of records in the Route 53 public or private hosted zone that AWS Cloud Map created automatically in step 1. When you register an instance in the next step, AWS Cloud Map creates records in the hosted zone. The record names are a combination of the name of the service (such as `backend`) and the name of the namespace (such as `example.com`): `backend.example.com`.

When you create a service, you can also choose whether you want to check the health of the resources that service instances point to:

- If you choose no health checking, AWS Cloud Map or Route 53 return service instances regardless of the health of the corresponding resources.
- If you choose Route 53 health checking (only available for public DNS namespaces), AWS Cloud Map automatically creates a Route 53 health check and associates it with the corresponding Route 53 record. Route 53 responds to DNS queries only with records for healthy resources.
- If you choose custom health checking, you use a third-party application to determine the health of your resources. Based on the results of the third-party health checks, you send [UpdateInstanceCustomHealthStatus](#) requests to AWS Cloud Map to update the status of the service instances.

If you configure health checking, either AWS Cloud Map or Route 53 returns only service instances for healthy resources in response to [DiscoverInstances](#) requests or DNS queries.

For more information, see the following topics:

- [Creating Services \(p. 14\)](#)
- [CreateService](#) in the *AWS Cloud Map API Reference*

4. Register one or more service instances. Each service instance contains information about how your application can contact one resource for an application.

For more information, see the following topics:

- [Registering Instances \(p. 20\)](#)
- [RegisterInstance](#) in the *AWS Cloud Map API Reference*

5. Write your application to discover instances using either the AWS Cloud Map [DiscoverInstances](#) API action or using DNS queries:

- If your application uses [DiscoverInstances](#), AWS Cloud Map returns information about the available instances that meet the specified criteria.
- If your application uses DNS queries, Route 53 returns one or more records.

If you specified settings for a health check when you created the service, AWS Cloud Map or Route 53 returns values only for healthy instances.

6. When you want to stop using a resource, deregister the corresponding service instance. AWS Cloud Map automatically deletes the associated Route 53 record and health check, if any.

For more information, see the following topics:

- [Deregistering Service Instances \(p. 24\)](#)
- [DeregisterInstance](#) in the *AWS Cloud Map API Reference*

7. If you don't need a service and namespace any longer, you can delete them. Note the following:
 - Before you can delete a service, you must deregister all instances that were registered using the service.
 - Before you can delete a namespace, you must delete all services that were created in the namespace.

For more information, see the following topics:

- [Deleting Services \(p. 20\)](#)
- [Deleting Namespaces \(p. 13\)](#)
- [DeleteService](#) in the *AWS Cloud Map API Reference*
- [DeleteNamespace](#) in the *AWS Cloud Map API Reference*

Configuring AWS Cloud Map

The following sections explain how to use the AWS Cloud Map console and AWS CLI to create, view, and delete namespaces and services, and register and deregister instances.

In a production environment, you'll probably perform most AWS Cloud Map actions programmatically. For more information about programmatic access to AWS Cloud Map, see the following pages for documentation and downloads:

- [Setting Up AWS Cloud Map \(p. 3\)](#)
- [Tools for Amazon Web Services](#) lists SDKs, command line tools, and other developer resources.
- [AWS Cloud Map API Reference](#) provides information about using the AWS Cloud Map API when you're using a programming language that AWS doesn't provide an SDK for.

Topics

- [Working with Namespaces \(p. 9\)](#)
- [Working with Services \(p. 13\)](#)
- [Working with Service Instances \(p. 20\)](#)
- [AWS Cloud Map Features That Aren't Available on the AWS Cloud Map Console \(p. 25\)](#)

Working with Namespaces

A namespace is a way to group services for an application. When you create a namespace, you specify how you want to discover service instances that you register with AWS Cloud Map: using API calls or using DNS queries. You also specify the name that you want your application to use to discover instances.

Topics

- [Creating Namespaces \(p. 10\)](#)
- [Values That You Specify When You Create Namespaces \(p. 10\)](#)
- [Viewing a List of Namespaces \(p. 12\)](#)
- [Deleting Namespaces \(p. 13\)](#)

Creating Namespaces

To create a namespace, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. Choose **Create namespace**.
3. On the **Create namespace** page, enter the applicable values. For more information, see [Values That You Specify When You Create Namespaces \(p. 10\)](#).
4. Choose **Create namespace**.

AWS CLI

- Create a namespace with the command for the instance discovery type you would prefer (replace the *red* values with your own):
 - Create an HTTP namespace using [create-http-namespace](#). Service instances registered using an HTTP namespace can be discovered using a `DiscoverInstances` request, but they can't be discovered using DNS.

```
aws servicediscovery create-http-namespace --name name-of-namespace
```

- Create a private namespace based on DNS and only visible inside a specified Amazon VPC using [create-private-dns-namespace](#). You can discover instances that were registered with a private DNS namespace by using either a `DiscoverInstances` request or using DNS.

```
aws servicediscovery create-private-dns-namespace --name name-of-namespace --  
vpc vpc-xxxxxxxx
```

- Create a public namespace based on DNS that is visible on the internet using [create-public-dns-namespace](#). You can discover instances that were registered with a public DNS namespace by using either a `DiscoverInstances` request or using DNS.

```
aws servicediscovery create-public-dns-namespace --name name-of-namespace
```

Values That You Specify When You Create Namespaces

When you create an AWS Cloud Map namespace, you specify the following values.

Note

After you create a namespace, you can change tags. However, you can't change any other values.

Values

- [Namespace name](#)

- [Namespace description](#)
- [Instance discovery](#)
- [Tags](#)
- [VPC](#)

Namespace name

The name that you specify for a namespace depends on how you want your application to discover instances. The method of how instances are discovered is determined by the option that you choose for **Instance discovery**. The options appear later on the current page in the console. They are as follows:

API calls

If you choose this option, your application discovers service instances by specifying the namespace name and service name in a [DiscoverInstances](#) request. For more information, see [DiscoverInstances](#) in the *AWS Cloud Map API Reference*.

You can specify a name that's up to 1,024 characters in length. A name can contain both uppercase and lowercase letters, numbers, underscores (`_`), and hyphens (`-`).

API calls and DNS queries in VPCs

Enter the domain name that you want your applications in a VPC to use when they discover instances by submitting DNS queries. AWS Cloud Map automatically creates an Amazon Route 53 private hosted zone that has this name. When you register service instances, AWS Cloud Map creates DNS records in the hosted zone that have names in the following format:

service-name.namespace-name

If you choose this option, your application can also discover instances by specifying the namespace name and service name in a [DiscoverInstances](#) request. For more information, see [DiscoverInstances](#) in the *AWS Cloud Map API Reference*.

You can specify an internationalized domain name (IDN) if you convert the name to Punycode first. For information about online converters, perform an internet search on "punycode converter".

You can also convert an internationalized domain name to Punycode when you create namespaces programmatically. For example, if you're using Java, you can convert a Unicode value to Punycode by using the `toASCII` method of the `java.net.IDN` library.

API calls and public DNS queries

Enter the domain name that you want your applications to use when they discover instances by submitting public DNS queries. This must be a domain name that you have registered. When you create the namespace, AWS Cloud Map automatically creates an Amazon Route 53 public hosted zone that has the same name. When you register service instances, AWS Cloud Map creates DNS records in the hosted zone that have names in the following format:

service-name.namespace-name

If you choose this option, your application can also discover instances by specifying the namespace name and service name in a [DiscoverInstances](#) request. For more information, see [DiscoverInstances](#) in the *AWS Cloud Map API Reference*.

You can specify an internationalized domain name (IDN) if you convert the name to Punycode first. For information about online converters, perform an internet search on "punycode converter".

You can also convert an internationalized domain name to Punycode when you create namespaces programmatically. For example, if you're using Java, you can convert a Unicode value to Punycode by using the `toASCII` method of the `java.net.IDN` library.

Namespace description

Enter a description for the namespace. The value that you enter here appears on the **Namespaces** page and on the detail page for each namespace.

Instance discovery

Choose how you want your application to discover registered instances:

API calls

Choose this option if you want your application to use only API calls to discover registered instances.

API calls and DNS queries in VPCs

Choose this option if you want your application to be able to discover instances using either API calls or using DNS queries in a VPC. You aren't required to use both methods.

API calls and public DNS queries

Choose this option if you want your application to be able to discover instances using either API calls or using public DNS queries. You aren't required to use both methods.

SOA TTL

For **API calls and DNS queries in VPCs** or **API calls and public DNS queries**, the time to live (TTL) value for the start of authority (SOA) DNS record of the Route 53 hosted zone created with your namespace. The value determines how long DNS resolvers cache information for this record before the resolvers forward another DNS query to Amazon Route 53 to get updated settings. A smaller value will also reduce the time a missing entry will be cached (negative caching) at the expense of additional queries for that namespace.

Tags

You can specify one or more tags to add to your namespace. A tag is an optional label that you can assign to an AWS resource. Each tag consists of a key and a value. For example, you can define a tag with Key = Environment and Value = Production. Tags enable you to categorize your AWS resources so you can more easily manage them.

You can update or remove tags on your namespaces after they have been created. For more information, see [Tagging your AWS Cloud Map resources \(p. 42\)](#).

VPC

When you choose **API calls and DNS queries in VPCs** for the value of **Instance discovery**, AWS Cloud Map creates an Amazon Route 53 private hosted zone that has the same name. AWS Cloud Map associates the VPC that you choose in the **VPC** list with that private hosted zone.

Route 53 Resolver resolves DNS queries that originate in the VPC using records in the private hosted zone. If the private hosted zone doesn't include a record that matches the domain name in a DNS query, Route 53 responds to the query with `NXDOMAIN` (non-existent domain).

You can associate additional VPCs with the private hosted zone. For more information, see [AssociateVPCWithHostedZone](#) in the *Amazon Route 53 API Reference*.

Viewing a List of Namespaces

To view a list of namespaces, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.

AWS CLI

- List namespaces with the `list-namespaces` command.

```
aws servicediscovery list-namespaces
```

Deleting Namespaces

When you delete a namespace, you can no longer use it to register or discover service instances. Note the following:

- Before you can delete a namespace, you must delete all the services that were created in the namespace. For more information, see [Deleting Services \(p. 20\)](#).
- Before you can delete a service, you must deregister all the service instances that were registered using the service. For more information, see [Deregistering Service Instances \(p. 24\)](#).
- When you create a namespace, if you specify that you want to discover service instances using either public DNS queries or DNS queries in VPCs, AWS Cloud Map creates an Amazon Route 53 public or private hosted zone. When you delete the namespace, AWS Cloud Map deletes the corresponding hosted zone.

To delete a namespace, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. Choose the option for the namespace that you want to delete.
4. Choose **Delete**.
5. Confirm that you want to delete the service.

AWS CLI

- Delete a namespace with the `delete-namespace` command (replace the `red` value with your own). If the namespace still contains one or more services, the request fails.

```
aws servicediscovery delete-namespace --id srv-xxxxxxx
```

Working with Services

A service is a template for registering service instances, which allow you to locate the resources for an application using DNS queries or the AWS Cloud Map [DiscoverInstances](#) API action, depending on how you configured the namespace.

Topics

- [Creating Services \(p. 14\)](#)
- [Values That You Specify When You Create Services \(p. 15\)](#)
- [Viewing a List of Services That You Created in a Namespace \(p. 19\)](#)
- [Deleting Services \(p. 20\)](#)

Creating Services

To create a service, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. On the **Namespaces** page, choose the namespace that you want to add the service to.
4. On the **Namespace: *namespace-name*** page, choose **Create service**.
5. On the **Create service** page, enter the applicable values. For more information, see [Values That You Specify When You Create Services \(p. 15\)](#).
6. Choose **Create service**.

AWS CLI

- Create a service with the `create-service` command (replace the `red` value with your own).

```
aws servicediscovery create-service \  
  --name service-name \  
  --namespace-id ns-xxxxxxxxxx \  
  --dns-config "NamespaceId=ns-  
xxxxxxxxxx,RoutingPolicy=MULTIVALUE,DnsRecords=[{Type=A,TTL=60}]"
```

Output:

```
{  
  "Service": {  
    "Id": "srv-xxxxxxxxxx",  
    "Arn": "arn:aws:servicediscovery:us-west-2:80364222207:service/srv-  
xxxxxxxxxx",  
    "Name": "service-name",  
    "NamespaceId": "ns-xxxxxxxxxx",  
    "DnsConfig": {  
      "NamespaceId": "ns-xxxxxxxxxx",  
      "RoutingPolicy": "MULTIVALUE",  
      "DnsRecords": [  
        {  
          "Type": "A",  
          "TTL": 60  
        }  
      ]  
    },  
    "CreateDate": 1587081768.334,  
    "CreatorRequestId": "567c1193-6b00-4308-bd57-ad38a8822d25"  
  }  
}
```

Note

For services that are accessible by DNS queries, you cannot create multiple services with names that differ only by case (such as EXAMPLE and example). Otherwise, these services will have the same DNS name. If you use a namespace that's only accessible by API calls, then you can create services that with names that differ only by case.

Values That You Specify When You Create Services

When you create an AWS Cloud Map service, you specify the following values.

Note

You can only change tags in a service after you create it.

Values

- [Service name](#)
- [Service description](#)
- [Service discovery configuration](#)
- [Routing policy](#)
- [Record type](#)
- [TTL](#)
- [Health check options](#)
- [Failure threshold](#)
- [Health check protocol](#)
- [Health check path](#)
- [Tags](#)

Service name

Enter a name that describes the instances that you register when using this service. The value is used to discover AWS Cloud Map service instances either in API calls or in DNS queries. This depends on the instance discovery method that you chose when you created the namespace. You can use one of the following methods:

- **API calls** – When your application calls [DiscoverInstances](#), the API call includes the namespace and service names.
- **API calls and DNS queries in VPCs or API calls and public DNS queries** – When you register service instances and create the namespace, AWS Cloud Map creates an Amazon Route 53 private or public hosted zone. It also create DNS records in that hosted zone. The names of the records are in the following format:

service-name.namespace-name

When your application submits a DNS query to discover service instances, the query is for a record that includes the name of the service in the record name.

Note

When creating a service in a namespace that supports DNS queries, you can choose to have the service instances for that service discoverable only with calls to the [DiscoverInstances](#) API operation and not DNS queries. See [Service discovery configuration](#).

If you want AWS Cloud Map to create an **SRV** record when you register an instance and you're using a system that requires a specific **SRV** format (such as [HAProxy](#)), specify the following for **Service name**:

- Start the name with an underscore (`_`), for example `_exampleservice`.
- End the name with `._protocol`, for example `._tcp`.

When you register an instance, AWS Cloud Map creates an **SRV** record and assigns a name by concatenating the service name and the namespace name, for example:

`_exampleservice._tcp.example.com`

Note

For services that are discoverable by DNS queries, you can't create multiple services with names that differ only by case (such as EXAMPLE and example). Otherwise, these services have the same DNS name and can't be distinguished.

Service description

Enter a description for the service. The value that you enter here appears on the **Services** page and on the detail page for each service.

Service discovery configuration

If the namespace supports DNS queries, AWS Cloud Map supports the following service discovery options:

API and DNS

AWS Cloud Map will create **SRV** records when you register an instance for the service. Service instances can also be discovered using the [DiscoverInstances](#) API operation.

API only

AWS Cloud Map will not create **SRV** records for instance for the service. Service instances can be discovered only using the [DiscoverInstances](#) API operation.

Routing policy (public and private DNS namespaces only)

If you're using a public or private DNS namespace to create the service, choose the Amazon Route 53 routing policy for the DNS records that AWS Cloud Map creates when you register instances. (Public DNS namespaces have a value of **API calls and public DNS queries** for **Instance discovery**, and private DNS namespaces have a value of **API calls and DNS queries in VPCs**.)

Note

You can't use the console to configure AWS Cloud Map to create a Route 53 alias record when you register an instance. If you want AWS Cloud Map to create alias records for Elastic Load Balancing load balancer when you register instances programmatically, choose **Weighted routing** for **Routing policy**.

AWS Cloud Map supports the following Route 53 routing policies:

Weighted routing

Route 53 returns the applicable value from one randomly selected instance from among the instances that you registered using the same service. All records have the same weight, so you can't route more or less traffic to any instances.

For example, suppose the service includes configurations for one **A** record and a health check, and you use the service to register 10 instances. Route 53 responds to DNS queries with the IP address for one randomly selected instance from among the healthy instances. If no instances are healthy, Route 53 responds to DNS queries as if all the instances were healthy.

If you don't define a health check for the service, Route 53 assumes that all instances are healthy and returns the applicable value for one randomly selected instance.

For more information, see [Weighted Routing](#) in the *Amazon Route 53 Developer Guide*.

Multivalue answer routing

If you define a health check for the service and the result of the health check is healthy, Route 53 returns the applicable value for up to eight instances.

For example, suppose that the service includes configurations for one **A** record and a health check. You use the service to register 10 instances. Route 53 responds to DNS queries with IP addresses for only a maximum of eight healthy instances. If fewer than eight instances are healthy, Route 53 responds to every DNS query with the IP addresses for all the healthy instances.

If you don't define a health check for the service, Route 53 assumes that all instances are healthy and returns the values for up to eight instances.

For more information, see [Multivalued Answer Routing](#) in the *Amazon Route 53 Developer Guide*.

Record type (public and private DNS namespaces only)

If you're using a public or private DNS namespace to create the service, choose the DNS record type for the records that AWS Cloud Map creates when you register instances. Amazon Route 53 returns the applicable value in response to DNS queries for registered instances.

The following record types are supported:

A

When you register an instance, you specify the IP address of the resource in IPv4 format, such as **192.0.2.44**.

AAAA

When you register an instance, you specify the IP address of the resource in IPv6 format, such as **2001:0db8:85a3:0000:0000:abcd:0001:2345**.

CNAME

When you register an instance, you specify the domain name of the resource (such as `www.example.com`). Note the following:

- If you want to choose **CNAME**, you must choose **Weighted routing** for **Routing policy**.
- If you choose **CNAME**, you can't choose **Route 53 health check** for **Health check options**.

SRV

The value for an **SRV** record uses the following values:

```
priority weight port service-hostname
```

Note the following about the values:

- The values of `priority` and `weight` are both set to 1 and can't be changed.
- For `port`, AWS Cloud Map uses the value that you specify for **Port** when you register an instance.
- The value of `service-hostname` is a concatenation of the following values:
 - The value that you specify for **Service instance ID** when you register an instance
 - The name of the service
 - The name of the namespace

For example, suppose you specify **test** for **Service instance ID** when you register an instance. The name of the service is **backend** and the name of the namespace is **example.com**. AWS Cloud Map assigns the following value to the `service-hostname` attribute in the **SRV** record:

```
test.backend.example.com
```

If you specify settings for an **SRV** record, note the following:

- If you specify values for **IPv4 address**, **IPv6 address**, or both, AWS Cloud Map automatically creates **A** and/or **AAAA** records that have the same name as the value of `service-hostname` in the **SRV** record.

- If you're using a system that requires a specific **SRV** format, such as [HAProxy](#), see [service name \(p. 15\)](#) for information about how to specify the correct name format.

You can specify record types in the following combinations:

- **A**
- **AAAA**
- **A** and **AAAA**
- **CNAME**
- **SRV**

If you specify **A** and **AAAA** record types, you can specify an IPv4 IP address, an IPv6 IP address, or both when you register an instance.

TTL (public and private DNS namespaces only)

If you're using a public or private DNS namespace to create the service, enter a value for **TTL**, or time to live. The value of **TTL** determines how long DNS resolvers cache information for this record before the resolvers forward another DNS query to Amazon Route 53 to get updated settings.

Health check options

No health check

If you don't configure a health check, traffic is routed to service instances regardless of whether they're healthy.

Route 53 health check (not supported for private DNS namespaces)

If you specify settings for an Amazon Route 53 health check, AWS Cloud Map creates a Route 53 health check whenever you register an instance and deletes the health check when you deregister the instance.

For public DNS namespaces, AWS Cloud Map associates the health check with the Route 53 record that AWS Cloud Map creates when you register an instance.

For namespaces that you use API calls to discover instances for, AWS Cloud Map creates a Route 53 health check. However, there's no DNS record for AWS Cloud Map to associate the health check with. To determine whether a health check is healthy, you can configure monitoring using either the Route 53 console or using Amazon CloudWatch. For more information about using the Route 53 console, see [Get Notified When a Health Check Fails](#) in the *Amazon Route 53 Developer Guide*. For more information about using CloudWatch, see [PutMetricAlarm](#) in the *Amazon CloudWatch API Reference*.

For information about the charges for Route 53 health checks, see [Route 53 Pricing](#).

Custom health check

If you configure AWS Cloud Map to use a custom health check when you register an instance, you must use a third-party health checker to evaluate the health of your resources. Custom health checks are useful in the following circumstances:

- You can't use a Route 53 health check because the resource isn't available over the internet. For example, suppose that you have an instance that's located in an Amazon VPC. You can use a custom health check for this instance. However, for the health check to work, your health checker must also be in the same VPC as your instance.
- You want to use a third-party health checker regardless of where your resources are.

Failure threshold (Route 53 health check only)

The number of consecutive Route 53 health checks that a resource must pass or fail for Amazon Route 53 to change the current status of the resource from healthy to unhealthy or the opposite

situation. For more information, see [How Amazon Route 53 Determines Whether a Health Check Is Healthy](#) *Amazon Route 53 Developer Guide*.

Health check protocol (Route 53 health check only)

The method that you want Amazon Route 53 to use to check the health of your resource:

HTTP

Route 53 tries to establish a TCP connection. If successful, Route 53 submits an HTTP request and waits for an HTTP status code of a 2xx or 3xx format.

HTTPS

Route 53 tries to establish a TCP connection. If successful, Route 53 submits an HTTPS request and waits for an HTTP status code of a 2xx or 3xx format.

Important

If you choose HTTPS, the resource must support TLS v1.0 or later.

If you choose HTTPS for the value of **Health check protocol**, additional charges apply. For more information, see [Route 53 Pricing](#).

TCP

Route 53 tries to establish a TCP connection.

For more information, see [How Amazon Route 53 Determines Whether a Health Check Is Healthy](#).

Health check path (Route 53 HTTP and HTTPS health checks only)

The path that you want Amazon Route 53 to request when performing health checks. The path can be any value such as the file `/docs/route53-health-check.html`. When the resource is healthy, the returned value is an HTTP status code of a 2xx or 3xx format. You can also include query string parameters, for example, `/welcome.html?language=jp&login=y`. The AWS Cloud Map console automatically adds a leading slash (/) character.

Tags

You can specify one or more tags to add to your service. A tag is an optional label that you can assign to an AWS resource. Each tag consists of a key and a value. For example, you can define a tag with Key = Environment and Value = Production. Using tags to categorize AWS resources can make managing those resources easier.

After your tags are created, you can always update or remove tags on your namespaces. For more information, see [Tagging your AWS Cloud Map resources](#) (p. 42).

Viewing a List of Services That You Created in a Namespace

To view a list of the services that you created in a namespace, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. Choose the name of the namespace that contains the services that you want to list.

AWS CLI

- List services with the `list-services` command.

```
aws servicediscovery list-services
```

Deleting Services

Before you can delete a service, you must deregister all service instances that were registered using the service. For more information, see [Deregistering Service Instances \(p. 24\)](#).

To delete a service, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. Choose the option for the namespace that contains the service that you want to delete.
4. On the **Namespace: namespace-name** page, choose the option for the service that you want to delete.
5. Choose **Delete**.
6. Confirm that you want to delete the service.

AWS CLI

- Delete a service with the `delete-service` command (replace the `red` value with your own).

```
aws servicediscovery delete-service --id srv-xxxxxx
```

Working with Service Instances

A service instance contains information about how to locate a resource, such as a web server, for an application. After you register instances, you locate them by using DNS queries or the AWS Cloud Map [DiscoverInstances](#) API action.

Topics

- [Registering Instances \(p. 20\)](#)
- [Values That You Specify When You Register or Update Instances \(p. 21\)](#)
- [Updating Instances \(p. 23\)](#)
- [Viewing a List of Service Instances \(p. 24\)](#)
- [Deregistering Service Instances \(p. 24\)](#)

Registering Instances

To register a service instance, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.

3. On the **Namespaces** page, choose the namespace that contains the service that you want to use as a template for registering a service instance.
4. On the **Namespace: *namespace-name*** page, choose the service that you want to use.
5. On the **Service: *service-name*** page, choose **Register service instance**.
6. On the **Register service instance** page, enter the applicable values. For more information, see [Values That You Specify When You Register or Update Instances \(p. 21\)](#).
7. Choose **Register service instance**.

AWS CLI

- When you submit a `RegisterInstance` request:
 - For each DNS record that you define in the service that's specified by `ServiceId`, a record is created or updated in the hosted zone that's associated with the corresponding namespace.
 - If the service includes `HealthCheckConfig`, a health check is created based on the settings in the health check configuration.
 - Any health checks are associated with each of the new or updated records.

Register a service instance with the `register-instance` command (replace the *red* values with your own).

```
aws servicediscovery register-instance \  
  --service-id srv-xxxxxxxx \  
  --instance-id myservice-xx \  
  --attributes=AWS_INSTANCE_IPV4=172.2.1.3,AWS_INSTANCE_PORT=808
```

Values That You Specify When You Register or Update Instances

When you register a service instance, you specify the following values.

Values

- [Instance type](#)
- [Service instance ID](#)
- [IPv4 address](#)
- [IPv6 address](#)
- [Port](#)
- [CNAME](#)
- [Custom attributes](#)

Instance type

Each of the following instance types is available for selected configurations only.

IP address

Choose this option when the resource that's associated with the service instance is accessible using an IP address.

You can choose this option for all three types of namespaces: HTTP, public DNS, and private DNS. For public and private DNS namespaces, it's available only when the service doesn't include a CNAME record.

Identifying information for another resource

Choose this option when the resource that's associated with the service instance is accessible using values other than an IP address or a domain name (CNAME). Specify the other values in **Custom attributes**.

You can choose this option only for HTTP namespaces. In addition, it's available only when the service that you're using to register the instance either doesn't include a health check or includes a custom health check.

Identifying information for another resource based on CNAME

Choose this option when the resource that's associated with the service instance is accessible using a domain name (CNAME).

You can choose this option only for public and private DNS namespaces and only when the service includes a **CNAME** record.

Service instance ID

An identifier that you want to associate with the instance. Note the following:

- To register a new instance, you must specify a value that's unique among instances that you register by using the same service.
- If the service that's specified by **Service instance ID** includes settings for an **SRV** record, the value of **Service instance ID** is automatically included as part of the value for the **SRV** record. For more information, see **Record type** in the section [Values That You Specify When You Create Services \(p. 15\)](#).
- You can update an existing instance programmatically. Call [RegisterInstance](#), specify the value of **Service instance ID** and **Service ID**, and specify the new settings for the service instance. If AWS Cloud Map created a health check when you registered the instance originally, AWS Cloud Map deletes the old health check and creates a new one.

Note

The health check isn't deleted immediately, so it will still appear for a while if you submit an Amazon Route 53 `ListHealthChecks` request, for example.

IPv4 address

The IPv4 IP address, if any, where your applications can access the resource that's associated with this service instance.

IPv6 address

The IPv6 IP address, if any, where your applications can access the resource that's associated with this service instance.

Port

The port, if any, that your applications must include to access the resource that's associated with this service instance. **Port** is required when the service includes an **SRV** record or an Amazon Route 53 health check.

CNAME

The domain or subdomain name that your applications can use to access the resource that's associated with this service instance.

Custom attributes

If a resource is accessible using some method other than an IP address or a domain or subdomain name, specify one or more custom attributes that your application can use to access the resource. You can also use custom attributes for a variety of other purposes.

You can add up to 30 custom attributes. Note the following:

- If you don't specify values for **IPv4 address**, **IPv6 address**, **Port**, or **CNAME**, you must specify at least one key-value pair for **Custom attributes**.
- You must specify both **Key** and **Value**.
- **Key** can be up to 255 characters long and can include the characters a-z, A-Z, 0-9 and other printable ASCII characters between 33 and 126 (Decimal). Spaces, tabs, and other whitespace characters are not allowed.
- **Value** can be up to 1,024 characters long and can include the characters a-z, A-Z, 0-9, other printable ASCII characters between 33 and 126 (Decimal), space, and tab.

Updating Instances

You can update service instances in two ways, depending on which values you want to update:

- **Update any values:** If you want to update any of the values that you specified for a service instance when you registered it, including custom attributes, you reregister the service instance and respecify all values. See [To update a service instance \(p. 23\)](#).
- **Update only custom attributes:** If you want to update only the custom attributes for a service instance, you don't need to reregister the instance. You can update only those values. See [To update only custom attributes for a service instance \(p. 23\)](#).

To update a service instance

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. On the **Namespaces** page, choose the namespace that contains the service that you originally used to register the service instance.
4. On the **Namespace: namespace-name** page, choose the service that you used to register the service instance.
5. On the **Service: service-name** page, copy the ID of the service instance that you want to update.
6. Choose **Register service instance**.
7. On the **Register service instance** page, paste the ID that you copied in step 5 into **Service instance ID**.
8. Enter all the other values that you want to apply to the service instance. The previous values for the service instance are not retained. For more information, see [Values That You Specify When You Register or Update Instances \(p. 21\)](#).
9. Choose **Register service instance**.

To update only custom attributes for a service instance

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. On the **Namespaces** page, choose the namespace that contains the service that you originally used to register the service instance.
4. On the **Namespace: namespace-name** page, choose the service that you used to register the service instance.
5. On the **Service: service-name** page, choose the name of the service instance that you want to update.
6. In the **Custom attributes** section, choose **Edit**.

7. On the **Edit service instance: *instance-name*** page, add, remove, or update custom attributes. You can update both keys and values for existing attributes.
8. Choose **Update service instance**.

Viewing a List of Service Instances

To view a list of the service instances that you registered using a service, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. Choose the name of the namespace that contains the service for which you want to list service instances.
4. Choose the name of the service that you used to create the service instances.

AWS CLI

- List service instances with the `list-instances` command (replace the *red* value with your own).

```
aws servicediscovery list-instances --service-id srv-xxxxxxxx
```

Deregistering Service Instances

Before you can delete a service, you must deregister all service instances that were registered using the service.

To deregister a service instance, perform the following procedure.

AWS Management Console

1. Sign in to the AWS Management Console and open the AWS Cloud Map console at <https://console.aws.amazon.com/cloudmap/>.
2. In the navigation pane, choose **Namespaces**.
3. Choose the option for the namespace that contains the service instance that you want to deregister.
4. On the **Namespace: *namespace-name*** page, choose the option for the service you used to register the service instance.
5. On the **Service: *service-name*** page, choose the option for the service instance that you want to deregister.
6. Choose **Deregister**.
7. Confirm that you want to deregister the service instance.

AWS CLI

- Deregister a service instance with the `deregister-instance` command (replace the *red* values with your own). This command deletes the Amazon Route 53 DNS records and any health checks that Cloud Map created for the specified instance.

```
aws servicediscovery deregister-instance \  
--service-id srv-xxxxxxxx \  
--instance-id myservice-53
```

AWS Cloud Map Features That Aren't Available on the AWS Cloud Map Console

The following AWS Cloud Map features aren't available on the AWS Cloud Map console. To use these features, you must use a programmatic method to access AWS Cloud Map:

Creating Route 53 alias records when you register service instances

When you register a service instance using the console, you can't create an alias record that routes traffic to an Elastic Load Balancing (ELB) load balancer. Note the following:

- When you create a service, you must specify `WEIGHTED` for `RoutingPolicy`. You can do this using the console. For more information, see [Creating Services \(p. 14\)](#).

For information about creating a service using the AWS Cloud Map API, see [CreateService](#) in the *AWS Cloud Map API Reference*.

- When you register an instance, you must include the `AWS_ALIAS_DNS_NAME` attribute. For more information, see [RegisterInstance](#) in the *AWS Cloud Map API Reference*.

Specifying the initial health status for custom health checks

If you register an instance using a service that includes a custom health check, you can't specify the initial status for the custom health check. By default, the initial status of a custom health check is **Healthy**. If you want the initial health status to be **Unhealthy**, register the instance programmatically and include the `AWS_INIT_HEALTH_STATUS` attribute. For more information, see [RegisterInstance](#) in the *AWS Cloud Map API Reference*.

Getting the status of an incomplete operation

If you close a browser window after you create a namespace but before creating the namespace has completed, the console doesn't provide a way to see the current status. You can get the status by using [ListOperations](#). For more information, see [ListOperations](#) in the *AWS Cloud Map API Reference*.

Security in AWS Cloud Map

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that's built to meet the requirements of the most security-sensitive organizations.

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

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

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

Topics

- [AWS Identity and Access Management in AWS Cloud Map \(p. 26\)](#)
- [Logging and Monitoring in AWS Cloud Map \(p. 38\)](#)
- [Compliance Validation for AWS Cloud Map \(p. 40\)](#)
- [Resilience in AWS Cloud Map \(p. 40\)](#)
- [Infrastructure Security in AWS Cloud Map \(p. 41\)](#)

AWS Identity and Access Management in AWS Cloud Map

To perform any action on AWS Cloud Map resources, such as registering a domain or updating a record, AWS Identity and Access Management (IAM) requires you to authenticate that you're an approved AWS user. If you're using the AWS Cloud Map console, you authenticate your identity by providing your AWS user name and a password. If you're accessing AWS Cloud Map programmatically, your application authenticates your identity for you by using access keys or by signing requests.

After you authenticate your identity, IAM controls your access to AWS by verifying that you have permissions to perform actions and to access resources. If you are an account administrator, you can use IAM to control the access of other users to the resources that are associated with your account.

This chapter explains how to use [IAM](#) and AWS Cloud Map to help secure your resources.

Topics

- [Authentication \(p. 27\)](#)
- [Access Control \(p. 27\)](#)

Authentication

You can access AWS as any of the following types of identities:

- **AWS account root user** – When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.
- **IAM user** – An [IAM user](#) is an identity within your AWS account that has specific custom permissions (for example, permissions to create an HTTP namespace in AWS Cloud Map). You can use an IAM user name and password to sign in to secure AWS webpages like the [AWS Management Console](#), [AWS Discussion Forums](#), or the [AWS Support Center](#).

In addition to a user name and password, you can also generate [access keys](#) for each user. You can use these keys when you access AWS services programmatically, either through [one of the several SDKs](#) or by using the [AWS Command Line Interface](#). The SDK and CLI tools use the access keys to cryptographically sign your request. If you don't use AWS tools, you must sign the request yourself. AWS Cloud Map supports *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see [Signature Version 4 Signing Process](#) in the *Amazon Web Services General Reference*.

- **IAM role** – An [IAM role](#) is an IAM identity that you can create in your account that has specific permissions. It is similar to an *IAM user*, but it isn't associated with a specific person. An IAM role enables you to obtain temporary access keys that can be used to access AWS services and resources. IAM roles with temporary credentials are useful in the following situations:
 - **Federated user access** – Instead of creating an IAM user, you can use existing user identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an [identity provider](#). For more information about federated users, see [Federated Users and Roles](#) in the *IAM User Guide*.
 - **AWS service access** – You can use an IAM role in your account to grant an AWS service permissions to access your account's resources. For example, you can create a role that allows Amazon Redshift to access an Amazon S3 bucket on your behalf and then load data from that bucket into an Amazon Redshift cluster. For more information, see [Creating a Role to Delegate Permissions to an AWS Service](#) in the *IAM User Guide*.
 - **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that's attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances](#) in the *IAM User Guide*.

Access Control

To create, update, delete, or list AWS Cloud Map resources, you need permissions to perform the action, and you need permission to access the corresponding resources. In addition, to perform the action programmatically, you need valid access keys.

The following sections describe how to manage permissions for AWS Cloud Map. We recommend that you read the overview first.

- [Overview of Managing Access Permissions to Your AWS Cloud Map Resources \(p. 28\)](#)

- [Using Identity-Based Policies \(IAM Policies\) for AWS Cloud Map \(p. 31\)](#)
- [AWS Cloud Map API Permissions: Actions, Resources, and Conditions Reference \(p. 34\)](#)

Overview of Managing Access Permissions to Your AWS Cloud Map Resources

Every AWS resource is owned by an AWS account, and permissions to create or access a resource are governed by permissions policies.

Note

An *account administrator* (or administrator user) is a user that has administrator privileges. For more information about administrators, see [IAM Best Practices](#) in the *IAM User Guide*.

When you grant permissions, you decide who gets the permissions, the resources they get permissions for, and the actions that they get permissions to perform.

Topics

- [ARNs for AWS Cloud Map Resources \(p. 28\)](#)
- [Understanding Resource Ownership \(p. 28\)](#)
- [Managing Access to Resources \(p. 28\)](#)
- [Specifying Policy Elements: Resources, Actions, Effects, and Principals \(p. 30\)](#)
- [Specifying Conditions in an IAM Policy \(p. 31\)](#)

ARNs for AWS Cloud Map Resources

You can grant or deny resource-level permissions for namespaces and services for selected operations. For more information, see [AWS Cloud Map API Permissions: Actions, Resources, and Conditions Reference \(p. 34\)](#).

Understanding Resource Ownership

An AWS account owns the resources that are created in the account, regardless of who created the resources. Specifically, the resource owner is the AWS account of the principal entity (that is, the root account, an IAM user, or an IAM role) that authenticates the resource creation request.

The following examples illustrate how this works:

- If you use the root account credentials of your AWS account to create an HTTP namespace, your AWS account is the owner of the resource.
- If you create an IAM user in your AWS account and grant permissions to create an HTTP namespace to that user, the user can create an HTTP namespace. However, your AWS account, to which the user belongs, owns the HTTP namespace resource.
- If you create an IAM role in your AWS account with permissions to create an HTTP namespace, anyone who can assume the role can create an HTTP namespace. Your AWS account, to which the role belongs, owns the HTTP namespace resource.

Managing Access to Resources

A *permissions policy* specifies who has access to what. This section explains the options for creating permissions policies for AWS Cloud Map. For general information about IAM policy syntax and descriptions, see the [IAM Policy Reference](#) in the *IAM User Guide*.

Policies attached to an IAM identity are referred to as *identity-based* policies (IAM policies), and policies attached to a resource are referred to as *resource-based* policies. AWS Cloud Map supports only identity-based policies (IAM policies).

Topics

- [Identity-Based Policies \(IAM Policies\) \(p. 29\)](#)
- [Resource-Based Policies \(p. 30\)](#)

Identity-Based Policies (IAM Policies)

You can attach policies to IAM identities. For example, you can do the following:

- **Attach a permissions policy to a user or a group in your account** – An account administrator can use a permissions policy that's associated with a particular user to grant permissions for that user to create AWS Cloud Map resources.
- **Attach a permissions policy to a role (grant cross-account permissions)** – You can grant permission to perform AWS Cloud Map actions to a user that was created by another AWS account. To do so, you attach a permissions policy to an IAM role, and then you allow the user in the other account to assume the role. The following example explains how this works for two AWS accounts, account A and account B:
 1. Account A administrator creates an IAM role and attaches to the role a permissions policy that grants permissions to create or access resources that are owned by account A.
 2. Account A administrator attaches a trust policy to the role. The trust policy identifies account B as the principal that can assume the role.
 3. Account B administrator can then delegate permissions to assume the role to users or groups in Account B. This allows users in account B to create or access resources in account A.

For more information about how to delegate permissions to users in another AWS account, see [Access Management](#) in the *IAM User Guide*.

The following example policy allows a user to perform the [CreatePublicDnsNamespace](#) action to create a public DNS namespace for any AWS account. The Amazon Route 53 permissions are required because when you create a public DNS namespace, AWS Cloud Map also creates a Route 53 hosted zone:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "servicediscovery:CreatePublicDnsNamespace",
        "route53:CreateHostedZone",
        "route53:GetHostedZone",
        "route53:ListHostedZonesByName"
      ],
      "Resource": "*"
    }
  ]
}
```

If you want the policy to instead apply to private DNS namespaces, you need to grant permissions to use the AWS Cloud Map [CreatePrivateDnsNamespace](#) action. In addition, you grant permission to use the same Route 53 actions as in the previous example because AWS Cloud Map creates a Route 53 private hosted zone. You also grant permission to use two Amazon EC2 actions, `DescribeVpcs` and `DescribeRegions`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "servicediscovery:CreatePrivateDnsNamespace",
        "route53:CreateHostedZone",
        "route53:GetHostedZone",
        "route53:ListHostedZonesByName"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeVpcs",
        "ec2:DescribeRegions"
      ],
      "Resource": "*"
    }
  ]
}
```

For more information about attaching policies to identities for AWS Cloud Map, see [Using Identity-Based Policies \(IAM Policies\) for AWS Cloud Map \(p. 31\)](#). For more information about users, groups, roles, and permissions, see [Identities \(Users, Groups, and Roles\)](#) in the *IAM User Guide*.

Resource-Based Policies

Other services, such as Amazon S3, also support attaching permissions policies to resources. For example, you can attach a policy to an S3 bucket to manage access permissions to that bucket. AWS Cloud Map doesn't support attaching policies to resources.

Specifying Policy Elements: Resources, Actions, Effects, and Principals

AWS Cloud Map includes API actions (see the [AWS Cloud Map API Reference](#)) that you can use on each AWS Cloud Map resource (see [ARNs for AWS Cloud Map Resources \(p. 28\)](#)). You can grant a user or a federated user permissions to perform any or all of these actions. Note that some API actions, such as creating a public DNS namespace, require permissions to perform more than one action.

The following are the basic policy elements:

- **Resource** – You use an Amazon Resource Name (ARN) to identify the resource that the policy applies to. For more information, see [ARNs for AWS Cloud Map Resources \(p. 28\)](#).
- **Action** – You use action keywords to identify resource actions that you want to allow or deny. For example, depending on the specified `Effect`, the `servicediscovery:CreateHttpNamespace` permission allows or denies a user the ability to perform the AWS Cloud Map [CreateHttpNamespace](#) action.
- **Effect** – You specify the effect, either allow or deny, when a user tries to perform the action on the specified resource. If you don't explicitly grant access to an action, access is implicitly denied. You can also explicitly deny access to a resource, which you might do to make sure that a user cannot access it, even if a different policy grants access.
- **Principal** – In identity-based policies (IAM policies), the user that the policy is attached to is the implicit principal. For resource-based policies, you specify the user, account, service, or other entity that you want to receive permissions (applies to resource-based policies only). AWS Cloud Map doesn't support resource-based policies.

For more information about IAM policy syntax and descriptions, see the [IAM Policy Reference](#) in the *IAM User Guide*.

For a list of the AWS Cloud Map API actions and the resources that they apply to, see [AWS Cloud Map API Permissions: Actions, Resources, and Conditions Reference](#) (p. 34).

Specifying Conditions in an IAM Policy

When you grant permissions, you can use the IAM policy language to specify when a policy should take effect. For example, you might want a policy to be applied only after a specified date, or you might want a policy to apply only to a specified namespace.

To express conditions, you use predefined condition keys. AWS Cloud Map defines its own set of condition keys and also supports using some global condition keys. For more information, see the following topics:

- For information about AWS Cloud Map condition keys, see [AWS Cloud Map API Permissions: Actions, Resources, and Conditions Reference](#) (p. 34).
- For information about AWS global condition keys, see [AWS Global Condition Context Keys](#) in the *IAM User Guide*.
- For information about specifying conditions in a policy language, [IAM JSON Policy Elements: Condition](#) in the *IAM User Guide*.

Using Identity-Based Policies (IAM Policies) for AWS Cloud Map

This topic provides examples of identity-based policies that demonstrate how an account administrator can attach permissions policies to IAM identities (users, groups, and roles) and thereby grant permissions to perform actions on AWS Cloud Map resources.

Important

We recommend that you first review the introductory topics that explain the basic concepts and options to manage access to your AWS Cloud Map resources. For more information, see [Overview of Managing Access Permissions to Your AWS Cloud Map Resources](#) (p. 28).

Topics

- [Permissions Required to Use the AWS Cloud Map Console](#) (p. 32)
- [AWS Managed \(Predefined\) Policies for AWS Cloud Map](#) (p. 33)
- [Customer Managed Policy Examples](#) (p. 33)

The following example shows a permissions policy that grants a user permission to register, deregister, and register service instances. The `Sid`, or statement ID, is optional:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowInstancePermissions",
      "Effect": "Allow",
      "Action": [
        "servicediscovery:RegisterInstance",
        "servicediscovery:DeregisterInstance",
        "servicediscovery:DiscoverInstances",
        "servicediscovery:Get*",
        "servicediscovery:List*",
        "route53:GetHostedZone",

```

```
        "route53:ListHostedZonesByName",
        "route53:ChangeResourceRecordSets",
        "route53:CreateHealthCheck",
        "route53:GetHealthCheck",
        "route53>DeleteHealthCheck",
        "route53:UpdateHealthCheck",
        "ec2:DescribeInstances"
    ],
    "Resource": "*"
}
]
```

The policy grants permissions to the actions that are required to register and manage service instances. The Route 53 permission is required if you're using public or private DNS namespaces because AWS Cloud Map creates, updates, and deletes Route 53 records and health checks when you register and deregister instances. The wildcard character (*) in `Resource` grants access to all AWS Cloud Map instances, and Route 53 records and health checks that are owned by the current AWS account.

For a list of actions and the ARN that you specify to grant or deny permission to use each action, see [AWS Cloud Map API Permissions: Actions, Resources, and Conditions Reference \(p. 34\)](#).

Permissions Required to Use the AWS Cloud Map Console

To grant full access to the AWS Cloud Map console, you grant the permissions in the following permissions policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "servicediscovery:*",
        "route53:GetHostedZone",
        "route53:ListHostedZonesByName",
        "route53:CreateHostedZone",
        "route53>DeleteHostedZone",
        "route53:ChangeResourceRecordSets",
        "route53:CreateHealthCheck",
        "route53:GetHealthCheck",
        "route53>DeleteHealthCheck",
        "route53:UpdateHealthCheck",
        "ec2:DescribeInstances",
        "ec2:DescribeVpcs",
        "ec2:DescribeRegions"
      ],
      "Resource": "*"
    }
  ]
}
```

Here's why the permissions are required:

servicediscovery:*

Lets you perform all AWS Cloud Map actions.

route53:CreateHostedZone, route53:GetHostedZone, route53:ListHostedZonesByName, route53>DeleteHostedZone

Lets AWS Cloud Map manage hosted zones when you create and delete public and private DNS namespaces.

route53:CreateHealthCheck, route53:GetHealthCheck, route53>DeleteHealthCheck, route53:UpdateHealthCheck

Lets AWS Cloud Map manage health checks when you include Amazon Route 53 health checks when you create a service.

ec2:DescribeVpcs and ec2:DescribeRegions

Let AWS Cloud Map manage private hosted zones.

AWS Managed (Predefined) Policies for AWS Cloud Map

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. These AWS managed policies grant necessary permissions for common use cases so that you can avoid having to investigate what permissions are needed. For more information, see [AWS Managed Policies](#) in the *IAM User Guide*. For AWS Cloud Map, IAM provides the following managed policies:

- **AWSCloudMapDiscoverInstanceAccess** – Grants access to the AWS Cloud Map [DiscoverInstances](#) API action
- **AWSCloudMapReadOnlyAccess** – Grants read-only access to all AWS Cloud Map actions
- **AWSCloudMapRegisterInstanceAccess** – Grants read-only access to namespaces and services, and grants permission to register and deregister service instances
- **AWSCloudMapFullAccess** – Provides full access to all AWS Cloud Map actions

Note

You can review these permissions policies by signing in to the IAM console and searching for specific policies there. You can also create your own custom IAM policies to allow permissions for AWS Cloud Map API actions. You can attach these custom policies to the IAM users or groups that require those permissions.

Customer Managed Policy Examples

You can create your own custom IAM policies to allow permissions for AWS Cloud Map actions. You can attach these custom policies to the IAM users or groups that require the specified permissions. These policies work when you are using the AWS Cloud Map API, the AWS SDKs, or the AWS CLI. The following examples show permissions for several common use cases. For the policy that grants a user full access to AWS Cloud Map, see [Permissions Required to Use the AWS Cloud Map Console](#) (p. 32).

Examples

- [Example 1: Allow Read Access to All AWS Cloud Map Resources](#) (p. 33)
- [Example 2: Allow Creation of All Types of Namespaces](#) (p. 34)

Example 1: Allow Read Access to All AWS Cloud Map Resources

The following permissions policy grants the user read-only access to all AWS Cloud Map resources:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "servicediscovery:Get*",
        "servicediscovery:List*"
      ]
    }
  ]
}
```

```
        "servicediscovery:DiscoverInstances"
      ],
      "Resource": "*"
    }
  ]
}
```

Example 2: Allow Creation of All Types of Namespaces

The following permissions policy allows users to create all types of namespaces:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "servicediscovery:CreateHttpNamespace",
        "servicediscovery:CreatePrivateDnsNamespace",
        "servicediscovery:CreatePublicDnsNamespace",
        "route53:CreateHostedZone",
        "route53:GetHostedZone",
        "route53:ListHostedZonesByName",
        "ec2:DescribeVpcs",
        "ec2:DescribeRegions"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS Cloud Map API Permissions: Actions, Resources, and Conditions Reference

When you set up [Access Control](#) (p. 27) and write a permissions policy that you can attach to an IAM identity (identity-based policies), you can use the following lists as a reference. The lists include each AWS Cloud Map API action, the actions that you must grant permissions access to, and the AWS resource that you must grant access to. You specify the actions in the `Action` field for the policy, and you specify the resource value in the `Resource` field for the policy.

You can use AWS Cloud Map–specific condition keys in your IAM policies for some operations. For more information, see [AWS Cloud Map Condition Keys Reference](#) (p. 37). You can also use AWS wide condition keys. For a complete list of AWS wide keys, see [Available Keys](#) in the *IAM User Guide*.

To specify an action, use the `servicediscovery` prefix followed by the API action name, for example, `servicediscovery:CreatePublicDnsNamespace` and `route53:CreateHostedZone`.

Topics

- [Required Permissions for AWS Cloud Map Actions](#) (p. 34)
- [AWS Cloud Map Condition Keys Reference](#) (p. 37)

Required Permissions for AWS Cloud Map Actions

[CreateHttpNamespace](#)

Required Permissions (API Action):

- `servicediscovery:CreateHttpNamespace`

Resources: *

CreatePrivateDnsNamespace

Required Permissions (API Action):

- `servicediscovery:CreatePrivateDnsNamespace`
- `route53:CreateHostedZone`
- `route53:GetHostedZone`
- `route53:ListHostedZonesByName`
- `ec2:DescribeVpcs`
- `ec2:DescribeRegions`

Resources: *

CreatePublicDnsNamespace

Required Permissions (API Action):

- `servicediscovery:CreatePublicDnsNamespace`
- `route53:CreateHostedZone`
- `route53:GetHostedZone`
- `route53:ListHostedZonesByName`

Resources: *

CreateService

Required Permissions (API Action): `servicediscovery:CreateService`

Resources: *

DeleteNamespace

Required Permissions (API Action):

- `servicediscovery>DeleteNamespace`
- `route53>DeleteHostedZone`

Resources: *, `arn:aws:servicediscovery:region:account-id:namespace/namespace-id`

DeleteService

Required Permissions (API Action): `servicediscovery>DeleteService`

Resources: *, `arn:aws:servicediscovery:region:account-id:service/service-id`

DeregisterInstance

Required Permissions (API Action):

- `servicediscovery:DeregisterInstance`
- `route53:GetHealthCheck`
- `route53>DeleteHealthCheck`
- `route53:UpdateHealthCheck`
- `route53:ChangeResourceRecordSets`

Resources: *

DiscoverInstances

Required Permissions (API Action): `servicediscovery:DiscoverInstances`

Resources: *

GetInstance

Required Permissions (API Action): `servicediscovery:GetInstance`

Resources: *

GetInstancesHealthStatus

Required Permissions (API Action): `servicediscovery:GetInstancesHealthStatus`

Resources: *

GetNamespace

Required Permissions (API Action): `servicediscovery:GetNamespace`

Resources: *, `arn:aws:servicediscovery:region:account-id:namespace/namespace-id`

GetOperation

Required Permissions (API Action): `servicediscovery:GetOperation`

Resources: *

GetService

Required Permissions (API Action): `servicediscovery:GetService`

Resources: *, `arn:aws:servicediscovery:region:account-id:service/service-id`

ListInstances

Required Permissions (API Action): `servicediscovery>ListInstances`

Resources: *

ListNamespaces

Required Permissions (API Action): `servicediscovery>ListNamespaces`

Resources: *

ListOperations

Required Permissions (API Action): `servicediscovery>ListOperations`

Resources: *

ListServices

Required Permissions (API Action): `servicediscovery>ListServices`

Resources: *

RegisterInstance

Required Permissions (API Action):

- `servicediscovery:RegisterInstance`
- `route53:GetHealthCheck`
- `route53:CreateHealthCheck`
- `route53:UpdateHealthCheck`
- `route53:ChangeResourceRecordSets`
- `ec2:DescribeInstances`

Resources: *

UpdateHttpNamespace

Required Permissions (API Action): `servicediscovery:UpdateHttpNamespace`

Resources: *, `arn:aws:servicediscovery:region:account-id:namespace/namespace-id`

UpdateInstanceCustomHealthStatus

Required Permissions (API Action): `servicediscovery:UpdateInstanceCustomHealthStatus`

Resources: *

UpdatePrivateDnsNamespace

Required Permissions (API Action):

- `servicediscovery:UpdatePrivateDnsNamespace`
- `route53:ChangeResourceRecordSets`

Resources: *, `arn:aws:servicediscovery:region:account-id:namespace/namespace-id`

UpdatePublicDnsNamespace

Required Permissions (API Action):

- `servicediscovery:UpdatePublicDnsNamespace`
- `route53:ChangeResourceRecordSets`

Resources: *, `arn:aws:servicediscovery:region:account-id:namespace/namespace-id`

UpdateService

Required Permissions (API Action):

- `servicediscovery:UpdateService`
- `route53:GetHealthCheck`
- `route53:CreateHealthCheck`
- `route53>DeleteHealthCheck`
- `route53:UpdateHealthCheck`
- `route53:ChangeResourceRecordSets`

Resources: *, `arn:aws:servicediscovery:region:account-id:service/service-id`

AWS Cloud Map Condition Keys Reference

AWS Cloud Map defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For more information, see [Specifying Conditions in an IAM Policy \(p. 31\)](#).

`servicediscovery:NamespaceArn`

A filter that lets you get objects by specifying the Amazon Resource Name (ARN) for the related namespace.

`servicediscovery:NamespaceName`

A filter that lets you get objects by specifying the name of the related namespace.

`servicediscovery:ServiceArn`

A filter that lets you get objects by specifying the Amazon Resource Name (ARN) for the related service.

servicediscovery:ServiceName

A filter that lets you get objects by specifying the name of the related service.

Logging and Monitoring in AWS Cloud Map

Monitoring is an important part of maintaining the reliability, availability, and performance of your AWS solutions. You should collect monitoring data from all of the parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. However, before you start monitoring, you should create a monitoring plan that includes answers to the following questions:

- What are your monitoring goals?
- What resources will you monitor?
- How often will you monitor these resources?
- What monitoring tools will you use?
- Who will perform the monitoring tasks?
- Who should be notified when something goes wrong?

Topics

- [Logging AWS Cloud Map API Calls with AWS CloudTrail \(p. 38\)](#)

Logging AWS Cloud Map API Calls with AWS CloudTrail

AWS Cloud Map is integrated with AWS CloudTrail, a service that provides a record of the actions that are taken by a user, a role, or an AWS service in AWS Cloud Map. CloudTrail captures all API calls for most AWS Cloud Map API actions as events. This includes calls from the AWS Cloud Map console and all programmatic access, such as the AWS Cloud Map API and AWS SDKs. (CloudTrail doesn't capture calls to the AWS Cloud Map [DiscoverInstances](#) API.)

If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Cloud Map. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to AWS Cloud Map, the IP address that the request was made from, who made the request, when it was made, and additional details.

Topics

- [AWS Cloud Map Information in CloudTrail \(p. 38\)](#)
- [Viewing AWS Cloud Map Events in Event History \(p. 39\)](#)
- [Understanding AWS Cloud Map Log File Entries \(p. 39\)](#)

AWS Cloud Map Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Cloud Map, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see [Viewing Events with CloudTrail Event History](#).

For an ongoing record of events in your AWS account, including events for AWS Cloud Map, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create

a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following topics:

- [Overview for Creating a Trail](#)
- [CloudTrail Supported Services and Integrations](#)
- [Configuring Amazon SNS Notifications for CloudTrail](#)
- [Receiving CloudTrail Log Files from Multiple Regions](#) and [Receiving CloudTrail Log Files from Multiple Accounts](#)

Most AWS Cloud Map actions are logged by CloudTrail and are documented in the [AWS Cloud Map API Reference](#). For example, calls to the [CreateHttpNamespace](#), [DeleteService](#), and [RegisterInstance](#) actions generate entries in the CloudTrail log files. (CloudTrail doesn't capture calls to the AWS Cloud Map [DiscoverInstances](#) API.)

Every event or log entry contains information about who generated the request. The identity information helps you determine the following situations:

- Whether the request was made with root or IAM user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the [CloudTrail userIdentity Element](#).

Viewing AWS Cloud Map Events in Event History

With CloudTrail, you can view recent events in **Event history**. To view events for AWS Cloud Map API requests, you need to choose the AWS Region where you created your namespaces in the Region selector at the top of the console. If you created namespaces in multiple AWS Regions, you must view the events for each Region separately. For more information, see [Viewing Events with CloudTrail Event History](#) in the *AWS CloudTrail User Guide*.

Understanding AWS Cloud Map Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The `eventName` element identifies the action that occurred. CloudTrail supports all AWS Cloud Map API actions. The following example shows a CloudTrail log entry for [CreatePublicDnsNamespace](#).

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "A1B2C3D4E5F6G7EXAMPLE",
        "arn": "arn:aws:iam:111122223333:user/smithj",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "smithj"
      },
    },
  ],
}
```

```
    "eventTime": "2018-01-16T00:44:17Z",
    "eventSource": "servicediscovery.amazonaws.com",
    "eventName": "CreatePublicDnsNamespace",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.92",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:52.0)
Gecko/20100101 Firefox/52.0",
    "requestParameters": {
      "description": "test",
      "creatorRequestId": "1234567890123456789",
      "name": "example.com"
    },
    "responseElements": {
      "operationId": "unmipghn37443trlkgpf4idvvithec6fw-2example"
    },
    "requestID": "35e1872d-c0dc-11e7-99e1-03e9fexample",
    "eventID": "409b4d91-34e6-41ee-bd97-a816dexample",
    "eventType": "AwsApiCall",
    "recipientAccountId": "444455556666"
  }
]
```

Compliance Validation for AWS Cloud Map

The security and compliance of AWS Cloud Map is assessed by third-party auditors as part of multiple AWS compliance programs, including Health Insurance Portability and Accountability Act (HIPAA), Payment Card Industry Data Security Standard (PCI DSS), ISO, and FIPS.

For a list of AWS services in scope of specific compliance programs, see [AWS Services in Scope by Compliance Program](#). For general information, see [AWS Compliance Programs](#).

You can download third-party audit reports using AWS Artifact. For more information, see [Downloading Reports in AWS Artifact](#).

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides resources to help with compliance:

- [Security and Compliance Quick Start Guides](#) – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- [Architecting for HIPAA Security and Compliance Whitepaper](#) – This paper describes how companies can use AWS to create HIPAA-compliant applications.
- [AWS Compliance Resources](#) – This collection of workbooks and guides might apply to your industry and location.
- [AWS Config](#) – This AWS service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- [AWS Security Hub](#) – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

Resilience in AWS Cloud Map

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency,

high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

AWS Cloud Map is primarily a global service. However, you can use AWS Cloud Map to create Route 53 health checks that check the health of resources in specific Regions, such as Amazon EC2 instances and Elastic Load Balancing load balancers.

For more information about AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

Infrastructure Security in AWS Cloud Map

As a managed service, AWS Cloud Map is protected by the AWS global network security procedures that are described in the [Amazon Web Services: Overview of Security Processes](#) paper.

You use AWS published API calls to access AWS Cloud Map through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that's associated with an IAM principal. Or you can use the [AWS Security Token Service](#) (AWS STS) to generate temporary security credentials to sign requests.

Tagging your AWS Cloud Map resources

To help you manage your AWS Cloud Map resources, you can assign your own metadata to each resource in the form of *tags*. This topic describes tags and shows you how to create them.

Contents

- [Tag basics \(p. 42\)](#)
- [Tagging your resources \(p. 42\)](#)
- [Tag restrictions \(p. 43\)](#)
- [Working with tags using the CLI or API \(p. 43\)](#)

Tag basics

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

Tags enable you to categorize your AWS resources by, for example, purpose, owner, or environment. When you have many resources of the same type, you can quickly identify a specific resource based on the tags you've assigned to it. For example, you can define a set of tags for your AWS Cloud Map services to help you track each service's owner and stack level. We recommend that you devise a consistent set of tag keys for each resource type.

Tags are not automatically assigned to your resources. After you add a tag, you can edit tag keys and values or remove tags from a resource at any time. If you delete a resource, any tags for the resource are also deleted.

Tags don't have any semantic meaning to AWS Cloud Map and are interpreted strictly as a string of characters. You can set the value of a tag to an empty string, but you can't set the value of a tag to null. If you add a tag that has the same key as an existing tag on that resource, the new value overwrites the old value.

You can work with tags using the AWS Management Console, the AWS CLI, and the AWS Cloud Map API.

If you're using AWS Identity and Access Management (IAM), you can control which users in your AWS account have permission to create, edit, or delete tags.

Tagging your resources

You can tag new or existing AWS Cloud Map namespaces and services..

If you're using the AWS Cloud Map console, you can apply tags to new resources when they are created or to existing resources at any time using the **Tags** tab on the relevant resource page.

If you're using the AWS Cloud Map API, the AWS CLI, or an AWS SDK, you can apply tags to new resources using the `tags` parameter on the relevant API action or to existing resources using the [TagResource](#) API action. For more information, see [TagResource](#).

Some resource-creating actions enable you to specify tags for a resource when the resource is created. If tags cannot be applied during resource creation, the resource creation process fails. This ensures that

resources you intended to tag on creation are either created with specified tags or not created at all. If you tag resources at the time of creation, you don't need to run custom tagging scripts after resource creation.

The following table describes the AWS Cloud Map resources that can be tagged, and the resources that can be tagged on creation.

Tagging support for AWS Cloud Map resources

Resource	Supports tags	Supports tag propagation	Supports tagging on creation (AWS Cloud Map API, AWS CLI, AWS SDK)
AWS Cloud Map namespaces	Yes	No. Namespace tags don't propagate to any other resources associated with the namespace.	Yes
AWS Cloud Map services	Yes	No. Service tags don't propagate to any other resources associated with the service.	Yes

Tag restrictions

The following basic restrictions apply to tags:

- Maximum number of tags for each resource – 50
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length – 128 Unicode characters in UTF-8
- Maximum value length – 256 Unicode characters in UTF-8
- If your tagging schema is used across multiple AWS services and resources, remember that other services might have restrictions on allowed characters. Generally allowed characters are letters, numbers, spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Don't use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for either keys or values, as it is reserved for AWS use. You can't edit or delete tag keys or values with this prefix. Tags with this prefix don't count against your tags-per-resource limit.

Working with tags using the CLI or API

Use the following AWS CLI commands or AWS Cloud Map API operations to add, update, list, and delete the tags for your resources.

Tagging support for AWS Cloud Map resources

Task	API action	AWS CLI	AWS Tools for Windows PowerShell
Add or overwrite one or more tags.	TagResource	<code>tag-resource</code>	<code>Add-SDResourceTag</code>

Task	API action	AWS CLI	AWS Tools for Windows PowerShell
Delete one or more tags.	UntagResource	<code>untag-resource</code>	<code>Remove-SDResourceTag</code>
List tags for a resource	ListTagsForResource	<code>list-tags-for-resource</code>	<code>Get-SDResourceTag</code>

The following examples show how to tag or untag resources using the AWS CLI.

Example 1: Tag an existing resource

The following command tags an existing resource.

```
aws servicediscovery tag-resource --resource-arn resource_ARN --tags team=devs
```

Example 2: Untag an existing resource

The following command deletes a tag from an existing resource.

```
aws servicediscovery untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Example 3: List tags for a resource

The following command lists the tags associated with an existing resource.

```
aws servicediscovery list-tags-for-resource --resource-arn resource_ARN
```

Some resource-creating actions enable you to specify tags when you create the resource. The following actions support tagging on creation.

Task	API action	AWS CLI	AWS Tools for Windows PowerShell
Create an HTTP namespace	CreateHttpNamespace	<code>create-http-namespace</code>	<code>New-SDHttpNamespace</code>
Create a private namespace based on DNS	CreatePrivateDnsNamespace	<code>create-private-dns-namespace</code>	<code>New-SDPrivateDnsNamespace</code>
Create a public namespace based on DNS	CreatePublicDnsNamespace	<code>create-public-dns-namespace</code>	<code>New-SDPublicDnsNamespace</code>
Create a service	CreateService	<code>create-service</code>	<code>New-SDService</code>

AWS Cloud Map quotas

AWS Cloud Map entities are subject to the following quotas. Each quota listed applies to each AWS Region where you create AWS Cloud Map resources. For example, each AWS account can create 50 namespaces in each Region.

Resource	Default quota
Namespaces	50 namespaces in each AWS Region * Request a higher quota
Service instances	1,000 instance for each service 2,000 instances in each namespace Request a higher quota
Custom attributes	30 for each service instance

* When you create a namespace, we automatically create an Amazon Route 53 hosted zone. This hosted zone counts against the quota on the number of hosted zones that you can create with an AWS account. For more information, see [Quotas on hosted zones](#) in the *Amazon Route 53 Developer Guide*.

AWS Cloud Map API request throttling quota

AWS Cloud Map throttles [DiscoverInstances](#) API requests for each AWS account on a per-Region basis. Throttling helps improve the performance of the service and helps provide fair usage for all AWS Cloud Map customers. Throttling ensures that calls to the AWS Cloud Map [DiscoverInstances](#) API doesn't exceed the maximum allowed [DiscoverInstances](#) API request quotas. [DiscoverInstances](#) API calls originating from any of the following sources are subject to the request quotas:

- A third-party application
- A command line tool
- The AWS Cloud Map console

If you exceed an API throttling quota, you get the `RequestLimitExceeded` error code. For more information, see [the section called "Request rate limiting" \(p. 45\)](#).

How throttling is applied

AWS Cloud Map uses the [token bucket algorithm](#) to implement API throttling. With this algorithm, your account has a *bucket* that holds a specific number of *tokens*. The number of tokens in the bucket represents your throttling quota at any given second. There is one bucket for a single Region, and it applies to all endpoints in the Region.

Request rate limiting

Throttling limits the number of [DiscoverInstances](#) API requests that you can make. Each request removes one token from the bucket. For example, the bucket size for the [DiscoverInstances](#) API operation is 2,000

tokens, so you can make up to 2,000 [DiscoverInstances](#) requests in one second. If you exceed 2,000 requests in one second, you're throttled and the remaining requests within that second fail.

Buckets automatically refill at a set rate. If the bucket isn't at capacity, a set number of tokens is added back every second until the bucket reaches capacity. If the bucket is at capacity when refill tokens arrive, then these tokens are discarded. The bucket size for the [DiscoverInstances](#) API operation is 2,000 tokens, and the refill rate is 1,000 tokens every second. If you make 2,000 [DiscoverInstances](#) API requests in a second, the bucket is immediately reduced to zero (0) tokens. The bucket is then refilled by up to 1,000 tokens every second until it reaches its maximum capacity of 2000 tokens.

You can use tokens as they are added to the bucket. You don't need to wait for the bucket to be at maximum capacity before you make API requests. If you deplete the bucket by making 2,000 [DiscoverInstances](#) API requests in one second, you can still make up to 1,000 [DiscoverInstances](#) API requests every second after that for as long as you need. This means that you can immediately use the refill tokens as they are added to your bucket. The bucket only starts to refill to the maximum capacity when you make fewer API requests every second than the refill rate.

Retries or batch processing

If an API request fails, your application might need to retry the request. To reduce the number of API requests, use an appropriate sleep interval between successive requests. For best results, use an increasing or variable sleep interval.

Calculating the sleep interval

When you have to poll or retry an API request, we recommend using an exponential backoff algorithm to calculate the sleep interval between API calls. By using progressively longer wait times between retries for consecutive error responses, you can reduce the number of failed requests. For more information and implementation examples of this algorithm, see [Error Retries and Exponential Backoff in AWS](#).

Adjusting API throttling quotas

You can request an increase to API throttling quotas for your AWS account. To request a quota adjustment, contact the [AWS Support Center](#).

Related Information

The following related resources can help you as you work with AWS Cloud Map.

Topics

- [AWS resources](#) (p. 47)
- [Third-Party Tools and Libraries](#) (p. 47)

AWS resources

The following related resources can help you as you work with this service.

- **Classes & Workshops** – Links to role-based and specialty courses, in addition to self-paced labs to help sharpen your AWS skills and gain practical experience.
- **AWS Developer Tools** – Links to developer tools, SDKs, IDE toolkits, and command line tools for developing and managing AWS applications.
- **AWS Whitepapers** – Links to a comprehensive list of technical AWS whitepapers, covering topics such as architecture, security, and economics and authored by AWS Solutions Architects or other technical experts.
- **AWS Support Center** – The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.
- **AWS Support** – The primary webpage for information about AWS Support, a one-on-one, fast-response support channel to help you build and run applications in the cloud.
- **Contact Us** – A central contact point for inquiries concerning AWS billing, account, events, abuse, and other issues.
- **AWS Site Terms** – Detailed information about our copyright and trademark; your account, license, and site access; and other topics.

Third-Party Tools and Libraries

In addition to AWS resources, the following third-party tools and libraries work with AWS Cloud Map.

- **Cloud Application Framework (AWS Cloud Map)** – Library that handles common cloud platform tasks, such as queuing messages, publishing events, and calling cloud functions, with the help of AWS Cloud Map.
- **ExternalDNS for Kubernetes** – Tool for configuring external DNS services including Amazon Route 53, and AWS Cloud Map for Kubernetes Ingresses and Services.

Document History for AWS Cloud Map

The following entries describe important changes in each release of the AWS Cloud Map documentation.

March 24, 2021

AWS Cloud Map added support for creating services in a namespace that supports DNS queries that are discoverable only using the [DiscoverInstances](#) API operation and not using DNS queries. For more information, see [Service discovery configuration](#).

February 08, 2021

AWS Cloud Map added support for adding metadata tags to your namespaces and services using the AWS Management Console. For more information, see [Tagging your AWS Cloud Map resources](#).

June 22, 2020

AWS Cloud Map added support for adding metadata tags to your namespaces and services using the AWS CLI and APIs. For more information, see [Tagging your AWS Cloud Map resources](#).

November 28, 2018

This is the first release of *AWS Cloud Map Developer Guide*.

AWS glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS General Reference*.