AWS Cloud Map services

An AWS Cloud Map service is a template for registering service instances that consists of the service name and DNS configuration, if applicable, for the service. You can also set up a health check to determine the health status of instances in the service and filter out unhealthy resources. A service can represent a component of your application. For example, you can create a service for resources that handle payments on your application and another for resources that manage users.

A service allows you to locate the resources for an application by getting back one or more endpoints that can be used to connect to the resource. The location of resources is done using DNS queries or the AWS Cloud Map `DiscoverInstances` API action, depending on how you've configured the namespace. You can use the AWS Cloud Map console to scope instance discovery at the service level.

You can also specify custom metadata as atttributes at the service level using the `UpdateServiceAttributes` API. You can set service attributes to avoid duplicating attributes across instances and modify these attributes without needing to make any changes to instance attributes. Information you can specify as attributes at the service level includes, but isn't limited to, the following:

- Endpoint weights for shifting traffic during progressive deployments.
- Service preferences such as API timeouts and suggested retry policies.

For more information, see UpdateServiceAttributes in the *AWS Cloud Map API reference*.

The following topics describe health check and DNS configurations for services and include instructions for creating, listing, updating, and deleting a service.

Topics

AWS Cloud Map service health check configuration

Health checks help determine whether service instances are healthy or not. If you don't configure a health check during service creation, traffic will be routed to service instances regardless of the instances' health status. When you configure a health check, AWS Cloud Map returns healthy resources by default. You can use the `HealthStatus` parameter of the `DiscoverInstances` API to filter resources by health status and get a list of unhealthy resources. You can also use the `GetInstancesHealthStatus` API to retrieve the health status of a particular service instance.

You can either configure a Route 53 health check or a custom, third-party health check when you create an AWS Cloud Map service.

Route 53 health checks

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.If you specify both `A` and `AAAA` record types in a service's DNS configuration, AWS Cloud Map creates a health check that uses the IPv4 address to check the health of the resource. If the endpoint that's specified by the IPv4 address is unhealthy, Route 53 considers both the `A` and `AAAA` records to be unhealthy. If you specify a `CNAME` record type in a service's DNS configuration, you can't configure a Route 53 health check.

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*.

Note

- You can't configure an Amazon Route 53 health check for a service created in a private DNS namespace.

- A Route 53 health checker in each health-checking AWS Region sends a health check request to an endpoint every 30 seconds. On average, your endpoint receives a health check request about every two seconds. However, health checkers don't coordinate with one another. Therefore, you might sometimes see several requests in one second that's followed by a few seconds with no health checks at all. For a list of health-checking regions, see Regions.

For information about the charges for Route 53 health checks, see Route 53 Pricing.

Custom health checks

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.

When you use a custom health checks, AWS Cloud Map doesn't check the health of a given resource directly. Instead, the third-party health checker checks the health of the resource and returns a status to your application. Your application will then need to submit a `UpdateInstanceCustomHealthStatus` request that relays this status to AWS Cloud Map. If the initial status relayed is `UNHEALTHY`, and if there isn't another `UpdateInstanceCustomHealthStatus` within 30 seconds that relays a status of `HEALTHY`, the resource is confirmed to be unhealthy. AWS Cloud Map stops routing traffic to that resource.

AWS Cloud Map service DNS configuration

When you create a service in a namespace that supports instance discovery by DNS queries, AWS Cloud Map creates Route 53 DNS records. You must specify a Route 53 routing policy and DNS record type that will apply to all Route 53 DNS records that AWS Cloud Map creates.

Routing policy

A routing policy determines how Route 53 responds to the DNS queries that are used for service instance discovery. Supported routing policies and how they relate to AWS Cloud Map are as follows.

Weighted routing

Route 53 returns the applicable value from one randomly selected AWS Cloud Map service instance from among the instances that you registered using the same AWS Cloud Map 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 Multivalue Answer Routing in the *Amazon Route 53 Developer Guide*.

Record type

A Route 53 DNS record type determines the type of value Route 53 returns in response to the DNS queries that are used for service instance discovery. The different DNS record types you can specify, and the associated values returned by Route 53 in response to queries are as follows.

A

If you specify this type, Route 53 returns the IP address of the resource in IPv4 format, such as **192.0.2.44**.

AAAA

If you specify this type, Route 53 returns the IP address of the resource in IPv6 format, such as **2001:0db8:85a3:0000:0000:abcd:0001:2345**.

CNAME

If you specify this type, Route 53 returns the domain name of the resource (such as www.example.com).

Note

- To configure a **CNAME** DNS record, you must specify the **Weighted routing** routing policy.
- When you configure a **CNAME** DNS record, you can't configure a Route 53 health check.

SRV

If you specify this type, Route 53 returns the value for an SRV record. The value for an **SRV** record uses the following values:

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

Consider the following:

- 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** (AWS_INSTANCE_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** (InstanceID) when you register an instance
  - The name of the service
  - The name of the namespace

  For example, suppose you specify **test** as an 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
  ```

  Note

  If you specify values an IPv4 address, an IPv6 address, or both when you register an instance, 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.

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.

Creating an AWS Cloud Map service for an application component

After creating a namespace, you can create services to represent different components of your application that serve particular purposes. For example, you can create a service for resources in your application that process payments.

Note

You can't create multiple services that are accessible by DNS queries with names that differ only by case (such as EXAMPLE and example). Trying to do so will result in these services having 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.

Follow these steps to create a service using the AWS Management Console, AWS CLI, and SDK for Python.

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. For **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.

   Note

   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](http://www.haproxy.org/)), 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**

6. (Optional) For **Service description**, enter a description for the service. The description that you enter here appears on the **Services** page and on the detail page for each service.

7. If the namespace supports DNS queries, under **Service discovery configuration**, you can configure discoverability at the service level. Choose between allowing both API calls and DNS queries or only API calls for the discovery of instances in this service.

   > **Note**
   >
   > If you choose **API calls**, AWS Cloud Map will not create SRV records when you register an instance.

   If you choose **API and DNS**, follow these steps to configure DNS records. You can add or remove DNS records.

   1. For **Routing policy**, select the Amazon Route 53 routing policy for the DNS records that AWS Cloud Map creates when you register instances. You can select between **Weighted routing** and **Multivalue answer routing**. For more information, see Routing policy.

      > **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 an Elastic Load Balancing load balancer when you register instances programmatically, choose **Weighted routing** for **Routing policy**.

   2. For **Record type**, choose the DNS record type that determines what Route 53 returns in response to DNS queries by AWS Cloud Map. For more information, see Record type.
   3. For **TTL**, specify a numerical value to define the time to live (TTL) value, in seconds, at the service level. 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.

8. Under **Health check configuration**, for **Health check options**, choose the type of health check applicable for service instances. You can choose not to configure any health checks, or you can choose between a Route 53 health check or an external health check for your instances. For more information, see AWS Cloud Map service health check configuration.

Note

Route 53 health checks are configurable only for services in public DNS namespaces.

If you choose **Route 53 health checks**, provide the following information.

1. For **Failure threshold**, provide a number between 1 and 10 that defines the number of consecutive Route 53 health checks a service instance must pass or fail for its health status to change.
2. For **Health check protocol**, select the method Route 53 will use to check the health of the service instances.
3. If you choose **HTTP** or **HTTPS** health check protocol, for **Health check path**, provide a 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.

For more information about Route 53 health checks, see How Amazon Route 53 Determines Whether a Health Check Is Healthy in the *Amazon Route 53 Developer Guide*.

9. (Optional) Under **Tags**, choose **Add tags** and then specify a key and a value to tag your namespace. You can specify one or more tags to add to your namespace. Tags allow you to categorize your AWS resources so you can more easily manage them. For more information, see Tagging your AWS Cloud Map resources.
10. Choose **Create service**.

AWS CLI

- Create a service with the `create-service` command. Replace the *red* values with your own.

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

Output:

```
{
        "Service": {
        "Id": "srv-xxxxxxxxxxx",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
xxxxxxxxxxx",
        "Name": "service-name",
        "NamespaceId": "ns-xxxxxxxxxxx",
        "DnsConfig": {
            "NamespaceId": "ns-xxxxxxxxxxx",
            "RoutingPolicy": "MULTIVALUE",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 60
                }
            ]
        },
        "CreateDate": 1587081768.334,
        "CreatorRequestId": "567c1193-6b00-4308-bd57-ad38a8822d25"
    }
}
```

AWS SDK for Python (Boto3)

If you don't already have `Boto3` installed, you can find instructions for installing, configuring, and using `Boto3` here.

1. Import `Boto3` and use `servicediscovery` as your service.

```
import boto3
client = boto3.client('servicediscovery')
```

2. Create a service with `create_service()`. Replace the *`red`* values with your own. For more information, see create_service.

```
response = client.create_service(
    DnsConfig={
        'DnsRecords': [
            {
                'TTL': 60,
                'Type': 'A',
            },
        ],
        'NamespaceId': 'ns-xxxxxxxxxxx',
        'RoutingPolicy': 'MULTIVALUE',
    },
    Name='service-name',
    NamespaceId='ns-xxxxxxxxxxx',
)
```

Example response output

```
{
    'Service': {
        'Arn': 'arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
xxxxxxxxxxx',
        'CreateDate': 1587081768.334,
        'DnsConfig': {
            'DnsRecords': [
                {
                    'TTL': 60,
                    'Type': 'A',
                },
            ],
            'NamespaceId': 'ns-xxxxxxxxxxx',
            'RoutingPolicy': 'MULTIVALUE',
        },
        'Id': 'srv-xxxxxxxxxxx',
        'Name': 'service-name',
        'NamespaceId': 'ns-xxxxxxxxxxx',
    },
    'ResponseMetadata': {
        '...': '...',
    },
}
```

Next steps

After creating a service, you can register your application resources as service instances that contain information about how your application can locate the resource. For more information about registering AWS Cloud Map service instances, see Registering a resource as an AWS Cloud Map service instance.

You can also specify custom metadata such as endpoint weights, API timeouts, and retry policies as service attributes after creating a service. For more information, see ServiceAttributes and `UpdateServiceAttributes` in the *AWS Cloud Map API Reference*.

Updating an AWS Cloud Map service

Depending on a service's configuration, you can update its tags, Route 53 health check failure threshold, and time to live (TTL) for DNS resolvers. To update a service, perform the following procedure.

Note

You can't update settings for services associated with HTTP namespaces.

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 in which the service is created.
4. On the **Namespace: *`namespace-name`*** page, select the service you want to edit and choose **View details**.
5. On the **Service: *`service-name`*** page, choose **Edit**.

   Note

   You can't use the **Edit** button workflow to edit values for services that allow only API calls for instance discovery. However, you can add or remove tags on the **Service: *`service-name`*** page.

6. On the **Edit service** page, under **Service description**, you can update any previously set description for the service or add a new description. You can also add tags and update **TTL** for DNS resolvers.
7. Under **DNS configuration**, for **TTL**, you can specify an updated period of time, in seconds, that 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.
8. If you've set up Route 53 health checks, for **Failure threshold**, you can specify a new number between 1 and 10 that defines the number of consecutive Route 53 health checks a service instance must pass or fail for its health status to change.
9. Choose **Update service**.

AWS CLI

- Update a service with the `update-service` command (replace the *red* value with your own).

```
aws servicediscovery update-service \
    --id  srv-xxxxxxxxxxx \
    --service "Description=new
 description,DnsConfig={DnsRecords=[{Type=A,TTL=60}]}"
```

Output:

```
{
    "OperationId": "l3pfx7f4ynndrbj3cfq5fm2qy2z37bms-5m6iaoty"
}
```

AWS SDK for Python (Boto3)

1. If you don't already have `Boto3` installed, you can find instructions for installing, configuring, and using `Boto3` here.
2. Import `Boto3` and use `servicediscovery` as your service.

```
import boto3
client = boto3.client('servicediscovery')
```

3. Update a service with `update_service()` (replace the *red* value with your own).

```
response = client.update_service(
    Id='srv-xxxxxxxxxxx',
    Service={
        'DnsConfig': {
            'DnsRecords': [
                {
                    'TTL': 300,
                    'Type': 'A',
                },
            ],
        },
        'Description': "new description",
    }
)
```

Example response output

```
{
    "OperationId": "l3pfx7f4ynndrbj3cfq5fm2qy2z37bms-5m6iaoty"
}
```

Listing AWS Cloud Map services 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 **Domain name** of the namespace that contains the services that you want to list. You can view a list of all services under **Services** and enter the service name or ID in the search field to find a specific service. You can identify the AWS account that created the service by using the **Created by** field and the account that owns the service by using the **Resource owner** field.

Note

If the namespace is a shared namespace, the AWS account ID under **Resource owner** is the account that created and shared the namespace. The account ID under **Created by** can differ from the ID under **Resource owner** if a namespace consumer created the service. The account IDs may not be the same as your account ID. For more information about shared namespaces, see Shared AWS Cloud Map namespaces.

AWS CLI

- List services with the `list-services` command. The following command lists all services in a namespace using the namespace ID as the filter. Replace the *red* value with your own.

```
aws servicediscovery list-services --filters
 Name=NAMESPACE_ID,Values=ns-1234567890abcdef,Condition=EQ
```

AWS SDK for Python (Boto3)

1. If you don't already have `Boto3` installed, you can find instructions for installing, configuring, and using `Boto3` here.
2. Import `Boto3` and use `servicediscovery` as your service.

```
import boto3
client = boto3.client('servicediscovery')
```

3. List services with `list_services()`.

```
response = client.list_services()
# If you want to see the response
print(response)
```

Example response output

```
{
    'Services': [
        {
            'Arn': 'arn:aws:servicediscovery:us-west-2:123456789012:service/srv-
xxxxxxxxxxxxxxxx',
            'CreateDate': 1587081768.334,
            'DnsConfig': {
                'DnsRecords': [
                    {
                        'TTL': 60,
                        'Type': 'A',
                    },
                ],
                'RoutingPolicy': 'MULTIVALUE',
            },
            'Id': 'srv-xxxxxxxxxxxxxxxx',
            'Name': 'myservice',
        },
    ],
    'ResponseMetadata': {
        '...': '...',
    },
}
```

Deleting an AWS Cloud Map service

Before you can delete a service, you must deregister all service instances that were registered using the service. For more information, see Deregistering an AWS Cloud Map service instance.

After deregistering all instances registered using the service, perform the following procedure to delete the service.

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
```

AWS SDK for Python (Boto3)

1. If you don't already have `Boto3` installed, you can find instructions for installing, configuring, and using `Boto3` here.
2. Import `Boto3` and use `servicediscovery` as your service.

```
import boto3
client = boto3.client('servicediscovery')
```

3. Delete a service with `delete_service()` (replace the *red* value with your own).

```
response = client.delete_service(
    Id='srv-xxxxxx',
)
# If you want to see the response
print(response)
```

Example response output

```
{
    'ResponseMetadata': {
        '...': '...',
    },
}
```