aws

Network Load Balancers

Elastic Load Balancing

Elastic Load Balancing: Network Load Balancers

Table of Contents

What is a Network Load Balancer?

Elastic Load Balancing automatically distributes your incoming traffic across multiple targets, such as EC2 instances, containers, and IP addresses, in one or more Availability Zones. It monitors the health of its registered targets, and routes traffic only to the healthy targets. Elastic Load Balancing scales your load balancer as your incoming traffic changes over time. It can automatically scale to the vast majority of workloads.

Elastic Load Balancing supports the following load balancers: Application Load Balancers, Network Load Balancers, Gateway Load Balancers, and Classic Load Balancers. You can select the type of load balancer that best suits your needs. This guide discusses Network Load Balancers. For more information about the other load balancers, see the User Guide for Application Load Balancers, the User Guide for Gateway Load Balancers, and the User Guide for Classic Load Balancers.

Network Load Balancer components

A *load balancer* serves as the single point of contact for clients. The load balancer distributes incoming traffic across multiple targets, such as Amazon EC2 instances. This increases the availability of your application. You add one or more listeners to your load balancer.

A *listener* checks for connection requests from clients, using the protocol and port that you configure, and forwards requests to a target group.

A *target group* routes requests to one or more registered targets, such as EC2 instances, using the protocol and the port number that you specify. Network Load Balancer target groups support the TCP, UDP, TCP_UDP, and TLS protocols. You can register a target with multiple target groups. You can configure health checks on a per target group basis. Health checks are performed on all targets registered to a target group that is specified in a listener rule for your load balancer.

For more information, see the following documentation:

- Load balancers
- Listeners
- Target groups

Network Load Balancer overview

A Network Load Balancer functions at the fourth layer of the Open Systems Interconnection (OSI) model. It can handle millions of requests per second. After the load balancer receives a connection request, it selects a target from the target group for the default rule. It attempts to open a TCP connection to the selected target on the port specified in the listener configuration.

When you enable an Availability Zone for the load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone. By default, each load balancer node distributes traffic across the registered targets in its Availability Zone only. If you enable cross-zone load balancing, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones. For more information, see Availability Zones.

To increase the fault tolerance of your applications, you can enable multiple Availability Zones for your load balancer and ensure that each target group has at least one target in each enabled Availability Zone. For example, if one or more target groups does not have a healthy target in an Availability Zone, we remove the IP address for the corresponding subnet from DNS, but the load balancer nodes in the other Availability Zones are still available to route traffic. If a client doesn't honor the time-to-live (TTL) and sends requests to the IP address after it is removed from DNS, the requests fail.

For TCP traffic, the load balancer selects a target using a flow hash algorithm based on the protocol, source IP address, source port, destination IP address, destination port, and TCP sequence number. The TCP connections from a client have different source ports and sequence numbers, and can be routed to different targets. Each individual TCP connection is routed to a single target for the life of the connection.

For UDP traffic, the load balancer selects a target using a flow hash algorithm based on the protocol, source IP address, source port, destination IP address, and destination port. A UDP flow has the same source and destination, so it is consistently routed to a single target throughout its lifetime. Different UDP flows have different source IP addresses and ports, so they can be routed to different targets.

Elastic Load Balancing creates a network interface for each Availability Zone you enable. Each load balancer node in the Availability Zone uses this network interface to get a static IP address. When you create an Internet-facing load balancer, you can optionally associate one Elastic IP address per subnet.

When you create a target group, you specify its target type, which determines how you register targets. For example, you can register instance IDs, IP addresses, or an Application Load Balancer. The target type also affects whether the client IP addresses are preserved. For more information, see the section called “Client IP preservation”.

You can add and remove targets from your load balancer as your needs change, without disrupting the overall flow of requests to your application. Elastic Load Balancing scales your load balancer as traffic to your application changes over time. Elastic Load Balancing can scale to the vast majority of workloads automatically.

You can configure health checks, which are used to monitor the health of the registered targets so that the load balancer can send requests only to the healthy targets.

For more information, see How Elastic Load Balancing works in the *Elastic Load Balancing User Guide*.

Benefits of migrating from a Classic Load Balancer

Using a Network Load Balancer instead of a Classic Load Balancer has the following benefits:

- Ability to handle volatile workloads and scale to millions of requests per second.
- Support for static IP addresses for the load balancer. You can also assign one Elastic IP address per subnet enabled for the load balancer.
- Support for registering targets by IP address, including targets outside the VPC for the load balancer.
- Support for routing requests to multiple applications on a single EC2 instance. You can register each instance or IP address with the same target group using multiple ports.
- Support for containerized applications. Amazon Elastic Container Service (Amazon ECS) can select an unused port when scheduling a task and register the task with a target group using this port. This enables you to make efficient use of your clusters.
- Support for monitoring the health of each service independently, as health checks are defined at the target group level and many Amazon CloudWatch metrics are reported at the target group level. Attaching a target group to an Auto Scaling group enables you to scale each service dynamically based on demand.

For more information about the features supported by each load balancer type, see Product comparisons for Elastic Load Balancing.

How to get started

To create a Network Load Balancer, try one of the following tutorials:

- Getting started with Network Load Balancers
- Tutorial: Create a Network Load Balancer using the AWS CLI

For demos of common load balancer configurations, see Elastic Load Balancing Demos.

Pricing

For more information, see Network Load Balancer Pricing.

Getting started with Network Load Balancers

This tutorial provides a hands-on introduction to Network Load Balancers through the AWS Management Console, a web-based interface. To create your first Network Load Balancer, complete the following steps.

Tasks

For demos of common load balancer configurations, see Elastic Load Balancing Demos.

Before you begin

- Decide which Availability Zones you will use for your EC2 instances. Configure your virtual private cloud (VPC) with at least one public subnet in each of these Availability Zones. These public subnets are used to configure the load balancer. You can launch your EC2 instances in other subnets of these Availability Zones instead.
- Launch at least one EC2 instance in each Availability Zone. Ensure that the security groups for these instances allow TCP access from clients on the listener port and health check requests from your VPC. For more information, see Target security groups.

Step 1: Configure your target group

Create a target group, which is used in request routing. The rule for your listener routes requests to the registered targets in this target group. The load balancer checks the health of targets in this target group using the health check settings defined for the target group.

To configure your target group using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose **Create target group**.
4. Keep the target type as **instances**.
5. For **Target group name**, enter a name for the new target group.
6. For **Protocol**, choose **TCP**, and for **Port**, choose **80**.
7. For **VPC**, select the VPC that contains your instances.
8. For **Health checks**, keep the default settings.
9. Choose **Next**.
10. On the **Register targets** page, complete the following steps. This is an optional step to create a target group. However, you must register your targets if you want to test your load balancer and ensure that it is routing traffic to your targets.
    a. For **Available instances**, select one or more instances.
    b. Keep the default port 80, and choose **Include as pending below**.
11. Choose **Create target group**.

Step 2: Choose a load balancer type

Elastic Load Balancing supports different types of load balancers. For this tutorial, you create a Network Load Balancer.

To create a Network Load Balancer using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation bar, choose a Region for your load balancer. Be sure to choose the same Region that you used for your EC2 instances.
3. In the navigation pane, under **Load Balancing**, choose **Load Balancers**.
4. Choose **Create load balancer**.
5. For **Network Load Balancer**, choose **Create**.

Step 3: Configure your load balancer and listener

To create a Network Load Balancer, you must first provide basic configuration information for your load balancer, such as a name, scheme, and IP address type. Then provide information about your

network, and one or more listeners. A listener is a process that checks for connection requests. It is configured with a protocol and a port for connections from clients to the load balancer. For more information about supported protocols and ports, see Listener configuration.

To configure your load balancer and listener

1. For **Load balancer name**, enter a name for your load balancer. For example, `my-nlb`.
2. For **Scheme** and **IP address type**, keep the default values.
3. For **Network mapping**, select the VPC that you used for your EC2 instances. For each Availability Zone that you used to launch your EC2 instances, select the Availability Zone and then select one public subnet for that Availability Zone.

   By default, AWS assigns an IPv4 address to each load balancer node from the subnet for its Availability Zone. Alternatively, when you create an internet-facing load balancer, you can select an Elastic IP address for each Availability Zone. This provides your load balancer with static IP addresses.
4. For **Security groups**, we preselect the default security group for your VPC. You can select other security groups as needed. If you don't have a suitable security group, choose **Create a new security group** and create one that meets your security needs. For more information, see Create a security group in the *Amazon VPC User Guide*.

   > **Warning**
   >
   > If you don't associate any security groups with your load balancer now, you can't associate them later on.
5. For **Listeners and routing**, keep the default protocol and port, and select the target group from the list. This configures a listener that accepts TCP traffic on port 80 and forwards traffic to the selected target group by default.
6. (Optional) Add tags to categorize your load balancer. Tag keys must be unique for each load balancer. Allowed characters are letters, spaces, numbers (in UTF-8), and the following special characters: + - = . _ : / @. Do not use leading or trailing spaces. Tag values are case-sensitive.
7. Review your configuration, and choose **Create load balancer**. A few default attributes are applied to your load balancer during creation. You can view and edit them after creating the load balancer. For more information, see Load balancer attributes.

Step 4: Test your load balancer

After creating the load balancer, verify that it's sending traffic to your EC2 instances.

To test your load balancer

1. After you are notified that your load balancer was created successfully, choose **Close**.
2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Select the newly created target group.
4. Choose **Targets** and verify that your instances are ready. If the status of an instance is `initial`, it's probably because the instance is still in the process of being registered, or it has not passed the minimum number of health checks to be considered healthy. After the status of at least one instance is `healthy`, you can test your load balancer.
5. In the navigation pane, under **Load Balancing**, choose **Load Balancers**.
6. Select the name of the newly created load balancer to open its details page.
7. Copy the DNS name of the load balancer (for example, my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com). Paste the DNS name into the address field of an internet-connected web browser. If everything is working, the browser displays the default page of your server.

Step 5: (Optional) Delete your load balancer

As soon as your load balancer becomes available, you are billed for each hour or partial hour that you keep it running. When you no longer need a load balancer, you can delete it. As soon as the load balancer is deleted, you stop incurring charges for it. Note that deleting a load balancer does not affect the targets registered with the load balancer. For example, your EC2 instances continue to run.

To delete your load balancer using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Load Balancers**.
3. Select the checkbox for the load balancer, and choose **Actions**, **Delete**.
4. When prompted for confirmation, enter **`confirm`** and choose **Delete**.

Tutorial: Create a Network Load Balancer using the AWS CLI

This tutorial provides a hands-on introduction to Network Load Balancers through the AWS CLI.

Before you begin

- Install the AWS CLI or update to the current version of the AWS CLI if you are using a version that does not support Network Load Balancers. For more information, see Installing the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.
- Decide which Availability Zones you will use for your EC2 instances. Configure your virtual private cloud (VPC) with at least one public subnet in each of these Availability Zones.
- Decide if you will create an IPv4 or dualstack load balancer. Use IPv4 if you want clients to communicate with the load balancer using IPv4 addresses only. Use dualstack if you want clients to communicate with the load balancer using IPv4 and IPv6 addresses. You can also use dualstack to communicate with backend targets, such as IPv6 applications or dualstack subnets, using IPv6.
- Launch at least one EC2 instance in each Availability Zone. Ensure that the security groups for these instances allow TCP access from clients on the listener port and health check requests from your VPC. For more information, see Target security groups.

Create your IPv4 load balancer

To create your first load balancer, complete the following steps.

To create a IPv4 load balancer

1. Use the create-load-balancer command to create an IPv4 load balancer, specifying a public subnet for each Availability Zone in which you launched instances. You can specify only one subnet per Availability Zone.

   By default, when Network Load Balancers are created using the AWS CLI, they do not automatically use the default security group for the VPC. If you don't associate any security groups with your load balancer during creation, you can't add them later. We recommend that

you specify security groups for your load balancer during creation by using the `--security-groups` option.

```
aws elbv2 create-load-balancer --name my-load-balancer --type network --subnets
 subnet-0e3f5cac72EXAMPLE --security-groups sg-0123456789EXAMPLE
```

The output includes the Amazon Resource Name (ARN) of the load balancer, with the following format:

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/net/my-load-
balancer/1234567890123456
```

2. Use the create-target-group command to create an IPv4 target group, specifying the same VPC that you used for your EC2 instances. IPv4 target groups support IP and instance type targets.

```
aws elbv2 create-target-group --name my-targets --protocol TCP --port 80 --vpc-id
 vpc-0598c7d356EXAMPLE
```

The output includes the ARN of the target group, with this format:

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-
targets/1234567890123456
```

3. Use the register-targets command to register your instances with your target group:

```
aws elbv2 register-targets --target-group-arn targetgroup-arn --targets
 Id=i-1234567890abcdef0 Id=i-0abcdef1234567890
```

4. Use the create-listener command to create a listener for your load balancer with a default rule that forwards requests to your target group:

```
aws elbv2 create-listener --load-balancer-arn loadbalancer-arn --protocol TCP --
port 80  \
--default-actions Type=forward,TargetGroupArn=targetgroup-arn
```

The output contains the ARN of the listener, with the following format:

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:listener/net/my-load-
balancer/1234567890123456/1234567890123456
```

5. (Optional) You can verify the health of the registered targets for your target group using this describe-target-health command:

```
aws elbv2 describe-target-health --target-group-arn targetgroup-arn
```

Create your dualstack load balancer

To create your first load balancer, complete the following steps.

To create a dualstack load balancer

1. Use the create-load-balancer command to create a dualstack load balancer, specifying a public subnet for each Availability Zone in which you launched instances. You can specify only one subnet per Availability Zone.

```
aws elbv2 create-load-balancer --name my-load-balancer --type network --subnets
 subnet-0e3f5cac72EXAMPLE --ip-address-type dualstack
```

   The output includes the Amazon Resource Name (ARN) of the load balancer, with the following format:

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/net/my-load-
balancer/1234567890123456
```

2. Use the create-target-group command to create a target group, specifying the same VPC that you used for your EC2 instances.

   You must use either a TCP or TLS target group with your dualstack load balancer.

   You can create IPv4 and IPv6 target groups to associate with dualstack load balancers. The target group's IP address type determines the IP version that the load balancer will use to both communicate with, and check the health of, your backend targets.

   IPv4 target groups support IP and instance type targets. IPv6 targets only support IP targets.

```
aws elbv2 create-target-group --name my-targets --protocol TCP --port 80 --vpc-id
 vpc-0598c7d356EXAMPLE  --ip-address-type [ipv4 or ipv6]
```

   The output includes the ARN of the target group, with this format:

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-
targets/1234567890123456
```

3. Use the register-targets command to register your instances with your target group:

```
aws elbv2 register-targets --target-group-arn targetgroup-arn --targets
 Id=i-1234567890abcdef0 Id=i-0abcdef1234567890
```

4. Use the create-listener command to create a listener for your load balancer with a default rule that forwards requests to your target group. Dualstack load balancers must have TCP or TLS listeners.

```
aws elbv2 create-listener --load-balancer-arn loadbalancer-arn --protocol TCP --
port 80  \
--default-actions Type=forward,TargetGroupArn=targetgroup-arn
```

The output contains the ARN of the listener, with the following format:

```
arn:aws:elasticloadbalancing:us-east-2:123456789012:listener/net/my-load-
balancer/1234567890123456/1234567890123456
```

5. (Optional) You can verify the health of the registered targets for your target group using this describe-target-health command:

```
aws elbv2 describe-target-health --target-group-arn targetgroup-arn
```

Specify an Elastic IP address for your load balancer

When you create a Network Load Balancer, you can specify one Elastic IP address per subnet using a subnet mapping.

```
aws elbv2 create-load-balancer --name my-load-balancer --type network \
--subnet-mappings SubnetId=subnet-0e3f5cac72EXAMPLE,AllocationId=eipalloc-12345678
```

Delete your load balancer

When you no longer need your load balancer and target group, you can delete them as follows:

```
aws elbv2 delete-load-balancer --load-balancer-arn loadbalancer-arn
aws elbv2 delete-target-group --target-group-arn targetgroup-arn
```

Network Load Balancers

A *load balancer* serves as the single point of contact for clients. Clients send requests to the load balancer, and the load balancer sends them to targets, such as EC2 instances, in one or more Availability Zones.

To configure your load balancer, you create target groups, and then register targets with your target groups. Your load balancer is most effective if you ensure that each enabled Availability Zone has at least one registered target. You also create listeners to check for connection requests from clients and route requests from clients to the targets in your target groups.

Network Load Balancers support connections from clients over VPC peering, AWS managed VPN, AWS Direct Connect, and third-party VPN solutions.

Contents

Load balancer state

A load balancer has one of the following states:

`provisioning`

The load balancer is being set up.

`active`

The load balancer is fully set up and ready to route traffic.

`failed`

The load balancer couldn't be set up.

Load balancer attributes

A load balancer has the following attributes:

`access_logs.s3.enabled`

Indicates whether access logs stored in Amazon S3 are enabled. The default is `false`.

`access_logs.s3.bucket`

The name of the Amazon S3 bucket for the access logs. This attribute is required if access logs are enabled. For more information, see Bucket requirements.

`access_logs.s3.prefix`

The prefix for the location in the Amazon S3 bucket.

`deletion_protection.enabled`

Indicates whether deletion protection is enabled. The default is `false`.

`ipv6.deny_all_igw_traffic`

Blocks internet gateway (IGW) access to the load balancer, preventing unintended access to your internal load balancer through an internet gateway. It is set to `false` for internet-facing load balancers and `true` for internal load balancers. This attribute does not prevent non-IGW internet access (for example, through peering, Transit Gateway, AWS Direct Connect, or AWS VPN).

`load_balancing.cross_zone.enabled`

Indicates whether cross-zone load balancing is enabled. The default is `false`.

`dns_record.client_routing_policy`

Indicates how traffic is distributed among the load balancer Availability Zones. The possible values are `availability_zone_affinity` with 100 percent zonal affinity, `partial_availability_zone_affinity` with 85 percent zonal affinity, and `any_availability_zone` with 0 percent zonal affinity.

IP address type

You can set the types of IP addresses that clients can use with your load balancer.

Network Load Balancers support the following IP address types:

`ipv4`

Clients must connect to the load balancer using IPv4 addresses (for example, 192.0.2.1). IPv4 enabled load balancers (both internet-facing and internal) support TCP, UDP, TCP_UDP, and TLS listeners.

`dualstack`

Clients can connect to the load balancer using both IPv4 addresses (for example, 192.0.2.1) and IPv6 addresses (for example, 2001:0db8:85a3:0:0:8a2e:0370:7334). Dualstack enabled load balancers (both internet-facing and internal) support TCP and TLS listeners.

Considerations

- The load balancer communicates with targets based on the IP address type of the target group.
- When you enable dualstack mode for the load balancer, Elastic Load Balancing provides an AAAA DNS record for the load balancer. Clients that communicate with the load balancer using IPv4 addresses resolve the A DNS record. Clients that communicate with the load balancer using IPv6 addresses resolve the AAAA DNS record.
- Access to your internal dualstack load balancers through the internet gateway is blocked to prevent unintended internet access. However, this does not prevent other internet access (for example, through peering, Transit Gateway, AWS Direct Connect, or AWS VPN).

For more information about IP address types, see IP address types for your Network Load Balancer.

Network Load Balancer resource map

The Network Load Balancer resource map provides an interactive display of your load balancer's architecture, including its associated listeners, target groups, and targets. The resource map also highlights the relationships and routing paths between all resources, producing a visual representation of your load balancer's configuration.

To view your Network Load Balancer's resource map using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, choose **Load Balancers**.
3. Select the load balancer.
4. Choose the **Resource map** tab to display the load balancer's resource map.

Resource map components

Map views

There are two views available in the Network Load Balancer resource map: **Overview**, and **Unhealthy Target Map**. **Overview** is selected by default and displays all of your load balancer's resources. Selecting the **Unhealthy Target Map** view will only display the unhealthy targets and the resources associated to them.

The **Unhealthy Target Map** view can be used to troubleshoot targets that are failing health checks. For more information, see Troubleshoot unhealthy targets using the resource map.

Resource columns

The Network Load Balancer resource map contains three resource columns, one for each resource type. The resource groups are **Listeners**, **Target groups**, and **Targets**.

Resource tiles

Each resource within a column has its own tile, which displays details about that specific resource.

- Hovering over a resource tile highlights the relationships between it and other resources.

- Selecting a resource tile highlights the relationships between it and other resources, and displays additional details about that resource.
  - **target group health summary:** The number of registered targets for each health status.
  - **target health status:** The target's current health status and description.

> **Note**
>
> You can turn off **Show resource details** to hide additional details within the resource map.

- Each resource tile contains a link that, when selected, navigates to that resource's details page.
  - **Listeners** - Select the listeners protocol:port. For example, `TCP:80`
  - **Target groups** - Select the target group name. For example, `my-target-group`
  - **Targets** - Select the targets ID. For example, `i-1234567890abcdef0`

Export the resource map

Selecting **Export** gives you the option of exporting the current view of your Network Load Balancer's resource map as a PDF.

Availability Zones

You enable one or more Availability Zones for your load balancer when you create it. If you enable multiple Availability Zones for your load balancer, this increases the fault tolerance of your applications. You can't disable Availability Zones for a Network Load Balancer after you create it, but you can enable additional Availability Zones.

When you enable an Availability Zone, you specify one subnet from that Availability Zone. Elastic Load Balancing creates a load balancer node in the Availability Zone and a network interface for the subnet (the description starts with "ELB net" and includes the name of the load balancer). Each load balancer node in the Availability Zone uses this network interface to get an IPv4 address. Note that you can view this network interface but you can't modify it.

When you create an internet-facing load balancer, you can optionally specify one Elastic IP address per subnet. If you do not choose one of your own Elastic IP addresses, Elastic Load Balancing provides one Elastic IP address per subnet for you. These Elastic IP addresses provide your load

balancer with static IP addresses that will not change during the life of the load balancer. You can't change these Elastic IP addresses after you create the load balancer.

When you create an internal load balancer, you can optionally specify one private IP address per subnet. If you do not specify an IP address from the subnet, Elastic Load Balancing chooses one for you. These private IP addresses provide your load balancer with static IP addresses that will not change during the life of the load balancer. You can't change these private IP addresses after you create the load balancer.

Considerations

- For internet-facing load balancers, the subnets that you specify must have at least 8 available IP addresses. For internal load balancers, this is only required if you let AWS select a private IPv4 address from the subnet.
- You can't specify a subnet in a constrained Availability Zone. The error message is "Load balancers with type 'network' are not supported in *az_name*". You can specify a subnet in another Availability Zone that is not constrained and use cross-zone load balancing to distribute traffic to targets in the constrained Availability Zone.
- You can specify subnets that were shared with you.
- You can't specify a subnet in a Local Zone.

After you enable an Availability Zone, the load balancer starts routing requests to the registered targets in that Availability Zone. Your load balancer is most effective if you ensure that each enabled Availability Zone has at least one registered target.

To add Availability Zones using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Network mapping** tab, choose **Edit subnets**.
5. To enable an Availability Zone, select the check box for that Availability Zone. If there is one subnet for that Availability Zone, it is selected. If there is more than one subnet for that Availability Zone, select one of the subnets. Note that you can select only one subnet per Availability Zone.

For an internet-facing load balancer, you can select an Elastic IP address for each Availability Zone. For an internal load balancer, you can assign a private IP address from the IPv4 range of each subnet instead of letting Elastic Load Balancing assign one.

6. Choose **Save changes**.

To add Availability Zones using the AWS CLI

Use the set-subnets command.

Cross-zone load balancing

By default, each load balancer node distributes traffic across the registered targets in its Availability Zone only. If you turn on cross-zone load balancing, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones. You can also turn on cross-zone load balancing at the target group level. For more information, see the section called "Cross-zone load balancing" and Cross-zone load balancing in the *Elastic Load Balancing User Guide*.

Deletion protection

To prevent your load balancer from being deleted accidentally, you can enable deletion protection. By default, deletion protection is disabled for your load balancer.

If you enable deletion protection for your load balancer, you must disable it before you can delete the load balancer.

To enable deletion protection using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Under **Configuration**, turn on **Deletion protection**.
6. Choose **Save changes**.

To disable deletion protection using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Under **Configuration**, turn on **Deletion protection**.
6. Choose **Save changes**.

To enable or disable deletion protection using the AWS CLI

Use the modify-load-balancer-attributes command with the `deletion_protection.enabled` attribute.

Connection idle timeout

For each TCP request that a client makes through a Network Load Balancer, the state of that connection is tracked. If no data is sent through the connection by either the client or target for longer than the idle timeout, the connection is closed. If a client or target sends data after the idle timeout period elapses, it receives a TCP RST packet to indicate that the connection is no longer valid.

We set the idle timeout value for TCP flows to 350 seconds. You can't modify this value. Clients or targets can use TCP keepalive packets to reset the idle timeout. Keepalive packets sent to maintain TLS connections can't contain data or payload.

When a TLS listener receives a TCP keepalive packet from either a client or a target, the load balancer generates TCP keepalive packets and sends them to both the front-end and back-end connections every 20 seconds. You can't modify this behavior.

While UDP is connectionless, the load balancer maintains UDP flow state based on the source and destination IP addresses and ports. This ensures that packets that belong to the same flow are consistently sent to the same target. After the idle timeout period elapses, the load balancer considers the incoming UDP packet as a new flow and routes it to a new target. Elastic Load Balancing sets the idle timeout value for UDP flows to 120 seconds.

EC2 instances must respond to a new request within 30 seconds in order to establish a return path.

DNS name

Each Network Load Balancer receives a default Domain Name System (DNS) name with the following syntax: *name*-*id*.elb.*region*.amazonaws.com. For example, my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com.

If you'd prefer to use a DNS name that is easier to remember, you can create a custom domain name and associate it with the DNS name for your load balancer. When a client makes a request using this custom domain name, the DNS server resolves it to the DNS name for your load balancer.

First, register a domain name with an accredited domain name registrar. Next, use your DNS service, such as your domain registrar, to create a DNS record to route requests to your load balancer. For more information, see the documentation for your DNS service. For example, if you use Amazon Route 53 as your DNS service, you create an alias record that points to your load balancer. For more information, see Routing traffic to an ELB load balancer in the *Amazon Route 53 Developer Guide*.

The load balancer has one IP address per enabled Availability Zone. These are the IP addresses of the load balancer nodes. The DNS name of the load balancer resolves to these addresses. For example, suppose that the custom domain name for your load balancer is `example.networkloadbalancer.com`. Use the following **dig** or **nslookup** command to determine the IP addresses of the load balancer nodes.

Linux or Mac

```
$ dig +short example.networkloadbalancer.com
```

Windows

```
C:\> nslookup example.networkloadbalancer.com
```

The load balancer has DNS records for its load balancer nodes. You can use DNS names with the following syntax to determine the IP addresses of the load balancer nodes: *az*.*name*-*id*.elb.*region*.amazonaws.com.

Linux or Mac

```
$ dig +short us-east-2b.my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com
```

Windows

```
C:\> nslookup us-east-2b.my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com
```

Availability Zone DNS affinity

When using the default client routing policy, requests sent to your Network Load Balancers DNS name will receive any healthy load balancer IP addresses. This leads to the distribution of client connections across the load balancers Availability Zones. With the Availability Zone affinity routing policies, client DNS queries favor load balancer IP addresses in their own Availability Zone. This helps improve both latency and resiliency, as clients do not need to cross Availability Zone boundaries when connecting to targets.

Client routing policies available to Network Load Balancers using Route 53 resolver:

- **Availability Zone affinity** – *100 percent zonal affinity*

  Client DNS queries will favor load balancer IP address in their own Availability Zone. Queries may resolve to other zones if there are no healthy load balancer IP addresses in their own zone.
- **Partial Availability Zone affinity** – *85 percent zonal affinity*

  85 percent of client DNS queries will favor load balancer IP addresses in their own Availability Zone, while the remaining queries resolve to any healthy zone. Queries may resolve to other healthy zones if there are no healthy IPs in their zone. When there are no healthy IPs in any zone, queries resolve to any zone.
- **Any Availability Zone** (default) – *0 percent zonal affinity*

  Client DNS queries are resolved among healthy load balancer IP addresses across all load balancer Availability Zones.

> **Note**
>
> Availability Zone affinity routing policies only apply to clients resolving the Network Load Balancers DNS name using Route 53 Resolver. For more information, see What is Amazon Route 53 Resolver? in the *Amazon Route 53 Developer Guide*

Availability Zone affinity helps route requests from the client to the load balancer, while cross-zone load balancing is used to help route requests from the load balancer to the targets. When

using Availability Zone affinity, cross-zone load balancing should be turned off, this ensures the load balancer traffic from clients to targets remains within the same Availability Zone. With this configuration, client traffic is sent to the same Network Load Balancer Availability Zone, so it's recommended to configure your application to scale independently in each Availability Zone. This is an important consideration when the number of clients per Availability zone, or the traffic per Availability Zone are not the same. For more information, see Cross-zone load balancing for target groups.

When an Availability Zone is considered unhealthy, or when a zonal shift is started, the zonal IP address will be considered unhealthy and not returned to clients unless fail open is in effect. Availability Zone affinity is maintained when the DNS record fails open. This helps keep Availability Zones independent and prevent potential cross zone failures.

When using Availability Zone affinity, times of imbalance between Availability Zones are expected. It's recommended ensuring your targets are scaling at the zonal level, to support each Availability Zones workload. In cases where these imbalances are significant, it's recommended turning off Availability Zone affinity. This allows even distribution of client connections between all the load balancers Availability Zones within 60 seconds, or the DNS TTL.

Before using Availability Zone affinity, consider the following:

- Availability Zone affinity causes changes on all of the Network Load Balancers clients who are using Route 53 Resolver.
  - Clients aren't able to decide between zonal-local and multi-zone DNS resolutions. Availability Zone affinity decides for them.
  - Clients aren't provided with a reliable method to determine when they're being impacted by Availability Zone affinity, or how to know which IP address is in which Availability Zone.
- Clients will remain assigned to their zone-local IP address until it is deemed fully unhealthy according to DNS health checks, and is removed from DNS.
- Using Availability Zone affinity with cross-zone load balancing on can lead to unbalanced distribution of client connections between Availability Zones. It's recommended to configure your application stack to scale independently in each Availability Zone, ensuring it can support zonal clients traffic.
- If cross-zone load balancing is on, the Network Load Balancer is subject to cross zone impact.
- The load on each of the Network Load Balancers Availability Zones will be proportional to the zonal locations of clients requests. If you don't configure how many clients are running in which Availability Zone, you will have to independently scale each Availability Zone reactively.

Monitoring

It is recommended to track the distribution of connections between Availability Zones, using the zonal load balancer metrics. You can use metrics to view the number of new and active connections per zone.

We recommend tracking the following:

- **`ActiveFlowCount`** – The total number of concurrent flows (or connections) from clients to targets.
- **`NewFlowCount`** – The total number of new flows (or connections) established from clients to targets in the time period.
- **`HealthyHostCount`** – The number of targets that are considered healthy.
- **`UnHealthyHostCount`** – The number of targets that are considered unhealthy.

For more information, see CloudWatch metrics for your Network Load Balancer

Turn on Availability Zone affinity

The steps in this procedure explain how to turn on Availability Zone affinity using the Amazon EC2 console.

To turn on Availability Zone affinity using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Under **Availability Zone routing configuration**, **Client routing policy (DNS record)**, select **Availability Zone affinity** or **Partial Availability Zone affinity**.
6. Choose **Save changes**.

To turn on Availability Zone affinity using the AWS CLI

Use the modify-load-balancer-attributes command with the `dns_record.client_routing_policy` attribute.

Turn off Availability Zone affinity

The steps in this procedure explain how to turn off Availability Zone affinity using the Amazon EC2 console.

To turn off Availability Zone affinity using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Under **Availability Zone routing configuration**, **Client routing policy (DNS record)**, select **Any Availability Zone**.
6. Choose **Save changes**.

To turn off Availability Zone affinity using the AWS CLI

Use the modify-load-balancer-attributes command with the `dns_record.client_routing_policy` attribute.

Create a Network Load Balancer

A load balancer takes requests from clients and distributes them across targets in a target group, such as EC2 instances.

Before you begin, ensure that the virtual private cloud (VPC) for your load balancer has at least one public subnet in each Availability Zone where you have targets. You must also configure a target group and register at least one target to set as default in order to route your traffic to the target group.

To create a load balancer using the AWS CLI, see Tutorial: Create a Network Load Balancer using the AWS CLI.

To create a load balancer using the AWS Management Console, complete the following tasks.

Tasks

- Step 1: Configure a target group

- Step 2: Register targets
- Step 3: Configure a load balancer and a listener
- Step 4: Test the load balancer

Step 1: Configure a target group

Configuring a target group allows you to register targets such as EC2 instances. The target group that you configure in this step is used as the target group in the listener rule when you configure your load balancer. For more information, see Target groups for your Network Load Balancers.

To configure your target group using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Target Groups**.
3. Choose **Create target group**.
4. For the **Basic configuration** pane, do the following:
   a. For **Choose a target type**, select **Instances** to register targets by instance ID, **IP addresses** to register targets by IP address, or **Application Load Balancer** to register an Application Load Balancer as a target.
   b. For **Target group name**, enter a name for the target group.
   c. For **Protocol**, choose a protocol as follows:
      - If the listener protocol is TCP, choose **TCP** or **TCP_UDP**.
      - If the listener protocol is TLS, choose **TCP** or **TLS**.
      - If the listener protocol is UDP, choose **UDP** or **TCP_UDP**.
      - If the listener protocol is TCP_UDP, choose **TCP_UDP**.
   d. (Optional) For **Port**, modify the default value as needed.
   e. For **IP address type**, choose **IPv4** or **IPv6**. This option is available only if the target type is **Instances** or **IP addresses** and the protocol is TCP or TLS.

      You must associate an IPv6 target group with a dualstack load balancer. All targets in the target group must have the same IP address type. You can't change the IP address type of a target group after you create it.
   f. For **VPC**, select the virtual private cloud (VPC) with the targets to register.

5. For the **Health checks** pane, modify the default settings as needed. For **Advanced health check settings**, choose the health check port, count, timeout, interval, and success codes. If health checks consecutively exceed the **Unhealthy threshold** count, the load balancer takes the target out of service. If health checks consecutively exceed the **Healthy threshold** count, the load balancer puts the target back in service. For more information, see Health checks for your target groups.
6. (Optional) To add a tag, expand **Tags**, choose **Add tag**, and enter a tag key and a tag value.
7. Choose **Next**.

Step 2: Register targets

You can register EC2 instances, IP addresses, or an Application Load Balancer with your target group. This is an optional step to create a load balancer. However, you must register your targets to ensure that your load balancer can route traffic to them.

1. On the **Register targets** page, add one or more targets as follows:
   - If the target type is **Instances**, select the instances, enter the ports, and then choose **Include as pending below**.
   - If the target type is **IP addresses**, select the network, enter the IP addresses and ports, and then choose **Include as pending below**.
   - If the target type is **Application Load Balancer**, select an Application Load Balancer.
2. Choose **Create target group**.

Step 3: Configure a load balancer and a listener

To create a Network Load Balancer, you must first provide basic configuration information for your load balancer, such as a name, scheme, and IP address type. Then provide information about your network and one or more listeners. A listener is a process that checks for connection requests. It is configured with a protocol and a port for connections from clients to the load balancer. For more information about supported protocols and ports, see Listener configuration.

To configure your load balancer and listener using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.

3. Choose **Create load balancer**.
4. Under **Network Load Balancer**, choose **Create**.
5. **Basic configuration**

   a. For **Load balancer name**, enter a name for your load balancer. For example, **`my-nlb`**. The name of your Network Load Balancer must be unique within your set of Application Load Balancers and Network Load Balancers for the Region. It can have a maximum of 32 characters, and contain only alphanumeric characters and hyphens. It must not begin or end with a hyphen, or with `internal-`.

   b. For **Scheme**, choose **Internet-facing** or **Internal**. An internet-facing load balancer routes requests from clients to targets over the internet. An internal load balancer routes requests to targets using private IP addresses.

   c. For **IP address type**, choose **IPv4** if your clients use IPv4 addresses to communicate with the load balancer or **Dualstack** if your clients use both IPv4 and IPv6 addresses to communicate with the load balancer.

6. **Network mapping**

   a. For **VPC**, select the VPC that you used for your EC2 instances.

   If you selected **Internet-facing** for **Scheme**, only VPCs with an internet gateway are available for selection.

   b. For **Mappings**, select one or more Availability Zones and corresponding subnets. Enabling multiple Availability Zones increases the fault tolerance of your applications. You can specify subnets that were shared with you.

   For internet-facing load balancers, you can select an Elastic IP address for each Availability Zone. This provides your load balancer with static IP addresses. Alternatively, for an internal load balancer, you can assign a private IP address from the IPv4 range of each subnet instead of letting AWS assign one for you.

7. For **Security groups**, we preselect the default security group for your VPC. You can select other security groups as needed. If you don't have a suitable security group, choose **Create a new security group** and create one that meets your security needs. For more information, see Create a security group in the *Amazon VPC User Guide*.

> **Warning**
>
> If you don't associate any security groups with your load balancer now, you can't associate them later on.

8. **Listeners and routing**

   a. The default is a listener that accepts TCP traffic on port 80. You can keep the default listener settings, or modify **Protocol** and **Port** as needed.

   b. For **Default action**, select a target group to forward traffic. If you didn't create a target group previously, you must create one now. You can optionally choose **Add listener** to add another listener (for example, a TLS listener).

   c. (Optional) Add tags to categorize your listener.

   d. For **Secure listener settings** (available only for TLS listeners), do the following:

      i. For **Security policy**, choose a security policy that meets your requirements.

      ii. For **ALPN policy**, choose a policy to enable ALPN or choose **None** to disable ALPN.

      iii. For **Default SSL certificate**, choose **From ACM** (recommended) and select a certificate. If you don't have an available certificate, you can import a certificate into ACM or use ACM to provision one for you. For more information, see Issuing and managing certificates in the *AWS Certificate Manager User Guide*.

9. (Optional) You can use **Add-on services** with your load balancer. For example, you can choose to have **AWS Global Accelerator** create an accelerator for you and associate your load balancer with the accelerator. The accelerator name can have the following characters (up to 64 characters): a-z, A-Z, 0-9, . (period), and - (hyphen). After the accelerator is created, go to the **AWS Global Accelerator** console to finish configuring it. For more information, see Add an accelerator when you create a load balancer

10. **Tags**

    (Optional) Add tags to categorize your load balancer. For more information, see Tags.

11. **Summary**

    Review your configuration, and choose **Create load balancer**. A few default attributes are applied to your load balancer during creation. You can view and edit them after creating the load balancer. For more information, see Load balancer attributes.

Step 4: Test the load balancer

After creating your load balancer, you can verify that your EC2 instances have passed the initial health check, and then test that the load balancer is sending traffic to your EC2 instances. To delete the load balancer, see Delete a Network Load Balancer.

To test the load balancer

1. After the load balancer is created, choose **Close**.
2. In the left navigation pane, choose **Target Groups**.
3. Select the new target group.
4. Choose **Targets** and verify that your instances are ready. If the status of an instance is `initial`, it's probably because the instance is still in the process of being registered or it has not passed the minimum number of health checks to be considered healthy. After the status of at least one instance is healthy, you can test your load balancer. For more information, see Target health status.
5. In the navigation pane, choose **Load Balancers**.
6. Select the new load balancer.
7. Copy the DNS name of the load balancer (for example, my-load-balancer-1234567890abcdef.elb.us-east-2.amazonaws.com). Paste the DNS name into the address field of an internet-connected web browser. If everything is working, the browser displays the default page of your server.

IP address types for your Network Load Balancer

You can configure your Network Load Balancer so that clients can communicate with the load balancer using IPv4 addresses only, or using both IPv4 and IPv6 addresses (dualstack). The load balancer communicates with targets based on the IP address type of the target group. For more information, see IP address type.

Dualstack requirements

- You can set the IP address type when you create the load balancer and update it at any time.
- The virtual private cloud (VPC) and subnets that you specify for the load balancer must have associated IPv6 CIDR blocks. For more information, see IPv6 addresses in the *Amazon EC2 User Guide*.

- The load balancer must have only TCP and TLS listeners.
- The route tables for the load balancer subnets must route IPv6 traffic.
- The network ACLs for the load balancer subnets must allow IPv6 traffic.

To set the IP address type at creation

Configure settings as described in Create a load balancer.

To update the IP address type using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the check box for the load balancer.
4. Choose **Actions**, **Edit IP address type**.
5. For **IP address type**, choose **IPv4** to support IPv4 addresses only or **Dualstack** to support both IPv4 and IPv6 addresses.
6. Choose **Save changes**.

To update the IP address type using the AWS CLI

Use the set-ip-address-type command.

Security groups for your Network Load Balancer

You can associate a security group with your Network Load Balancer to control the traffic that is allowed to reach and leave the load balancer. You specify the ports, protocols, and sources to allow for inbound traffic and the ports, protocols, and destinations to allow for outbound traffic. If you don't assign a security group to your load balancer, all client traffic can reach the load balancer listeners and all traffic can leave the load balancer.

You can add a rule to the security groups associated with your targets that references the security group associated with your Network Load Balancer. This allows clients to send traffic to your targets through your load balancer, but prevents them from sending traffic directly to your targets. Referencing the security group associated with your Network Load Balancer in the security groups associated with your targets ensures that your targets accept traffic from your load balancer even if you enable client IP preservation for your load balancer.

You are not charged for traffic that is blocked by inbound security group rules.

Contents

Considerations

- You can associate security groups with a Network Load Balancer when you create it. If you create a Network Load Balancer without associating any security groups, you can't associate them with the load balancer later on. We recommend that you associate a security group with your load balancer when you create it.
- After you create a Network Load Balancer with associated security groups, you can change the security groups associated with the load balancer at any time.
- Health checks are subject to outbound rules, but not inbound rules. You must ensure that outbound rules don't block health check traffic. Otherwise, the load balancer considers the targets unhealthy.
- You can control whether PrivateLink traffic is subject to inbound rules. If you enable inbound rules on PrivateLink traffic, the source of the traffic is the private IP address of the client, not the endpoint interface.

Example: Filter client traffic

The following inbound rules in the security group associated with your Network Load Balancer allow only traffic that comes from the specified address range. If this is an internal load balancer, you can specify a VPC CIDR range as the source to allow only traffic from a specific VPC. If this is an internet-facing load balancer that must accept traffic from anywhere on the internet, you can specify 0.0.0.0/0 as the source.

Inbound

Protocol	Source	Port range	Comment
protocol	*client IP address range*	*listener port*	Allows inbound traffic from the source CIDR on the listener port
ICMP	0.0.0.0/0	All	Allows inbound ICMP traffic to support MTU or Path MTU Discovery †

† For more information, see Path MTU Discovery in the *Amazon EC2 User Guide*.

Outbound

Protocol	Destination	Port range	Comment
All	Anywhere	All	Allows all outbound traffic

Example: Accept traffic only from the load balancer

Suppose that your Network Load Balancer has a security group sg-111112222233333. Use the following rules in the security groups associated with your target instances to ensure that they accept traffic only from the Network Load Balancer. You must ensure that the targets accept traffic from the load balancer on both the target port and the health check port. For more information, see the section called "Target security groups".

Inbound

Protocol	Source	Port range	Comment
protocol	sg-111112222233333	*target port*	Allows inbound traffic from the load balancer on the target port
protocol	sg-111112222233333	*health check*	Allows inbound traffic from the load balancer on the health check port

Outbound

Protocol	Destination	Port range	Comment
All	Anywhere	Any	Allows all outbound traffic

Update the associated security groups

If you associated at least one security group with a load balancer when you created it, you can update the security groups for that load balancer at any time.

To update security groups using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Load Balancers**.
3. Select the load balancer.
4. On the **Security** tab, choose **Edit**.
5. To associate a security group with your load balancer, select it. To remove a security group from your load balancer, clear it.
6. Choose **Save changes**.

To update security groups using the AWS CLI

Use the set-security-groups command.

Update the security settings

By default, we apply the inbound security group rules to all traffic sent to the load balancer. However, you might not want to apply these rules to traffic sent to the load balancer through AWS PrivateLink, which can originate from overlapping IP addresses. In this case, you can configure the load balancer so that we do not apply the inbound rules for traffic sent to the load balancer through AWS PrivateLink.

To update the security settings using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Load Balancers**.

3. Select the load balancer.
4. On the **Security** tab, choose **Edit**.
5. Under **Security setting**, clear **Enforce inbound rules on PrivateLink traffic**.
6. Choose **Save changes**.

To update the security settings using the AWS CLI

Use the set-security-groups command.

Monitor load balancer security groups

Use the `SecurityGroupBlockedFlowCount_Inbound` and `SecurityGroupBlockedFlowCount_Outbound` CloudWatch metrics to monitor the count of flows that are blocked by the load balancer security groups. Blocked traffic is not reflected in other metrics. For more information, see the section called "CloudWatch metrics".

Use VPC flow logs to monitor traffic that is accepted or rejected by the load balancer security groups. For more information, see VPC flow logs in the *Amazon VPC User Guide*.

Tags for your Network Load Balancer

Tags help you to categorize your load balancers in different ways. For example, you can tag a resource by purpose, owner, or environment.

You can add multiple tags to each load balancer. If you add a tag with a key that is already associated with the load balancer, it updates the value of that tag.

When you are finished with a tag, you can remove it from your load balancer.

Restrictions

- Maximum number of tags per resource—50
- Maximum key length—127 Unicode characters
- Maximum value length—255 Unicode characters
- Tag keys and values are case-sensitive. Allowed characters are letters, spaces, and numbers representable in UTF-8, plus the following special characters: + - = . _ : / @. Do not use leading or trailing spaces.

- Do not use the `aws:` prefix in your tag names or values because it is reserved for AWS use. You can't edit or delete tag names or values with this prefix. Tags with this prefix do not count against your tags per resource limit.

To update the tags for a load balancer using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Tags** tab, choose **Manage tags**.
5. To add a tag, choose **Add tag** and enter the tag key and tag value. Allowed characters are letters, spaces, numbers (in UTF-8), and the following special characters: + - = . _ : / @. Do not use leading or trailing spaces. Tag values are case-sensitive.
6. To update a tag, enter new values in **Key** and **Value**.
7. To delete a tag, choose the **Remove** button next to the tag.
8. When you have finished, choose **Save changes**.

To update the tags for a load balancer using the AWS CLI

Use the add-tags and remove-tags commands.

Delete a Network Load Balancer

As soon as your load balancer becomes available, you are billed for each hour or partial hour that you keep it running. When you no longer need the load balancer, you can delete it. As soon as the load balancer is deleted, you stop incurring charges for it.

You can't delete a load balancer if deletion protection is enabled. For more information, see Deletion protection.

You can't delete a load balancer if it is in use by another service. For example, if the load balancer is associated with a VPC endpoint service, you must delete the endpoint service configuration before you can delete the associated load balancer.

Deleting a load balancer also deletes its listeners. Deleting a load balancer does not affect its registered targets. For example, your EC2 instances continue to run and are still registered to their target groups. To delete your target groups, see Delete a target group.

To delete a load balancer using the console

1. If you have a DNS record for your domain that points to your load balancer, point it to a new location and wait for the DNS change to take effect before deleting your load balancer.

   Example:

   - If the record is a CNAME record with a Time To Live (TTL) of 300 seconds, wait at least 300 seconds before continuing to the next step.
   - If the record is a Route 53 Alias(A) record, wait at least 60 seconds.
   - If using Route 53, the record change takes 60 seconds to propagate to all global Route 53 name servers. Add this time to the TTL value of the record that is being updated.
2. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
3. In the navigation pane, choose **Load Balancers**.
4. Select the check box for the load balancer.
5. Choose **Actions**, **Delete load balancer**.
6. When prompted for confirmation, enter **`confirm`** and choose **Delete**.

To delete a load balancer using the AWS CLI

Use the delete-load-balancer command.

Zonal shift

Zonal shift is a capability in Amazon Route 53 Application Recovery Controller (Route 53 ARC). With zonal shift, you can shift a load balancer resource away from an impaired Availability Zone with a single action. This way, you can continue operating from other healthy Availability Zones in an AWS Region.

When you start a zonal shift, your load balancer stops sending traffic for the resource to the affected Availability Zone. Route 53 ARC creates the zonal shift immediately. However, it can take a short time, typically up to a few minutes, to complete existing, in-progress connections in the affected Availability Zone. For more information, see How a zonal shift works: health checks and zonal IP addresses in the *Amazon Route 53 Application Recovery Controller Developer Guide*.

Zonal shifts are only supported on Application Load Balancers and Network Load Balancers with cross-zone load balancing turned off. If you turn on cross-zone load balancing, you can't start a

zonal shift. For more information, see Resources supported for zonal shifts in the *Amazon Route 53 Application Recovery Controller Developer Guide*.

Before you use a zonal shift, review the following:

- Cross-zone load balancing isn't supported with zonal shifts. You must turn off cross-zone load balancing to use this capability.
- Zonal shift isn't supported when you use an Application Load Balancer as an accelerator endpoint in AWS Global Accelerator.
- You can start a zonal shift for a specific load balancer only for a single Availability Zone. You can't start a zonal shift for multiple Availability Zones.
- AWS proactively removes zonal load balancer IP addresses from DNS when multiple infrastructure issues impact services. Always check current Availability Zone capacity before you start a zonal shift. If your load balancers have cross-zone load balancing turned off and you use a zonal shift to remove a zonal load balancer IP address, the Availability Zone affected by the zonal shift also loses target capacity.
- When an Application Load Balancer is a target of a Network Load Balancer, always start the zonal shift from the Network Load Balancer. If you start a zonal shift from the Application Load Balancer, the Network Load Balancer doesn't recognize the shift and continues to send traffic to the Application Load Balancer.

For more guidance and information, see Best practices with Route 53 ARC zonal shifts in the *Amazon Route 53 Application Recovery Controller Developer Guide*.

Start a zonal shift

The steps in this procedure explain how to start a zonal shift using the Amazon EC2 console. For steps to start a zonal shift using the Route 53 ARC console, see Starting a zonal shift in the *Amazon Route 53 Application Recovery Controller Developer Guide*.

To start a zonal shift using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Load Balancers**.
3. Select the load balancer name.
4. On the **Integrations** tab, under **Route 53 Application Recovery Controller**, choose **Start zonal shift**.

5. Select the Availability Zone that you want to move traffic away from.
6. Choose or enter an expiration for the zonal shift. A zonal shift can initially be set from 1 minute up to three days (72 hours).

   All zonal shifts are temporary. You must set an expiration, but you can update active shifts later to set a new expiration.
7. Enter a comment. You can update the zonal shift later to edit the comment, if you like.
8. Select the check box to acknowledge that starting a zonal shift will reduce capacity for your application by shifting traffic away from the Availability Zone.
9. Choose **Start**.

To start a zonal shift using the AWS CLI

To work with zonal shift programmatically, see the Zonal Shift API Reference Guide.

Update a zonal shift

The steps in this procedure explain how to update a zonal shift using the Amazon EC2 console. For steps to update a zonal shift using the Amazon Route 53 Application Recovery Controller console, see Updating a zonal shift in the *Amazon Route 53 Application Recovery Controller Developer Guide*.

To update a zonal shift using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Load Balancers**.
3. Select a load balancer name that has an active zonal shift.
4. On the **Integrations** tab, under **Route 53 Application Recovery Controller**, choose **Update zonal shift**.

   This opens the Route 53 ARC console to continue the update.
5. For **Set zonal shift expiration**, optionally select or enter an expiration.
6. For **Comment**, optionally edit the existing comment or enter a new comment.
7. Choose **Update**.

To update a zonal shift using the AWS CLI

To work with zonal shift programmatically, see the Zonal Shift API Reference Guide.

Cancel a zonal shift

The steps in this procedure explain how to cancel a zonal shift using the Amazon EC2 console. For steps to cancel a zonal shift using the Amazon Route 53 Application Recovery Controller console, see Canceling a zonal shift in the *Amazon Route 53 Application Recovery Controller Developer Guide*.

To cancel a zonal shift using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Load Balancers**.
3. Select a load balancer name that has an active zonal shift.
4. On the **Integrations** tab, under **Route 53 Application Recovery Controller**, choose **Cancel zonal shift**.

   This opens the Route 53 ARC console to continue the cancelation.
5. Choose **Cancel zonal shift**.
6. On the confirmation dialog, choose **Confirm**.

To cancel a zonal shift using the AWS CLI

To work with zonal shift programmatically, see the Zonal Shift API Reference Guide.

Listeners for your Network Load Balancers

A *listener* is a process that checks for connection requests, using the protocol and port that you configure. Before you start using your Network Load Balancer, you must add at least one listener. If your load balancer has no listeners, it can't receive traffic from clients. The rule that you define for a listener determines how the load balancer routes requests to the targets that you register, such as EC2 instances.

Contents

Listener configuration

Listeners support the following protocols and ports:

- **Protocols**: TCP, TLS, UDP, TCP_UDP
- **Ports**: 1-65535

You can use a TLS listener to offload the work of encryption and decryption to your load balancer so that your applications can focus on their business logic. If the listener protocol is TLS, you must deploy exactly one SSL server certificate on the listener. For more information, see TLS listeners for your Network Load Balancer.

If you must ensure that the targets decrypt TLS traffic instead of the load balancer, you can create a TCP listener on port 443 instead of creating a TLS listener. With a TCP listener, the load balancer passes encrypted traffic through to the targets without decrypting it.

To support both TCP and UDP on the same port, create a TCP_UDP listener. The target groups for a TCP_UDP listener must use the TCP_UDP protocol.

For dualstack Network Load Balancers, only TCP and TLS protocols are supported.

You can use WebSockets with your listeners.

All network traffic sent to a configured listener is classified as intended traffic. Network traffic that does not match a configured listener is classified as unintended traffic. ICMP requests other than Type 3 are also considered unintended traffic. Network Load Balancers drop unintended traffic without forwarding it to any targets. TCP data packets sent to the listener port for a configured listeners that are not new connections or part of an active TCP connection are rejected with a TCP reset (RST).

For more information, see Request routing in the *Elastic Load Balancing User Guide*.

Listener rules

When you create a listener, you specify a rule for routing requests. This rule forwards requests to the specified target group. To update this rule, see Update a listener for your Network Load Balancer.

Create a listener for your Network Load Balancer

A listener is a process that checks for connection requests. You define a listener when you create your load balancer, and you can add listeners to your load balancer at any time.

Prerequisites

- You must specify a target group for the listener rule. For more information, see Create a target group for your Network Load Balancer.
- You must specify an SSL certificate for a TLS listener. The load balancer uses the certificate to terminate the connection and decrypt requests from clients before routing them to targets. For more information, see Server certificates.

Add a listener

You configure a listener with a protocol and a port for connections from clients to the load balancer, and a target group for the default listener rule. For more information, see Listener configuration.

To add a listener using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Listeners** tab, choose **Add listener**.
5. For **Protocol**, choose **TCP**, **UDP**, **TCP_UDP**, or **TLS**. Keep the default port or type a different port. For dualstack Network Load Balancers, only the TCP and TLS protocols are supported.
6. For **Default action**, choose an available target group.
7. [TLS listeners] For **Security policy**, we recommend that you keep the default security policy.
8. [TLS listeners] For **Default SSL certificate**, do one of the following:
   - If you created or imported a certificate using AWS Certificate Manager, choose **From ACM** and choose the certificate.
   - If you uploaded a certificate using IAM, choose **From IAM** and choose the certificate.
9. [TLS listeners] For **ALPN policy**, choose a policy to enable ALPN or choose **None** to disable ALPN. For more information, see ALPN policies.
10. Choose **Add**.
11. [TLS listeners] To add an optional certificate list for use with the SNI protocol, see Add certificates to the certificate list.

To add a listener using the AWS CLI

Use the create-listener command to create the listener.

TLS listeners for your Network Load Balancer

To use a TLS listener, you must deploy at least one server certificate on your load balancer. The load balancer uses a server certificate to terminate the front-end connection and then to decrypt requests from clients before sending them to the targets. Note that if you need to pass encrypted traffic to the targets without the load balancer decrypting it, create a TCP listener on port 443 instead of creating a TLS listener. The load balancer passes the request to the target as is, without decrypting it.

Elastic Load Balancing uses a TLS negotiation configuration, known as a security policy, to negotiate TLS connections between a client and the load balancer. A security policy is a

combination of protocols and ciphers. The protocol establishes a secure connection between a client and a server and ensures that all data passed between the client and your load balancer is private. A cipher is an encryption algorithm that uses encryption keys to create a coded message. Protocols use several ciphers to encrypt data over the internet. During the connection negotiation process, the client and the load balancer present a list of ciphers and protocols that they each support, in order of preference. The first cipher on the server's list that matches any one of the client's ciphers is selected for the secure connection.

Network Load Balancers do not support TLS renegotiation or mutual TLS authentication (mTLS). For mTLS support, create a TCP listener instead of a TLS listener. The load balancer passes the request through as is, so you can implement mTLS on the target.

To create a TLS listener, see Add a listener. For related demos, see TLS Support on Network Load Balancer and SNI Support on Network Load Balancer.

Server certificates

The load balancer requires X.509 certificates (server certificate). Certificates are a digital form of identification issued by a certificate authority (CA). A certificate contains identification information, a validity period, a public key, a serial number, and the digital signature of the issuer.

When you create a certificate for use with your load balancer, you must specify a domain name. The domain name on the certificate must match the custom domain name record so that we can verify the TLS connection. If they do not match, the traffic is not encrypted.

You must specify a fully qualified domain name (FQDN) for your certificate, such as `www.example.com` or an apex domain name such as `example.com`. You can also use an asterisk (*) as a wild card to protect several site names in the same domain. When you request a wild-card certificate, the asterisk (*) must be in the leftmost position of the domain name and can protect only one subdomain level. For instance, `*.example.com` protects `corp.example.com`, and `images.example.com`, but it cannot protect `test.login.example.com`. Also note that `*.example.com` protects only the subdomains of `example.com`, it does not protect the bare or apex domain (`example.com`). The wild-card name appears in the **Subject** field and in the **Subject Alternative Name** extension of the certificate. For more information about public certificates, see Requesting a public certificate in the *AWS Certificate Manager User Guide*.

We recommend that you create certificates for your load balancers using AWS Certificate Manager (ACM). ACM integrates with Elastic Load Balancing so that you can deploy the certificate on your load balancer. For more information, see the AWS Certificate Manager User Guide.

Alternatively, you can use TLS tools to create a certificate signing request (CSR), then get the CSR signed by a CA to produce a certificate, then import the certificate into ACM or upload the certificate to AWS Identity and Access Management (IAM). For more information, see Importing certificates in the *AWS Certificate Manager User Guide* or Working with server certificates in the *IAM User Guide*.

Contents

Supported key algorithms

- RSA 1024-bit
- RSA 2048-bit
- RSA 3072-bit
- ECDSA 256-bit
- ECDSA 384-bit
- ECDSA 521-bit

Default certificate

When you create a TLS listener, you must specify exactly one certificate. This certificate is known as the *default certificate*. You can replace the default certificate after you create the TLS listener. For more information, see Replace the default certificate.

If you specify additional certificates in a certificate list, the default certificate is used only if a client connects without using the Server Name Indication (SNI) protocol to specify a hostname or if there are no matching certificates in the certificate list.

If you do not specify additional certificates but need to host multiple secure applications through a single load balancer, you can use a wildcard certificate or add a Subject Alternative Name (SAN) for each additional domain to your certificate.

Certificate list

After you create a TLS listener, it has a default certificate and an empty certificate list. You can optionally add certificates to the certificate list for the listener. Using a certificate list enables the load balancer to support multiple domains on the same port and provide a different certificate for each domain. For more information, see Add certificates to the certificate list.

The load balancer uses a smart certificate selection algorithm with support for SNI. If the hostname provided by a client matches a single certificate in the certificate list, the load balancer selects this certificate. If a hostname provided by a client matches multiple certificates in the certificate list, the load balancer selects the best certificate that the client can support. Certificate selection is based on the following criteria in the following order:

- Hashing algorithm (prefer SHA over MD5)
- Key length (prefer the largest)
- Validity period

The load balancer access log entries indicate the hostname specified by the client and the certificate presented to the client. For more information, see Access log entries.

Certificate renewal

Each certificate comes with a validity period. You must ensure that you renew or replace each certificate for your load balancer before its validity period ends. This includes the default certificate and certificates in a certificate list. Renewing or replacing a certificate does not affect in-flight requests that were received by the load balancer node and are pending routing to a healthy target. After a certificate is renewed, new requests use the renewed certificate. After a certificate is replaced, new requests use the new certificate.

You can manage certificate renewal and replacement as follows:

- Certificates provided by AWS Certificate Manager and deployed on your load balancer can be renewed automatically. ACM attempts to renew certificates before they expire. For more information, see Managed renewal in the *AWS Certificate Manager User Guide*.
- If you imported a certificate into ACM, you must monitor the expiration date of the certificate and renew it before it expires. For more information, see Importing certificates in the *AWS Certificate Manager User Guide*.

- If you imported a certificate into IAM, you must create a new certificate, import the new certificate to ACM or IAM, add the new certificate to your load balancer, and remove the expired certificate from your load balancer.

Security policies

When you create a TLS listener, you must select a security policy. You can update the security policy as needed. For more information, see Update the security policy.

Considerations:

- The `ELBSecurityPolicy-TLS13-1-2-2021-06` policy is the default security policy for TLS listeners created using the AWS Management Console.
  - We recommend the `ELBSecurityPolicy-TLS13-1-2-2021-06` security policy, which includes TLS 1.3, and is backwards compatible with TLS 1.2.
- The `ELBSecurityPolicy-2016-08` policy is the default security policy for TLS listeners created using the AWS CLI.
- You can choose the security policy that is used for front-end connections, but not backend connections.
  - For backend connections, if your TLS listener is using a TLS 1.3 security policy, the `ELBSecurityPolicy-TLS13-1-0-2021-06` security policy is used. Otherwise, the `ELBSecurityPolicy-2016-08` security policy is used for backend connections.
- To meet compliance and security standards that require disabling certain TLS protocol versions, or to support legacy clients requiring deprecated ciphers, you can use one of the `ELBSecurityPolicy-TLS-` security policies. You can enable access logs for information about the TLS requests sent to your Network Load Balancer, analyze TLS traffic patterns, manage security policy upgrades, and troubleshoot issues. Enable access logging for your load balancer and examine the corresponding access log entries. For more information, see Access Logs and Network Load Balancer Example Queries.
- You can restrict which security policies are available to users across your AWS accounts and AWS Organizations by using the Elastic Load Balancing condition keys in your IAM and service control policies (SCPs), respectively. For more information, see Service control policies (SCPs) in the *AWS Organizations User Guide*

TLS 1.3 security policies

Elastic Load Balancing provides the following TLS 1.3 security policies for Network Load Balancers:

- `ELBSecurityPolicy-TLS13-1-2-2021-06` **(Recommended)**
- `ELBSecurityPolicy-TLS13-1-2-Res-2021-06`
- `ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06`
- `ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06`
- `ELBSecurityPolicy-TLS13-1-1-2021-06`
- `ELBSecurityPolicy-TLS13-1-0-2021-06`
- `ELBSecurityPolicy-TLS13-1-3-2021-06`

FIPS security policies

The Federal Information Processing Standard (FIPS) is a US and Canadian government standard that specifies the security requirements for cryptographic modules that protect sensitive information. To learn more, see Federal Information Processing Standard (FIPS) 140 on the *AWS Cloud Security Compliance* page.

All FIPS policies leverage the AWS-LC FIPS validated cryptographic module. To learn more, see the AWS-LC Cryptographic Module page on the *NIST Cryptographic Module Validation Program* site.

Elastic Load Balancing provides the following FIPS security policies for Network Load Balancer:

- `ELBSecurityPolicy-TLS13-1-3-FIPS-2023-04`
- `ELBSecurityPolicy-TLS13-1-2-Res-FIPS-2023-04`
- `ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04` **(Recommended)**
- `ELBSecurityPolicy-TLS13-1-2-Ext0-FIPS-2023-04`
- `ELBSecurityPolicy-TLS13-1-2-Ext1-FIPS-2023-04`
- `ELBSecurityPolicy-TLS13-1-2-Ext2-FIPS-2023-04`
- `ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04`
- `ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04`

FS supported policies

Elastic Load Balancing provides the following FS (Forward Secrecy) supported security policies for Network Load Balancers:

- `ELBSecurityPolicy-FS-1-2-Res-2020-10`
- `ELBSecurityPolicy-FS-1-2-Res-2019-08`
- `ELBSecurityPolicy-FS-1-2-2019-08`
- `ELBSecurityPolicy-FS-1-1-2019-08`
- `ELBSecurityPolicy-FS-2018-06`

TLS 1.0 - 1.2 security policies

Elastic Load Balancing provides the following TLS 1.0 - 1.2 security policies for Network Load Balancers:

- `ELBSecurityPolicy-TLS-1-2-Ext-2018-06`
- `ELBSecurityPolicy-TLS-1-2-2017-01`
- `ELBSecurityPolicy-TLS-1-1-2017-01`
- `ELBSecurityPolicy-2016-08`
- `ELBSecurityPolicy-TLS-1-0-2015-04`
- `ELBSecurityPolicy-2015-05` **(identical to** `ELBSecurityPolicy-2016-08`**)**

TLS protocols and ciphers

TLS 1.3

The following table describes the supported TLS protocols and ciphers for the available TLS 1.3 security policies.

Note: The `ELBSecurityPolicy-` prefix has been removed from the policy names in the security policies row.

Example: Security policy `ELBSecurityPolicy-TLS13-1-2-2021-06` is displayed as `TLS13-1-2-2021-06`.

Security policies	TLS13-1-2-2021-06	TLS13-1-3-2021-06	TLS13-1-2-Res-2021-06	TLS13-1-2-Ext2-2021-06	TLS13-1-2-Ext1-2021-06	TLS13-1-1-2021-06	TLS13-1-0-2021-06
TLS Protocols							
Protocol-TLSv1							✓
Protocol-TLSv1.1						✓	✓
Protocol-TLSv1.2	✓		✓	✓	✓	✓	✓
Protocol-TLSv1.3	✓	✓	✓	✓	✓	✓	✓
TLS Ciphers							
TLS_AES_128_GCM_SHA256	✓	✓	✓	✓	✓	✓	✓
TLS_AES_256_GCM_SHA384	✓	✓	✓	✓	✓	✓	✓
TLS_CHACHA20_POLY1305_SHA256	✓	✓	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128	✓		✓	✓	✓	✓	✓

Security policies	TLS13-1-2-2021-06	TLS13-1-3-2021-06	TLS13-1-2-Res-2021-06	TLS13-1-2-Ext2-2021-06	TLS13-1-2-Ext1-2021-06	TLS13-1-1-2021-06	TLS13-1-0-2021-06
-GCM-SHA256							
ECDHE-RSA-AES128-GCM-SHA256	✓		✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA256	✓			✓	✓	✓	✓
ECDHE-RSA-AES128-SHA256	✓			✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA				✓		✓	✓
ECDHE-RSA-AES128-SHA				✓		✓	✓

Security policies	TLS13-1-2-2021-06	TLS13-1-3-2021-06	TLS13-1-2-Res-2021-06	TLS13-1-2-Ext2-2021-06	TLS13-1-2-Ext1-2021-06	TLS13-1-1-2021-06	TLS13-1-0-2021-06
ECDHE-ECDSA-AES256-GCM-SHA384	✓		✓	✓	✓	✓	✓
ECDHE-RSA-AES256-GCM-SHA384	✓		✓	✓	✓	✓	✓
ECDHE-ECDSA-AES256-SHA384	✓			✓	✓	✓	✓
ECDHE-RSA-AES256-SHA384	✓			✓	✓	✓	✓
ECDHE-RSA-AES256-SHA				✓		✓	✓
ECDHE-ECDSA-AES256-SHA				✓		✓	✓

Security policies	TLS13-1-2-2021-06	TLS13-1-3-2021-06	TLS13-1-2-Res-2021-06	TLS13-1-2-Ext2-2021-06	TLS13-1-2-Ext1-2021-06	TLS13-1-1-2021-06	TLS13-1-0-2021-06
AES128-GCM-SHA256				✓	✓	✓	✓
AES128-SHA256				✓	✓	✓	✓
AES128-SHA				✓		✓	✓
AES256-GCM-SHA384				✓	✓	✓	✓
AES256-SHA256				✓	✓	✓	✓
AES256-SHA				✓		✓	✓

To create a TLS listener that uses a TLS 1.3 policy using the CLI

Use the create-listener command with any TLS 1.3 security policy.

The example uses the `ELBSecurityPolicy-TLS13-1-2-2021-06` security policy.

```
aws elbv2 create-listener --name my-listener \
--protocol TLS --port 443 \
--ssl-policy ELBSecurityPolicy-TLS13-1-2-2021-06
```

To modify a TLS listener to use a TLS 1.3 policy using the CLI

Use the modify-listener command with any TLS 1.3 security policy.

The example uses the ELBSecurityPolicy-TLS13-1-2-2021-06 security policy.

```
aws elbv2 modify-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0 \
--ssl-policy ELBSecurityPolicy-TLS13-1-2-2021-06
```

To view the security policies used by a listener using the CLI

Use the describe-listener command with the arn of your listener.

```
aws elbv2 describe-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0
```

To view the configuration of a TLS 1.3 security policy using the CLI

Use the describe-ssl-policies command with any TLS 1.3 security policy.

The example uses the ELBSecurityPolicy-TLS13-1-2-2021-06 security policy.

```
aws elbv2 describe-ssl-policies \
--names ELBSecurityPolicy-TLS13-1-2-2021-06
```

FIPS

> **Important**
>
> Policies ELBSecurityPolicy-TLS13-1-1-FIPS-2023-04 and ELBSecurityPolicy-TLS13-1-0-FIPS-2023-04 are provided for legacy compatibility only. While they utilize FIPS cryptography using the FIPS140 module, they may not conform to the latest NIST guidance for TLS configuration.

The following table describes the supported TLS protocols and ciphers for the available FIPS security policies.

Note: The ELBSecurityPolicy- prefix has been removed from the policy names in the security policies row.

Example: Security policy `ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04` is displayed as `TLS13-1-2-FIPS-2023-04`.

Security policies	TLS13-1-3-FIPS-2023-04	TLS13-1-2-Res-FIPS-2023-04	TLS13-1-2-FIPS-2023-04	TLS13-1-2-Ext0-FIPS-2023-04	TLS13-1-2-Ext1-FIPS-2023-04	TLS13-1-2-Ext2-FIPS-2023-04	TLS13-1-1-FIPS-2023-04	TLS13-1-0-FIPS-2023-04
TLS Protocols								
Protocol-TLSv1								✓
Protocol-TLSv1.1							✓	✓
Protocol-TLSv1.2		✓	✓	✓	✓	✓	✓	✓
Protocol-TLSv1.3	✓	✓	✓	✓	✓	✓	✓	✓
TLS Ciphers								
TLS_AES_128_GCM_SHA256	✓	✓	✓	✓	✓	✓	✓	✓
TLS_AES_256_GCM_SHA384	✓	✓	✓	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128-GCM-		✓	✓	✓	✓	✓	✓	✓

Security policies	TLS13-1-3-FIPS-2023-04	TLS13-1-2-Res-FIPS-2023-04	TLS13-1-2-FIPS-2023-04	TLS13-1-2-Ext0-FIPS-2023-04	TLS13-1-2-Ext1-FIPS-2023-04	TLS13-1-2-Ext2-FIPS-2023-04	TLS13-1-1-FIPS-2023-04	TLS13-1-0-FIPS-2023-04
SHA256								
ECDHE-RSA-AES128-GCM-SHA256		✓	✓	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA256			✓	✓	✓	✓	✓	✓
ECDHE-RSA-AES128-SHA256			✓	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA				✓		✓	✓	✓
ECDHE-RSA-AES128-SHA				✓		✓	✓	✓

Security policies	TLS13-1-3-FIPS-2023-04	TLS13-1-2-Res-FIPS-2023-04	TLS13-1-2-FIPS-2023-04	TLS13-1-2-Ext0-FIPS-2023-04	TLS13-1-2-Ext1-FIPS-2023-04	TLS13-1-2-Ext2-FIPS-2023-04	TLS13-1-1-FIPS-2023-04	TLS13-1-0-FIPS-2023-04
ECDHE-ECDSA-AES256-GCM-SHA384		✓	✓	✓	✓	✓	✓	✓
ECDHE-RSA-AES256-GCM-SHA384		✓	✓	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES256-SHA384			✓	✓	✓	✓	✓	✓
ECDHE-RSA-AES256-SHA384			✓	✓	✓	✓	✓	✓
ECDHE-RSA-AES256-SHA				✓		✓	✓	✓

Security policies	TLS13-1-3-FIPS-2023-04	TLS13-1-2-Res-FIPS-2023-04	TLS13-1-2-FIPS-2023-04	TLS13-1-2-Ext0-FIPS-2023-04	TLS13-1-2-Ext1-FIPS-2023-04	TLS13-1-2-Ext2-FIPS-2023-04	TLS13-1-1-FIPS-2023-04	TLS13-1-0-FIPS-2023-04
ECDHE-ECDSA-AES256-SHA				✓		✓	✓	✓
AES128-GCM-SHA256					✓	✓	✓	✓
AES128-SHA256					✓	✓	✓	✓
AES128-SHA						✓	✓	✓
AES256-GCM-SHA384					✓	✓	✓	✓
AES256-SHA256					✓	✓	✓	✓
AES256-SHA						✓	✓	✓

To create a TLS listener that uses a FIPS policy using the CLI

Use the create-listener command with any FIPS security policy.

The example uses the ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 security policy.

```
aws elbv2 create-listener --name my-listener \
--protocol TLS --port 443 \
--ssl-policy ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04
```

To modify a TLS listener to use a FIPS policy using the CLI

Use the modify-listener command with any FIPS security policy.

The example uses the ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 security policy.

```
aws elbv2 modify-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0 \
--ssl-policy ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04
```

To view the security policies used by a listener using the CLI

Use the describe-listener command with the arn of your listener.

```
aws elbv2 describe-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0
```

To view the configuration of a FIPS security policy using the CLI

Use the describe-ssl-policies command with any FIPS security policy.

The example uses the ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04 security policy.

```
aws elbv2 describe-ssl-policies \
--names ELBSecurityPolicy-TLS13-1-2-FIPS-2023-04
```

FS

The following table describes the supported TLS protocols and ciphers for the available FS supported security policies.

Note: The ELBSecurityPolicy- prefix has been removed from the policy names in the security policies row.

Example: Security policy ELBSecurityPolicy-FS-2018-06 is displayed as FS-2018-06.

Security policies	Default	FS-1-2-Res-2020-10	FS-1-2-Res-2019-08	FS-1-2-2019-08	FS-1-1-2019-08	FS-2018-06
TLS Protocols						
Protocol-TLSv1	✓					✓
Protocol-TLSv1.1	✓				✓	✓
Protocol-TLSv1.2	✓	✓	✓	✓	✓	✓
TLS Ciphers						
ECDHE-ECDSA-AES128-GCM-SHA256	✓	✓	✓	✓	✓	✓
ECDHE-RSA-AES128-GCM-SHA256	✓	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA256	✓		✓	✓	✓	✓

Security policies	Default	FS-1-2-Res-2020-10	FS-1-2-Res-2019-08	FS-1-2-2019-08	FS-1-1-2019-08	FS-2018-06
ECDHE-RSA-AES128-SHA256	✓		✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA	✓			✓	✓	✓
ECDHE-RSA-AES128-SHA	✓			✓	✓	✓
ECDHE-ECDSA-AES256-GCM-SHA384	✓	✓	✓	✓	✓	✓
ECDHE-RSA-AES256-GCM-SHA384	✓	✓	✓	✓	✓	✓

Security policies	Default	FS-1-2-Res-2020-10	FS-1-2-Res-2019-08	FS-1-2-2019-08	FS-1-1-2019-08	FS-2018-06
ECDHE-ECDSA-AES256-SHA384	✓		✓	✓	✓	✓
ECDHE-RSA-AES256-SHA384	✓		✓	✓	✓	✓
ECDHE-RSA-AES256-SHA	✓			✓	✓	✓
ECDHE-ECDSA-AES256-SHA	✓			✓	✓	✓
AES128-GCM-SHA256	✓					
AES128-SHA256	✓					
AES128-SHA	✓					

Security policies	Default	FS-1-2-Res-2020-10	FS-1-2-Res-2019-08	FS-1-2-2019-08	FS-1-1-2019-08	FS-2018-06
AES256-GCM-SHA384	✓					
AES256-SHA256	✓					
AES256-SHA	✓					

To create a TLS listener that uses a FS supported policy using the CLI

Use the create-listener command with any FS supported security policy.

The example uses the `ELBSecurityPolicy-FS-2018-06` security policy.

```
aws elbv2 create-listener --name my-listener \
--protocol TLS --port 443 \
--ssl-policy ELBSecurityPolicy-FS-2018-06
```

To modify a TLS listener to use a FS supported policy using the CLI

Use the modify-listener command with any FS supported security policy.

The example uses the `ELBSecurityPolicy-FS-2018-06` security policy.

```
aws elbv2 modify-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0 \
--ssl-policy ELBSecurityPolicy-FS-2018-06
```

To view the security policies used by a listener using the CLI

Use the describe-listener command with the `arn` of your listener.

```
aws elbv2 describe-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0
```

To view the configuration of a FS supported security policy using the CLI

Use the describe-ssl-policies command with any FS supported security policy.

The example uses the `ELBSecurityPolicy-FS-2018-06` security policy.

```
aws elbv2 describe-ssl-policies \
--names ELBSecurityPolicy-FS-2018-06
```

TLS 1.0 - 1.2

The following table describes the supported TLS protocols and ciphers for the available TLS 1.0-1.2 security policies.

Note: The `ELBSecurityPolicy-` prefix has been removed from the policy names in the security policies row.

Example: Security policy `ELBSecurityPolicy-TLS-1-2-Ext-2018-06` is displayed as `TLS-1-2-Ext-2018-06`.

Security policies	Default	TLS-1-2-Ext-2018-06	TLS-1-2-2017-01	TLS-1-1-2017-01	TLS-1-0-2015-04*
TLS Protocols					
Protocol-TLSv1	✓				✓

Security policies	Default	TLS-1-2-Ext-2018-06	TLS-1-2-2017-01	TLS-1-1-2017-01	TLS-1-0-2015-04*
Protocol-TLSv1.1	✓			✓	✓
Protocol-TLSv1.2	✓	✓	✓	✓	✓
TLS Ciphers					
ECDHE-ECDSA-AES128-GCM-SHA256	✓	✓	✓	✓	✓
ECDHE-RSA-AES128-GCM-SHA256	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA256	✓	✓	✓	✓	✓
ECDHE-RSA-AES128-SHA256	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES128-SHA	✓	✓		✓	✓

Security policies	Default	TLS-1-2-Ext-2018-06	TLS-1-2-2017-01	TLS-1-1-2017-01	TLS-1-0-2015-04*
ECDHE-RSA-AES128-SHA	✓	✓		✓	✓
ECDHE-ECDSA-AES256-GCM-SHA384	✓	✓	✓	✓	✓
ECDHE-RSA-AES256-GCM-SHA384	✓	✓	✓	✓	✓
ECDHE-ECDSA-AES256-SHA384	✓	✓	✓	✓	✓
ECDHE-RSA-AES256-SHA384	✓	✓	✓	✓	✓
ECDHE-RSA-AES256-SHA	✓	✓		✓	✓
ECDHE-ECDSA-AES256-SHA	✓	✓		✓	✓
AES128-GCM-SHA256	✓	✓	✓	✓	✓

Security policies	Default	TLS-1-2-Ext-2018-06	TLS-1-2-2017-01	TLS-1-1-2017-01	TLS-1-0-2015-04*
AES128-SHA256	✓	✓	✓	✓	✓
AES128-SHA	✓	✓		✓	✓
AES256-GCM-SHA384	✓	✓	✓	✓	✓
AES256-SHA256	✓	✓	✓	✓	✓
AES256-SHA	✓	✓		✓	✓
DES-CBC3-SHA					✓

* Do not use this policy unless you must support a legacy client that requires the DES-CBC3-SHA cipher, which is a weak cipher.

To create a TLS listener that uses a TLS 1.0-1.2 policy using the CLI

Use the create-listener command with any TLS 1.0-1.2 supported security policy.

The example uses the `ELBSecurityPolicy-2016-08` security policy.

```
aws elbv2 create-listener --name my-listener \
--protocol TLS --port 443 \
--ssl-policy ELBSecurityPolicy-2016-08
```

To modify a TLS listener to use a TLS 1.0-1.2 policy using the CLI

Use the modify-listener command with any TLS 1.0-1.2 supported security policy.

The example uses the `ELBSecurityPolicy-2016-08` security policy.

```
aws elbv2 modify-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0 \
--ssl-policy ELBSecurityPolicy-2016-08
```

To view the security policies used by a listener using the CLI

Use the describe-listener command with the `arn` of your listener.

```
aws elbv2 describe-listener \
--listener-arn arn:aws:elasticloadbalancing:us-east-1:123456789012:listener/app/my-load-balancer/abcdef01234567890/1234567890abcdef0
```

To view the configuration of a TLS 1.0-1.2 security policy using the CLI

Use the describe-ssl-policies command with any TLS 1.0-1.2 supported security policy.

The example uses the `ELBSecurityPolicy-2016-08` security policy.

```
aws elbv2 describe-ssl-policies \
--names ELBSecurityPolicy-2016-08
```

ALPN policies

Application-Layer Protocol Negotiation (ALPN) is a TLS extension that is sent on the initial TLS handshake hello messages. ALPN enables the application layer to negotiate which protocols should be used over a secure connection, such as HTTP/1 and HTTP/2.

When the client initiates an ALPN connection, the load balancer compares the client ALPN preference list with its ALPN policy. If the client supports a protocol from the ALPN policy, the load balancer establishes the connection based on the preference list of the ALPN policy. Otherwise, the load balancer does not use ALPN.

Supported ALPN Policies

The following are the supported ALPN policies:

`HTTP10nly`

Negotiate only HTTP/1.*. The ALPN preference list is http/1.1, http/1.0.

`HTTP20nly`

Negotiate only HTTP/2. The ALPN preference list is h2.

`HTTP20ptional`

Prefer HTTP/1.* over HTTP/2 (which can be useful for HTTP/2 testing). The ALPN preference list is http/1.1, http/1.0, h2.

`HTTP2Preferred`

Prefer HTTP/2 over HTTP/1.*. The ALPN preference list is h2, http/1.1, http/1.0.

None

Do not negotiate ALPN. This is the default.

Enable ALPN Connections

You can enable ALPN connections when you create or modify a TLS listener. For more information, see Add a listener and Update the ALPN policy.

Update a listener for your Network Load Balancer

You can update the listener protocol, listener port or the target group which receives traffic from the forwarding action. The default action, also known as the default rule, forwards requests to the selected target group.

If you change the protocol from TCP or UDP to TLS, you must specify a security policy and server certificate. If you change the protocol from TLS to TCP or UDP, the security policy and server certificate are removed.

When the target group for the default action of the listener is updated, new connections are routed to the newly configured target group. However, this has no effect on any active connections that were created prior to this change. These active connections remain associated to the target in the original target group for up to one hour if traffic is being sent, or up to when the idle-timeout period elapses if no traffic is sent, whichever occurs first. The parameter `Connection termination on deregistration` is not applied when updating the listener, as it's applied when deregistering targets.

To update your listener using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Choose the name of the load balancer to open its detail page.
4. On the **Listeners** tab, choose the text in the **Protocol:Port** column to open the detail page for the listener.
5. Choose **Edit**.
6. (Optional) Change the specified values for **Protocol** and **Port** as needed.
7. (Optional) Choose a different target group for **Default action**.
8. (Optional) Add, update, or remove tags as needed.
9. Choose **Save changes**.

To update your listener using the AWS CLI

Use the modify-listener command.

Update a TLS listener for your Network Load Balancer

After you create a TLS listener, you can replace the default certificate, add or remove certificates from the certificate list, update the security policy, or update the ALPN policy.

Tasks

- Replace the default certificate
- Add certificates to the certificate list
- Remove certificates from the certificate list
- Update the security policy
- Update the ALPN policy

Replace the default certificate

You can replace the default certificate for your TLS listener using the following procedure. For more information, see Default certificate.

To replace the default certificate using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, choose **Load Balancers**.
3. Choose the name of the load balancer to open its detail page.
4. On the **Listeners** tab, choose the text in the **Protocol:Port** column to open the detail page for the listener.
5. For **Default SSL certificate**, do one of the following:
   - If you created or imported a certificate using AWS Certificate Manager, choose **From ACM** and choose the certificate.
   - If you uploaded a certificate using IAM, choose **From IAM** and choose the certificate.
6. Choose **Save changes**.

To replace the default certificate using the AWS CLI

Use the modify-listener command with the **--certificates** option.

Add certificates to the certificate list

You can add certificates to the certificate list for your listener using the following procedure. When you first create a TLS listener, the certificate list is empty. You can add one or more certificates. You can optionally add the default certificate to ensure that this certificate is used with the SNI protocol even if it is replaced as the default certificate. For more information, see Certificate list.

To add certificates to the certificate list using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Choose the name of the load balancer to open its detail page.
4. On the **Listeners** tab, choose the text in the **Protocol:Port** column to open the detail page for the listener.
5. Select the check box for the listener and choose **Actions**, **Add SSL certificates for SNI**.
6. To add certificates that are already managed by ACM or IAM, select the check boxes for the certificates and choose **Include as pending below**.

7. If you have a certificate that isn't managed by ACM or IAM, choose **Import certificate**, complete the form, and choose **Import**.
8. Choose **Add pending certificates**.

To add a certificate to the certificate list using the AWS CLI

Use the add-listener-certificates command.

Remove certificates from the certificate list

You can remove certificates from the certificate list for a TLS listener using the following procedure. To remove the default certificate for a TLS listener, see Replace the default certificate.

To remove certificates from the certificate list using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Choose the name of the load balancer to open its detail page.
4. On the **Listeners** tab, choose the text in the **Protocol:Port** column to open the detail page for the listener.
5. Select the check box for the listener and choose **Actions**, **Add SSL certificates for SNI**.
6. Select the check boxes for the certificates and choose **Remove**.
7. When prompted for confirmation, enter **`confirm`** and choose **Remove**.

To remove a certificate from the certificate list using the AWS CLI

Use the remove-listener-certificates command.

Update the security policy

When you create a TLS listener, you can select the security policy that meets your needs. When a new security policy is added, you can update your TLS listener to use the new security policy. Network Load Balancers do not support custom security policies. For more information, see Security policies.

To update the security policy using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

2. In the navigation pane, choose **Load Balancers**.
3. Choose the name of the load balancer to open its detail page.
4. On the **Listeners** tab, choose the text in the **Protocol:Port** column to open the detail page for the listener.
5. Choose **Edit**.
6. For **Security policy**, choose a security policy.
7. Choose **Save changes**.

To update the security policy using the AWS CLI

Use the modify-listener command with the **--ssl-policy** option.

Update the ALPN policy

You can update the ALPN policy for your TLS listener using the following procedure. For more information, see ALPN policies.

To update the ALPN policy using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Choose the name of the load balancer to open its detail page.
4. On the **Listeners** tab, choose the text in the **Protocol:Port** column to open the detail page for the listener.
5. Choose **Edit**.
6. For **ALPN policy**, choose a policy to enable ALPN or choose **None** to disable ALPN.
7. Choose **Save changes**.

To update the ALPN policy using the AWS CLI

Use the modify-listener command with the **--alpn-policy** option.

Delete a listener for your Network Load Balancer

You can delete a listener at any time.

To delete a listener using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the check box for load balancer.
4. On the **Listeners** tab, select the check box for the listener, and then choose **Actions**, **Delete listener**.
5. When prompted for confirmation, enter **`confirm`** and choose **Delete**.

To delete a listener using the AWS CLI

Use the delete-listener command.

Target groups for your Network Load Balancers

Each *target group* is used to route requests to one or more registered targets. When you create a listener, you specify a target group for its default action. Traffic is forwarded to the target group specified in the listener rule. You can create different target groups for different types of requests. For example, create one target group for general requests and other target groups for requests to the microservices for your application. For more information, see Network Load Balancer components.

You define health check settings for your load balancer on a per target group basis. Each target group uses the default health check settings, unless you override them when you create the target group or modify them later on. After you specify a target group in a rule for a listener, the load balancer continually monitors the health of all targets registered with the target group that are in an Availability Zone enabled for the load balancer. The load balancer routes requests to the registered targets that are healthy. For more information, see Health checks for your target groups.

Contents

- Delete a target group

Routing configuration

By default, a load balancer routes requests to its targets using the protocol and port number that you specified when you created the target group. Alternatively, you can override the port used for routing traffic to a target when you register it with the target group.

Target groups for Network Load Balancers support the following protocols and ports:

- **Protocols**: TCP, TLS, UDP, TCP_UDP
- **Ports**: 1-65535

If a target group is configured with the TLS protocol, the load balancer establishes TLS connections with the targets using certificates that you install on the targets. The load balancer does not validate these certificates. Therefore, you can use self-signed certificates or certificates that have expired. Because the load balancer is in a virtual private cloud (VPC), traffic between the load balancer and the targets is authenticated at the packet level, so it is not at risk of man-in-the-middle attacks or spoofing even if the certificates on the targets are not valid.

The following table summarizes the supported combinations of listener protocol and target group settings.

Listener protocol	Target group protocol	Target group type	Health check protocol
TCP	TCP | TCP_UDP	instance | ip	HTTP | HTTPS | TCP
TCP	TCP	alb	HTTP | HTTPS
TLS	TCP | TLS	instance | ip	HTTP | HTTPS | TCP
UDP	UDP | TCP_UDP	instance | ip	HTTP | HTTPS | TCP
TCP_UDP	TCP_UDP	instance | ip	HTTP | HTTPS | TCP

Target type

When you create a target group, you specify its target type, which determines how you specify its targets. After you create a target group, you can't change its target type.

The following are the possible target types:

`instance`

The targets are specified by instance ID.

`ip`

The targets are specified by IP address.

`alb`

The target is an Application Load Balancer.

When the target type is `ip`, you can specify IP addresses from one of the following CIDR blocks:

- The subnets of the VPC for the target group
- 10.0.0.0/8 (RFC 1918)
- 100.64.0.0/10 (RFC 6598)
- 172.16.0.0/12 (RFC 1918)
- 192.168.0.0/16 (RFC 1918)

Important

You can't specify publicly routable IP addresses.

All of the supported CIDR blocks enable you to register the following targets with a target group:

- AWS resources that are addressable by IP address and port (for example, databases).
- On-premises resources linked to AWS through AWS Direct Connect or a Site-to-Site VPN connection.

When client IP preservation is disabled for your target groups, the load balancer can support about 55,000 connections per minute for each combination of Network Load Balancer IP address and unique target (IP address and port). If you exceed these connections, there is an increased chance of port allocation errors. If you get port allocation errors, add more targets to the target group.

When launching a Network Load Balancer in a shared Amazon VPC (as a participant), you can only register targets in subnets that have been shared with you.

When the target type is `alb`, you can register a single Application Load Balancer as a target. For more information, see Application Load Balancers as targets.

Network Load Balancers do not support the `lambda` target type. Application Load Balancers are the only load balancers that support the `lambda` target type. For more information, see Lambda functions as targets in the *User Guide for Application Load Balancers*.

If you have microservices on instances that are registered with a Network Load Balancer, you can't use the load balancer to provide communication between them unless the load balancer is internet-facing or the instances are registered by IP address. For more information, see Connections time out for requests from a target to its load balancer.

Request routing and IP addresses

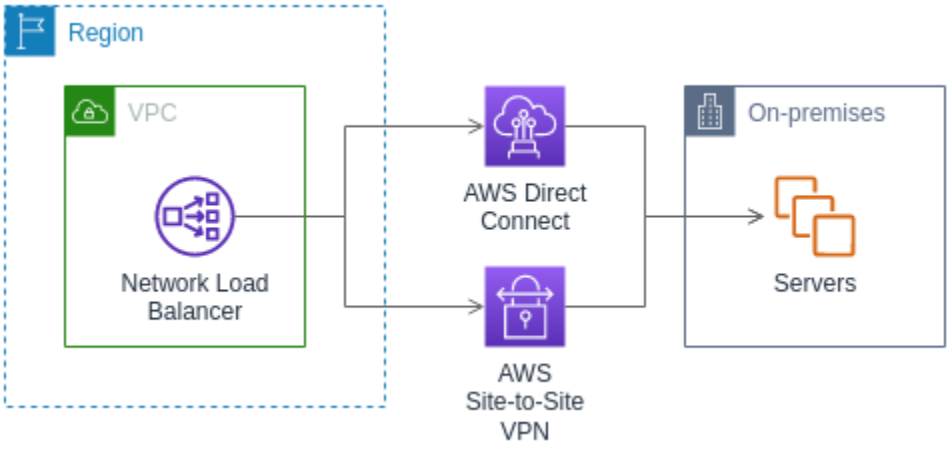
If you specify targets using an instance ID, traffic is routed to instances using the primary private IP address that is specified in the primary network interface for the instance. The load balancer rewrites the destination IP address from the data packet before forwarding it to the target instance.

If you specify targets using IP addresses, you can route traffic to an instance using any private IP address from one or more network interfaces. This enables multiple applications on an instance to use the same port. Note that each network interface can have its own security group. The load balancer rewrites the destination IP address before forwarding it to the target.

For more information about allowing traffic to your instances, see Target security groups.

On premises resources as targets

On premises resources linked through AWS Direct Connect or a Site-to-Site VPN connection can serve as a target, when the target type is `ip`.



When using on premises resources, the IP addresses of these targets must still come from one of the following CIDR blocks:

- 10.0.0.0/8 (RFC 1918)
- 100.64.0.0/10 (RFC 6598)
- 172.16.0.0/12 (RFC 1918)
- 192.168.0.0/16 (RFC 1918)

For more information about AWS Direct Connect, see What is AWS Direct Connect?

For more information about AWS Site-to-Site VPN, see What is AWS Site-to-Site VPN?

IP address type

When creating a new target group, you can select the IP address type of your target group. This controls the IP version used to communicate with targets and check their health status.

Network Load Balancers support both IPv4 and IPv6 target groups. The default selection is IPv4. IPv6 target groups can only be associated with dualstack Network Load Balancers.

Considerations

- All IP addresses within a target group must have the same IP address type. For example, you can't register an IPv4 target with an IPv6 target group.
- IPv6 target groups can only be used with `dualstack` load balancers with TCP or a TLS listeners.

- IPv6 target groups support IP and Instance type targets.

Registered targets

Your load balancer serves as a single point of contact for clients and distributes incoming traffic across its healthy registered targets. Each target group must have at least one registered target in each Availability Zone that is enabled for the load balancer. You can register each target with one or more target groups.

If demand on your application increases, you can register additional targets with one or more target groups in order to handle the demand. The load balancer starts routing traffic to a newly registered target as soon as the registration process completes and the target passes the first initial health check, irrespective of the configured threshold.

If demand on your application decreases, or if you need to service your targets, you can deregister targets from your target groups. Deregistering a target removes it from your target group, but does not affect the target otherwise. The load balancer stops routing traffic to a target as soon as it is deregistered. The target enters the `draining` state until in-flight requests have completed. You can register the target with the target group again when you are ready for it to resume receiving traffic.

If you are registering targets by instance ID, you can use your load balancer with an Auto Scaling group. After you attach a target group to an Auto Scaling group, Auto Scaling registers your targets with the target group for you when it launches them. For more information, see Attaching a load balancer to your Auto Scaling group in the *Amazon EC2 Auto Scaling User Guide*.

Requirements and considerations

- You can't register instances by instance ID if they use one of the following instance types: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3, or T1.
- When registering targets by instance ID for a IPv6 target group, the targets must have an assigned primary IPv6 address. To learn more, see IPv6 addresses in the *Amazon EC2 User Guide*
- When registering targets by instance ID, instances must be in the same Amazon VPC as the Network Load Balancer. You can't register instances by instance ID if they are in an VPC that is peered to the load balancer VPC (same Region or different Region). You can register these instances by IP address.
- If you register a target by IP address and the IP address is in the same VPC as the load balancer, the load balancer verifies that it is from a subnet that it can reach.

- The load balancer routes traffic to targets only in Availability Zones that are enabled. Targets in zones that are not enabled are unused.
- For UDP and TCP_UDP target groups, do not register instances by IP address if they reside outside of the load balancer VPC or if they use one of the following instance types: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3, or T1. Targets that reside outside the load balancer VPC or use an unsupported instance type might be able to receive traffic from the load balancer but then be unable to respond.

Target group attributes

The following target group attributes are supported. You can modify these attributes only if the target group type is `instance` or `ip`. If the target group type is `alb`, these attributes always use their default values.

`deregistration_delay.timeout_seconds`

The amount of time for Elastic Load Balancing to wait before changing the state of a deregistering target from `draining` to `unused`. The range is 0-3600 seconds. The default value is 300 seconds.

`deregistration_delay.connection_termination.enabled`

Indicates whether the load balancer terminates connections at the end of the deregistration timeout. The value is `true` or `false`. For new UDP/TCP_UDP target groups the default is `true`. Otherwise, the default is `false`.

`load_balancing.cross_zone.enabled`

Indicates whether cross zone load balancing is enabled. The value is `true`, `false` or `use_load_balancer_configuration`. The default is `use_load_balancer_configuration`.

`preserve_client_ip.enabled`

Indicates whether client IP preservation is enabled. The value is `true` or `false`. The default is disabled if the target group type is IP address and the target group protocol is TCP or TLS. Otherwise, the default is enabled. Client IP preservation can't be disabled for UDP and TCP_UDP target groups.

`proxy_protocol_v2.enabled`

Indicates whether proxy protocol version 2 is enabled. By default, proxy protocol is disabled.

`stickiness.enabled`

Indicates whether sticky sessions are enabled.

`stickiness.type`

The type of stickiness. The possible value is `source_ip`.

`target_group_health.dns_failover.minimum_healthy_targets.count`

The minimum number of targets that must be healthy. If the number of healthy targets is below this value, mark the zone as unhealthy in DNS, so that traffic is routed only to healthy zones. The possible values are `off`, or an integer from 1 to the maximum number of targets. When `off`, DNS fail away is disabled, meaning each target group independently contributes to DNS fail over. The default is 1.

`target_group_health.dns_failover.minimum_healthy_targets.percentage`

The minimum percentage of targets that must be healthy. If the percentage of healthy targets is below this value, mark the zone as unhealthy in DNS, so that traffic is routed only to healthy zones. The possible values are `off`, or an integer from 1 to 100. When `off`, DNS fail away is disabled, meaning each target group independently contributes to DNS fail over. The default is 1.

`target_group_health.unhealthy_state_routing.minimum_healthy_targets.count`

The minimum number of targets that must be healthy. If the number of healthy targets is below this value, send traffic to all targets, including unhealthy targets. The range is 1 to the maximum number of targets. The default is 1.

`target_group_health.unhealthy_state_routing.minimum_healthy_targets.percentage`

The minimum percentage of targets that must be healthy. If the percentage of healthy targets is below this value, send traffic to all targets, including unhealthy targets. The possible values are `off` or an integer from 1 to 100. The default is `off`.

`target_health_state.unhealthy.connection_termination.enabled`

Indicates whether the load balancer terminates connections to unhealthy targets. The value is `true` or `false`. The default is `true`.

`target_health_state.unhealthy.draining_interval_seconds`

The amount of time for Elastic Load Balancing to wait before changing the state of an unhealthy target from `unhealthy.draining` to `unhealthy`. The range is 0-360000 seconds. The default value is 0 seconds.

Note: This attribute can only be configured when `target_health_state.unhealthy.connection_termination.enabled` is `false`.

Client IP preservation

Network Load Balancers can preserve the source IP address of clients when routing requests to backend targets. When you disable client IP preservation, the source IP address is the private IP address of the Network Load Balancer.

By default, client IP preservation is enabled (and can't be disabled) for instance and IP type target groups with UDP and TCP_UDP protocols. However, you can enable or disable client IP preservation for TCP and TLS target groups using the `preserve_client_ip.enabled` target group attribute.

Default settings

- Instance type target groups: Enabled
- IP type target groups (UDP, TCP_UDP): Enabled
- IP type target groups (TCP, TLS): Disabled

Requirements and considerations

- When client IP preservation is enabled, the traffic must flow directly from the Network Load Balancer to the target. The target must be located in the same VPC as the Network Load Balancer or in a peered VPC in the same region.
- Client IP preservation is not supported when using a Gateway Load Balancer endpoint to inspect traffic between the Network Load Balancer and the target (instance or IP), even if the target is in the same Amazon VPC as the Network Load Balancer.
- The following instance types do not support client IP preservation: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3, and T1. We recommend that you register these instance types as IP addresses with client IP preservation disabled.
- Client IP preservation has no effect on inbound traffic from AWS PrivateLink. The source IP of the AWS PrivateLink traffic is always the private IP address of the Network Load Balancer.
- Client IP preservation is not supported when a target group contains AWS PrivateLink ENIs, or the ENI of another Network Load Balancer. This will cause loss of communication to those targets.

- Client IP preservation has no effect on traffic converted from IPv6 to IPv4. The source IP of this type of traffic is always the private IP address of the Network Load Balancer.
- When you specify targets by Application Load Balancer type, the client IP of all incoming traffic is preserved by the Network Load Balancer and is sent to the Application Load Balancer. The Application Load Balancer then appends the client IP to the `X-Forwarded-For` request header before sending it to the target.
- Client IP preservation changes take effect only for new TCP connections.
- NAT loopback, also known as hairpinning, is not supported when client IP preservation is enabled. When enabled you might encounter TCP/IP connection limitations related to observed socket reuse on the targets. These connection limitations can occur when a client, or a NAT device in front of the client, uses the same source IP address and source port when connecting to multiple load balancer nodes simultaneously. If the load balancer routes these connections to the same target, the connections appear to the target as if they come from the same source socket, which results in connection errors. If this happens, the clients can retry (if the connection fails) or reconnect (if the connection is interrupted). You can reduce this type of connection error by increasing the number of source ephemeral ports or by increasing the number of targets for the load balancer. You can prevent this type of connection error, by disabling client IP preservation or disabling cross-zone load balancing.
- When client IP preservation is disabled, a Network Load Balancer supports 55,000 simultaneous connections or about 55,000 connections per minute to each unique target (IP address and port). If you exceed these connections, there is an increased chance of port allocation errors, resulting in failures to establish new connections. Port allocation errors can be tracked using the `PortAllocationErrorCount` metric. To fix port allocation errors, add more targets to the target group. For more information, see CloudWatch metrics for your Network Load Balancer.

To configure client IP preservation using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. To enable client IP preservation, turn on **Preserve client IP addresses**. To disable client IP preservation, turn off **Preserve client IP addresses**.
6. Choose **Save changes**.

To enable or disable client IP preservation using the AWS CLI

Use the modify-target-group-attributes command with the `preserve_client_ip.enabled` attribute.

For example, use the following command to disable client IP preservation.

```
aws elbv2 modify-target-group-attributes --attributes
 Key=preserve_client_ip.enabled,Value=false --target-group-arn ARN
```

Your output should be similar to the following example.

```
{
    "Attributes": [
      {
        "Key": "proxy_protocol_v2.enabled",
        "Value": "false"
      },
      {
        "Key": "preserve_client_ip.enabled",
        "Value": "false"
      },
      {
        "Key": "deregistration_delay.timeout_seconds",
        "Value": "300"
      }
    ]
}
```

Deregistration delay

When you deregister a target, the load balancer stops creating new connections to the target. The load balancer uses connection draining to ensure that in-flight traffic completes on the existing connections. If the deregistered target stays healthy and an existing connection is not idle, the load balancer can continue to send traffic to the target. To ensure that existing connections are closed, you can do one of the following: enable the target group attribute for connection termination, ensure that the instance is unhealthy before you deregister it, or periodically close client connections.

The initial state of a deregistering target is `draining`. By default, the load balancer changes the state of a deregistering target to `unused` after 300 seconds. To change the amount of time that

the load balancer waits before changing the state of a deregistering target to unused, update the deregistration delay value. We recommend that you specify a value of at least 120 seconds to ensure that requests are completed.

If you enable the target group attribute for connection termination, connections to deregistered targets are closed shortly after the end of the deregistration timeout.

To update the deregistration attributes using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On **Attributes** tab, choose **Edit**.
5. To change the deregistration timeout, enter a new value for **Deregistration delay**. To ensure that existing connections are closed after you deregister targets, select **Terminate connections on deregistration**.
6. Choose **Save changes**.

To update the deregistration attributes using the AWS CLI

Use the modify-target-group-attributes command.

Proxy protocol

Network Load Balancers use proxy protocol version 2 to send additional connection information such as the source and destination. Proxy protocol version 2 provides a binary encoding of the proxy protocol header. With TCP listeners, the load balancer prepends a proxy protocol header to the TCP data. It does not discard or overwrite any existing data, including any incoming proxy protocol headers sent by the client or any other proxies, load balancers, or servers in the network path. Therefore, it is possible to receive more than one proxy protocol header. Also, if there is another network path to your targets outside of your Network Load Balancer, the first proxy protocol header might not be the one from your Network Load Balancer.

If you specify targets by IP address, the source IP addresses provided to your applications depend on the protocol of the target group as follows:

- TCP and TLS: By default, client IP preservation is disabled, and the source IP addresses provided to your applications are the private IP addresses of the load balancer nodes. To preserve the

client's IP address, ensure that the target is located within the same VPC or a peered VPC and enable client IP preservation. If you need the IP address of the client and these conditions are not met, enable the proxy protocol and get the client IP address from the proxy protocol header.

- UDP and TCP_UDP: The source IP addresses are the IP addresses of the clients, as client IP preservation is enabled by default for these protocols and cannot be disabled. If you specify targets by instance ID, the source IP addresses provided to your applications are the client IP addresses. However, if you prefer, you can enable proxy protocol and get the client IP addresses from the proxy protocol header.

If you specify targets by instance ID, the source IP addresses provided to your applications are the client IP addresses. However, if you prefer, you can enable proxy protocol and get the client IP addresses from the proxy protocol header.

> **Note**
>
> TLS listeners do not support incoming connections with proxy protocol headers sent by the client or any other proxies.

Health check connections

After you enable proxy protocol, the proxy protocol header is also included in health check connections from the load balancer. However, with health check connections, the client connection information is not sent in the proxy protocol header.

VPC endpoint services

For traffic coming from service consumers through a VPC endpoint service, the source IP addresses provided to your applications are the private IP addresses of the load balancer nodes. If your applications need the IP addresses of the service consumers, enable proxy protocol and get them from the proxy protocol header.

The proxy protocol header also includes the ID of the endpoint. This information is encoded using a custom Type-Length-Value (TLV) vector as follows.

Field	Length (in octets)	Description
Type	1	PP2_TYPE_AWS (0xEA)

Field	Length (in octets)	Description
Length	2	The length of value
Value	1	PP2_SUBTYPE_AWS_VPCE_ID (0x01)
	variable (value length minus 1)	The ID of the endpoint

For an example that parses TLV type 0xEA, see https://github.com/aws/elastic-load-balancing-tools/tree/master/proprot.

Enable proxy protocol

Before you enable proxy protocol on a target group, make sure that your applications expect and can parse the proxy protocol v2 header, otherwise, they might fail. For more information, see PROXY protocol versions 1 and 2.

To enable proxy protocol v2 using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name the target group to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. On the **Edit attributes** page, select **Proxy protocol v2**.
6. Choose **Save changes**.

To enable proxy protocol v2 using the AWS CLI

Use the modify-target-group-attributes command.

Sticky sessions

Sticky sessions are a mechanism to route client traffic to the same target in a target group. This is useful for servers that maintain state information in order to provide a continuous experience to clients.

Considerations

- Using sticky sessions can lead to an uneven distribution of connections and flows, which might impact the availability of your targets. For example, all clients behind the same NAT device have the same source IP address. Therefore, all traffic from these clients is routed to the same target.
- The load balancer might reset the sticky sessions for a target group if the health state of any of its targets changes or if you register or deregister targets with the target group.
- When the stickiness attribute is turned on for a target group, passive health checks are not supported. For more information, see Health checks for your target groups.
- Sticky sessions are not supported for TLS listeners.

To enable sticky sessions using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Under **Target selection configuration**, turn on **Stickiness**.
6. Choose **Save changes**.

To enable sticky sessions using the AWS CLI

Use the modify-target-group-attributes command with the `stickiness.enabled` attribute.

Create a target group for your Network Load Balancer

You register targets for your Network Load Balancer with a target group. By default, the load balancer sends requests to registered targets using the port and protocol that you specified for the target group. You can override this port when you register each target with the target group.

After you create a target group, you can add tags.

To route traffic to the targets in a target group, create a listener and specify the target group in the default action for the listener. For more information, see Listener rules. You can specify the same target group in multiple listeners, but these listeners must belong to the same Network Load

Balancer. To use a target group with a load balancer, you must verify that the target group is not in use by a listener for any other load balancer.

You can add or remove targets from your target group at any time. For more information, see Register targets with your target group. You can also modify the health check settings for your target group. For more information, see Modify the health check settings of a target group.

To create a target group using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Target Groups**.
3. Choose **Create target group**.
4. For the **Basic configuration** pane, do the following:
   a. For **Choose a target type**, select **Instances** to register targets by instance ID, **IP addresses** to register targets by IP address, or **Application Load Balancer** to register an Application Load Balancer as a target.
   b. For **Target group name**, enter a name for the target group. This name must be unique per Region per account, can have a maximum of 32 characters, must contain only alphanumeric characters or hyphens, and must not begin or end with a hyphen.
   c. For **Protocol**, choose a protocol as follows:
      - If the listener protocol is TCP, choose **TCP** or **TCP_UDP**.
      - If the listener protocol is TLS, choose **TCP** or **TLS**.
      - If the listener protocol is UDP, choose **UDP** or **TCP_UDP**.
      - If the listener protocol is TCP_UDP, choose **TCP_UDP**.
   d. (Optional) For **Port**, modify the default value as needed.
   e. For **IP address type**, choose **IPv4** or **IPv6**. This option is available only if the target type is **Instances** or **IP addresses** and the protocol is TCP or TLS.

      You must associate an IPv6 target group with a dualstack load balancer. All targets in the target group must have the same IP address type. You can't change the IP address type of a target group after you create it.
   f. For **VPC**, select the virtual private cloud (VPC) with the targets to register.
5. For the **Health checks** pane, modify the default settings as needed. For **Advanced health check settings**, choose the health check port, count, timeout, interval, and specify success

codes. If health checks consecutively exceed the **Unhealthy threshold** count, the load balancer takes the target out of service. If health checks consecutively exceed the **Healthy threshold** count, the load balancer puts the target back in service. For more information, see Health checks for your target groups.

6. (Optional) To add a tag, expand **Tags**, choose **Add tag**, and enter a tag key and a tag value.
7. Choose **Next**.
8. On the **Register targets** page, add one or more targets as follows:
   - If the target type is **Instances**, select the instances, enter the ports, and then choose **Include as pending below**.

     Note: The instances must have an assigned primary IPv6 address to be registered with a IPv6 target group.
   - If the target type is **IP addresses**, select the network, enter the IP addresses and ports, and then choose **Include as pending below**.
9. Choose **Create target group**.

To create a target group using the AWS CLI

Use the create-target-group command to create the target group, the add-tags command to tag your target group, and the register-targets command to add targets.

Health checks for your target groups

You register your targets with one or more target groups. The load balancer starts routing requests to a newly registered target as soon as the registration process completes. It can take a few minutes for the registration process to complete and health checks to start.

Network Load Balancers use active and passive health checks to determine whether a target is available to handle requests. By default, each load balancer node routes requests only to the healthy targets in its Availability Zone. If you enable cross-zone load balancing, each load balancer node routes requests to the healthy targets in all enabled Availability Zones. For more information, see Cross-zone load balancing.

With passive health checks, the load balancer observes how targets respond to connections. Passive health checks enable the load balancer to detect an unhealthy target before it is reported as unhealthy by the active health checks. You cannot disable, configure, or monitor passive health

checks. Passive health checks are not supported for UDP traffic, and target groups with stickiness turned on. For more information, see Sticky sessions.

If a target becomes unhealthy, the load balancer sends a TCP RST for packets received on the client connections associated with the target, unless the unhealthy target triggers the load balancer to fail open.

If target groups don't have a healthy target in an enabled Availability Zone, we remove the IP address for the corresponding subnet from DNS so that requests cannot be routed to targets in that Availability Zone. If all targets fail health checks at the same time in all enabled Availability Zones, the load balancer fails open. Network Load Balancers will also fail open when you have an empty target group. The effect of the fail open is to allow traffic to all targets in all enabled Availability Zones, regardless of their health status.

If a target group is configured with HTTPS health checks, its registered targets fail health checks if they support only TLS 1.3. These targets must support an earlier version of TLS, such as TLS 1.2.

For HTTP or HTTPS health check requests, the host header contains the IP address of the load balancer node and the listener port, not the IP address of the target and the health check port.

If you add a TLS listener to your Network Load Balancer, we perform a listener connectivity test. As TLS termination also terminates a TCP connection, a new TCP connection is established between your load balancer and your targets. Therefore, you might see the TCP connections for this test sent from your load balancer to the targets that are registered with your TLS listener. You can identify these TCP connections because they have the source IP address of your Network Load Balancer and the connections do not contain data packets.

For a UDP service, target availability can be tested using non-UDP health checks on your target group. You can use any available health check (TCP, HTTP, or HTTPS), and any port on your target to verify the availability of a UDP service. If the service receiving the health check fails, your target is considered unavailable. To improve the accuracy of health checks for a UDP service, configure the service listening to the health check port to track the status of your UDP service and fail the health check if the service is unavailable.

Health check settings

You configure active health checks for the targets in a target group using the following settings. If the health checks exceed **UnhealthyThresholdCount** consecutive failures, the load balancer takes the target out of service. When the health checks exceed **HealthyThresholdCount** consecutive successes, the load balancer puts the target back in service.

Setting	Description	Default
HealthCheckProtocol	The protocol the load balancer uses when performing health checks on targets. The possible protocols are HTTP, HTTPS, and TCP. The default is the TCP protocol. If the target type is `alb`, the supported health check protocols are HTTP and HTTPS.	TCP
HealthCheckPort	The port the load balancer uses when performing health checks on targets. The default is to use the port on which each target receives traffic from the load balancer.	Port on which each target receives traffic from the load balancer.
HealthCheckPath	[HTTP/HTTPS health checks] The health check path that is the destination on the targets for health checks. The default is /.	/
HealthCheckTimeoutSeconds	The amount of time, in seconds, during which no response from a target means a failed health check. The range is 2–120 seconds. The default values are 6 seconds for HTTP and 10 seconds for TCP and HTTPS health checks.	6 seconds for HTTP health checks and 10 seconds for TCP and HTTPS health checks.
HealthCheckIntervalSeconds	The approximate amount of time, in seconds, between health checks of an individual target. The range is 5–300 seconds. The default is 30 seconds.	30 seconds

Setting	Description	Default
	Important Health checks for a Network Load Balancer are distributed and use a consensus mechanism to determine target health. Therefore, targets receive more than the configured number of health checks. To reduce the impact to your targets if you are using HTTP health checks, use a simpler destination on the targets, such as a static HTML file, or switch to TCP health checks.	
HealthyThresholdCount	The number of consecutive successful health checks required before considering an unhealthy target healthy. The range is 2–10. The default is 5.	5
UnhealthyThresholdCount	The number of consecutive failed health checks required before considering a target unhealthy. The range is 2–10. The default is 2.	2
Matcher	[HTTP/HTTPS health checks] The HTTP codes to use when checking for a successful response from a target. The range is 200 to 599. The default is 200-399.	200-399

Target health status

Before the load balancer sends a health check request to a target, you must register it with a target group, specify its target group in a listener rule, and ensure that the Availability Zone of the target is enabled for the load balancer.

The following table describes the possible values for the health status of a registered target.

Value	Description
`initial`	The load balancer is in the process of registering the target or performing the initial health checks on the target. Related reason codes: `Elb.RegistrationInProgress` | `Elb.InitialHealthChecking`
`healthy`	The target is healthy. Related reason codes: None
`unhealthy`	The target did not respond to a health check, failed the health check, or the target is in stopped state. Related reason code: `Target.FailedHealthChecks`
`draining`	The target is deregistering and connection draining is in process. Related reason code: `Target.DeregistrationInProgress`
`unhealthy.draining`	The target did not respond to health checks or failed the health checks and enters a grace period. The target supports existing connections and will not accept any new connections during this grace period. Related reason code: `Target.FailedHealthChecks`
`unavailable`	Target health is unavailable. Related reason code: `Elb.InternalError`
`unused`	The target is not registered with a target group, the target group is not used in a listener rule, or the target is in an Availability Zone that is not enabled.

Value	Description
	Related reason codes: `Target.NotRegistered` | `Target.NotInUse` | `Target.InvalidState` | `Target.IpUnusable`

Health check reason codes

If the status of a target is any value other than `Healthy`, the API returns a reason code and a description of the issue, and the console displays the same description in a tooltip. Note that reason codes that begin with `Elb` originate on the load balancer side and reason codes that begin with `Target` originate on the target side.

Reason code	Description
`Elb.InitialHealthChecking`	Initial health checks in progress
`Elb.InternalError`	Health checks failed due to an internal error
`Elb.RegistrationInProgress`	Target registration is in progress
`Target.DeregistrationInProgress`	Target deregistration is in progress
`Target.FailedHealthChecks`	Health checks failed
`Target.InvalidState`	Target is in the stopped state Target is in the terminated state Target is in the terminated or stopped state Target is in an invalid state
`Target.IpUnusable`	The IP address cannot be used as a target, as it is in use by a load balancer

Reason code	Description
`Target.NotInUse`	Target group is not configured to receive traffic from the load balancer Target is in an Availability Zone that is not enabled for the load balancer
`Target.NotRegistered`	Target is not registered to the target group

Check the health of your targets

You can check the health status of the targets registered with your target groups.

To check the health of your targets using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. The **Details** pane displays the total number of targets, plus the number of targets for each health status.
5. On the **Targets** tab, the **Health status** column indicates the status of each target.
6. If the status of a target is any value other than `Healthy`, the **Health status details** column contains more information.

To check the health of your targets using the AWS CLI

Use the describe-target-health command. The output of this command contains the target health state. It includes a reason code if the status is any value other than `Healthy`.

To receive email notifications about unhealthy targets

Use CloudWatch alarms to trigger a Lambda function to send details about unhealthy targets. For step-by-step instructions, see the following blog post: Identifying unhealthy targets of your load balancer.

Modify the health check settings of a target group

You can modify the health check settings for your target group at any time.

To modify health check settings for a target group using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On the **Health checks** tab, choose **Edit**.
5. On the **Edit health check settings** page, modify the settings as needed, and then choose **Save changes**.

To modify health check settings for a target group using the AWS CLI

Use the modify-target-group command.

Cross-zone load balancing for target groups

The nodes for your load balancer distribute requests from clients to registered targets. When cross-zone load balancing is on, each load balancer node distributes traffic across the registered targets in all registered Availability Zones. When cross-zone load balancing is off, each load balancer node distributes traffic across only the registered targets in its Availability Zone. This could be used if zonal failure domains are preferred over regional, ensuring that a healthy zone isn't impacted by an unhealthy zone, or for overall latency improvements.

With Network Load Balancers, cross-zone load balancing is off by default at the load balancer level, but you can turn it on at any time. For target groups, the default is to use the load balancer setting, but you can override the default by explicitly turning cross-zone load balancing on or off at the target group level.

Considerations

- When enabling cross-zone load balancing for a Network Load Balancer, EC2 data transfer charges apply. For more information, see Understanding data transfer charges in the *AWS Data Exports User Guide*

- The target group setting determines the load balancing behavior for the target group. For example, if cross-zone load balancing is enabled at the load balancer level and disabled at the target group level, traffic sent to the target group is not routed across Availability Zones.
- When cross-zone load balancing is off, ensure that you have enough target capacity in each of the load balancer Availability Zones, so that each zone can serve its associated workload.
- When cross-zone load balancing is off, ensure that all target groups participate in the same Availability Zones. An empty Availability Zone is considered unhealthy.

Modify cross-zone load balancing for a load balancer

You can turn cross-zone load balancing on or off at the load balancer level at any time.

To modify cross-zone load balancing for a load balancer using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Load Balancers**.
3. Select the name of the load balancer to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. On the **Edit load balancer attributes** page, turn **Cross-zone load balancing** on or off.
6. Choose **Save changes**.

To modify cross-zone load balancing for your load balancer using the AWS CLI

Use the modify-load-balancer-attributes command with the `load_balancing.cross_zone.enabled` attribute.

Modify cross-zone load balancing for a target group

The cross-zone load balancing setting at the target group level overrides the setting at the load balancer level.

You can turn cross-zone load balancing on or off at the target group level if the target group type is `instance` or `ip`. If the target group type is `alb`, the target group always inherits the cross-zone load balancing setting from the load balancer.

To modify cross-zone load balancing for a target group using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, select **Target Groups**.
3. Select the name of the target group to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. On the **Edit target group attributes** page, select **On** for **Cross-zone load balancing**.
6. Choose **Save changes**.

To modify cross-zone load balancing for a target group using the AWS CLI

Use the modify-target-group-attributes command with the `load_balancing.cross_zone.enabled` attribute.

Target group health

By default, a target group is considered healthy as long as it has at least one healthy target. If you have a large fleet, having only one healthy target serving traffic is not sufficient. Instead, you can specify a minimum count or percentage of targets that must be healthy, and what actions the load balancer takes when the healthy targets fall below the specified threshold. This improves availability.

Unhealthy state actions

You can configure healthy thresholds for the following actions:

- DNS failover – When the healthy targets in a zone fall below the threshold, we mark the IP addresses of the load balancer node for the zone as unhealthy in DNS. Therefore, when clients resolve the load balancer DNS name, the traffic is routed only to healthy zones.
- Routing failover – When the healthy targets in a zone fall below the threshold, the load balancer sends traffic to all targets that are available to the load balancer node, including unhealthy targets. This increases the chances that a client connection succeeds, especially when targets temporarily fail to pass health checks, and reduces the risk of overloading the healthy targets.

Requirements and considerations

- If you specify both types of thresholds for an action (count and percentage), the load balancer takes the action when either threshold is breached.
- If you specify thresholds for both actions, the threshold for DNS failover must be greater than or equal to the threshold for routing failover, so that DNS failover occurs either with or before routing failover.
- If you specify the threshold as a percentage, we calculate the value dynamically, based on the total number of targets that are registered with the target groups.
- The total number of targets is based on whether cross-zone load balancing is off or on. If cross-zone load balancing is off, each node sends traffic only to the targets in its own zone, which means that the thresholds apply to the number of targets in each enabled zone separately. If cross-zone load balancing is on, each node sends traffic to all targets in all enabled zones, which means that the specified thresholds apply to the total number targets in all enabled zones. For more information, see Cross-zone load balancing.
- With DNS failover, we remove the IP addresses for the unhealthy zones from the DNS hostname for the load balancer. However, the local client DNS cache might contain these IP addresses until the time-to-live (TTL) in the DNS record expires (60 seconds).
- When DNS failover occurs, this impacts all target groups associated with the load balancer. Ensure that you have enough capacity in your remaining zones to handle this additional traffic, especially if cross-zone load balancing is off.
- With DNS failover, if all load balancer zones are considered unhealthy, the load balancer sends traffic to all zones, including the unhealthy zones.
- There are factors other than whether there are enough healthy targets that might lead to DNS failover, such as the health of the zone.

Example

The following example demonstrates how target group health settings are applied.

Scenario

- A load balancer that supports two Availability Zones, A and B
- Each Availability Zone contains 10 registered targets
- The target group has the following target group health settings:

  - DNS failover - 50%
  - Routing failover - 50%
- Six targets fail in Availability Zone B

If cross-zone load balancing is off

- The load balancer node in each Availability Zone can send traffic only to the 10 targets in its Availability Zone.
- There are 10 healthy targets in Availability Zone A, which meets the required percentage of healthy targets. The load balancer continues to distribute traffic between the 10 healthy targets.
- There are only 4 healthy targets in Availability Zone B, which is 40% of the targets for the load balancer node in Availability Zone B. Because this is less than the required percentage of healthy targets, the load balancer takes the following actions:
  - DNS failover - Availability Zone B is marked as unhealthy in DNS. Because clients can't resolve the load balancer name to the load balancer node in Availability Zone B, and Availability Zone A is healthy, clients send new connections to Availability Zone A.
  - Routing failover - When new connections are sent explicitly to Availability Zone B, the load balancer distributes traffic to all targets in Availability Zone B, including the unhealthy targets. This prevents outages among the remaining healthy targets.

If cross-zone load balancing is on

- Each load balancer node can send traffic to all 20 registered targets across both Availability Zones.
- There are 10 healthy targets in Availability Zone A and 4 healthy targets in Availability Zone B, for a total of 14 healthy targets. This is 70% of the targets for the load balancer nodes in both Availability Zones, which meets the required percentage of healthy targets.
- The load balancer distributes traffic between the 14 healthy targets in both Availability Zones.

Modify target group health settings

You can modify the target group health settings for your target group as follows.

To modify target group health settings using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Check whether cross-zone load balancing is turned on or turned off. Update this setting as needed to ensure that you have enough capacity to handle the additional traffic if a zone fails.
6. Expand **Target group health requirements**.
7. For **Configuration type**, we recommend that you choose **Unified configuration**, which sets the same threshold for both actions.
8. For **Healthy state requirements**, do one of the following:
   - Choose **Minimum healthy target count**, and then enter a number from 1 to the maximum number of targets for your target group.
   - Choose **Minimum healthy target percentage**, and then enter a number from 1 to 100.
9. Choose **Save changes**.

To modify target group health settings using the AWS CLI

Use the modify-target-group-attributes command. The following example sets the healthy threshold for both unhealthy state actions to 50%.

```
aws elbv2 modify-target-group-attributes \
--target-group-arn arn:aws:elasticloadbalancing:region:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \
--attributes
 Key=target_group_health.dns_failover.minimum_healthy_targets.percentage,Value=50 \

 Key=target_group_health.unhealthy_state_routing.minimum_healthy_targets.percentage,Value=50
```

Connection termination for unhealthy targets

Connection termination is enabled by default. When the target of a Network Load Balancer fails the configured health checks and is deemed unhealthy, the load balancer terminates established connections and stops routing new connections to the target. With connection termination

disabled the target is still considered unhealthy and won't receive new connections, but established connections are kept active, allowing them to gracefully close.

Connection termination for unhealthy targets can be set individually for each target group.

To modify the connection termination setting using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Under **Target unhealthy state management**, choose whether **Terminate connections when targets become unhealthy** is enabled or disabled.
6. Choose **Save changes**.

To modify the connection termination setting using the AWS CLI

Use the modify-target-group-attributes command with the `target_health_state.unhealthy.connection_termination.enabled` attribute.

Unhealthy draining interval

> **Important**
>
> Connection termination must be disabled before enabling unhealthy draining interval.

Targets in the `unhealthy.draining` state are considered unhealthy, do not receive new connections, but retain established connections for the configured interval. The unhealthy connection interval determines the amount of time the target remains in the `unhealthy.draining` state before it's state becomes `unhealthy`. If the target passes health checks during the unhealthy connection interval, it's state becomes `healthy` again. If a deregistration is triggered, the targets state becomes `draining` and the deregistration delay timeout begins.

The unhealthy draining interval can be set individually for each target group.

To modify the unhealthy draining interval using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. Under **Target unhealthy state management**, make sure **Terminate connections when targets become unhealthy** is turned off.
6. Enter a value for **Unhealthy draining interval**.
7. Choose **Save changes**.

To modify the unhealthy draining interval using the AWS CLI

Use the modify-target-group-attributes command with the `target_health_state.unhealthy.draining_interval_seconds` attribute.

Using Route 53 DNS failover for your load balancer

If you use Route 53 to route DNS queries to your load balancer, you can also configure DNS failover for your load balancer using Route 53. In a failover configuration, Route 53 checks the health of the target group targets for the load balancer to determine whether they are available. If there are no healthy targets registered with the load balancer, or if the load balancer itself is unhealthy, Route 53 routes traffic to another available resource, such as a healthy load balancer or a static website in Amazon S3.

For example, suppose that you have a web application for `www.example.com`, and you want redundant instances running behind two load balancers residing in different Regions. You want the traffic to be primarily routed to the load balancer in one Region, and you want to use the load balancer in the other Region as a backup during failures. If you configure DNS failover, you can specify your primary and secondary (backup) load balancers. Route 53 directs traffic to the primary load balancer if it is available, or to the secondary load balancer otherwise.

Using evaluate target health

- When evaluate target health is set to `Yes` on an alias record for a Network Load Balancer, Route 53 evaluates the health of the resource specified by the `alias target` value. For a Network Load Balancer, Route 53 uses the target group health checks associated with the load balancer.

- When all the target groups in a Network Load Balancer are healthy, Route 53 marks the alias record as healthy. If a target group contains at least one healthy target, the target group health check passes. Route 53 then returns records according to your routing policy. If the failover routing policy is used, Route 53 returns the primary record.
- If any of the target groups in a Network Load Balancer are unhealthy, the alias record fails the Route 53 health check (fail-open). If using evaluate target health, this would fail the failover routing policy.
- If all of the target groups in a Network Load Balancer are empty (no targets), then Route 53 considers the record unhealthy (fail-open). If using evaluate target health, this would fail the failover routing policy.

For more information, see Configuring DNS failover in the *Amazon Route 53 Developer Guide*.

Register targets with your target group

When your target is ready to handle requests, you register it with one or more target groups. The target type of the target group determines how you register targets. For example, you can register instance IDs, IP addresses, or an Application Load Balancer. Your Network Load Balancer starts routing requests to targets as soon as the registration process completes and the targets pass the initial health checks. It can take a few minutes for the registration process to complete and health checks to start. For more information, see Health checks for your target groups.

If demand on your currently registered targets increases, you can register additional targets in order to handle the demand. If demand on your registered targets decreases, you can deregister targets from your target group. It can take a few minutes for the deregistration process to complete and for the load balancer to stop routing requests to the target. If demand increases subsequently, you can register targets that you deregistered with the target group again. If you need to service a target, you can deregister it and then register it again when servicing is complete.

When you deregister a target, Elastic Load Balancing waits until in-flight requests have completed. This is known as *connection draining*. The status of a target is `draining` while connection draining is in progress. After deregistration is complete, status of the target changes to `unused`. For more information, see Deregistration delay.

If you are registering targets by instance ID, you can use your load balancer with an Auto Scaling group. After you attach a target group to an Auto Scaling group and the group scales out, the instances launched by the Auto Scaling group are automatically registered with the target group.

If you detach the load balancer from the Auto Scaling group, the instances are automatically deregistered from the target group. For more information, see Attaching a load balancer to your Auto Scaling group in the *Amazon EC2 Auto Scaling User Guide*.

Target security groups

Before adding targets to your target group, configure the security groups associated with the targets to accept traffic from your Network Load Balancer.

Recommendations for target security groups if the load balancer has an associated security group

- **To allow client traffic**: Add a rule that references the security group associated with the load balancer.
- **To allow PrivateLink traffic**: If you configured the load balancer to evaluate inbound rules for traffic sent through AWS PrivateLink, add a rule that accepts traffic from the load balancer security group on the traffic port. Otherwise, add a rule that accepts traffic from the load balancer private IP addresses on the traffic port.
- **To accept load balancer health checks**: Add a rule that accepts health check traffic from the load balancer security groups on the health check port.

Recommendations for target security groups if the load balancer is not associated with a security group

- **To allow client traffic**: If your load balancer preserves client IP addresses, add a rule that accepts traffic from the IP addresses of approved clients on the traffic port. Otherwise, add a rule that accepts traffic from the load balancer private IP addresses on the traffic port.
- **To allow PrivateLink traffic**: Add a rule that accepts traffic from the load balancer private IP addresses on the traffic port.
- **To accept load balancer health checks**: Add a rule that accepts health check traffic from the load balancer private IP addresses on the health check port.

How client IP preservation works

Network Load Balancers don't preserve client IP addresses unless you set the `preserve_client_ip.enabled` attribute to `true`. Also, with dualstack Network Load Balancers,

we preserve client IP addresses when translating IPv4 addresses to IPv6. However, when translating IPv6 addresses to IPv4 the source IP is always the private IP address of the Network Load Balancer.

To find the load balancer private IP addresses using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Network Interfaces**.
3. In the search field, enter the name of your Network Load Balancer. There is one network interface per load balancer subnet.
4. On the **Details** tab for each network interface, copy the address from **Private IPv4 address**.

For more information, see Security groups for your Network Load Balancer.

Network ACLs

When you register EC2 instances as targets, you must ensure that the network ACLs for the subnets for your instances allow traffic on both the listener port and the health check port. The default network access control list (ACL) for a VPC allows all inbound and outbound traffic. If you create custom network ACLs, verify that they allow the appropriate traffic.

The network ACLs associated with the subnets for your instances must allow the following traffic for an internet-facing load balancer.

Recommended rules for instance subnets

Inbound			
Source	**Protocol**	**Port Range**	**Comment**
Client IP addresses	*listener*	*target port*	Allow client traffic (IP Preservation: ON)
VPC CIDR	*listener*	*target port*	Allow client traffic (IP Preservation: OFF)
VPC CIDR	*health check*	*health check*	Allow health check traffic
Outbound			

Destination	Protocol	Port Range	Comment
Client IP addresses	*listener*	1024-65535	Allow return traffic to client (IP Preservation: ON)
VPC CIDR	*listener*	1024-65535	Allow return traffic to client (IP Preservation: OFF)
VPC CIDR	*health check*	1024-65535	Allow health check traffic

The network ACLs associated with the subnets for your load balancer must allow the following traffic for an internet-facing load balancer.

Recommended rules for load balancer subnets

Inbound			
Source	**Protocol**	**Port Range**	**Comment**
Client IP addresses	*listener*	*listener*	Allow client traffic
VPC CIDR	*listener*	1024-65535	Allow response from target
VPC CIDR	*health check*	1024-65535	Allow health check traffic
Outbound			
Destination	**Protocol**	**Port Range**	**Comment**
Client IP addresses	*listener*	1024-65535	Allow responses to clients
VPC CIDR	*listener*	*target port*	Allow requests to targets

VPC CIDR	*health check*	*health check*	Allow health check to targets

For an internal load balancer, the network ACLs for the subnets for your instances and load balancer nodes must allow both inbound and outbound traffic to and from the VPC CIDR, on the listener port and ephemeral ports.

Shared subnets

Participants can create a Network Load Balancer in a shared VPC. Participants can't register a target that runs in a subnet that is not shared with them.

Shared subnets for Network Load Balancers is supported in all AWS Regions, excluding:

- Asia Pacific (Osaka) `ap-northeast-3`
- Asia Pacific (Hong Kong) `ap-east-1`
- Middle East (Bahrain) `me-south-1`
- AWS China (Beijing) `cn-north-1`
- AWS China (Ningxia) `cn-northwest-1`

Register or deregister targets

Each target group must have at least one registered target in each Availability Zone that is enabled for the load balancer.

The target type of your target group determines how you register targets with that target group. For more information, see Target type.

Requirements and considerations

- You can't register instances by instance ID if they use one of the following instance types: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3, or T1.
- When registering targets by instance ID for a IPv6 target group, the targets must have an assigned primary IPv6 address. To learn more, see IPv6 addresses in the *Amazon EC2 User Guide*
- When registering targets by instance ID, instances must be in the same Amazon VPC as the Network Load Balancer. You can't register instances by instance ID if they are in an VPC that

is peered to the load balancer VPC (same Region or different Region). You can register these instances by IP address.

- If you register a target by IP address and the IP address is in the same VPC as the load balancer, the load balancer verifies that it is from a subnet that it can reach.
- For UDP and TCP_UDP target groups, do not register instances by IP address if they reside outside of the load balancer VPC or if they use one of the following instance types: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3, or T1. Targets that reside outside the load balancer VPC or use an unsupported instance type might be able to receive traffic from the load balancer but then be unable to respond.

Contents

Register or deregister targets by instance ID

An instance must be in the `running` state when you register it.

To register or deregister targets by instance ID using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. Choose the **Targets** tab.
5. To register instances, choose **Register targets**. Select one or more instances, enter the default instance port as needed, and then choose **Include as pending below**. When you are finished adding instances, choose **Register pending targets**.

   Note:

   - The instances must have an assigned primary IPv6 address to be registered with a IPv6 target group.

- AWS GovCloud (US) Regions don't support assigning a primary IPv6 address using the console. You must use the API to assign primary IPv6 addresses in AWS GovCloud (US) Regions.

6. To deregister instances, select the instance and then choose **Deregister**.

Register or deregister targets by IP address

IPv4 targets

An IP address that you register must be from one of the following CIDR blocks:

- The subnets of the VPC for the target group
- 10.0.0.0/8 (RFC 1918)
- 100.64.0.0/10 (RFC 6598)
- 172.16.0.0/12 (RFC 1918)
- 192.168.0.0/16 (RFC 1918)

The IP address type can't be changed after the target group is created.

When launching a Network Load Balancer in a shared Amazon VPC as a participant, you can only register targets in subnets that have been shared with you.

IPv6 targets

- The IP addresses that you register must be within the VPC CIDR block or within a peered VPC CIDR block.
- The IP address type can't be changed after the target group is created.
- You can associate IPv6 target groups only to a dualstack load balancer with TCP or a TLS listeners.

To register or deregister targets by IP address using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. Choose the **Targets** tab.

5. To register IP addresses, choose **Register targets**. For each IP address, select the network, Availability Zone, IP address (IPv4 or IPv6), and port, and then choose **Include as pending below**. When you are finished specifying addresses, choose **Register pending targets**.
6. To deregister IP addresses, select the IP addresses and then choose **Deregister**. If you have many registered IP addresses, you might find it helpful to add a filter or change the sort order.

Register or deregister targets using the AWS CLI

Use the register-targets command to add targets and the deregister-targets command to remove targets.

Application Load Balancers as targets

You can create a target group with a single Application Load Balancer as the target, and configure your Network Load Balancer to forward traffic to it. In this scenario, the Application Load Balancer takes over the load balancing decision as soon as traffic reaches it. This configuration combines the features of both load balancers and offers the following advantages:

- You can use the layer 7 request-based routing feature of the Application Load Balancer in combination with features that the Network Load Balancer supports, such as endpoint services (AWS PrivateLink) and static IP addresses.
- You can use this configuration for applications that need a single endpoint for multi-protocols, such as media services using HTTP for signaling and RTP to stream content.

You can use this feature with an internal or internet-facing Application Load Balancer as the target of an internal or internet-facing Network Load Balancer.

Considerations

- To associate an Application Load Balancer as a target of a Network Load Balancer, they must be in the same Amazon VPC within the same account.
- You can associate an Application Load Balancer as a target of multiple Network Load Balancers. To do this, register the Application Load Balancer with a separate target group for each individual Network Load Balancer.
- Each Application Load Balancer that you register with a Network Load Balancer decreases the maximum number of targets per Availability Zone per Network Load Balancer by 50 (if cross-

zone load balancing is disabled) or 100 (if cross-zone load balancing is enabled). You can disable cross-zone load balancing in both load balancers to minimize latency and avoid Regional data transfer charges. For more information, see Quotas for your Network Load Balancers.

- When the target group type is `alb`, you can't modify the target group attributes. These attributes always use their default values.
- After you register an Application Load Balancer as a target, you can't delete the Application Load Balancer until you deregister it from all target groups.

Step 1: Create the Application Load Balancer

Before you begin, configure the target groups that this Application Load Balancer will use. Ensure that you have a virtual private cloud (VPC) with the targets that you will register with the target group. This VPC must have at least one public subnet in each of the Availability Zones used by your targets.

To create an Application Load Balancer using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Load Balancers**.
3. Choose **Create load balancer**.
4. Under **Application Load Balancer**, choose **Create**.
5. On the **Create Application Load Balancer** page, under **Basic configuration**, specify the **Load balancer name**, **Scheme**, and **IP address type**.
6. For **Listeners**, you can create an HTTP or HTTPS listener on any port. However, you must ensure that the port number of this listener matches the port of the target group in which this Application Load Balancer will reside.
7. Under **Availability Zones**, do the following:

    a. For **VPC**, select a virtual private cloud (VPC) with instances or IP addresses that you included as targets of your Application Load Balancer. You must use the same VPC that you would use for your Network Load Balancer in Step 3: Create a Network Load Balancer, and configure the Application Load Balancer as its target.

    b. Select two or more **Availability Zones** and corresponding subnets. Ensure that these Availability Zones match those enabled for your Network Load Balancer to optimize availability, scaling, and performance.

8. You can **Assign a security group** to your load balancer by creating a new security group or by selecting an existing one.

   The security group that you select should contain a rule that allows traffic to the listener port for this load balancer. Use the CIDR blocks (IP address range) of the client's computers as the traffic source in the inbound rules for security groups. This allows clients to send traffic through this Application Load Balancer. For more information about configuring security groups for an Application Load Balancer as a target of a Network Load Balancer, see Security groups for your Application Load Balancer in the *User Guide for Application Load Balancers*.
9. For **Configure Routing**, select the target group that you configured for this Application Load Balancer. If you don't have an available target group, and want to configure a new one, see Create a target group in the *User Guide for Application Load Balancers*.
10. Review your configuration, and choose **Create load balancer**.

To create the Application Load Balancer using the AWS CLI

Use the create-load-balancer command.

Step 2: Create the target group with the Application Load Balancer as the target

Creating a target group allows you to register a new or existing Application Load Balancer as a target. You can only add one Application Load Balancer per target group. The same Application Load Balancer can also be used in a separate target group, as the target of up to two Network Load Balancers.

To create a target group and register the Application Load Balancer as a target, using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose **Create target group**.
4. On the **Specify group details** page, under **Basic configuration**, choose **Application Load Balancer**.
5. For **Target group name**, enter a name for the Application Load Balancer target group.

6. For **Protocol**, only TCP is allowed. Select the **Port** for your target group. This target group port must match the listener port of the Application Load Balancer. Alternatively, you can add or edit the listener port on the Application Load Balancer to match this port.
7. For **VPC**, select the virtual private cloud (VPC) with the Application Load Balancer to register with the target group.
8. For **Health checks**, choose HTTP or HTTPS as the **Health check protocol**. Health checks are sent to the Application Load Balancer and forwarded to its targets using the specified port, protocol, and ping path. Ensure that your Application Load Balancer can receive these health checks by having a listener with a port and protocol that matches the health check port and protocol.
9. (Optional) Add one or more tags as required.
10. Choose **Next**.
11. On the **Register targets** page, choose the Application Load Balancer that you want to register as the target. The Application Load Balancer you choose from the list must have a listener on the same port as the target group you're creating. You can add or edit a listener on this load balancer to match the target group's port or return to the previous step and change the port that's specified for the target group. If you're unsure about which Application Load Balancer to add as the target, or don't want to add it at this point, you can choose to add the Application Load Balancer later.
12. Choose **Create target group**.

To create a target group and register the Application Load Balancer as a target, using the AWS CLI

Use the create-target-group and register-targets command.

Step 3: Create a Network Load Balancer, and configure the Application Load Balancer as its target

Use the following steps to create the Network Load Balancer and then configure the Application Load Balancer as its target using the console.

To create your Network Load Balancer and listener using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Load Balancers**.

3. Choose **Create load balancer**.
4. Under **Network Load Balancer**, choose **Create**.
5. **Basic configuration**

   On the **Basic configuration** pane, configure **Load balancer name**, **Scheme**, and **IP address type**.
6. **Network mapping**

   a. For **VPC**, select the same VPC that you used for your Application Load Balancer target. If you selected **Internet-facing** for **Scheme**, only VPCs with an internet gateway are available for selection.

   b. For **Mappings**, select one or more Availability Zones and corresponding subnets. We recommend that you select the same Availability Zones as your Application Load Balancer target to optimize availability, scaling, and performance.

      (Optional) To use static IP addresses, choose **Use an Elastic IP address** in the **IPv4 settings** for each Availability Zone. With static IP addresses you can add certain IP addresses to an allow list for firewalls, or you can hard code IP addresses with clients.
7. **Listeners and routing**

   a. The default is a listener that accepts TCP traffic on port 80. Only TCP listeners can forward traffic to an Application Load Balancer target group. You must keep **Protocol** as **TCP**, but you can modify **Port** as needed.

      With this configuration, you can use HTTPS listeners on the Application Load Balancer to terminate TLS traffic.

   b. For **Default action**, select the Application Load Balancer target group to forward traffic. If you don't see it in the list, or can't select a target group (because it is already in use by another Network Load Balancer), you can create an Application Load Balancer target group as shown in Step 2: Create the target group with the Application Load Balancer as the target.
8. **Tags**

   (Optional) Add tags to categorize your load balancer. For more information, see Tags.
9. **Summary**

   Review your configuration, and choose **Create load balancer**.

To create the Network Load Balancer using the AWS CLI

Use the create-load-balancer command.

Step 4: (Optional) Create a VPC endpoint service

To use the Network Load Balancer that you set up in the previous step as an endpoint for private connectivity, you can enable AWS PrivateLink. This establishes a private connection to your load balancer as an endpoint service.

To create a VPC endpoint service using your Network Load Balancer

1. On the navigation pane, choose **Load Balancers**.
2. Select the name of the Network Load Balancer to open its details page.
3. On the **Integrations** tab, expand **VPC Endpoint Services (AWS PrivateLink)**.
4. Choose **Create endpoint services** to open the **Endpoint services** page. For the remaining steps, see Create an endpoint service in the *AWS PrivateLink Guide*.

Tags for your target group

Tags help you to categorize your target groups in different ways, for example, by purpose, owner, or environment.

You can add multiple tags to each target group. Tag keys must be unique for each target group. If you add a tag with a key that is already associated with the target group, it updates the value of that tag.

When you are finished with a tag, you can remove it.

Restrictions

- Maximum number of tags per resource—50
- Maximum key length—127 Unicode characters
- Maximum value length—255 Unicode characters
- Tag keys and values are case sensitive. Allowed characters are letters, spaces, and numbers representable in UTF-8, plus the following special characters: + - = . _ : / @. Do not use leading or trailing spaces.

- Do not use the `aws:` prefix in your tag names or values because it is reserved for AWS use. You can't edit or delete tag names or values with this prefix. Tags with this prefix do not count against your tags per resource limit.

To update the tags for a target group using console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Choose the name of the target group to open its details page.
4. On the **Tags** tab, choose **Manage tags** and do one or more of the following:
   a. To update a tag, enter new values for **Key** and **Value**.
   b. To add a tag, choose **Add tag** and enter values for **Key** and **Value**.
   c. To delete a tag, choose **Remove** next to the tag.
5. When you have finished updating tags, choose **Save changes**.

To update the tags for a target group using the AWS CLI

Use the add-tags and remove-tags commands.

Delete a target group

You can delete a target group if it is not referenced by the forward actions of any listener rules. Deleting a target group does not affect the targets registered with the target group. If you no longer need a registered EC2 instance, you can stop or terminate it.

To delete a target group using console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Load Balancing**, choose **Target Groups**.
3. Select the target group and choose **Actions**, **Delete**.
4. When prompted for confirmation, choose **Yes, delete**.

To delete a target group using the AWS CLI

Use the delete-target-group command.

Monitor your Network Load Balancers

You can use the following features to monitor your load balancers, analyze traffic patterns, and troubleshoot issues with your load balancers and targets.

CloudWatch metrics

You can use Amazon CloudWatch to retrieve statistics about data points for your load balancers and targets as an ordered set of time-series data, known as *metrics*. You can use these metrics to verify that your system is performing as expected. For more information, see CloudWatch metrics for your Network Load Balancer.

VPC Flow Logs

You can use VPC Flow Logs to capture detailed information about the traffic going to and from your Network Load Balancer. For more information, see VPC flow logs in the *Amazon VPC User Guide*.

Create a flow log for each network interface for your load balancer. There is one network interface per load balancer subnet. To identify the network interfaces for a Network Load Balancer, look for the name of the load balancer in the description field of the network interface.

There are two entries for each connection through your Network Load Balancer, one for the frontend connection between the client and the load balancer and the other for the backend connection between the load balancer and the target. If the target group's client IP preservation attribute is enabled, the connection appears to the instance as a connection from the client. Otherwise, the connection's source IP is the load balancer's private IP address. If the security group of the instance doesn't allow connections from the client but the network ACLs for the load balancer subnet allow them, the logs for the network interface for the load balancer show "ACCEPT OK" for the frontend and backend connections, while the logs for the network interface for the instance show "REJECT OK" for the connection.

If a Network Load Balancer has associated security groups, your flow logs contain entries for traffic that is allowed or rejected by the security groups. For Network Load Balancers with TLS listeners, your flow logs entries reflect only the rejected entries.

Access logs

You can use access logs to capture detailed information about TLS requests made to your load balancer. The log files are stored in Amazon S3. You can use these access logs to analyze traffic

patterns and to troubleshoot issues with your targets. For more information, see Access logs for your Network Load Balancer.

CloudTrail logs

You can use AWS CloudTrail to capture detailed information about the calls made to the Elastic Load Balancing API and store them as log files in Amazon S3. You can use these CloudTrail logs to determine which calls were made, the source IP address where the call came from, who made the call, when the call was made, and so on. For more information, see Logging API calls for your Network Load Balancer using AWS CloudTrail.

CloudWatch metrics for your Network Load Balancer

Elastic Load Balancing publishes data points to Amazon CloudWatch for your load balancers and your targets. CloudWatch enables you to retrieve statistics about those data points as an ordered set of time-series data, known as *metrics*. Think of a metric as a variable to monitor, and the data points as the values of that variable over time. For example, you can monitor the total number of healthy targets for a load balancer over a specified time period. Each data point has an associated time stamp and an optional unit of measurement.

You can use metrics to verify that your system is performing as expected. For example, you can create a CloudWatch alarm to monitor a specified metric and initiate an action (such as sending a notification to an email address) if the metric goes outside what you consider an acceptable range.

Elastic Load Balancing reports metrics to CloudWatch only when requests are flowing through the load balancer. If there are requests flowing through the load balancer, Elastic Load Balancing measures and sends its metrics in 60-second intervals. If there are no requests flowing through the load balancer or no data for a metric, the metric is not reported. For Network Load Balancers with security groups, traffic rejected by the security groups is not captured in the CloudWatch metrics.

For more information, see the Amazon CloudWatch User Guide.

Contents

Network Load Balancer metrics

The AWS/NetworkELB namespace includes the following metrics.

Metric	Description
ActiveFlowCount	The total number of concurrent flows (or connections) from clients to targets. This metric includes connections in the SYN_SENT and ESTABLISHED states. TCP connections are not terminated at the load balancer, so a client opening a TCP connection to a target counts as a single flow. **Reporting criteria**: Always reported. **Statistics**: The most useful statistics are Average, Maximum, and Minimum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
ActiveFlowCount_TCP	The total number of concurrent TCP flows (or connections) from clients to targets. This metric includes connections in the SYN_SENT and ESTABLISHED state. TCP connections are not terminated at the load balancer, so a client opening a TCP connection to a target counts as a single flow. **Reporting criteria**: There is a nonzero value **Statistics**: The most useful statistics are Average, Maximum, and Minimum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
ActiveFlowCount_TLS	The total number of concurrent TLS flows (or connections) from clients to targets. This metric includes connections in the SYN_SENT and ESTABLISHED state. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistics are Average, Maximum, and Minimum. **Dimensions** • LoadBalancer • AavailabilityZone ,LoadBalancer
ActiveFlowCount_UDP	The total number of concurrent UDP flows (or connections) from clients to targets. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistics are Average, Maximum, and Minimum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
ClientTLSNegotiationErrorCount	The total number of TLS handshakes that failed during negotiation between a client and a TLS listener. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer

Metric	Description
ConsumedLCUs	The number of load balancer capacity units (LCU) used by your load balancer. You pay for the number of LCUs that you use per hour. For more information, see Elastic Load Balancing Pricing. **Reporting criteria**: Always reported. **Statistics**: All **Dimensions** • LoadBalancer
ConsumedLCUs_TCP	The number of load balancer capacity units (LCU) used by your load balancer for TCP. You pay for the number of LCUs that you use per hour. For more information, see Elastic Load Balancing Pricing. **Reporting criteria**: There is a nonzero value. **Statistics**: All **Dimensions** • LoadBalancer
ConsumedLCUs_TLS	The number of load balancer capacity units (LCU) used by your load balancer for TLS. You pay for the number of LCUs that you use per hour. For more information, see Elastic Load Balancing Pricing. **Reporting criteria**: There is a nonzero value. **Statistics**: All **Dimensions** • LoadBalancer

Metric	Description
ConsumedLCUs_UDP	The number of load balancer capacity units (LCU) used by your load balancer for UDP. You pay for the number of LCUs that you use per hour. For more information, see Elastic Load Balancing Pricing. **Reporting criteria**: There is a nonzero value. **Statistics**: All **Dimensions** • LoadBalancer
HealthyHostCount	The number of targets that are considered healthy. This metric does not include any Application Load Balancers registered as targets. **Reporting criteria**: Reported if there are registered targets. **Statistics**: The most useful statistics are Maximum and Minimum. **Dimensions** • LoadBalancer ,TargetGroup • AvailabilityZone ,LoadBalancer ,TargetGroup
NewFlowCount	The total number of new flows (or connections) established from clients to targets in the time period. **Reporting criteria**: Always reported. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
NewFlowCount_TCP	The total number of new TCP flows (or connections) established from clients to targets in the time period. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
NewFlowCount_TLS	The total number of new TLS flows (or connections) established from clients to targets in the time period. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
NewFlowCount_UDP	The total number of new UDP flows (or connections) established from clients to targets in the time period. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
PeakPacketsPerSecond	Highest average packet rate (packets processed per second), calculated every 10 seconds during the sampling window. This metric includes health check traffic. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Maximum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
PortAllocationErrorCount	The total number of ephemeral port allocation errors during a client IP translation operation. A non-zero value indicates dropped client connections. Note: Network Load Balancers support 55,000 simultaneous connections or about 55,000 connections per minute to each unique target (IP address and port) when performing client address translation. To fix port allocation errors, add more targets to the target group. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
ProcessedBytes	The total number of bytes processed by the load balancer, including TCP/IP headers. This count includes traffic to and from targets, minus health check traffic. **Reporting criteria**: Always reported. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
ProcessedBytes_TCP	The total number of bytes processed by TCP listeners. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
ProcessedBytes_TLS	The total number of bytes processed by TLS listeners. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
ProcessedBytes_UDP	The total number of bytes processed by UDP listeners. **Reporting criteria**: There is a nonzero value **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
ProcessedPackets	The total number of packets processed by the load balancer. This count includes traffic to and from targets, including health check traffic. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
SecurityGroupBlockedFlowCount_Inbound_ICMP	The number of new ICMP messages rejected by the inbound rules of the load balancer security groups. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
SecurityGroupBlockedFlowCount_Inbound_TCP	The number of new TCP flows rejected by the inbound rules of the load balancer security groups. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
SecurityGroupBlockedFlowCount_Inbound_UDP	The number of new UDP flows rejected by the inbound rules of the load balancer security groups. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
SecurityGroupBlockedFlowCount_Outbound_ICMP	The number of new ICMP messages rejected by the outbound rules of the load balancer security groups. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
SecurityGroupBlockedFlowCount_Outbound_TCP	The number of new TCP flows rejected by the outbound rules of the load balancer security groups. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
SecurityGroupBlockedFlowCount_Outbound_UDP	The number of new UDP flows rejected by the outbound rules of the load balancer security groups. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
TargetTLSNegotiationErrorCount	The total number of TLS handshakes that failed during negotiation between a TLS listener and a target. **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer

Metric	Description
TCP_Client_Reset_Count	The total number of reset (RST) packets sent from a client to a target. These resets are generated by the client and forwarded by the load balancer. **Reporting criteria**: Always reported. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
TCP_ELB_Reset_Count	The total number of reset (RST) packets generated by the load balancer. For more information, see Troubleshooting. **Reporting criteria**: Always reported. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer
TCP_Target_Reset_Count	The total number of reset (RST) packets sent from a target to a client. These resets are generated by the target and forwarded by the load balancer. **Reporting criteria**: Always reported. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric	Description
UnHealthyHostCount	The number of targets that are considered unhealthy. This metric does not include any Application Load Balancers registered as targets. **Reporting criteria**: Reported if there are registered targets. **Statistics**: The most useful statistics are Maximum and Minimum. **Dimensions** • LoadBalancer ,TargetGroup • AvailabilityZone ,LoadBalancer ,TargetGroup
UnhealthyRoutingFlowCount	The number of flows (or connections) that are routed using the routing failover action (fail open). **Reporting criteria**: There is a nonzero value. **Statistics**: The most useful statistic is Sum. **Dimensions** • LoadBalancer • AvailabilityZone ,LoadBalancer

Metric dimensions for Network Load Balancers

To filter the metrics for your load balancer, use the following dimensions.

Dimension	Description
AvailabilityZone	Filters the metric data by Availability Zone.

Dimension	Description
LoadBalancer	Filters the metric data by load balancer. Specify the load balancer as follows: net/*load-balancer-name*/*1234567890123456* (the final portion of the load balancer ARN).
TargetGroup	Filters the metric data by target group. Specify the target group as follows: targetgroup/*target-group-name*/*1234567890123456* (the final portion of the target group ARN).

Statistics for Network Load Balancer metrics

CloudWatch provides statistics based on the metric data points published by Elastic Load Balancing. Statistics are metric data aggregations over specified period of time. When you request statistics, the returned data stream is identified by the metric name and dimension. A dimension is a name/value pair that uniquely identifies a metric. For example, you can request statistics for all the healthy EC2 instances behind a load balancer launched in a specific Availability Zone.

The `Minimum` and `Maximum` statistics reflect the minimum and maximum values of the data points reported by the individual load balancer nodes in each sampling window. Increases in the maximum of `HealthyHostCount` correspond to decreases in the minimum of `UnHealthyHostCount`. It's recommended to monitor maximum `HealthyHostCount`, invoking the alarm when the maximum `HealthyHostCount` falls below your required minimum, or being `0`. This can help in identifying when your targets have become unhealthy. It's also recommended to monitor minimum `UnHealthyHostCount`, invoking the alarm when the minimum `UnHealthyHostCount` rises above `0`. This allows you to become aware when there are no longer any registered targets.

The `Sum` statistic is the aggregate value across all load balancer nodes. Because metrics include multiple reports per period, `Sum` is only applicable to metrics that are aggregated across all load balancer nodes.

The `SampleCount` statistic is the number of samples measured. Because metrics are gathered based on sampling intervals and events, this statistic is typically not useful. For example, with `HealthyHostCount`, `SampleCount` is based on the number of samples that each load balancer node reports, not the number of healthy hosts.

View CloudWatch metrics for your load balancer

You can view the CloudWatch metrics for your load balancers using the Amazon EC2 console. These metrics are displayed as monitoring graphs. The monitoring graphs show data points if the load balancer is active and receiving requests.

Alternatively, you can view metrics for your load balancer using the CloudWatch console.

To view metrics using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. To view metrics filtered by target group, do the following:
   a. In the navigation pane, choose **Target Groups**.
   b. Select your target group and choose **Monitoring**.
   c. (Optional) To filter the results by time, select a time range from **Showing data for**.
   d. To get a larger view of a single metric, select its graph.
3. To view metrics filtered by load balancer, do the following:
   a. In the navigation pane, choose **Load Balancers**.
   b. Select your load balancer and choose **Monitoring**.
   c. (Optional) To filter the results by time, select a time range from **Showing data for**.
   d. To get a larger view of a single metric, select its graph.

To view metrics using the CloudWatch console

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Metrics**.
3. Select the **NetworkELB** namespace.
4. (Optional) To view a metric across all dimensions, type its name in the search field.

To view metrics using the AWS CLI

Use the following list-metrics command to list the available metrics:

```
aws cloudwatch list-metrics --namespace AWS/NetworkELB
```

To get the statistics for a metric using the AWS CLI

Use the following get-metric-statistics command get statistics for the specified metric and dimension. Note that CloudWatch treats each unique combination of dimensions as a separate metric. You can't retrieve statistics using combinations of dimensions that were not specially published. You must specify the same dimensions that were used when the metrics were created.

```
aws cloudwatch get-metric-statistics --namespace AWS/NetworkELB \
--metric-name UnHealthyHostCount --statistics Average  --period 3600 \
--dimensions Name=LoadBalancer,Value=net/my-load-balancer/50dc6c495c0c9188 \
Name=TargetGroup,Value=targetgroup/my-targets/73e2d6bc24d8a067 \
--start-time 2017-04-18T00:00:00Z --end-time 2017-04-21T00:00:00Z
```

The following is example output:

```
{
    "Datapoints": [
        {
            "Timestamp": "2017-04-18T22:00:00Z",
            "Average": 0.0,
            "Unit": "Count"
        },
        {
            "Timestamp": "2017-04-18T04:00:00Z",
            "Average": 0.0,
            "Unit": "Count"
        },
        ...
    ],
    "Label": "UnHealthyHostCount"
}
```

Access logs for your Network Load Balancer

Elastic Load Balancing provides access logs that capture detailed information about the TLS connections established with your Network Load Balancer. You can use these access logs to analyze traffic patterns and troubleshoot issues.

> **Important**
>
> Access logs are created only if the load balancer has a TLS listener, and the logs contain information about TLS requests only. Access logs record requests on a best-effort basis. We recommend that you use access logs to understand the nature of the requests, not as a complete accounting of all requests.

Access logging is an optional feature of Elastic Load Balancing that is disabled by default. After you enable access logging for your load balancer, Elastic Load Balancing captures the logs as compressed files and stores them in the Amazon S3 bucket that you specify. You can disable access logging at any time.

You can enable server-side encryption with Amazon S3-managed encryption keys (SSE-S3), or using Key Management Service with Customer Managed Keys (SSE-KMS CMK) for your S3 bucket. Each access log file is automatically encrypted before it is stored in your S3 bucket and decrypted when you access it. You do not need to take any action as there is no difference in the way you access encrypted or unencrypted log files. Each log file is encrypted with a unique key, which is itself encrypted with a KMS key that is regularly rotated. For more information, see Specifying Amazon S3 encryption (SSE-S3) and Specifying server-side encryption with AWS KMS (SSE-KMS) in the *Amazon S3 User Guide*.

There is no additional charge for access logs. You are charged storage costs for Amazon S3, but not charged for the bandwidth used by Elastic Load Balancing to send log files to Amazon S3. For more information about storage costs, see Amazon S3 Pricing.

Access log files

Elastic Load Balancing publishes a log file for each load balancer node every 5 minutes. Log delivery is eventually consistent. The load balancer can deliver multiple logs for the same period. This usually happens if the site has high traffic.

The file names of the access logs use the following format:

```
bucket[/prefix]/AWSLogs/aws-account-id/elasticloadbalancing/region/yyyy/mm/dd/aws-account-id_elasticloadbalancing_region_net.load-balancer-id_end-time_random-string.log.gz
```

bucket

The name of the S3 bucket.

prefix

The prefix (logical hierarchy) in the bucket. If you don't specify a prefix, the logs are placed at the root level of the bucket.

aws-account-id

The AWS account ID of the owner.

region

The Region for your load balancer and S3 bucket.

yyyy/mm/dd

The date that the log was delivered.

load-balancer-id

The resource ID of the load balancer. If the resource ID contains any forward slashes (/), they are replaced with periods (.).

end-time

The date and time that the logging interval ended. For example, an end time of 20181220T2340Z contains entries for requests made between 23:35 and 23:40.

random-string

A system-generated random string.

The following is an example log file name:

```
s3://my-bucket/prefix/AWSLogs/123456789012/elasticloadbalancing/us-
east-2/2020/05/01/123456789012_elasticloadbalancing_us-east-2_net.my-
loadbalancer.1234567890abcdef_20200501T0000Z_20sg8hgm.log.gz
```

You can store your log files in your bucket for as long as you want, but you can also define Amazon S3 lifecycle rules to archive or delete log files automatically. For more information, see Manage your storage lifecycle in the *Amazon S3 User Guide*.

Access log entries

The following table describes the fields of an access log entry, in order. All fields are delimited by spaces. When new fields are introduced, they are added to the end of the log entry. When processing the log files, you should ignore any fields at the end of the log entry that you were not expecting.

Field	Description
type	The type of listener. The supported value is `tls`.
version	The version of the log entry. The current version is 2.0.
time	The time recorded at the end of the TLS connection, in ISO 8601 format.
elb	The resource ID of the load balancer.
listener	The resource ID of the TLS listener for the connection.
client:port	The IP address and port of the client.
destination:port	The IP address and port of the destination. If the client connects directly to the load balancer, the destination is the listener. If the client connects using a VPC endpoint service, the destination is the VPC endpoint.
connection_time	The total time for the connection to complete, from start to closure, in milliseconds.
tls_handshake_time	The total time for the TLS handshake to complete after the TCP connection is established, including client-side delays, in milliseconds. This time is included in the connection_time field.
received_bytes	The count of bytes received by the load balancer from the client, after decryption.
sent_bytes	The count of bytes sent by the load balancer to the client, before encryption.

Field	Description
incoming_tls_alert	The integer value of TLS alerts received by the load balancer from the client, if present. Otherwise, this value is set to -.
chosen_cert_arn	The ARN of the certificate served to the client. If no valid client hello message is sent, this value is set to -.
chosen_cert_serial	Reserved for future use. This value is always set to -.
tls_cipher	The cipher suite negotiated with the client, in OpenSSL format. If TLS negotiation does not complete, this value is set to -.
tls_protocol_version	The TLS protocol negotiated with the client, in string format. The possible values are `tlsv10`, `tlsv11`, `tlsv12`, and `tlsv13`. If TLS negotiation does not complete, this value is set to -.
tls_named_group	Reserved for future use. This value is always set to -.
domain_name	The value of the server_name extension in the client hello message. This value is URL-encoded. If no valid client hello message is sent or the extension is not present, this value is set to -.
alpn_fe_protocol	The application protocol negotiated with the client, in string format. The possible values are h2, `http/1.1`, and `http/1.0`. If no ALPN policy is configured in the TLS listener, no matching protocol is found, or no valid protocol list is sent, this value is set to -.
alpn_be_protocol	The application protocol negotiated with the target, in string format. The possible values are h2, `http/1.1`, and `http/1.0`. If no ALPN policy is configured in the TLS listener, no matching protocol is found, or no valid protocol list is sent, this value is set to -.
alpn_client_preference_list	The value of the application_layer_protocol_negotiation extension in the client hello message. This value is URL-encoded. Each protocol is enclosed in double quotes and protocols are separated by a comma. If no ALPN policy is configured in the TLS listener, no valid client hello message is sent, or the extension is not present, this value is set to -. The string is truncated if it is longer than 256 bytes.

Field	Description
tls_connection_creation_time	The time recorded at the beginning of the TLS connection, in ISO 8601 format.

Example log entries

The following are example log entries. Note that the text appears on multiple lines only to make it easier to read.

The following is an example for a TLS listener without an ALPN policy.

```
tls 2.0 2018-12-20T02:59:40 net/my-network-loadbalancer/c6e77e28c25b2234
 g3d4b5e8bb8464cd
72.21.218.154:51341 172.100.100.185:443 5 2 98 246 -
arn:aws:acm:us-east-2:671290407336:certificate/2a108f19-aded-46b0-8493-c63eb1ef4a99 -
ECDHE-RSA-AES128-SHA tlsv12 -
my-network-loadbalancer-c6e77e28c25b2234.elb.us-east-2.amazonaws.com
- - - 2018-12-20T02:59:30
```

The following is an example for a TLS listener with an ALPN policy.

```
tls 2.0 2020-04-01T08:51:42 net/my-network-loadbalancer/c6e77e28c25b2234
 g3d4b5e8bb8464cd
72.21.218.154:51341 172.100.100.185:443 5 2 98 246 -
arn:aws:acm:us-east-2:671290407336:certificate/2a108f19-aded-46b0-8493-c63eb1ef4a99 -
ECDHE-RSA-AES128-SHA tlsv12 -
my-network-loadbalancer-c6e77e28c25b2234.elb.us-east-2.amazonaws.com
h2 h2 "h2","http/1.1" 2020-04-01T08:51:20
```

Bucket requirements

When you enable access logging, you must specify an S3 bucket for the access logs. The bucket can be owned by a different account than the account that owns the load balancer. The bucket must meet the following requirements.

Requirements

- The bucket must be located in the same Region as the load balancer.

- The prefix that you specify must not include `AWSLogs`. We add the portion of the file name starting with `AWSLogs` after the bucket name and prefix that you specify.
- The bucket must have a bucket policy that grants permission to write the access logs to your bucket. Bucket policies are a collection of JSON statements written in the access policy language to define access permissions for your bucket. The following is an example policy.

```
{
    "Version": "2012-10-17",
    "Id": "AWSLogDeliveryWrite",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryAclCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::my-bucket",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": ["012345678912"]
                },
                "ArnLike": {
                    "aws:SourceArn": ["arn:aws:logs:us-east-1:012345678912:*"]
                }
            }
        },
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-bucket/AWSLogs/account-ID/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": ["012345678912"]
                },
                "ArnLike": {
                    "aws:SourceArn": ["arn:aws:logs:us-east-1:012345678912:*"]
```

```
                }
            }
        }
    ]
}
```

In the previous policy, for `aws:SourceAccount`, specify the list of account numbers for which logs are being delivered to this bucket. For `aws:SourceArn`, specify the list of ARNs of the resource that generates the logs, in the form `arn:aws:logs:`*`source-region`*`:`*`source-account-id`*`:*`.

Encryption

You can enable server-side encryption for your Amazon S3 access log bucket in one of the following ways:

- Amazon S3-Managed Keys (SSE-S3)
- AWS KMS keys stored in AWS Key Management Service (SSE-KMS) †

† With Network Load Balancer access logs, you can't use AWS managed keys, you must use customer managed keys.

For more information, see Specifying Amazon S3 encryption (SSE-S3) and Specifying server-side encryption with AWS KMS (SSE-KMS) in the *Amazon S3 User Guide*.

The key policy must allow the service to encrypt and decrypt the logs. The following is an example policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
```

```
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

Enable access logging

When you enable access logging for your load balancer, you must specify the S3 bucket where the load balancer will store the logs. Be sure that you own this bucket and that you configured the required bucket policy for this bucket. For more information, see Bucket requirements.

To enable access logging using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Load Balancers**.
3. Select the name of your load balancer to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. On the **Edit load balancer attributes** page, do the following:
   a. For **Monitoring**, turn on **Access logs**.
   b. Choose **Browse S3** and select a bucket to use. Alternatively, enter the location of your S3 bucket, including any prefix.
   c. Choose **Save changes**.

To enable access logging using the AWS CLI

Use the modify-load-balancer-attributes command.

Disable access logging

You can disable access logging for your load balancer at any time. After you disable access logging, your access logs remain in your S3 bucket until you delete the them. For more information, see Working with buckets in the *Amazon Simple Storage Service User Guide*.

To disable access logging using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

2. In the navigation pane, choose **Load Balancers**.
3. Select the name of your load balancer to open its details page.
4. On the **Attributes** tab, choose **Edit**.
5. For **Monitoring**, turn off **Access logs**.
6. Choose **Save changes**.

To disable access logging using the AWS CLI

Use the modify-load-balancer-attributes command.

Processing access log files

The access log files are compressed. If you open the files using the Amazon S3 console, they are uncompressed and the information is displayed. If you download the files, you must uncompress them to view the information.

If there is a lot of demand on your website, your load balancer can generate log files with gigabytes of data. You might not be able to process such a large amount of data using line-by-line processing. Therefore, you might have to use analytical tools that provide parallel processing solutions. For example, you can use the following analytical tools to analyze and process access logs:

- Amazon Athena is an interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL. For more information, see Querying Network Load Balancer logs in the *Amazon Athena User Guide*.
- Loggly
- Splunk
- Sumo Logic

Logging API calls for your Network Load Balancer using AWS CloudTrail

Elastic Load Balancing is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Elastic Load Balancing. CloudTrail captures all API calls for Elastic Load Balancing as events. The calls captured include calls from the AWS Management

Console and code calls to the Elastic Load Balancing API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Elastic Load Balancing. 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 Elastic Load Balancing, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

Elastic Load Balancing information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Elastic Load Balancing, 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 Elastic Load Balancing, 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:

- 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

All Elastic Load Balancing actions for Network Load Balancers are logged by CloudTrail and are documented in the Elastic Load Balancing API Reference version 2015-12-01. For example, calls to the `CreateLoadBalancer` and `DeleteLoadBalancer` actions generate entries in the CloudTrail log files.

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

- Whether the request was made with root or 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.

Understanding Elastic Load Balancing 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, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The log files include events for all AWS API calls for your AWS account, not just Elastic Load Balancing API calls. You can locate calls to the Elastic Load Balancing API by checking for `eventSource` elements with the value `elasticloadbalancing.amazonaws.com`. To view a record for a specific action, such as `CreateLoadBalancer`, check for `eventName` elements with the action name.

The following are example CloudTrail log records for Elastic Load Balancing for a user who created a Network Load Balancer and then deleted it using the AWS CLI. You can identify the CLI using the `userAgent` elements. You can identify the requested API calls using the `eventName` elements. Information about the user (`Alice`) can be found in the `userIdentity` element.

Example Example: CreateLoadBalancer

```
{
    "eventVersion": "1.03",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Alice"
    },
    "eventTime": "2016-04-01T15:31:48Z",
    "eventSource": "elasticloadbalancing.amazonaws.com",
    "eventName": "CreateLoadBalancer",
    "awsRegion": "us-west-2",
```

```
    "sourceIPAddress": "198.51.100.1",
    "userAgent": "aws-cli/1.10.10 Python/2.7.9 Windows/7 botocore/1.4.1",
    "requestParameters": {
        "subnets": ["subnet-8360a9e7","subnet-b7d581c0"],
        "securityGroups": ["sg-5943793c"],
        "name": "my-load-balancer",
        "scheme": "internet-facing",
        "type": "network"
    },
    "responseElements": {
        "loadBalancers":[{
            "type": "network",
            "ipAddressType": "ipv4",
            "loadBalancerName": "my-load-balancer",
            "vpcId": "vpc-3ac0fb5f",
            "securityGroups": ["sg-5943793c"],
            "state": {"code":"provisioning"},
            "availabilityZones": [
               {"subnetId":"subnet-8360a9e7","zoneName":"us-west-2a"},
               {"subnetId":"subnet-b7d581c0","zoneName":"us-west-2b"}
            ],
            "dNSName": "my-load-balancer-1836718677.us-west-2.elb.amazonaws.com",
            "canonicalHostedZoneId": "Z2P70J7HTTTPLU",
            "createdTime": "Apr 11, 2016 5:23:50 PM",
            "loadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/net/my-load-balancer/ffcddace1759e1d0",
            "scheme": "internet-facing"
        }]
    },
    "requestID": "b9960276-b9b2-11e3-8a13-f1ef1EXAMPLE",
    "eventID": "6f4ab5bd-2daa-4d00-be14-d92efEXAMPLE",
    "eventType": "AwsApiCall",
    "apiVersion": "2015-12-01",
    "recipientAccountId": "123456789012"
}
```

Example Example: DeleteLoadBalancer

```
{
    "eventVersion": "1.03",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "123456789012",
```

```
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Alice"
    },
    "eventTime": "2016-04-01T15:31:48Z",
    "eventSource": "elasticloadbalancing.amazonaws.com",
    "eventName": "DeleteLoadBalancer",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "198.51.100.1",
    "userAgent": "aws-cli/1.10.10 Python/2.7.9 Windows/7 botocore/1.4.1",
    "requestParameters": {
        "loadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/net/my-load-balancer/ffcddace1759e1d0"
    },
    "responseElements": null,
    "requestID": "349598b3-000e-11e6-a82b-298133eEXAMPLE",
    "eventID": "75e81c95-4012-421f-a0cf-babdaEXAMPLE",
    "eventType": "AwsApiCall",
    "apiVersion": "2015-12-01",
    "recipientAccountId": "123456789012"
}
```

Troubleshoot your Network Load Balancer

The following information can help you troubleshoot issues with your Network Load Balancer.

A registered target is not in service

If a target is taking longer than expected to enter the `InService` state, it might be failing health checks. Your target is not in service until it passes one health check. For more information, see Health checks for your target groups.

Verify that your instance is failing health checks and then check for the following:

A security group does not allow traffic

The security groups associated with an instance must allow traffic from the load balancer using the health check port and health check protocol. For more information, see Target security groups.

A network access control list (ACL) does not allow traffic

The network ACL associated with the subnets for your instances and the subnets for your load balancer must allow traffic and health checks from the load balancer. For more information, see Network ACLs.

Requests are not routed to targets

Check for the following:

A security group does not allow traffic

The security groups associated with the instances must allow traffic on the listener port from client IP addresses (if targets are specified by instance ID) or load balancer nodes (if targets are specified by IP address). For more information, see Target security groups.

A network access control list (ACL) does not allow traffic

The network ACLs associated with the subnets for your VPC must allow the load balancer and targets to communicate in both directions on the listener port. For more information, see Network ACLs.

The targets are in an Availability Zone that is not enabled

If you register targets in an Availability Zone but do not enable the Availability Zone, these registered targets do not receive traffic from the load balancer.

The instance is in a peered VPC

If you have instances in a VPC that is peered with the load balancer VPC, you must register them with your load balancer by IP address, not by instance ID.

Targets receive more health check requests than expected

Health checks for a Network Load Balancer are distributed and use a consensus mechanism to determine target health. Therefore, targets receive more than the number of health checks configured through the `HealthCheckIntervalSeconds` setting.

Targets receive fewer health check requests than expected

Check whether `net.ipv4.tcp_tw_recycle` is enabled. This setting is known to cause issues with load balancers. The `net.ipv4.tcp_tw_reuse` setting is considered a safer alternative.

Unhealthy targets receive requests from the load balancer

This occurs when all registered targets are unhealthy. If there is at least one healthy registered target, your Network Load Balancer routes requests only to its healthy registered targets.

When there are only unhealthy registered targets, the Network Load Balancer routes requests to all the registered targets, known as fail-open mode. The Network Load Balancer does this instead of removing all the IP addresses from DNS when all the targets are unhealthy and respective Availability Zones do not have healthy target to send request to.

Target fails HTTP or HTTPS health checks due to host header mismatch

The HTTP host header in the health check request contains the IP address of the load balancer node and the listener port, not the IP address of the target and the health check port. If you are mapping incoming requests by host header, you must ensure that health checks match any HTTP host header. Another option is to add a separate HTTP service on a different port and configure

the target group to use that port for health checks instead. Alternatively, consider using TCP health checks.

Unable to associate a security group with a load balancer

If the Network Load Balancer was created without security groups, it can't support security groups after creation. You can only associate a security group to a load balancer during creation, or to an existing load balancer that was originally created with security groups.

Unable to remove all security groups

If the Network Load Balancer was created with security groups, there must be at least one security group associated with it at all times. You cannot remove all security groups from the load balancer at the same time.

Increase in TCP_ELB_Reset_Count metric

For each TCP request that a client makes through a Network Load Balancer, the state of that connection is tracked. If no data is sent through the connection by either the client or the target for longer than the idle timeout, the connection is closed. If a client or a target sends data after the idle timeout period elapses, it receives a TCP RST packet to indicate that the connection is no longer valid. Additionally, if a target becomes unhealthy, the load balancer sends a TCP RST for packets received on the client connections associated with the target, unless the unhealthy target triggers the load balancer to fail open.

If you see a spike in the `TCP_ELB_Reset_Count` metric just before or just as the `UnhealthyHostCount` metric increases, it is likely that the TCP RST packets were sent because the target was starting to fail but hadn't been marked unhealthy. If you see persistent increases in `TCP_ELB_Reset_Count` without targets being marked unhealthy, you can check the VPC flow logs for clients sending data on expired flows.

Connections time out for requests from a target to its load balancer

Check whether client IP preservation is enabled on your target group. NAT loopback, also known as hairpinning, is not supported when client IP preservation is enabled. If an instance is a client of a load balancer that it's registered with, and it has client IP preservation enabled, the connection

succeeds only if the request is routed to a different instance. If the request is routed to the same instance it was sent from, the connection times out because the source and destination IP addresses are the same.

If an instance must send requests to a load balancer that it's registered with, do one of the following:

- Disable client IP preservation.
- Ensure that containers that must communicate, are on different container instances.

Performance decreases when moving targets to a Network Load Balancer

Both Classic Load Balancers and Application Load Balancers use connection multiplexing, but Network Load Balancers do not. Therefore, your targets can receive more TCP connections behind a Network Load Balancer. Be sure that your targets are prepared to handle the volume of connection requests they might receive.

Port allocation errors connecting through AWS PrivateLink

If your Network Load Balancer is associated with a VPC endpoint service, it supports 55,000 simultaneous connections or about 55,000 connections per minute to each unique target (IP address and port). If you exceed these connections, there is an increased chance of port allocation errors. Port allocation errors can be tracked using the `PortAllocationErrorCount` metric. To fix port allocation errors, add more targets to the target group. For more information, see CloudWatch metrics for your Network Load Balancer.

Intermittent connection failure when client IP preservation is enabled

When client IP preservation is enabled, you might encounter TCP/IP connection limitations related to observed socket reuse on the targets. These connection limitations can occur when a client, or a NAT device in front of the client, uses the same source IP address and source port when connecting to multiple load balancer nodes simultaneously. If the load balancer routes these connections to the same target, the connections appear to the target as if they come from the same source socket, which results in connection errors. If this happens, clients can retry (if the connection fails)

or reconnect (if the connection is interrupted). You can reduce this type of connection error by increasing the number of source ephemeral ports or by increasing the number of targets for the load balancer. You can prevent this type of connection error by disabling client IP preservation or by disabling cross-zone load balancing.

Additionally, when client IP preservation is enabled, connectivity might fail if the clients that are connecting to the Network Load Balancer are also connected to targets behind the load balancer. To resolve this, you can disable client IP preservation on the affected target groups. Alternatively, have your clients connect only to the Network Load Balancer, or only to the targets, but not both.

TCP connection delays

When both cross-zone load balancing and client IP preservation are enabled, a client connecting to different IPs on the same load balancer may be routed to the same target. If the client uses the same source port for both of these connections, the target will receive what appears to be a duplicate connection, which can lead to connection errors and TCP delays in establishing new connections. You can prevent this type of connection error by disabling cross-zone load balancing. For more information, see Cross-zone load balancing.

Potential failure when the load balancer is being provisioned

One of the reasons a Network Load Balancer could fail when it is being provisioned is if you use an IP address that is already assigned or allocated elsewhere (for example, assigned as a secondary IP address for an EC2 instance). This IP address prevents the load balancer from being set up, and its state is `failed`. You can resolve this by de-allocating the associated IP address and retrying the creation process.

DNS name resolution contains fewer IP addresses than enabled Availability Zones

Ideally your Network Load Balancer provides one IP address per enabled Availability Zone, when they have at least one healthy host in the Availability Zone. When there are no healthy host in a particular Availability Zone, and cross-zone load balancing is disabled, the IP address of the Network Load Balancer respective of that AZ will be removed from DNS.

For example, suppose your Network Load Balancer has three Availability Zones enabled, all of which have at least one healthy registered target instance.

- If the registered target instance(s) in Availability Zone A become unhealthy, the corresponding IP address of Availability Zone A for the Network Load Balancer is removed from DNS.
- If any two of the enabled Availability Zones have no healthy registered target instance(s), the respective two IP addresses of the Network Load Balancer will be removed from DNS.
- If there are no healthy registered target instance(s) in all the enabled Availability Zones, the fail-open mode is enabled and DNS will provide all the IP addresses from the three enabled AZs in the result.

Troubleshoot unhealthy targets using the resource map

If your Network Load Balancer targets are failing health checks, you can use the resource map to find unhealthy targets and take actions based on the failure reason code. For more information, see Network Load Balancer resource map.

Resource map provides two views: **Overview**, and **Unhealthy Target Map**. **Overview** is selected by default and displays all of your load balancer's resources. Selecting the **Unhealthy Target Map** view will display only the unhealthy targets in each target group associated to the Network Load Balancer.

> **Note**
>
> **Show resource details** must be enabled to view the health check summary and error messages for all applicable resources within the resource map. When not enabled, you must select each resource to view its details.

The **Target groups** column displays a summary of the healthy and unhealthy targets for each target group. This can help determine if all the targets are failing health checks, or only specific targets are failing. If all targets in a target group are failing health checks, check the target group's health check settings. Select a target group's name to open its detail page in a new tab.

The **Targets** column displays the TargetID and the current health check status for each target. When a target is unhealthy, the health check failure reason code is displayed. When a single target is failing a health check, verify the target has sufficient resources. Select a target's ID to open its detail page in a new tab.

Selecting **Export** gives you the option of exporting the current view of your Network Load Balancer's resource map as a PDF.

Verify that your instance is failing health checks and then based on the failure reason code check for the following issues:

- **Unhealthy: Request timed out**
  - Verify the security groups and network access control lists (ACL) associated with your targets and Network Load Balancer are not blocking connectivity.
  - Verify the target has sufficient capacity available to accept connections from the Network Load Balancer.
  - The Network Load Balancer's health check responses can be viewed in each target's application logs. For more information, see Health check reason codes.
- **Unhealthy: FailedHealthChecks**
  - Verify the target is listening for traffic on the health check port.

    When using a TLS listener

    You choose which security policy is used for front-end connections. The security policy used for back-end connections is automatically selected based on the front-end security policy in use.

    - If your TLS listener is using a TLS 1.3 security policy for front-end connections, the `ELBSecurityPolicy-TLS13-1-0-2021-06` security policy is used for back-end connections.
    - If your TLS listener is not using a TLS 1.3 security policy for front-end connections, the `ELBSecurityPolicy-2016-08` security policy is used for back-end connections.

    For more information, see Security policies.
  - Verify the target is providing a server certificate and key in the correct format specified by the security policy.
  - Verify the target supports one or more matching ciphers, and a protocol provided by the Network Load Balancer to establish TLS handshakes.

Quotas for your Network Load Balancers

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is Region-specific. You can request increases for some quotas, and other quotas cannot be increased.

To view the quotas for your Network Load Balancers, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **Elastic Load Balancing**. You can also use the describe-account-limits (AWS CLI) command for Elastic Load Balancing.

To request a quota increase, see Requesting a quota increase in the *Service Quotas User Guide*. If the quota is not yet available in Service Quotas, use the Elastic Load Balancing limit increase form.

Load balancer

Your AWS account has the following quotas related to Network Load Balancers.

Name	Default	Adjustable
Certificates per Network Load Balancer	25	Yes
Listeners per Network Load Balancer	50	No
Network Load Balancer ENIs per VPC	1,200 [1]	Yes
Network Load Balancers per Region	50	Yes
Target Groups per Action per Network Load Balancer	1	No
Targets per Availability Zone per Network Load Balancer	500 [2, 3]	Yes
Targets per Network Load Balancer	3,000 [3]	Yes

[1] Each Network Load Balancer uses one network interface per zone. The quota is set at the VPC level. When sharing subnets or VPCs, the usage is calculated across all tenants.

[2] If a target is registered with *N* target groups, it counts as *N* targets toward this limit. Each Application Load Balancer that is a target of the Network Load Balancer counts as 50 targets if cross-zone load balancing is disabled or 100 targets if cross-zone load balancing is enabled.

[3] If cross-zone load balancing is enabled, the maximum is 500 targets per load balancer, regardless of the number of Availability Zones.

Target groups

The following quotas are for target groups.

Name	Default	Adjustable
Target Groups per Region	3,000 [1]	Yes
Targets per Target Group per Region (instances or IP addresses)	1,000	Yes
Targets per Target Group per Region (Application Load Balancers)	1	No

[1] This quota is shared by Application Load Balancers and Network Load Balancers.

Document history for Network Load Balancers

The following table describes the releases for Network Load Balancers.

Change	Description	Date
RSA 3072-bit and ECDSA 256/384/521-bit certificates	This release adds support for RSA 3072-bit certificates, and Elliptic Curve Digital Signature Algorithm (ECDSA) 256, 384 and 521-bit certificates via AWS Certificate Manager (ACM).	January 19, 2024
FIPS 140-3 TLS termination	This release adds security policies that use FIPS 140-3 cryptographic modules when terminating TLS connections.	November 20, 2023
Zonal DNS affinity	This release adds support for clients resolving the load balancer DNS to receive an IP address in the same Availability Zone (AZ) they are in.	October 12, 2023
Disable unhealthy target connection termination	This release adds support to maintain active connections to targets that fail health checks.	October 12, 2023
Default UDP connection termination	This release adds support to terminate UDP connections at the end of the deregistration timeout by default.	October 12, 2023

Register targets using IPv6	This release adds support to register instances as targets when addressed by IPv6.	October 2, 2023
Security groups for your Network Load Balancer	This release adds support to associate security groups with your Network Load Balancers at creation.	August 10, 2023
Target group health	This release adds support to configure the minimum count or percentage of targets that must be healthy, and what actions the load balancer takes when the threshold is not met.	November 17, 2022
Health check configuration	This release provides improvements to health check configuration.	November 17, 2022
Cross-zone load balancing	This release adds support to configure cross-zone load balancing at the target group level.	November 17, 2022
IPv6 target groups	This release adds support to configure IPv6 target groups for Network Load Balancers.	November 23, 2021
IPv6 internal load balancers	This release adds support to configure IPv6 target groups for Network Load Balancers.	November 23, 2021
TLS 1.3	This release adds security policies supporting TLS version 1.3.	October 14, 2021

Application Load Balancers as targets	This release adds support to configure an Application Load Balancer as the target of a Network Load Balancer.	September 27, 2021
Client IP preservation	This release adds support to configure client IP preservation.	February 4, 2021
Security policy for FS supporting TLS version 1.2	This release adds a security policy for Forward Secrecy (FS) supporting TLS version 1.2.	November 24, 2020
Dual-stack mode	This release adds support for dual-stack mode, which enables clients to connect to the load balancer using both IPv4 addresses and IPv6 addresses.	November 13, 2020
Connection termination on deregistration	This release adds support to close connections to deregistered targets after the end of the deregistration timeout.	November 13, 2020
ALPN policies	This release adds support for Application-Layer Protocol Negotiation (ALPN) preference lists.	May 27, 2020
Sticky sessions	This release adds support for sticky sessions based on source IP address and protocol.	February 28, 2020

Shared subnets	This release adds support for specifying subnets that were shared with you by another AWS account.	November 26, 2019
Private IP addresses	This release enables you to provide a private IP address from the IPv4 address range of the subnet you specify when you enable an Availability Zone for an internal load balancer.	November 25, 2019
Add subnets	This release adds support for enabling additional Availability Zones after you create your load balancer.	November 25, 2019
Security policies for FS	This release adds support for three additional predefined forward secrecy security policies.	October 8, 2019
SNI support	This release adds support for Server Name Indication (SNI).	September 12, 2019
UDP protocol	This release adds support for the UDP protocol.	June 24, 2019
Available in new region	This release adds support for Network Load Balancers in the Asia Pacific (Osaka) Region.	June 12, 2019
TLS protocol	This release adds support for the TLS protocol.	January 24, 2019

Cross-zone load balancing	This release adds support for enabling cross-zone load balancing.	February 22, 2018
Proxy protocol	This release adds support for enabling Proxy Protocol.	November 17, 2017
IP addresses as targets	This release adds support for registering IP addresses as targets.	September 21, 2017
New load balancer type	This release of Elastic Load Balancing introduces Network Load Balancers.	September 7, 2017