Elastic Load Balancing

Network Load Balancers

Elastic Load Balancing: Network Load Balancers

Table of Contents

What Is a Network Load Balancer?

Elastic Load Balancing supports the following types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers. This guide discusses Network Load Balancers. For more information about the other load balancers, see the User Guide for Application 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.

Each *target group* routes requests to one or more registered targets, such as EC2 instances, using the TCP protocol and the port number that you specify. 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 (p. 9)
- Listeners (p. 15)
- Target Groups (p. 18)

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 Cross-Zone Load Balancing in the *Elastic Load Balancing User Guide*.

If you enable multiple Availability Zones for your load balancer and ensure that each target group has at least one target in each enabled Availability Zone, this increases the fault tolerance of your applications. 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.

A load balancer node 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.

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.

You can configure a target group so that you register targets by instance ID or IP address. If you specify targets using an instance ID, the source IP addresses of the clients are preserved and provided to your applications. If you specify targets by IP address, the source IP addresses are the private IP addresses of the load balancer nodes.

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 Comparison of Elastic Load Balancing Products.

How to Get Started

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

- Getting Started with Network Load Balancers (p. 3)
- Tutorial: Create a Network Load Balancer Using the AWS CLI (p. 6)

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

Alternatively, to create an Application Load Balancer, see Getting Started with Application Load Balancers in the *User Guide for Application Load Balancers*. To create a Classic Load Balancer, see Create a Classic Load Balancer in the *User Guide for Classic Load Balancers*.

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 (p. 26).

Step 1: Choose a Load Balancer Type

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

To create a Network Load Balancer

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 2: Configure Your Load Balancer and Listener

On the **Configure Load Balancer** page, complete the following procedure.

To configure your load balancer and listener

1. For **Name**, type a name for your load balancer.

   The name of your Network Load Balancer must be unique within your set of Application Load Balancers and Network Load Balancers for the region, can have a maximum of 32 characters, can contain only alphanumeric characters and hyphens, must not begin or end with a hyphen, and must not begin with "internal-".
2. For **Scheme**, keep the default value, **internet-facing**.
3. For **Listeners**, keep the default, which is a listener that accepts TCP traffic on port 80.
4. For **Availability Zones**, 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 the public subnet for that Availability Zone.

   When you create an internet-facing load balancer, you can optionally select an Elastic IP address from **Elastic IP**. This provides your load balancer node with a static IPv4 address.
5. Choose **Next: Configure Routing**.

Step 3: 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. On the **Configure Routing** page, complete the following procedure.

To configure your target group

1. For **Target group**, keep the default, **New target group**.
2. For **Name**, type a name for the new target group.
3. Keep **Protocol** as TCP, **Port** as 80, and **Target type** as instance.
4. For **Health checks**, keep the default protocol.
5. Choose **Next: Register Targets**.

Step 4: Register Targets with Your Target Group

On the **Register Targets** page, complete the following procedure.

To register targets with the target group

1. For **Instances**, select one or more instances.
2. Keep the default port, 80, and choose **Add to registered**.
3. When you have finished selecting instances, choose **Next: Review**.

Step 5: Create and Test Your Load Balancer

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

To create and test your load balancer

1. On the **Review** page, choose **Create**.
2. After you are notified that your load balancer was created successfully, choose **Close**.
3. In the navigation pane, under **LOAD BALANCING**, choose **Target Groups**.
4. Select the newly created target group.
5. 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.
6. In the navigation pane, under **LOAD BALANCING**, choose **Load Balancers**.
7. Select the newly created load balancer.
8. Choose **Description** and copy the DNS name of the load balancer (for example, my-load-balancer-1234567890.us-west-2.elb.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 6: Delete Your Load Balancer (Optional)

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

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 load balancer and choose **Actions**, **Delete**.
4. When prompted for confirmation, choose **Yes, 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.
- 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 (p. 26).

Create Your Load Balancer

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

To create a load balancer

1. Use the create-load-balancer command to create a 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-12345678
```

   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:

```
aws elbv2 create-target-group --name my-targets --protocol TCP --port 80 --vpc-id
 vpc-12345678
```

   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-12345678
 Id=i-23456789
```

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
```

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-12345678,AllocationId=eipalloc-12345678
```

Add Targets Using Port Overrides

If you have a microservices architecture with multiple services on a single instance, each service accepts connections on a different port. You can register the instance with the target group multiple times, each time with a different port.

To add targets using port overrides

1. Use the create-target-group command to create a target group:

```
aws elbv2 create-target-group --name my-targets --protocol TCP --port 80 \
--vpc-id vpc-12345678
```

2. Use the register-targets command to register your instances with your target group. Notice that the instance IDs are the same for each container, but the ports are different.

```
aws elbv2 register-targets --target-group-arn targetgroup-arn  \
--targets Id=i-12345678,Port=80 Id=i-12345678,Port=766
```

3. 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
```

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 (p. 18), 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 (p. 15) 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 inter-region VPC peering, AWS managed VPN, and third-party VPN solutions. Network Load Balancers support connections from clients over intra-region VPC peering only if those clients are Nitro instances.

Contents

Load Balancer State

A load balancer can be in 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 could not be set up.

Load Balancer Attributes

The following are the load balancer attributes:

`deletion_protection.enabled`

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

`load_balancing.cross_zone.enabled`

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

Availability Zones

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

When you enable an Availability Zone, you specify one subnet from that Availability Zone. The subnet must have at least 8 available IP addresses. 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 a static IP address. Note that you can view this network interface but you cannot modify it.

When you create an Internet-facing load balancer, you can optionally associate one Elastic IP address per subnet. You cannot add or change Elastic IP addresses for your subnets after you create the load balancer.

Cross-Zone Load Balancing

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 Cross-Zone Load Balancing in the *Elastic Load Balancing User Guide*.

To enable cross-zone load balancing 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 load balancer.
4. Choose **Description**, **Edit attributes**.
5. On the **Edit load balancer attributes** page, select **Enable** for **Cross-Zone Load Balancing**, and choose **Save**.

To enable cross-zone load balancing using the AWS CLI

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

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, under **LOAD BALANCING**, choose **Load Balancers**.
3. Select the load balancer.
4. Choose **Description**, **Edit attributes**.

5. On the **Edit load balancer attributes** page, select **Enable** for **Delete Protection**, and choose **Save**.

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, under **LOAD BALANCING**, choose **Load Balancers**.
3. Select the load balancer.
4. Choose **Description**, **Edit attributes**.
5. On the **Edit load balancer attributes** page, clear **Enable delete protection** and choose **Save**.

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 request that a client makes through a Network Load Balancer, the state of that connection is tracked. The connection is terminated by the target. 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 sends data after the idle timeout period elapses, it receives a TCP RST packet to indicate that the connection is no longer valid.

Elastic Load Balancing sets the idle timeout value to 350 seconds. You cannot modify this value. Your targets can use TCP keepalive packets to reset the idle timeout.

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, launch EC2 instances in at least one Availability Zone. Ensure that the virtual private cloud (VPC) has at least one public subnet in each of these Availability Zones.

To create a load balancer using the AWS CLI, see Tutorial: Create a Network Load Balancer Using the AWS CLI (p. 6).

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

Tasks

Step 1: Configure a Load Balancer and a Listener

First, provide some basic configuration information for your load balancer, such as a name, a 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 (p. 15).

To configure your load balancer and listener

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. For **Network Load Balancer**, choose **Create**.
5. For **Name**, type a name for your load balancer. For example, **`my-nlb`**.
6. For **Scheme**, an internet-facing load balancer routes requests from clients over the internet to targets. An internal load balancer routes requests to targets using private IP addresses.
7. For **Listeners**, the default is a listener that accepts TCP traffic on port 80. You can keep the default listener settings, modify the protocol, or modify the port. Choose **Add** to add another listener.
8. For **Availability Zones**, select the VPC that you used for your EC2 instances. For each Availability Zone that you used to launch your EC2 instances, select an Availability Zone and then select the public subnet for that Availability Zone. To associate an Elastic IP address with the subnet, select it from **Elastic IP**.
9. Choose **Next: Configure Routing**.

Step 2: Configure a Target Group

You register targets, such as EC2 instances, with a target group. The target group that you configure in this step is used as the target group in the listener rule, which forwards requests to the target group. For more information, see Target Groups for Your Network Load Balancers (p. 18).

To configure your target group

1. For **Target group**, keep the default, **New target group**.
2. For **Name**, type a name for the target group.
3. Set **Protocol** and **Port** as needed.
4. For **Target type**, select `instance` to specify targets by instance ID or `ip` to specify targets by IP address.
5. For **Health checks**, keep the default health check settings.
6. Choose **Next: Register Targets**.

Step 3: Register Targets with the Target Group

You can register EC2 instances as targets in a target group.

To register targets by instance ID

1. For **Instances**, select one or more instances.
2. Keep the default instance listener port or type a new one and choose **Add to registered**.
3. When you have finished registering instances, choose **Next: Review**.

To register targets by IP address

1. For each IP address to register, do the following:

   a. For **Network**, if the IP address is from a subnet of the target group VPC, select the VPC. Otherwise, select **Other private IP address**.
   b. For **Availability Zone**, select an Availability Zone or **all**. This determines whether the target receives traffic from the load balancer nodes in the specified Availability Zone only or from all enabled Availability Zones. This field is not displayed if you are registering IP addresses from the VPC. In this case, the Availability Zone is automatically detected.
   c. For **IP**, type the address.
   d. For **Port**, type the port.
   e. Choose **Add to list**.
2. When you have finished adding IP addresses to the list, choose **Next: Review**.

Step 4: Create 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. When you are finished with your load balancer, you can delete it. For more information, see Delete a Network Load Balancer (p. 14).

To create the load balancer

1. On the **Review** page, choose **Create**.
2. After the load balancer is created, choose **Close**.
3. On the navigation pane, under **LOAD BALANCING**, choose **Target Groups**.
4. Select the newly created target group.
5. 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.

Tags for Your Network Load Balancer

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

You can add multiple tags to each load balancer. Tag keys must be unique for 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, under **LOAD BALANCING**, choose **Load Balancers**.
3. Select the load balancer.
4. Choose **Tags**, **Add/Edit Tags**, and then do one or more of the following:

   a. To update a tag, edit the values of **Key** and **Value**.
   b. To add a new tag, choose **Create Tag**. For **Key** and **Value**, type values.
   c. To delete a tag, choose the delete icon (X) next to the tag.
5. When you have finished updating tags, choose **Save**.

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 (p. 10).

Note that 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 (p. 29).

To delete a load balancer using the console

1. If you have a CNAME 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.
2. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
3. In the navigation pane, under **LOAD BALANCING**, choose **Load Balancers**.
4. Select the load balancer.
5. Choose **Actions**, **Delete**.
6. When prompted for confirmation, choose **Yes, Delete**.

To delete a load balancer using the AWS CLI

Use the delete-load-balancer command.

Listeners for Your Network Load Balancers

Before you start using your Network Load Balancer, you must add one or more *listeners*. A listener is a process that checks for connection requests, using the protocol and port that you configure. The rules that you define for a listener determine how the load balancer routes requests to the targets in one or more target groups.

Contents

Listener Configuration

Listeners support the following protocols and ports:

- **Protocols**: TCP
- **Ports**: 1-65535

You can use WebSockets with your TCP listeners.

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 (p. 16).

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 (p. 22).

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 (p. 15).

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, under **LOAD BALANCING**, choose **Load Balancers**.
3. Select the load balancer.
4. Choose **Listener**, **Add listener**.
5. For **Protocol**, keep **TCP**.
6. For **Port**, type the listener port.
7. For **Default target group**, select an available target group with the TCP protocol.
8. Choose **Create**.

To add a listener using the AWS CLI

Use the create-listener command to create the listener.

Update a Listener for Your Network Load Balancer

You can update the listener port and the listener rule. Rules determine how your TCP listeners route requests to your target groups. You initially define the rule for a listener when you create the listener. For more information, see Listener Rules (p. 15).

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, under **LOAD BALANCING**, choose **Load Balancers**.
3. Select the load balancer.
4. Choose **Listener**, select the check box for the listener, and then choose **Actions**, **Edit**.
5. (Optional) Change the specified value for **Port**.
6. (Optional) Select a different target group from **Default target group**.
7. Choose **Save**.

To update your listener using the AWS CLI

Use the modify-listener command.

Delete a Listener for Your Network Load Balancer

You can delete a listener at any time. When you delete a load balancer, its listeners are deleted.

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, under **LOAD BALANCING**, choose **Load Balancers** and select the load balancer.
3. Choose **Listener**, select the check box for the listener, and then choose **Actions**, **Delete**.
4. When prompted for confirmation, choose **Yes, 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 each listener rule, you specify a target group and conditions. When a rule condition is met, traffic is forwarded to the corresponding target group. 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 (p. 1).

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.

Contents

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
- **Ports**: 1-65535

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 cannot 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.

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)

These supported CIDR blocks enable you to register the following with a target group: ClassicLink instances, AWS resources that are addressable by IP address and port (for example, databases), and on-premises resources linked to AWS through AWS Direct Connect or a software VPN connection.

Important
You can't specify publicly routable IP addresses.

Request Routing and IP Addresses

If you specify targets using an instance ID, traffic is routed to instances using the primary private IP address specified in the primary network interface for the 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.

Source IP Preservation

If you specify targets using an instance ID, the source IP addresses of the clients are preserved and provided to your applications.

If you specify targets by IP address, the source IP addresses are the private IP addresses of the load balancer nodes. If you need the IP addresses of the clients, enable Proxy Protocol and get the client IP addresses from the Proxy Protocol header.

If you have micro services on instances registered with a Network Load Balancer, you cannot 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 it's load balancer (p. 38).

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. You can register each EC2 instance or IP address with the same target group multiple times using different ports, which enables the load balancer to route requests to microservices.

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.

If demand on your application decreases, or 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*.

Limits

- You cannot register instances by instance ID if they have the following instance types: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3, and T1. You can register instances of these types by IP address.
- You cannot register instances by instance ID if they are in a peered VPC that is in a different region than the load balancer unless they are Nitro instances. You can register non-Nitro instances in a peered VPC that is in a different region than the load balancer by IP address.
- You cannot register instances in a peered VPC that is in the same region as the load balancer unless they are Nitro instances.

Target Group Attributes

The following are the target group attributes:

`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.

`proxy_protocol_v2.enabled`

Indicates whether Proxy Protocol version 2 is enabled. By default, Proxy Protocol is disabled.

Deregistration Delay

Elastic Load Balancing stops sending requests to instances that are deregistering. Connection draining ensures that in-flight requests complete before existing connections are closed. The initial state of a deregistering target is `draining`. By default, the state of a deregistering target changes to `unused` after 300 seconds. To change the amount of time that Elastic Load Balancing waits before changing the state 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.

To update the deregistration delay value 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. Select the target group.
4. Choose **Description**, **Edit attributes**.

5. Change the value of **Deregistration delay** as needed, and then choose **Save**.

To update the deregistration delay value 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.

If you specify targets by IP address, 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 clients, 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.

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)
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 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. Select the target group.
4. Choose **Description**, **Edit attributes**.
5. Select **Enable proxy protocol v2**, and then choose **Save**.

To enable Proxy Protocol using the AWS CLI

Use the modify-target-group-attributes command.

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 TCP listener and specify the target group in the default action for the listener. For more information, see Listener Rules (p. 15).

You can add or remove targets from your target group at any time. For more information, see Register Targets with Your Target Group (p. 26). 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 (p. 26).

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, under **LOAD BALANCING**, choose **Target Groups**.
3. Choose **Create target group**.
4. For **Target group name**, type a name for the target group.
5. For **Protocol**, select **TCP**.
6. (Optional) For **Port**, modify the default value as needed.
7. For **Target type**, select `instance` to specify targets by instance ID or `ip` to specify targets by IP address.
8. For **VPC**, select a virtual private cloud (VPC).
9. (Optional) For **Health check settings** and **Advanced health check settings**, modify the default settings as needed. Choose **Create**.
10. (Optional) Add one or more tags as follows:

    a. Select the newly created target group.
    b. Choose **Tags**, **Add/Edit Tags**.
    c. On the **Add/Edit Tags** page, for each tag that you add, choose **Create Tag** and then specify the tag key and tag value. When you have finished adding tags, choose **Save**.
11. (Optional) To add targets to the target group, see Register Targets with Your Target Group (p. 26).

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

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 (p. 10).

With active health checks, the load balancer periodically sends a request to each registered target to check its status. Each load balancer node checks the health of each target, using the health check settings for the target group with which the target is registered. After each health check is completed, the load balancer node closes the connection that was established for the health check.

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.

If one or more target groups does not 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 there are no enabled Availability Zones with a healthy target in each target group, requests are routed to targets in all enabled Availability Zones.

Health Check Settings

You configure active health checks for the targets in a target group using the following settings. The load balancer sends a health check request to each registered target every **HealthCheckIntervalSeconds** seconds, using the specified port, protocol, and ping path. It waits for the target to respond within the response timeout period. If the health checks exceed the threshold for consecutive failed responses, the load balancer takes the target out of service. When the health checks exceed the threshold for consecutive successful responses, the load balancer puts the target back in service.

Setting	Description
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.
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.
HealthCheckPath	[HTTP/HTTPS health checks] The ping 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. This is 10 seconds for TCP and HTTPS health checks and 6 seconds for HTTP health checks.

Setting	Description
HealthCheckIntervalSeconds	The approximate amount of time, in seconds, between health checks of an individual target. This value can be 10 seconds or 30 seconds. The default is 30 seconds. **Important** Health checks for a Network Load Balancer are distributed and use a consensus mechanism to determine target health. Therefore, targets can receive more than the configured number of health checks. To reduce the number of health checks to your targets, use a static HTML file as the HTTP target or use TCP health checks.
HealthyThresholdCount	The number of consecutive successful health checks required before considering an unhealthy target healthy. The range is 2 to 10. The default is 3.
UnhealthyThresholdCount	The number of consecutive failed health checks required before considering a target unhealthy. This value must be the same as the healthy threshold count.
Matcher	[HTTP/HTTPS health checks] The HTTP codes to use when checking for a successful response from a target. This value must be 200 to 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.
`healthy`	The target is healthy.
`unhealthy`	The target did not respond to a health check or failed the health check.
`unused`	The target is not registered with a target group, the target group is not used in a listener rule for the load balancer, or the target is in an Availability Zone that is not enabled for the load balancer.
`draining`	The target is deregistering and connection draining is in process.

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.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
`Target.ResponseCodeMismatch`	Health checks failed with these codes: [*code*]
`Target.Timeout`	Request timed out

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. Select the target group.
4. Choose **Targets**, and view the status of each target in the **Status** column. If the status is any value other than `Healthy`, view the tooltip for 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`.

Modify the Health Check Settings of a Target Group

You can modify some of the health check settings for your target group. If the protocol of the target group is TCP, you can't modify the health check protocol, interval, timeout, or success codes.

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. Select the target group.
4. Choose **Health checks**, **Edit**.
5. On the **Edit target group** page, modify the settings as needed, and then choose **Save**.

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

Use the modify-target-group command.

Register Targets with Your Target Group

You register your targets with one or more target groups. Each target group must have at least one registered target in each Availability Zone that is enabled for the load balancer. You can register targets by instance ID or by IP address. For more information, see Target Groups for Your Network Load Balancers (p. 18).

If demand on your currently registered targets increases, you can register additional targets in order to handle the demand. When your target is ready to handle requests, register it with your target group. The load balancer starts routing requests to the target as soon as the registration process completes and the target passes the initial health checks.

If demand on your registered targets decreases, or you need to service a target, you can deregister it from your target group. The load balancer stops routing requests to a target as soon as you deregister it. When the target is ready to receive requests, you can register it with the target group again.

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.

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

When you register EC2 instances as targets, you must ensure that the security groups for these instances allow traffic on both the listener port and the health check port.

Limits

- Network Load Balancers do not have associated security groups. Therefore, the security groups for your targets must use IP addresses to allow traffic from the load balancer.

- You cannot allow traffic from clients to targets through the load balancer using the security groups for the clients in the security groups for the targets. Use the client CIDR blocks in the target security groups instead.

Recommended Rules

Inbound		
Source	**Port Range**	**Comment**
Client IP addresses	*instance listener*	Allow traffic from clients on the instance listener port
VPC CIDR	*health check*	Allow traffic from the load balancer on the health check port

If you do not want to grant access to the entire VPC CIDR, you can grant access to the private IP addresses used by the load balancer nodes. There is one IP address per load balancer subnet. To find these addresses, use the following procedure.

To find the private IP addresses to whitelist

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, type 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 **Primary private IPv4 IP**.

Targets and Internet-facing Load Balancers

With an Internet-facing load balancer, targets that are registered by instance ID must have a route to the Internet to provide connectivity. The targets in a public subnet have a route to the Internet through the Internet gateway. If a target in a private subnet is registered by instance ID, ensure that the route table for the subnet has a route to the Internet (for example, through a NAT gateway or an EC2 instance).

Network ACLs

The default network access control list (ACL) for a VPC allows all inbound and outbound traffic. If you create custom network ACLs, they must allow the load balancer and instances to communicate in both directions on the listener port, health check port, and ephemeral ports (1024-65535).

Register or Deregister Targets

When you create a target group, you specify whether you must register targets by instance ID or IP address.

Limits

- You cannot register instances by instance ID if they have the following instance types: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3, and T1. You can register instances of these types by IP address.
- You cannot register instances by instance ID if they are in a peered VPC that is in a different region than the load balancer unless they are Nitro instances. You can register non-Nitro instances in a peered VPC that is in a different region than the load balancer by IP address.

- You cannot register instances in a peered VPC that is in the same region as the load balancer unless they are Nitro instances.

Register or Deregister Targets by Instance ID

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

To register or deregister targets by instance ID

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.
4. Choose **Targets**, **Edit**.
5. (Optional) For **Registered instances**, select any instances to be deregistered and choose **Remove**.
6. (Optional) For **Instances**, select any running instances to be registered, modify the default instance port as needed, and then choose **Add to registered**.
7. Choose **Save**.

Register or Deregister Targets by IP Address

The IP addresses that you register must be from the subnets of the VPC for the target group, the RFC 1918 range (10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16), and the RFC 6598 range (100.64.0.0/10). You cannot register publicly routable IP addresses.

To register or deregister targets by IP address

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 **Targets**, **Edit**.
4. To register IP addresses, choose the **Register targets** icon (the plus sign) in the menu bar. For each IP address, specify the network, Availability Zone, IP address, and port, and then choose **Add to list**. When you are finished specifying addresses, choose **Register**.
5. To deregister IP addresses, choose the **Deregister targets** icon (the minus sign) in the menu bar. If you have many registered IP addresses, you might find it helpful to add a filter or change the sort order. Select the IP addresses and choose **Deregister**.
6. To leave this screen, choose the **Back to target group** icon (the back button) in the menu bar.

To register or deregister targets using the AWS CLI

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

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 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. Select the target group.
4. On the **Tags** tab, choose **Add/Edit Tags**, and then do one or more of the following:
    a. To update a tag, edit the values of **Key** and **Value**.
    b. To add a new tag, choose **Create Tag** and then type values for **Key** and **Value**.
    c. To delete a tag, choose the delete icon (X) next to the tag.
5. When you have finished updating tags, choose **Save**.

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 any actions. Deleting a target group does not affect the targets registered with the target group. If you no longer need the EC2 instances, you can stop or terminate them.

To delete 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. Select the target group and choose **Actions**, **Delete**.
4. When prompted for confirmation, choose **Yes**.

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 (p. 30).

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 is registered by instance ID, the connection appears to the instance as a connection from the client. 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.

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 (p. 34).

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 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 TCP 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. **Statistics**: The most useful statistics are `Average`, `Maximum`, and `Minimum`.
`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.
`HealthyHostCount`	The number of targets that are considered healthy. **Statistics**: The most useful statistics are `Maximum` and `Minimum`.
`NewFlowCount`	The total number of new TCP flows (or connections) established from clients to targets in the time period. **Statistics**: The most useful statistic is `Sum`.
`ProcessedBytes`	The total number of bytes processed by the load balancer, including TCP/IP headers. **Statistics**: The most useful statistic is `Sum`.
`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. **Statistics**: The most useful statistic is `Sum`.
`TCP_ELB_Reset_Count`	The total number of reset (RST) packets generated by the load balancer.

Metric	Description
	Statistics: The most useful statistic is `Sum`.
`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. **Statistics**: The most useful statistic is `Sum`.
`UnHealthyHostCount`	The number of targets that are considered unhealthy. **Statistics**: The most useful statistics are `Maximum` and `Minimum`.

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.
`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 reported by the individual load balancer nodes. For example, suppose there are 2 load balancer nodes. One node has `HealthyHostCount` with a `Minimum` of 2, a `Maximum` of 10, and an `Average` of 6, while the other node has `HealthyHostCount` with a `Minimum` of 1, a `Maximum` of 5, and an `Average` of 3. Therefore, the load balancer has a `Minimum` of 1, a `Maximum` of 10, and an `Average` of about 4.

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 Amazon EC2 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"
}
```

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 AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.

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 (p. 23).

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.

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

The network ACL associated with the subnets for your instances must allow inbound traffic on the health check port and outbound traffic on the ephemeral ports (1024-65535). The network ACL associated with the subnets for your load balancer nodes must allow inbound traffic on the ephemeral ports and outbound traffic on the health check and ephemeral ports.

The instance is in a peered VPC

If the peered VPC is in a different region than the load balancer, you can register Nitro instances by IP address or instance ID, but you must register non-Nitro instances by IP address.

If the peered VPC is in the same region as the load balancer, you can register Nitro instances by IP address or instance ID, but you cannot register non-Nitro instances.

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).

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.

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.

There is no return route for traffic

The targets registered with an Internet-facing load balancer must have a route to the internet.

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 can 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

If there is at least one healthy registered target for your load balancer, the load balancer routes requests only to its healthy registered targets. If there are only unhealthy registered targets, the load balancer routes requests to all registered targets.

Connections time out for requests from a target to it's load balancer

Check whether you have an internal load balancer with targets registered by instance ID. Internal load balancers do not support hairpinning or loopback. When you register targets by instance ID, the source IP addresses of clients are preserved. If an instance is a client of an internal load balancer that it's registered with by instance ID, the connection succeeds only if the request is routed to a different instance. Otherwise, the source and destination IP addresses are the same and the connection times out.

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

- Register instances by IP address instead of instance ID. When using Amazon Elastic Container Service, use the `awsvpc` network mode with your tasks to ensure that target groups require registration by IP address.
- Ensure that containers that must communicate are on different container instances.
- Use an Internet-facing load balancer.

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.

Limits for Your Network Load Balancers

To view the current limits for your Network Load Balancers, use the **Limits** page of the Amazon EC2 console, or the describe-account-limits (AWS CLI) command. To request a limit increase, use the Elastic Load Balancing Limits form.

Your AWS account has the following limits related to Network Load Balancers.

Regional Limits

- Network Load Balancers per region: 20
- Target groups per region: 3000 *****

Load Balancer Limits

- Listeners per load balancer: 50
- Subnets per Availability Zone per load balancer: 1
- [Cross-zone load balancing disabled] Targets per Availability Zone per load balancer: 500
- [Cross-zone load balancing enabled] Targets per load balancer: 500
- Load balancers per target group: 1

***** This limit is shared by target groups for your Application Load Balancers and Network Load Balancers.

Document History for Network Load Balancers

The following table describes the releases for Network Load Balancers.

update-history-change	update-history-description	update-history-date
Cross-zone load balancing (p. 41)	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 (p. 41)	This release of Elastic Load Balancing introduces Network Load Balancers.	September 7, 2017