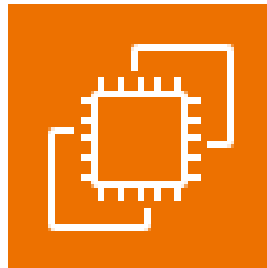


## **AWS Solution Architect Training with AWS Cloud Practitioner Global Certification Training**

**Trainer: Aravindraj.G- N minds Academy**

# **Configure Application Load Balancer with 3 Linux Web Servers in AWS**



**Pradeep Kumar R**

**B.E - Computer Science and Engineering**

**Mepco Schlenk Engineering College**

**Sivakasi**



## Objective

An AWS Application Load Balancer (ALB) is a fully managed Layer 7 (HTTP/HTTPS) load balancing service provided by Amazon Web Services (AWS). It is designed to handle HTTP and HTTPS traffic and offers advanced routing capabilities, allowing you to direct traffic to different resources based on the content of the request (e.g., URL paths, hostnames, query parameters, HTTP headers, etc.).

**When to Use AWS Application Load Balancer:**

1. **Web Applications:** If you are building a traditional web application or microservices application, ALB can route HTTP/HTTPS traffic to different services based on URL paths, hostnames, or headers.
2. **Microservices Architecture:** ALB is a good choice for microservices, where different services are hosted on different target groups. It can route traffic to the appropriate service based on the URL or other request attributes.
3. **Content-based Routing:** If you need to route traffic based on URL paths, hostnames, HTTP headers, or query parameters, ALB's advanced routing features are very useful.
4. **SSL/TLS Termination:** If you need to offload SSL/TLS decryption from your backend servers, ALB is a great option for handling HTTPS traffic securely.
5. **WebSocket Applications:** ALB supports WebSocket connections, making it suitable for applications that require persistent, real-time, full-duplex communication.
6. **Global Applications:** If you have applications that require high availability across multiple regions or availability zones, ALB's support for cross-zone load balancing helps ensure fault tolerance.

**Nomenclature and Components:**

1. **Listeners:** A listener checks for connection requests, defined by a protocol (HTTP or HTTPS) and a port (typically port 80 for HTTP or port 443 for HTTPS). It forwards traffic to one or more target groups based on rules you define.
2. **Target Groups:** A target group is a set of backend resources (such as EC2 instances or containers) that the ALB forwards traffic to. Each



target group can be associated with specific health check settings. Targets can be registered or deregistered as needed.

3. **Rules:** ALB allows you to define rules that determine how traffic is routed. You can configure rules to route traffic based on hostnames, path patterns, HTTP headers, query strings, or even the HTTP method (GET, POST, etc.).
4. **Health Checks:** ALB checks the health of targets by sending HTTP(S) requests to a specific path (e.g., /health). If a target is unhealthy, ALB stops routing traffic to it and forwards traffic to healthy targets.
5. **Target Types:** ALB supports three types of targets:
  - **Instance:** EC2 instances registered directly to the target group.
  - **IP:** IP addresses, such as instances in a private subnet or on-premise servers.
  - **Lambda functions:** AWS Lambda functions can also be used as targets.

#### Example Use Case:

Consider an e-commerce application where different services handle user authentication, product search, and order processing. You can use an ALB to route traffic as follows:

- Requests to /auth/\* go to the authentication service.
- Requests to /products/\* go to the product search service.
- Requests to /orders/\* go to the order processing service.

This routing setup can be achieved by configuring the ALB with path-based routing rules, improving the scalability and organization of the application.

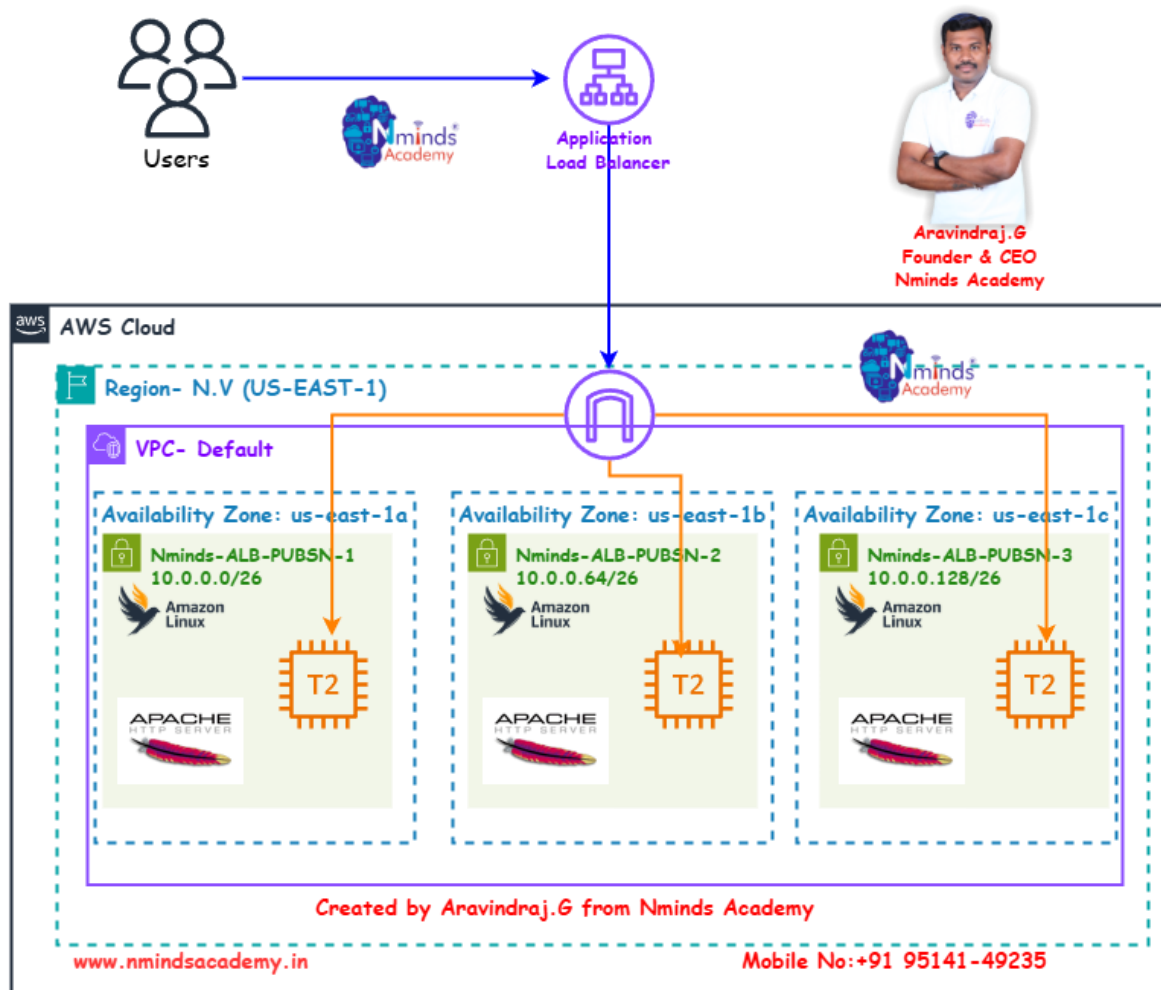
#### Benefits:

- **Advanced Routing:** ALB provides powerful content-based routing, making it suitable for modern applications and microservices.
- **SSL/TLS Termination:** Offload SSL/TLS decryption to the load balancer, reducing the overhead on your application servers.
- **Improved Fault Tolerance:** Health checks and automatic rerouting to healthy targets ensure high availability and reliability.
- **Ease of Use:** Fully managed and integrates easily with other AWS services like EC2, ECS, EKS, and Lambda.



# Topology

Configure High Availability with 3 Webserver using Application Load Balancer in AWS



Aravindraj.G  
Founder & CEO  
Nminds Academy



## Execution Tasks:

### Step 1: Launch Web-Server-1

The screenshot shows the AWS Management Console interface for launching an EC2 instance. At the top, a green banner indicates "Success: Successfully initiated launch of instance (i-0d1961d30dee5b57d)". Below this, the "Launch log" section is visible. The "Next Steps" section provides a search bar and a list of recommended actions:

- Create billing and free tier usage alerts:** To manage costs and avoid surprise bills, set up email notifications for billing and free tier usage thresholds. [Create billing alerts](#)
- Connect to your instance:** Once your instance is running, log into it from your local computer. [Connect to instance](#) [Learn more](#)
- Connect an RDS database:** Configure the connection between an EC2 instance and a database to allow traffic flow between them. [Connect an RDS database](#) [Create a new RDS database](#) [Learn more](#)
- Create EBS snapshot policy:** Create a policy that automates the creation, retention, and deletion of EBS snapshots. [Create EBS snapshot policy](#)
- Manage detailed monitoring:** Enable or disable detailed monitoring for the instance. If you enable detailed monitoring, the [Amazon CloudWatch](#) console displays the instance's CPU utilization metrics. [Manage detailed monitoring](#)
- Create Load Balancer:** Create an application, network gateway or classic Elastic Load Balancing. [Create Load Balancer](#)
- Create AWS budget:** AWS Budgets allows you to create budgets, forecast spend, and take action on your costs and [AWS Budgets](#)
- Manage CloudWatch alarms:** Create or update Amazon CloudWatch alarms for the instance. [Manage CloudWatch alarms](#)

The footer of the console shows the copyright notice: © 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

### Step 2: Launch Web-Server-2 and Web-Server-3:

This screenshot is identical to the one above, showing the AWS Management Console interface for launching an EC2 instance. The instance ID shown is i-0742249b85ffdbdba. The "Next Steps" section provides the same list of recommended actions as in the first screenshot.





```
ec2-user@ip-172-31-22-7:~$ ssh-keygen -f /dev/null -t rsa -b 4096 -C "ec2-user@ip-172-31-22-7"
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.172.152.76' (ED25519) to the list of known hosts.
#
      _ _ _ _ _
     _ _ _ _ _
    _ _ _ _ _
   _ _ _ _ _
  _ _ _ _ _
 _ _ _ _ _
_ _ _ _ _

Amazon Linux 2023

https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-172-31-22-7 ~]$ icacels "my-key-pair.pem" /inheritance:r
-bash: icacels: command not found
[ec2-user@ip-172-31-22-7 ~]$ icacels "my-key-pair.pem" /grant:r "donpr:R"
-bash: icacels: command not found
[ec2-user@ip-172-31-22-7 ~]$ ssh -i keyname.pem ec2-user@54.172.152.76
Warning: Identity file keyname.pem not accessible: No such file or directory.
The authenticity of host '54.172.152.76 (54.172.152.76)' can't be established.
ED25519 key fingerprint is SHA256:1I2Z0b0bIicI9PADJcJ2+plGosdp0iH4GSI2IGZzF7w.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.172.152.76' (ED25519) to the list of known hosts.
ec2-user@54.172.152.76: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[ec2-user@ip-172-31-22-7 ~]$ sudo yum update -y
sudo yum install httpd -y
sudo systemctl start httpd
sudo systemctl enable httpd
systemctl status httpd
Amazon Linux 2023 Kernel Livepatch repository
142 kB/s | 15 kB 00:00
=====
WARNING:
A newer release of "Amazon Linux" is available.

Available Versions:

Version 2023.7.20250414:
Run the following command to upgrade to 2023.7.20250414:

dnf upgrade --releasever=2023.7.20250414

Release notes:
https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.7.20250414.html
```

```
ec2-user@ip-172-31-22-7:~$ sudo yum install httpd -y
=====
Installing:
httpd x86_64 2.4.62-1.amzn2023 amazonlinux 48 k
Installing dependencies:
apr x86_64 1.7.5-1.amzn2023.0.4 amazonlinux 129 k
apr-util x86_64 1.6.3-1.amzn2023.0.1 amazonlinux 98 k
generic-logos-httpd noarch 18.0-0-12.amzn2023.0.3 amazonlinux 19 k
httpd-core x86_64 2.4.62-1.amzn2023 amazonlinux 1.4 M
httpd-filesystem noarch 2.4.62-1.amzn2023 amazonlinux 14 k
httpd-tools x86_64 2.4.62-1.amzn2023 amazonlinux 81 k
libbrotli x86_64 1.0.9-4.amzn2023.0.2 amazonlinux 315 k
mailcap noarch 2.1.49-3.amzn2023.0.3 amazonlinux 33 k
Installing weak dependencies:
apr-util-openssl x86_64 1.6.3-1.amzn2023.0.1 amazonlinux 17 k
mod_http2 x86_64 2.0.27-1.amzn2023.0.3 amazonlinux 166 k
mod_lua x86_64 2.4.62-1.amzn2023 amazonlinux 61 k
Transaction Summary
=====
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Downloading Packages:
(1/12): apr-1.7.5-1.amzn2023.0.4.x86_64.rpm 3.6 MB/s | 129 kB 00:00
(2/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64.rpm 432 kB/s | 17 kB 00:00
(3/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm 2.2 MB/s | 98 kB 00:00
(4/12): generic-logos-httpd-18.0-0-12.amzn2023.0.3.noarch.rpm 827 kB/s | 19 kB 00:00
(5/12): httpd-2.4.62-1.amzn2023.x86_64.rpm 2.0 MB/s | 48 kB 00:00
(6/12): httpd-core-2.4.62-1.amzn2023.x86_64.rpm 31 MB/s | 1.4 MB 00:00
(7/12): httpd-filesystem-2.4.62-1.amzn2023.noarch.rpm 395 kB/s | 14 kB 00:00
(8/12): httpd-tools-2.4.62-1.amzn2023.x86_64.rpm 2.3 MB/s | 81 kB 00:00
(9/12): mailcap-2.1.49-3.amzn2023.0.3.noarch.rpm 1.4 MB/s | 33 kB 00:00
(10/12): libbrotli-1.0.9-4.amzn2023.0.2.x86_64.rpm 8.2 MB/s | 315 kB 00:00
(11/12): mod_http2-2.0.27-1.amzn2023.0.3.x86_64.rpm 4.6 MB/s | 166 kB 00:00
(12/12): mod_lua-2.4.62-1.amzn2023.x86_64.rpm 2.8 MB/s | 61 kB 00:00
-----
Total 14 MB/s | 2.3 MB 00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing :
1/1
```



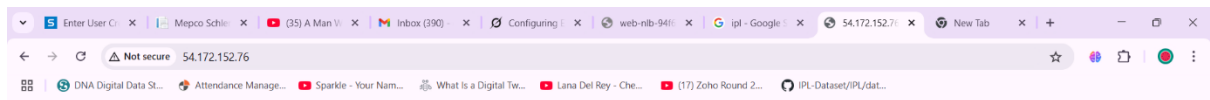


```
ec2-user@ip-172-31-22-7:~$  
===== WARNING: =====  
A newer release of "Amazon Linux" is available.  
  
Available Versions:  
  
Version 2023.7.20250414:  
Run the following command to upgrade to 2023.7.20250414:  
  
dnf upgrade --releasever=2023.7.20250414  
  
Release notes:  
https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.7.20250414.html  
===== Installed: =====  
apr-1.7.5-1.amzn2023.0.4.x86_64          apr-util-1.6.3-1.amzn2023.0.1.x86_64    apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64  
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch  httpd-2.4.62-1.amzn2023.x86_64          httpd-core-2.4.62-1.amzn2023.x86_64  
httpd-filesystem-2.4.62-1.amzn2023.noarch  httpd-tools-2.4.62-1.amzn2023.x86_64    libbrotli-1.0.9-4.amzn2023.0.2.x86_64  
mailcap-2.1.49-3.amzn2023.0.3.noarch      mod_httpd-2.0.27-1.amzn2023.0.3.x86_64  mod_lua-2.4.62-1.amzn2023.x86_64  
  
Complete!  
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.  
● httpd.service - The Apache HTTP Server  
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)  
   Active: active (running) since Thu 2025-04-17 15:29:37 UTC; 545ms ago  
     Docs: man:httpd.service(8)  
  Main PID: 4033 (httpd)  
    Status: "Started, listening on: port 80"  
    Tasks: 177 (limit: 1111)  
   Memory: 12.9M  
      CPU: 52ms  
   CGroup: /system.slice/httpd.service  
           └─4033 /usr/sbin/httpd -DFOREGROUND  
             └─4135 /usr/sbin/httpd -DFOREGROUND  
               └─4145 /usr/sbin/httpd -DFOREGROUND  
                 └─4146 /usr/sbin/httpd -DFOREGROUND  
                   └─4147 /usr/sbin/httpd -DFOREGROUND  
  
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...  
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.  
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal httpd[4033]: Server configured, listening on: port 80  
[ec2-user@ip-172-31-22-7 ~]$
```

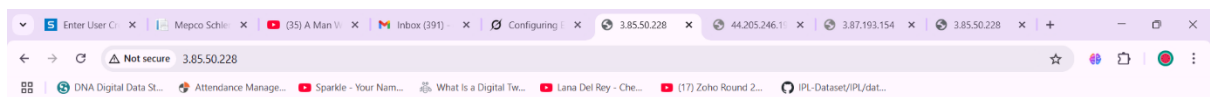
```
ec2-user@ip-172-31-22-7:~$  
===== Available Versions: =====  
  
Version 2023.7.20250414:  
Run the following command to upgrade to 2023.7.20250414:  
  
dnf upgrade --releasever=2023.7.20250414  
  
Release notes:  
https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.7.20250414.html  
===== Installed: =====  
apr-1.7.5-1.amzn2023.0.4.x86_64          apr-util-1.6.3-1.amzn2023.0.1.x86_64    apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64  
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch  httpd-2.4.62-1.amzn2023.x86_64          httpd-core-2.4.62-1.amzn2023.x86_64  
httpd-filesystem-2.4.62-1.amzn2023.noarch  httpd-tools-2.4.62-1.amzn2023.x86_64    libbrotli-1.0.9-4.amzn2023.0.2.x86_64  
mailcap-2.1.49-3.amzn2023.0.3.noarch      mod_httpd-2.0.27-1.amzn2023.0.3.x86_64  mod_lua-2.4.62-1.amzn2023.x86_64  
  
Complete!  
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.  
● httpd.service - The Apache HTTP Server  
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)  
   Active: active (running) since Thu 2025-04-17 15:29:37 UTC; 545ms ago  
     Docs: man:httpd.service(8)  
  Main PID: 4033 (httpd)  
    Status: "Started, listening on: port 80"  
    Tasks: 177 (limit: 1111)  
   Memory: 12.9M  
      CPU: 52ms  
   CGroup: /system.slice/httpd.service  
           └─4033 /usr/sbin/httpd -DFOREGROUND  
             └─4135 /usr/sbin/httpd -DFOREGROUND  
               └─4145 /usr/sbin/httpd -DFOREGROUND  
                 └─4146 /usr/sbin/httpd -DFOREGROUND  
                   └─4147 /usr/sbin/httpd -DFOREGROUND  
  
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...  
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.  
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal httpd[4033]: Server configured, listening on: port 80  
[ec2-user@ip-172-31-22-7 ~]$ echo "<h1>Web Server 1</h1>" | sudo tee /var/www/html/index.html  
<h1>Web Server 1</h1>  
[ec2-user@ip-172-31-22-7 ~]$ curl http://localhost  
<h1>Web Server 1</h1>  
[ec2-user@ip-172-31-22-7 ~]$
```







## Web Server 1



## Web Server 2



## Step 4: Configure Network Load Balancer

The screenshot shows the AWS Management Console interface. At the top, a green notification banner states "Elastic IP address allocated successfully. Elastic IP address 54.152.212.100" with a button to "Associate this Elastic IP address". Below this, the "Elastic IP addresses (1)" section is visible, showing a table with one entry: a Public IPv4 address 54.152.212.100, allocated to the user. The left sidebar shows the navigation menu with categories like EC2, Images, Elastic Block Store, and Network & Security. The bottom of the console shows the footer with copyright information and links to Privacy, Terms, and Cookie preferences.

```
Complete!
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
• httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
  Active: active (running) since Thu 2025-04-17 15:29:37 UTC; 545ms ago
  Docs: man:httpd.service(8)
  Main PID: 4033 (httpd)
  Status: "Started, listening on: port 80"
  Tasks: 177 (limit: 1111)
  Memory: 12.9M
  CPU: 52ms
  CGroup: /system.slice/httpd.service
          └─4033 /usr/sbin/httpd -DFOREGROUND
             └─4135 /usr/sbin/httpd -DFOREGROUND
                └─4145 /usr/sbin/httpd -DFOREGROUND
                   └─4146 /usr/sbin/httpd -DFOREGROUND
                      └─4147 /usr/sbin/httpd -DFOREGROUND

Apr 17 15:29:37 ip-172-31-22-7.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Apr 17 15:29:37 ip-172-31-22-7.ec2.internal httpd[4033]: Server configured, listening on: port 80
[ec2-user@ip-172-31-22-7 ~]$ echo "<h1>Web Server 1</h1>" | sudo tee /var/www/html/index.html
<h1>Web Server 1</h1>
[ec2-user@ip-172-31-22-7 ~]$ curl http://localhost
<h1>Web Server 1</h1>
[ec2-user@ip-172-31-22-7 ~]$ client_loop: send disconnect: Connection reset

C:\Users\donpr\Downloads>icacls "my-key-pair.pem" /inheritance:r
processed file: my-key-pair.pem
Successfully processed 1 files; Failed processing 0 files

C:\Users\donpr\Downloads>icacls "my-key-pair.pem" /grant:r "donpr:R" icacls "my-key-pair.pem" /inheritance:r
Invalid parameter "donpr:R"

C:\Users\donpr\Downloads>icacls "my-key-pair.pem" /grant:r "donpr:R"
processed file: my-key-pair.pem
Successfully processed 1 files; Failed processing 0 files

C:\Users\donpr\Downloads>sudo systemctl status httpd.
'sudo' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\donpr\Downloads>
```



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateTargetGroup:protocol=TCP;vpc=vpc-04869f3ce79a982fc

EC2 > Target groups > Create target group

Step 1  
Specify group details

Step 2  
Register targets

### Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (3/4)

Filter instances

<input type="checkbox"/>	Instance ID	Name	State	Security groups	Zone
<input checked="" type="checkbox"/>	i-04a9132f74d9a33f4	Web-Server-3	Running	launch-wizard-4	us-east-1b
<input checked="" type="checkbox"/>	i-0f673f9532c150eb4	Web-Server-2	Running	launch-wizard-3	us-east-1b
<input checked="" type="checkbox"/>	i-017bca58ea92714f	Web-Server-1	Running	launch-wizard-2	us-east-1b
<input type="checkbox"/>	i-04c5b5f557e14eba4	Linux-Web	Running	launch-wizard-1	us-east-1b

3 selected

Ports for the selected instances

Ports for routing traffic to the selected instances.

80

1-65535 (separate multiple ports with commas)

Include as pending below

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#TargetGroup:targetGroupArn=arn:aws:elasticloadbalancing:us-east-1:120742835816:targetgroup/Web-Targets/2a546460...

EC2 > Target groups > Web-Targets

Successfully created the target group: Web-Targets. Anomaly detection is automatically applied to all registered targets. Results can be viewed in the Targets tab.

### Web-Targets

Details

Target type: Instance

Protocol : Port: HTTP: 80

Protocol version: HTTP1

VPC: vpc-04869f3ce79a982fc

IP address type: IPv4

Load balancer: None associated

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (0) Info

Anomaly mitigation: Not applicable

Deregister | Register targets

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Filter targets

<input type="checkbox"/>	Instance ID	Name	Port	Zone	Health status	Health status details	Admini...	Over
Loading...								



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateALBWizard:

EC2 > Load balancers > Create Application Load Balancer

**Application Load Balancers now support public IPv4 IP Address Management (IPAM)**  
You can get started with this feature by configuring IP pools in the Network mapping section.

## Create Application Load Balancer [Info](#)

The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.

► How Application Load Balancers work

### Basic configuration

**Load balancer name**  
Name must be unique within your AWS account and can't be changed after the load balancer is created.

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

**Scheme** [Info](#)  
Scheme can't be changed after the load balancer is created.

☒ **Internet-facing**

- Serves Internet-facing traffic.
- Has public IP addresses.
- DNS name resolves to public IPs.
- Requires a public subnet.

☐ **Internal**

- Serves internal traffic.
- Has private IP addresses.
- DNS name resolves to private IPs.
- Compatible with the IPv4 and Dualstack IP address types.

**Load balancer IP address type** [Info](#)  
Select the front-end IP address type to assign to the load balancer. The VPC and subnets mapped to this load balancer must include the selected IP address types. Public IPv4 addresses have an additional cost.

☒ **IPv4**

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

## Step 5: Test Load Balancer

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#TargetGroup:targetGroupArn=arn:aws:elasticloadbalancing:us-east-1:120742835816:targetgroup/Web-NLB/d2e57645e17231d8

EC2 > Target groups > Web-NLB

Successfully created the target group: Web-NLB.

### Web-NLB

[Actions](#)

**Details**

arn:aws:elasticloadbalancing:us-east-1:120742835816:targetgroup/Web-NLB/d2e57645e17231d8

<b>Target type</b> Instance	<b>Protocol : Port</b> TCP: 80	<b>VPC</b> <a href="#">vpc-04869f3ce79a982fc</a>	<b>IP address type</b> IPv4
--------------------------------	-----------------------------------	---	--------------------------------

**Load balancer**  
[None associated](#)

<b>Total targets</b>	<b>Healthy</b>	<b>Unhealthy</b>	<b>Unused</b>	<b>Initial</b>	<b>Draining</b>
0	0	0	0	0	0

**Targets** | Monitoring | Health checks | Attributes | Tags

**Registered targets (0)**

Instance ID	Name	Port	Zone	Health s...	Health stat...	Administ...	Override...	Launch ...
No registered targets								

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#SecurityGroup:groupId=sg-0071f50f4a359fafc

EC2 > Security Groups > sg-0071f50f4a359fafc - ALB-SG

Security group (sg-0071f50f4a359fafc [ ALB-SG]) was created successfully

sg-0071f50f4a359fafc - ALB-SG

Details

Security group name ALB-SG	Security group ID sg-0071f50f4a359fafc	Description Allow HTTP for ALB	VPC ID vpc-04869f3ce79a982fc
Owner 120742835816	Inbound rules count 0 Permission entries	Outbound rules count 1 Permission entry	

Inbound rules

No security group rules found

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:us-east-1:120742835816:loadbalancer/net/Web-NLB/94f6729844900316

EC2 > Load balancers > Web-NLB

Successfully created load balancer

Web-NLB

Details

Load balancer type Network	Status Provisioning	VPC vpc-04869f3ce79a982fc	Load balancer IP address type IPv4
Scheme Internet-facing	Hosted zone Z26RNL4JYFTOT1	Availability Zones subnet-043ee758b31cacc02 us-east-1b (use1-az2) subnet-0f3b8dccc250c544f us-east-1a (use1-az1)	Date created April 17, 2025, 20:14 (UTC+05:30)

Load balancer ARN  
arn:aws:elasticloadbalancing:us-east-1:120742835816:loadbalancer/net/Web-NLB/94f6729844900316

DNS name info  
Web-NLB-94f6729844900316.elb.us-east-1.amazonaws.com (A Record)

Listeners (1)

A listener checks for connection requests using the protocol and port that you configure. Traffic received by a Network Load Balancer listener is forwarded to the selected target group.

