

LOAD BALANCING

The Load Balancer is a special service that sits between your users and your servers. When someone tries to access your website or application, their request goes to the load balancer first. The load balancer then decides which server or instance should handle the request. It does this by distributing the incoming requests evenly among the available servers.

AWS load balancing helps you manage incoming traffic to your website or application by distributing it across multiple servers, ensuring better performance, scalability, and reliability.

Types:

<p><u>1) Classic Load Balancer (CLB):</u></p> <ul style="list-style-type: none">Traditional load balancer is an old service.Operates at request and connection levels.Balances traffic across EC2 instances in multiple AZs.Supports both internet-facing and internal load balancing.Basic load balancing without advanced features.	<p><u>3) Network Load Balancer (NLB):</u></p> <ul style="list-style-type: none">Operates at Layer 4 (transport layer).Handles millions of requests with low latency.Supports TCP/UDP traffic.Ideal for high-performance and static IP needs.Automatically scales to handle high traffic.
<p><u>2) Application Load Balancer (ALB):</u></p> <ul style="list-style-type: none">Operates at Layer 7 (application layer).Routes HTTP/HTTPS traffic based on content.Supports host-based and path-based routing.Ideal for modern web apps and microservices.Integrates with container services like ECS and EKS.	<p><u>4) Gateway Load Balancer (GWLB):</u></p> <ul style="list-style-type: none">Works with VPC traffic.Distributes ingress traffic across multiple targets.Supports TLS termination and centralized security.Acts as a single entry point for VPC services.Integrates with AWS Network Firewall for enhanced security.

Target Groups

Target groups are an essential component of the Application Load Balancer (ALB) and Network Load Balancer (NLB) in AWS. They serve as a logical grouping of targets, such as EC2 instances, containers, IP addresses, or Lambda functions, that receive incoming traffic from the load balancer.

Functionality:

- When a load balancer receives a request, it forwards the request to one or more target groups based on routing rules.
- Each target group maintains health checks to ensure that only healthy targets receive traffic.
- Target groups support both instance-level and IP-level health checks.
- They can be associated with multiple listeners on the same load balancer.

Features:

- Health checks: Target groups periodically check the health of targets to ensure they are capable of handling traffic.
- Sticky sessions: ALB supports sticky sessions, allowing the load balancer to direct requests from the same client to the same target.
- Weighted routing: NLB supports weighted routing, enabling traffic distribution based on weights assigned to targets within a target group.
- Path-based routing: ALB supports path-based routing, allowing different paths of a URL to be routed to different target groups.

Use Cases:

- Target groups are used to distribute incoming traffic to multiple targets, improving availability and fault tolerance.
- They are essential for scaling applications horizontally by adding or removing targets dynamically.
- Target groups are used in conjunction with auto-scaling groups to automatically register and deregister instances based on demand.

Now we Perform Application load balancer Practical.

- 1) We launch a total of 9 Instances with a pre-configured HTTPD package using the shell script, write into user data option.
- 2) The 4 instances for a Home page, 3 instances for Product list pages, and 2 instances for a Price list page.
- 3) We will give the default path [/var/www/html/index.html] to the Home page.
- 4) For the product list page we will create a new directory in [/var/www/product], and the same as for the price list. These directories are our HTML index pages.
- 5) We need to add text to these pages to identify it, with text we add the variable \$HOSTNAME to verify load balancer is working or not.
- 6) Then we create an Application load balancer. With the same VPC in and mapping availability zone where our Instances.
- 7) Select the security group that has HTTP traffic allowed
- 8) Then create target groups 1st is home in the default path, 2nd is Product in the product directory path, and 3rd is the price in the price directory.
- 9) Then attach these target groups to the load balancer in the add Listener option.

After all steps are done we copy the load balancer DNS link, paste it into the new browser, and check it by refreshing, And giving the path of pages. If the IP address changes after refreshing the page that means our load balancer is successfully working.

Now we start

Note: Must allow HTTP traffic for all instances. And are in the same VPC.

Step 1) Launch 4 Instances for the Home page by writing the below shell script in the user data option.

```
#!/bin/bash
sudo -i
yum install httpd -y
systemctl start httpd
systemctl enable httpd
echo "Hello everyone, this Home Page of the website has a $HOSTNAME" > /var/www/html/index.html
```

Step 2) Launch 3 Instances for the Product list page by writing the below shell script in the user data option.

```
#!/bin/bash
sudo -i
yum install httpd -y
systemctl start httpd
systemctl enable httpd
mkdir /var/www/html/product
echo "Hello everyone, this Product List Page of the website has a $HOSTNAME" > /var/www/html/product/index.html
```

Step 3)) Launch 2 Instances for the Price list page by writing the below shell script in the user data option.

```
#!/bin/bash
sudo -i
yum install httpd -y
systemctl start httpd
systemctl enable httpd
mkdir /var/www/html/price
echo "Hello everyone, this Price List Page of the website has a $HOSTNAME" > /var/www/html/price/index.html
```

Instances (9) Info										Refresh Connect Instance state ▾ Actions ▾ Launch instances ▾
<input type="text"/> Find Instance by attribute or tag (case-sensitive)					Running ▾	< 1 > ⚙️				
<input type="checkbox"/>	Name ▾	Instance ID	Instance state ▾	Instanc... ▾	Status check	Alarm status	Availability Zo... ▾	Public IPv4 DNS ▾	Public IPv4	
<input type="checkbox"/>	Price page	i-08b3bec262d3d...	Running	t2.micro	Initializing	View alarms +	us-east-1b	ec2-54-198-4-176.com...	54.198.4.17	
<input type="checkbox"/>	Price page	i-074304894cc61...	Running	t2.micro	Initializing	View alarms +	us-east-1b	ec2-52-86-214-160.co...	52.86.214.1	
<input type="checkbox"/>	Product Page	i-02380c117b1f4...	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-18-232-110-167.co...	18.232.110.	
<input type="checkbox"/>	Product Page	i-0d9e9a12bf463...	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-54-164-147-216.co...	54.164.147.	
<input type="checkbox"/>	Product Page	i-07008e7bf8085...	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-54-91-47-203.com...	54.91.47.20	
<input type="checkbox"/>	Home Page	i-09617e50368e1...	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-54-172-90-5.comp...	54.172.90.5	
<input type="checkbox"/>	Home Page	i-0256aab4a2aa9...	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-3-94-251-40.comp...	3.94.251.40	
<input type="checkbox"/>	Home Page	i-0a8ea25f576f66...	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-54-80-142-226.co...	54.80.142.2	
<input type="checkbox"/>	Home Page	i-03ba08bd5920a...	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-3-87-109-245.com...	3.87.109.24	

Step 4) After Launching instances, copy their Public IP address, paste it into the browser, and check web page is running or not. If it's running everything is correct or not then check the shell scripting or HTTP traffic.

54.205.215.127
Not secure 54.205.215.127

“Hello everyone, this Home Page of the website has a ip-172-31-35-171.ec2.internal”

54.226.111.61/product/
Not secure 54.226.111.61/product/

“Hello everyone, this Product List Page of the website has a ip-172-31-37-217.ec2.internal”

23.20.243.237/price/
Not secure 23.20.243.237/price/

“Hello everyone, this Price List Page of the website has a ip-172-31-47-163.ec2.internal”

Now we will see our web pages are running.

Now we create Target groups

Step 5) Go to Load Balancing Menu in Ec2 dashboard, Select the Target Groups option and click on Create Target Group button.

Step 6) Our target type is Ec2 Instance, type target group name rest all options as by default, click on next button, and select your instance that you created for home page then click Includes as pending below button, and click on Create Target Group button.

Targets (4)								Remove all pending
<input type="text"/> Filter targets					<input type="checkbox"/> Show only pending	< 1 > ⚙️		
Instance ID	Name	Port	State	Security groups	Zone	Private IPv4 address	Subnet ID	
i-09617e50368e135b9	Home Page	80	Running	launch-wizard-4	us-east-1b	172.31.42.75	subnet-0e4e8a76d237d5	
i-0a8ea25f576f66601	Home Page	80	Running	launch-wizard-4	us-east-1b	172.31.36.43	subnet-0e4e8a76d237d5	
i-0256aab4a2aa91a5a	Home Page	80	Running	launch-wizard-4	us-east-1b	172.31.43.244	subnet-0e4e8a76d237d5	
i-03ba08bd5920a2525	Home Page	80	Running	launch-wizard-4	us-east-1b	172.31.39.146	subnet-0e4e8a76d237d5	

4 pending
Cancel
Previous
Create target group

As same creates target groups for Product and Price list Instances but in the above steps we gave the default path for the Home page, now we need to give that path where we create our Product and Price List.

Health check protocol

HTTP

Health check path

Use the default path of "/" to perform health checks on the root, or specify a custom path if preferred.

/product

Up to 1024 characters allowed.

▶ Advanced health check settings

Health check protocol

HTTP

Health check path

Use the default path of "/" to perform health checks on the root, or specify a custom path if preferred.

/price

Up to 1024 characters allowed.

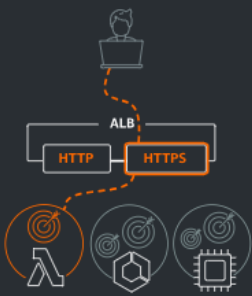
▶ Advanced health check settings

Our Target Groups are created now we create an Application load Balancer and attach these target groups to it.

Step 7) Go to the Create load balancer option and Click on the create button of Application Load Balancer

Load balancer types

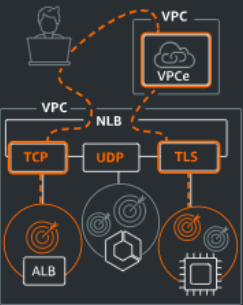
Application Load Balancer



Choose an Application Load Balancer when you need a flexible feature set for your applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.

Create


Network Load Balancer



Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your applications. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low latencies.

Create

Gateway Load Balancer



Choose a Gateway Load Balancer when you need to deploy and manage a fleet of third-party virtual appliances that support GENEVE. These appliances enable you to improve security, compliance, and policy controls.

Create

Type your ALB name, and Select VPC, and select the Availability zone where Instances are launched.

VPC

Info

Select the virtual private cloud (VPC) for your targets or you can [create a new VPC](#). Only VPCs with an internet gateway are enabled for selection. The selected VPC can't be changed after the load balancer is created. To confirm the VPC for your targets, view your [target groups](#).

-

vpc-0b61408ca59d3f247

IPv4: 172.31.0.0/16

Mappings

Info

Select at least two Availability Zones and one subnet per zone. The load balancer routes traffic to targets in these Availability Zones only. Availability Zones that are not supported by the load balancer or the VPC are not available for selection.

☒

us-east-1a (use1-az4)

Subnet

subnet-05722504926d6dec4

IPv4 address

Assigned by AWS

☒

us-east-1b (use1-az6)

Subnet

subnet-0e4e8a76d237d9dae

IPv4 address

Assigned by AWS

☒

us-east-1c (use1-az1)

Subnet

subnet-04496339d5f219a6f

IPv4 address

Assigned by AWS

☒

us-east-1d (use1-az2)

Subnet

subnet-008527ce3ecaebe76

IPv4 address

Assigned by AWS

☒

us-east-1e (use1-az3)

Subnet

subnet-0a7fcbaa05abb5f51

IPv4 address

Assigned by AWS

☒

us-east-1f (use1-az5)

Subnet

subnet-0d269a381e3d7b4f3

IPv4 address

Assigned by AWS

Step 8) In the Listener and routing option, the Default option Select our Home page target group.

Listeners and routing

Info

A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

▼ Listener HTTP:80

Remove

Protocol

Port

Default action

Info

HTTP

:

80

Forward to

Homepage

HTTP

1-65535

Target type: Instance, IPv4

[Create target group](#)

Listener tags - optional

Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

Add listener tag

You can add up to 50 more tags.

Add listener

Step 9) Click on the Create Load Balancer Button, It will take some time to provision, once in Active status then go to the target groups and check the targets health Status.

Details

arn:aws:elasticloadbalancing:us-east-1:971249973962:targetgroup/HomeTarget/f39aca8f34203cf2

Target type Instance	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC vpc-0b61408ca59d3f247
IP address type IPv4	Load balancer None associated		

4

Total targets

4

Healthy

0

Unhealthy

0

Unused

0

Initial

0

Draining

0 Anomalous

Distribution of targets by Availability Zone (AZ)
Select values in this table to see corresponding filters applied to the Registered targets table below.

Step 10) Copy DNS of load balancer and paste it into another browser.

Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers

1

Actions

Create load balancer

<input checked="" type="checkbox"/>	Name	DNS name copied	State	VPC ID	Availability Zone
<input checked="" type="checkbox"/>	Test-ALB	Test-ALB-901536978.us-east-1.elb.amazonaws.com	Active	vpc-0b61408ca59d3f247	6 Availability Zones

test-alb-901536978.us-east-1.elb.amazonaws.com

test-alb-901536978.us-east-1.elb.amazonaws.com

“Hello everyone, this Home Page of the website has a ip-172-31-43-213.ec2.internal”

Our load balancer DNS is working, now we refresh it, if the IP changes, that means the load balancer is distributing the traffic on other instances.

test-alb-901536978.us-east-1.elb.amazonaws.com

test-alb-901536978.us-east-1.elb.amazonaws.com

“Hello everyone, this Home Page of the website has a ip-172-31-45-171.ec2.internal”

test-alb-901536978.us-east-1.elb.amazonaws.com

test-alb-901536978.us-east-1.elb.amazonaws.com

“Hello everyone, this Home Page of the website has a ip-172-31-42-172.ec2.internal”

test-alb-901536978.us-east-1.elb.amazonaws.com

test-alb-901536978.us-east-1.elb.amazonaws.com

“Hello everyone, this Home Page of the website has a ip-172-31-35-171.ec2.internal”

Traffic Distribution is properly working

Step 10) We have only Added the Home Target group in Default Action, now we add the rules for Product Target and List Target Groups in the Home Target listener's add rule option.

- Select our Load Balancer
- Click on the Listener and Rules option
- Select our default listener, click the manage rule, and click on the add rule button.

The screenshot shows the AWS Management Console interface for a Load Balancer named 'Test-ALB'. The 'Listeners and rules' tab is selected. The 'Listeners and rules' section shows a table with one listener: 'HTTP:80'. The 'Add rule' button is highlighted in the 'Manage rules' dropdown menu. The 'HTTP:80' listener is highlighted in the table below.

- Enter the rule name, click on the next button
- Click on the add condition button, and Select the condition type as a path. Then enter the path, click confirm, and next.

The screenshot shows the 'Edit condition' dialog box. The 'Rule condition types' section shows 'Path' selected. The 'Path' field contains '/product/'. The 'Add new value' button is highlighted. The 'Confirm' button is highlighted at the bottom right.

We have given different paths for our Product and Price pages and all have the same protocol port **HTTP 80**. So here we doing **Path Base routing**. It allows to direct incoming requests to different target groups based on the URL path. In our scenario for the **Home** server **Product**, and **Price** are two different **backend Servers** hosted on the same **ALB** but route requests to different backend resources depending on the requested path. This setup enables to host multiple applications or services behind one ALB, simplifying your infrastructure while ensuring each application handles requests directed to its specific path.

- Select the Forward to target groups option, select target group, and click on next button.

Actions

Action types

Routing actions

☒ Forward to target groups ☐ Redirect to URL ☐ Return fixed response

Forward to target group [Info](#)
Choose a target group and specify routing weight or [Create target group](#)

Target group

ProductTarget [Info](#)
Target type: Instance, IPv4

Weight **Percent**

1 100%

0-999

Add target group

You can add up to 4 more target groups.

Group-level stickiness [Info](#)
If a target group is sticky, requests routed to it remain in that target group for the duration of the session. Individual target stickiness is a configuration of the target group.

☐ Turn on group-level stickiness

Cancel **Previous** **Next**

- Enter priority and click on the next button, then review and click on the create button.
- Now do the same for the price target group.

Rules **Tags**

Listener rules (3) [Info](#) [Rule limits](#) [Actions](#) [Add rule](#)

Traffic received by the listener is routed according to the default action and any additional rules. Rules are evaluated in priority order from the lowest value to the highest value.

<input type="checkbox"/>	Name tag	Priority	Conditions (If)	Actions (Then)	ARN	Tags
<input type="checkbox"/>	My Rule 1	1	Path Pattern is /product/	Forward to target group <ul style="list-style-type: none"> ProductTarget: 1 (100%) Group-level stickiness: Off 	ARN	1 tag
<input type="checkbox"/>	My Rule 2	2	Path Pattern is /price/	Forward to target group <ul style="list-style-type: none"> PriceTarget: 1 (100%) Group-level stickiness: Off 	ARN	1 tag
<input type="checkbox"/>	Default	Last (default)	If no other rule applies	Forward to target group <ul style="list-style-type: none"> HomeTarget: 1 (100%) Group-level stickiness: Off 	ARN	0 tags

[Activate Windows](#)

Step 11) Now go target group and check the health status of all target group's Instances. These health checks verify that the targets are responsive and able to handle incoming requests.

Step 12) Now copy the DNS URL of ALB and paste it into the new Browser and add the path of our Product and Price Pages.

Step 13) Refresh the web, the IP address will change. It means the load balancer is Distributing traffic among the backend servers.



