# ****Elastic Load Balancing****

Elastic Load Balancing automatically distributes your incoming application traffic across multiple targets, such as EC2 instances. It monitors the health of registered targets and routes traffic only to the healthy targets.

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers

## **Load Balancer Benefits**

A load balancer distributes workloads across multiple compute resources, such as virtual servers. Using a load balancer increases the availability and fault tolerance of your applications.

You can add and remove compute resources from your load balancer as your needs change, without disrupting the overall flow of requests to your applications.

You can configure health checks, which monitor the health of the compute resources, so that the load balancer sends requests only to the healthy ones. You can also offload the work of encryption and decryption to your load balancer so that your compute resources can focus on their main work.

## **Accessing Elastic Load Balancing**

**AWS Management Console AWS Command Line Interface (AWS CLI) AWS SDKs Query API**

## **How Elastic Load Balancing Works**

A load balancer accepts incoming traffic from clients and routes requests to its registered targets (such as EC2 instances) in one or more Availability Zones. The load balancer also monitors the health of its registered targets and ensures that it routes traffic only to healthy targets. When the load balancer detects an unhealthy target, it stops routing traffic to that target. It then resumes routing traffic to that target when it detects that the target is healthy again.

You configure your load balancer to accept incoming traffic by specifying one or more listeners. A listener is a process that checks for connection requests. It is configured with a protocol and port number for connections from clients to the load balancer. Likewise, it is configured with a protocol and port number for connections from the load balancer to the targets.

## **Availability Zones and Load Balancer Nodes**

When you enable an Availability Zone for your load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone. If you register targets in an Availability Zone but do not enable the Availability Zone, these registered targets do not receive traffic. Your load balancer is most effective when you ensure that each enabled Availability Zone has at least one registered target.

We recommend that you enable multiple Availability Zones. (With an Application Load Balancer, we require you to enable multiple Availability Zones.) This configuration helps ensure that the load balancer can continue to route traffic. If one Availability Zone becomes unavailable or has no healthy targets, the load balancer can route traffic to the healthy targets in another Availability Zone.

After you disable an Availability Zone, the targets in that Availability Zone remain registered with the load balancer. However, even though they remain registered, the load balancer does not route traffic to them.

## **Application Load Balancer**

## **Components**

A load balancer serves as the single point of contact for clients. The load balancer distributes incoming application traffic across multiple targets, such as EC2 instances, in multiple Availability Zones. 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. The rules that you define for a listener determine how the load balancer routes requests to its registered targets. Each rule consists of a priority, one or more actions, and one or more conditions. When the conditions for a rule are met, then its actions are performed. You must define a default rule for each listener, and you can optionally define additional rules.

Each target group routes requests to one or more registered targets, such as EC2 instances, using the protocol and 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.

The following diagram illustrates the basic components. Notice that each listener contains a default rule, and one listener contains another rule that routes requests to a different target group. One target is registered with two target groups.


                The components of a basic Application Load Balancer
            

## **Overview**

An Application Load Balancer functions at the application layer, the seventh layer of the Open Systems Interconnection (OSI) model. After the load balancer receives a request, it evaluates the listener rules in priority order to determine which rule to apply, and then selects a target from the target group for the rule action. You can configure listener rules to route requests to different target groups based on the content of the application traffic. Routing is performed independently for each target group, even when a target is registered with multiple target groups. You can configure the routing algorithm used at the target group level. The default routing algorithm is round robin; alternatively, you can specify the least outstanding requests routing algorithm.

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.

## **Benefits of Migrating from a Classic Load Balancer**

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

* Support for path-based routing. You can configure rules for your listener that forward requests based on the URL in the request. This enables you to structure your application as smaller services, and route requests to the correct service based on the content of the URL.
* Support for host-based routing. You can configure rules for your listener that forward requests based on the host field in the HTTP header. This enables you to route requests to multiple domains using a single load balancer.
* Support for routing based on fields in the request, such as standard and custom HTTP headers and methods, query parameters, and source IP addresses.
* 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 redirecting requests from one URL to another.
* Support for returning a custom HTTP response.
* Support for registering targets by IP address, including targets outside the VPC for the load balancer.
* Support for registering Lambda functions as targets.
* Support for the load balancer to authenticate users of your applications through their corporate or social identities before routing requests.
* 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 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.
* Access logs contain additional information and are stored in compressed format.
* Improved load balancer performance.

## **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.

## **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.

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.

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

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

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

When you create a target group, you specify its target type, which determines whether you register targets by instance ID or IP address. If you register targets by instance ID, the source IP addresses of the clients are preserved and provided to your applications. If you register 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.

## **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.

## **Practical Knowledge**

* Whenever we create a LB (ALB for HTTP, HTTPS), it allows us to choose or create a security Group for the same.
* We must create new target group as Configure Routing. We can customize the default settings for Health Check on targets.
* Need to register target (ex: EC2 instances) with our above created target group.
* Now we could see a new EC2 instance (with configuration of EC2 instance that we have chosen earlier) with Load Balancer.
* We can check Health for the instance that we have chosen, under Targets tab of Target Group.
* We know that we have 2 security group (1st is for EC2 instance that we created earlier, and another is for Load Balancer). We can configure them in such a way that the traffic on EC2 instance SG can come only from Load Balancer SG. So now the request from web can’t directly go to the pubic URL mentioned for EC2 instance, it has to come through Load Balancer.

## **Auto Scaling Groups**

An Auto Scaling group contains a collection of EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An Auto Scaling group also enables you to use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies. Both maintaining the number of instances in an Auto Scaling group and automatic scaling are the core functionality of the Amazon EC2 Auto Scaling service.

The size of an Auto Scaling group depends on the number of instances that you set as the desired capacity. You can adjust its size to meet demand, either manually or by using automatic scaling.

An Auto Scaling group starts by launching enough instances to meet its desired capacity. It maintains this number of instances by performing periodic health checks on the instances in the group. The Auto Scaling group continues to maintain a fixed number of instances even if an instance becomes unhealthy. If an instance becomes unhealthy, the group terminates the unhealthy instance and launches another instance to replace it.

You can use scaling policies to increase or decrease the number of instances in your group dynamically to meet changing conditions. When the scaling policy is in effect, the Auto Scaling group adjusts the desired capacity of the group, between the minimum and maximum capacity values that you specify and launches or terminates the instances as needed. You can also scale on a schedule.

An Auto Scaling group can launch On-Demand Instances, Spot Instances, or both. You can specify multiple purchase options for your Auto Scaling group only when you configure the group to use a launch template. (We recommend that you use launch templates instead of launch configurations to make sure that you can use the latest features of Amazon EC2.)

Spot Instances provide you with access to unused Amazon EC2 capacity at steep discounts relative to On-Demand prices.There are key differences between Spot Instances and On-Demand Instances:

* The price for Spot Instances varies based on demand
* Amazon EC2 can terminate an individual Spot Instance as the availability of, or price for, Spot Instances changes

When a Spot Instance is terminated, the Auto Scaling group attempts to launch a replacement instance to maintain the desired capacity for the group.

When instances are launched, if you specified multiple Availability Zones, the desired capacity is distributed across these Availability Zones. If a scaling action occurs, Amazon EC2 Auto Scaling automatically maintains balance across all of the Availability Zones that you specify.

## **Practical Knowledge**

* Create a Launch Template and then create an Auto Scaling Group.
* There is no extra charges for ASG, we have to pay only for EC2 instances.
* Here we must choose an AMI for our ASG and proceed.
* If we want to have predownloaded application in our instance, we can give it as “User Data” like

#!bin/bash

yum update -y

yum install -y httpd.x86\_64

systemctl start httpd.service

systemctl enable httpd.service

echo “Hello world from $(hostname -f)” > /var/www/html/index.com

* Once we have successfully created template, now we can create ASG. ASG can be configured in Advanced details as:

Choose “Receive traffic from one or more load balancers”

Provide Targets group (as we are using Application Load Balancer for HTTP application)

Choose ELB as health check

We can configure scaling policy (like minimum and maximum instance for load)

If we choose metric type as CPU utilization, Target value = 40, it means once CPU is utilized by 40%, it will scale it out. (create instances up to 3 else minimum 1)