# AWS ALB and AutoScaling

***CS516 – Cloud Computing***

***Computer Science Department***

***Maharishi International University***

## Maharishi International University - Fairfield, Iowa

## Content

- Types of ELB (OSI Model)
  - Application Load Balancer (ALB)
  - Network Load Balancer (NLB)
  - Classic Load Balancer
  - Gateway Load Balancer
- Application Load Balancer (ALB)
  - ALB listener
  - ALB listener rule
  - ALB target groups
- Auto Scaling (with CloudWatch)
- Scaling policies

## Elastic Load Balancer (ELB)

A tool that distributes incoming web traffic (visitors to a web site) and equally across multiple EC2 instances that are running a web site.

ELB automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables **fault-tolerance** in applications, seamlessly providing the required amount of load-balancing capacity needed to route application traffic.

Helps prevent one server from being overloaded while another server can handle more visitors. It makes the app more **reliable**.

## ELB features

- Helps scaling out. You can run N number of servers behind an ELB.
- It is like a gateway to your application(s). There will be one URL to all your servers when using ELB.
- Improves fault-tolerance and reliability. Because it knows if the servers are healthy or not. Then routes to only healthy instances. ELB is used in conjunction with ASG.

## OSI Layers

- Layer 7 (The application layer) – It is the layer that directly interacts with a user. These are applications such as email and browsers that shows data in a human-readable format.
- Layer 6 (The presentation layer) – This layer prepares the human-readable data by decrypting and decompressing data if required.
- Layer 5 (The session layer) – This layer is responsible for opening and closing communication between the local and remote applications. For instance, if a user is downloading 50 MiB data, this layer checks if it is done every time 5 MiB data is downloaded.

## OSI Layers

- Layer 4 (The transport layer) – It receives data from the session layer (5) and breaks the data down into smaller chunks and sends the data to the network layer (3). It also assembles data from layer 3 and passes that to layer 5.

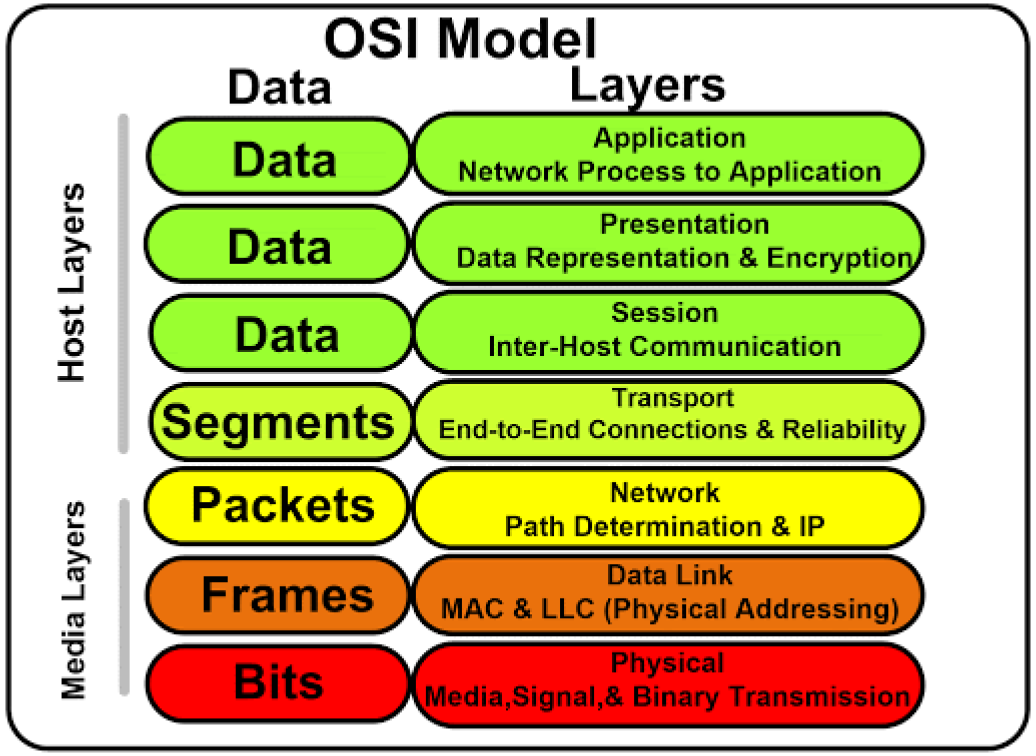
This layer transports data between 2 devices. The connection-oriented TCP protocol is used in this layer that makes sure all data is transferred and is more reliable. There is also a connectionless UDP protocol in this layer. UDP is faster than TCP used in streaming and gaming applications.

## OSI Layers

- Layer 3 (The network layer) – This layer is used to connect different networks. Common protocols in this layer are IP and ICMP. In real-life, developers need to check the connection between 2 servers in 2 different networks. In that case, they check the connection by making an ICMP call. You will see the ICMP protocol is open in some SG groups and now you understand what the rule is for.
- Layer 2 (The data link layer) – This layer is similar to the network layer but between devices in the same network.
- Layer 1 (The physical layer) – It is the physical networking matter you see all around you such as cables and switches. It transfers machine data in bits (one-zero, light-dark, high-low frequencies).

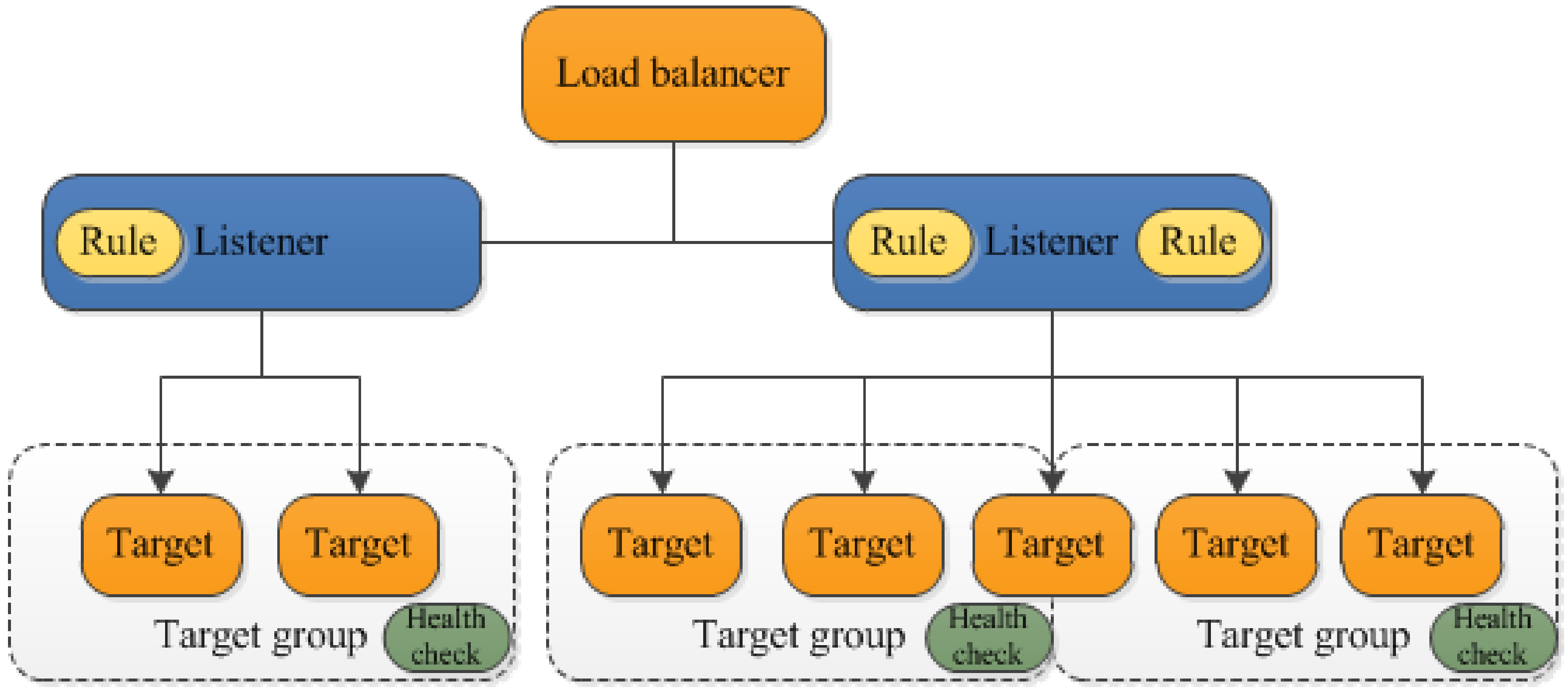
# Types of ELB

1. **Classic Load Balancer** (CLB) – Old generation, not recommended for new apps. Performs routing at Layer 4 and Layer 7.
2. **Network Load Balancer** (NLB) – Routes connections based on IP protocol data (layer 4). Ultra high performance and low latency. Supports UDP and static IP addresses as targets.
3. **Application Load Balancer** (ALB) – Routes based on the content of the request (layer 7). Supports path-based, host-based, query string, parameter-based, and source IP based routing. Supports IP addresses, Lambda, containers as targets.
4. **Gateway Load Balancer** – Operates at Layer 3 (Network). Gateway Load Balancers enable you to deploy, scale, and manage virtual appliances, such as firewalls, intrusion detection and prevention systems, and deep packet inspection systems.



# Application Load Balancer (ALB)

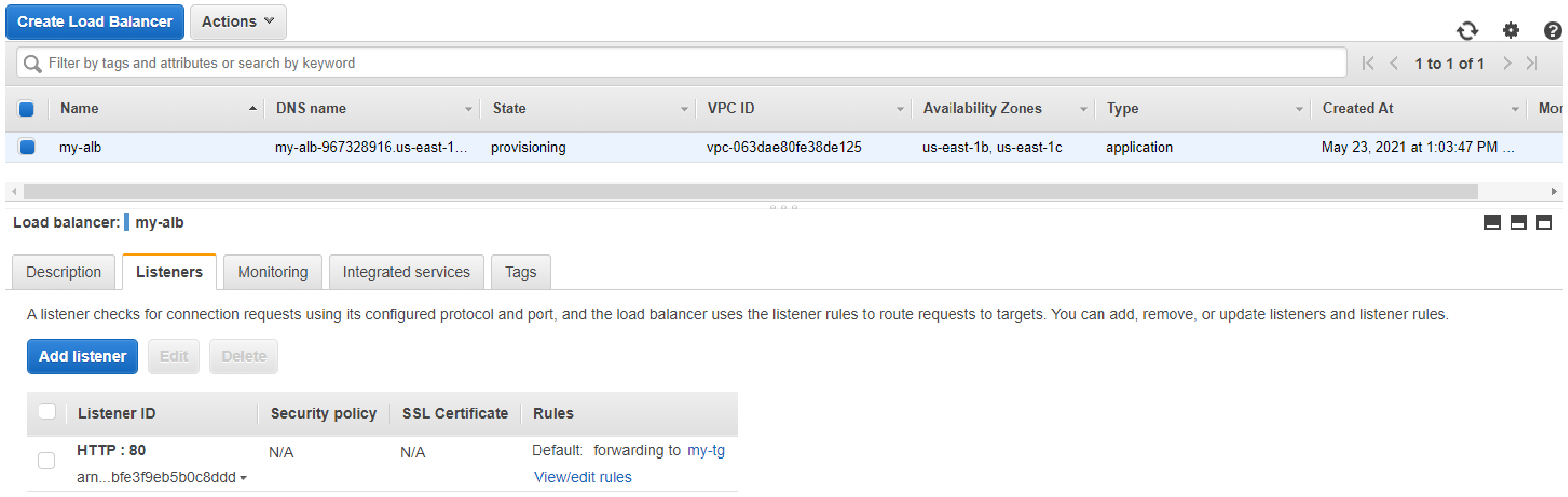
A load balancer serves as the single point of contact for clients. The load balancer distributes incoming application traffic across multiple targets, such as EC2 instances, in multiple Availability Zones. This increases the availability of your application. You add one or more listeners to your load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



# ALB Listener

A listener checks for connection requests from clients, using the **protocol** and **port** that you configure.

A **certificate** is attached to the listener. You must define a default certificate if using https. AWS and custom certificates are stored on Amazon Certificate Manager (ACM). AWS certificates are free!



| Name | DNS name | State | VPC ID | Availability Zones | Type | Created At | Mor |
|---|---|---|---|---|---|---|---|
| my-alb | my-alb-967328916.us-east-1... | provisioning | vpc-063dae80fe38de125 | us-east-1b, us-east-1c | application | May 23, 2021 at 1:03:47 PM ... | |

Load balancer: my-alb

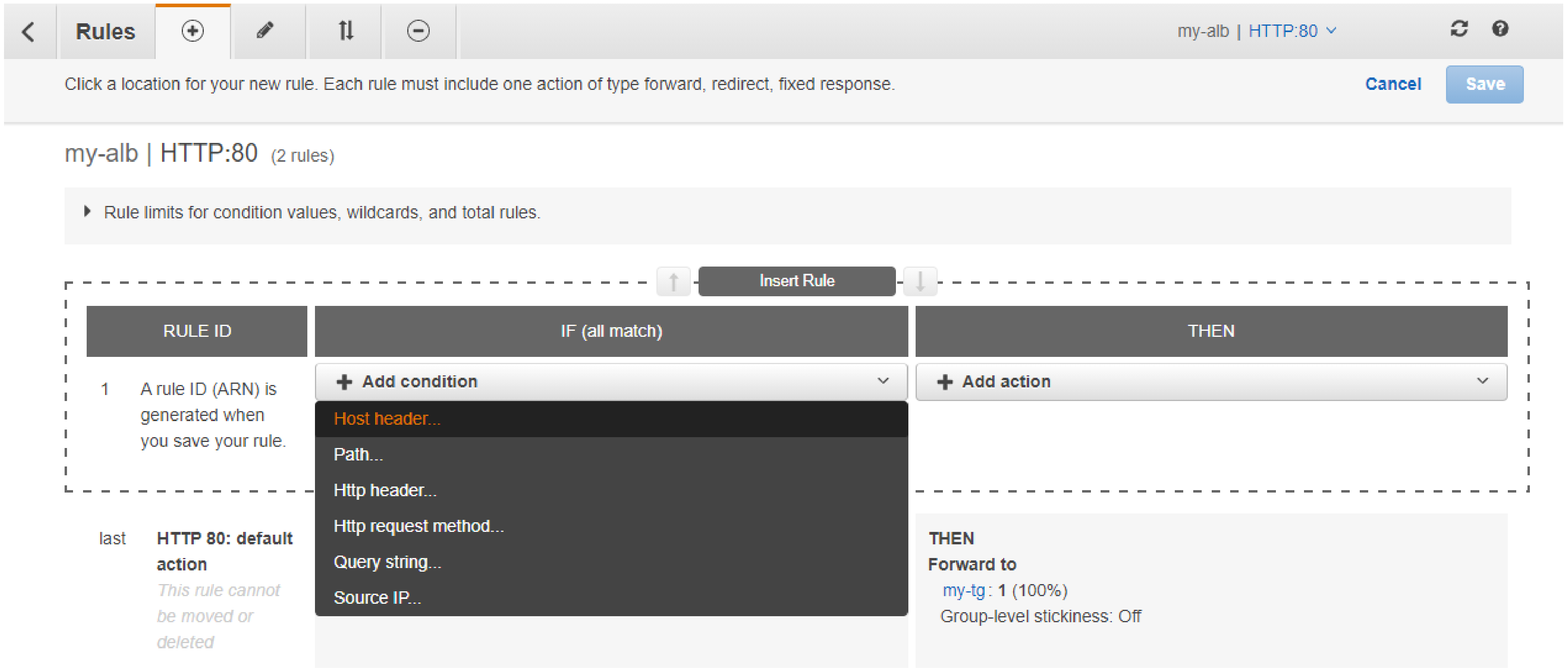
A listener checks for connection requests using its configured protocol and port, and the load balancer uses the listener rules to route requests to targets. You can add, remove, or update listeners and listener rules.

| Listener ID | Security policy | SSL Certificate | Rules |
|---|---|---|---|
| HTTP : 80<br>arn...bfe3f9eb5b0c8ddd | N/A | N/A | Default: forwarding to my-tg<br>View/edit rules |

# ALB Listener Rules

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, actions, 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.



Click a location for your new rule. Each rule must include one action of type forward, redirect, fixed response.

my-alb | HTTP:80 (2 rules)

Rule limits for condition values, wildcards, and total rules.

1 A rule ID (ARN) is generated when you save your rule.

last **HTTP 80: default action**
*This rule cannot be moved or deleted*

**THEN**
**Forward to**
my-tg: **1** (100%)
Group-level stickiness: Off

## ALB Target Groups

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 configure health checks on a per target group basis. Health checks are performed on all targets registered to a target group.

**Note:** When you associate a resource with an ALB, you associate it with its target group. When creating an ALB, target group can have no resources. But the important thing is you must specify the right target group type (ip, instance, lambda).

## Reasons that the instance is unhealthy

- When the web server is not started. If it is the case, SSH into the instance and start the web server like “service httpd start”. Or you enter a start up script in the User Data.
- SG that the instance SG doesn’t allow access from the ALB. In that case, you must add an inbound rule that allows access from ALB, source as ALB SG.

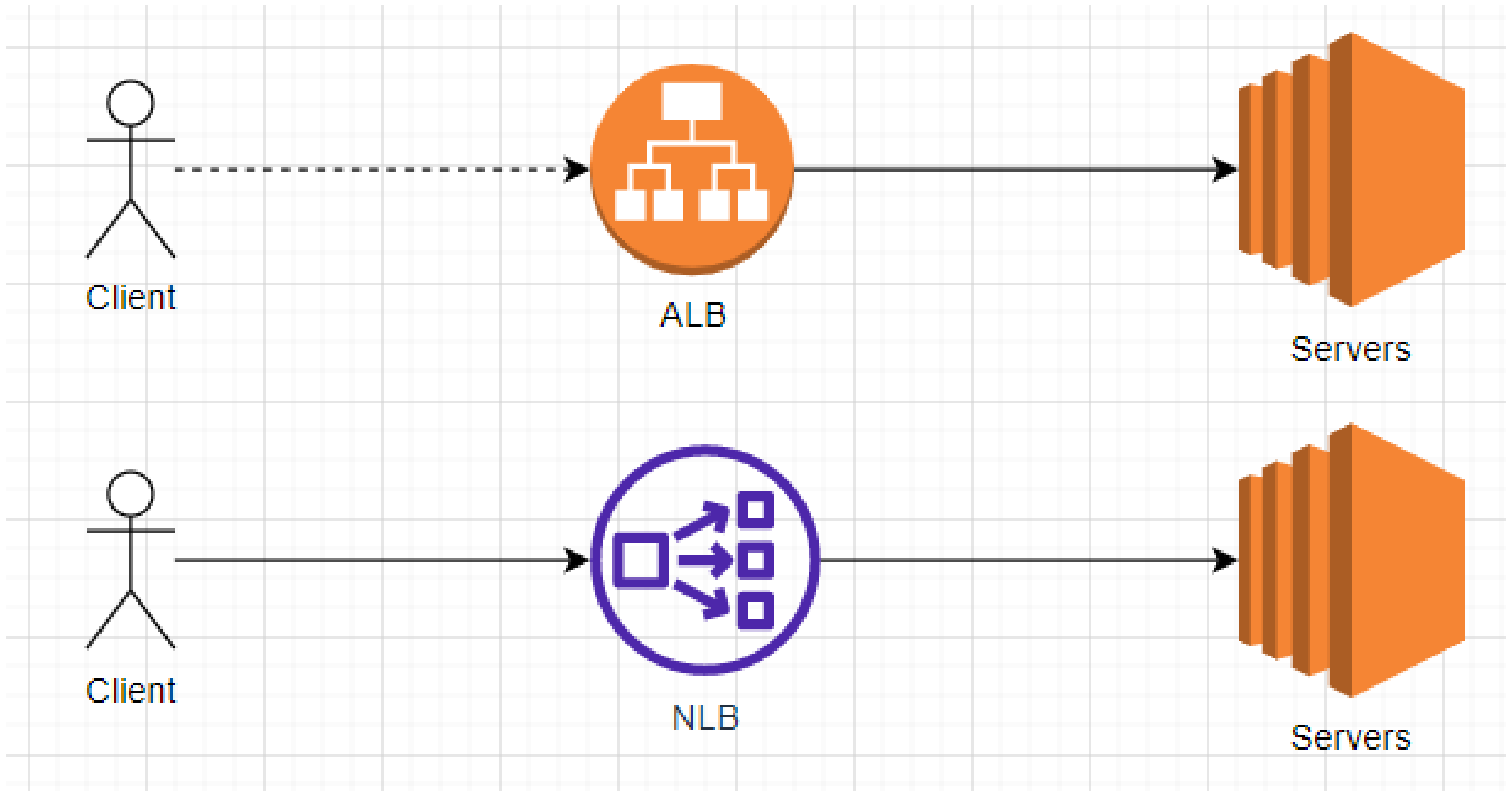
## ALB features that NLB doesn’t support

- Web sockets.
- Authentication with well-known identity providers such as Microsoft, Google, and Facebook. So you can offload the authentication part from your applications.
- Can send a fixed response without a backend.

## The difference between ALB and NLB

| **ALB** | **NLB** |
|---|---|
| Operates at OSI Layer 7 (Application) | Operates at OSI Layer 4 (Transport) |
| Lower performance | High performance |
| IP address, ECS, EC2, and Lambda as a target | Only IP address as a target |
| Routing rules based on hostname, path, query string parameter, HTTP method, HTTP headers, source IP, or port number. | Routing is only based on port number. |
| The source IP is the ALB node IP. To get the client IP, enable preserve client IP and it will be in the header. | The client IP is preserved by default as the source IP. |
| Pricing model is complex | Pricing model is straighforward |
| HTTP and HTTPS | TCP and UDP |
| Great for web apps | Great for streaming, near-real-time applications. |
| Has SG. | There is no SG. |

- The client connection terminates at ALB whereas the client connection terminates at the server in NLB.
- The source IP in server is ALB where the source IP in server is Client IP in NLB.
- NLB doesn’t have SG. To whitelist NLB, you can use an elastic IP. Deselect preserve client IP.



## Routing algorithm

The routing algorithms choose the target in the target group.

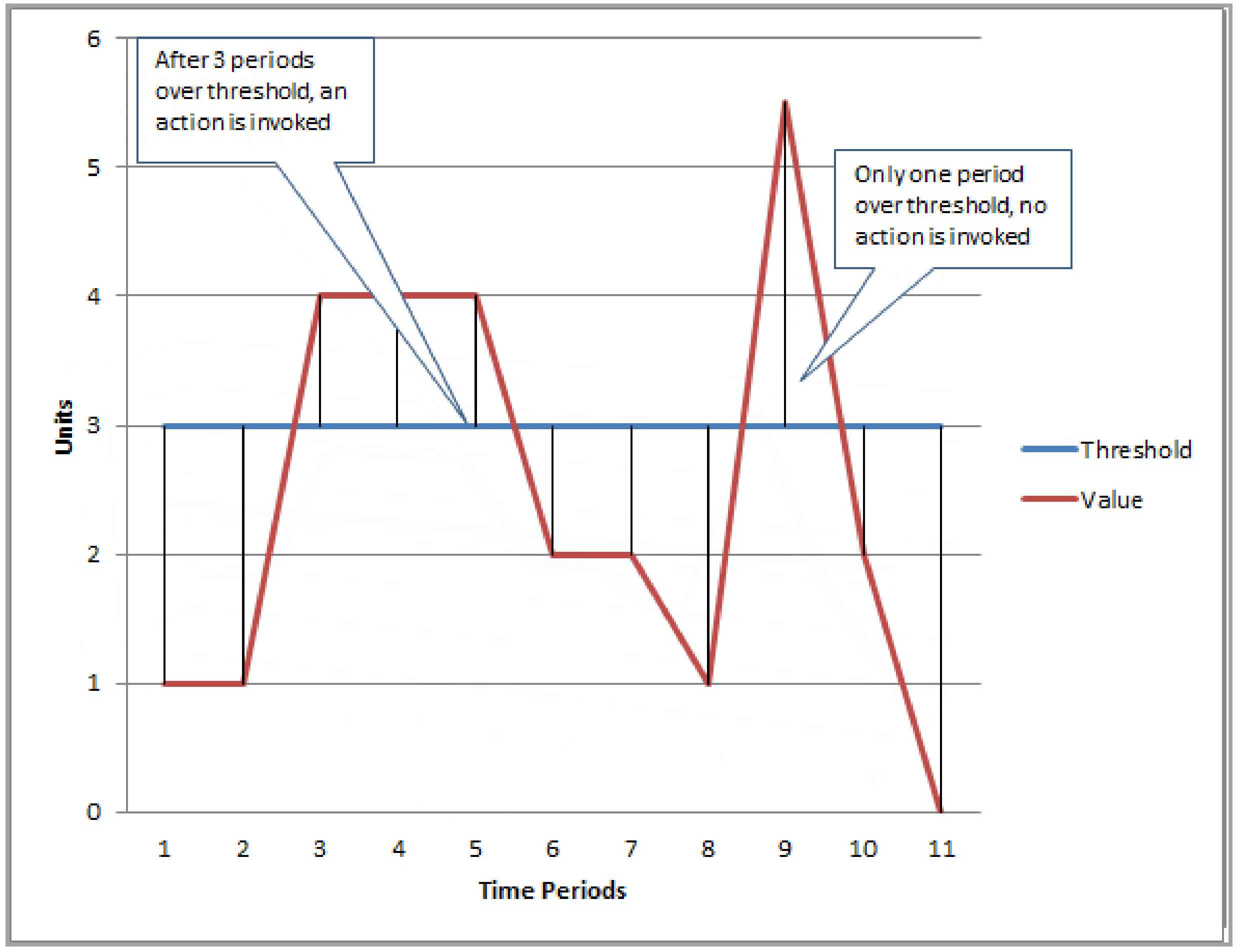
**Round-robin** (default) load balancing is one of the simplest methods for distributing client requests across a group of servers. Going down the list of servers in the group, the round-robin load balancer forwards a client request to each server in turn. When it reaches the end of the list, the load balancer loops back and goes down the list again.

**Least outstanding requests** is an algorithm that choses which instance receives the next request by selecting the instance that, at that moment, has the lowest number of outstanding (pending, unfinished) requests.

## Auto Scaling benefits

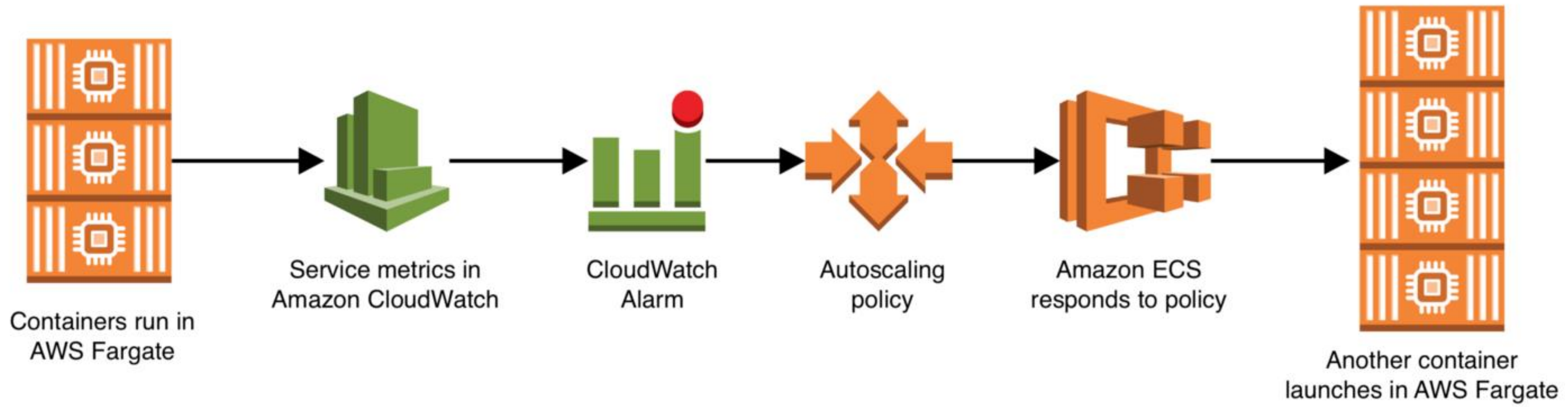
- **Better fault tolerance** - Auto Scaling can detect when a resource is unhealthy, terminate it, and launch a resource to replace it. You can also configure resources to use multiple AZs. If one AZ becomes unavailable, Auto Scaling can launch resources in another one to compensate.
- **Better availability** - Auto Scaling helps ensure that your application always has the right amount of capacity to handle the current traffic demand.
- **Better cost management** - Auto Scaling can dynamically increase and decrease capacity as needed. Because you pay for the resources you use.

## CloudWatch Alarm for AutoScaling



# Application Auto Scaling

Application Auto Scaling is a web service for developers and system administrators who need a solution for automatically scaling their scalable resources for individual AWS services beyond Amazon EC2. You can scale Elastic Container Service (ECS), Lambda, Aurora replicas (RDS), DynamoDB, and more.


Containers run in AWS Fargate
Service metrics in Amazon CloudWatch
CloudWatch Alarm
Autoscaling policy
Amazon ECS responds to policy
Another container launches in AWS Fargate

## EC2 Auto Scaling components

**Groups** - Your EC2 instances are organized into groups so that they can be treated as a logical unit for the purposes of scaling and management.

**Configuration templates** - Your group uses a launch (configuration) template where you can specify information such as the AMI ID, instance type, key pair, security groups, and block device mapping for your instances.

**Scaling options** - Amazon EC2 Auto Scaling provides several ways for you to scale your Auto Scaling groups. For example, you can configure a group to scale based on the occurrence of specified conditions (dynamic scaling) or on a schedule.

# Scaling policies

Application Auto Scaling allows you to automatically scale your scalable resources according to conditions that you define:

1. **Target tracking scaling** - Scale a resource based on a target value for a specific CloudWatch metric. Like thermostat at your home.
2. **Step scaling** - Scale a resource based on a set of scaling adjustments that vary based on the size of the alarm breach.
3. **Scheduled scaling** - Scale a resource based on the date and time.
4. **Predictive scaling** - Uses machine learning to analyze each resource's historical workload and regularly forecasts the future load for the next two days.

## Scale in algorithm

You can define a termination policy in case of scaling in. By default, it selects the AZ with two instances, and

- terminates the instance that was launched from the oldest launch template or launch configuration.
- If the instances were launched from the same launch template or launch configuration, Amazon EC2 Auto Scaling selects the instance that is closest to the next billing hour and terminates it.

Read more: Control instance termination