

# ELASTIC LOAD BALANCING

(MAPPING ELB with Auto scaling group and setting criteria along with cloud watch)

- Achieve fault tolerance for any application by ensuring scalability, performance, and security
- Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, and Lambda functions. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones. Elastic Load Balancing offers three types of load balancers that all feature the high availability, automatic scaling, and robust security necessary to make your applications fault tolerant

Types of load balancers

1. Application Load balancer
2. Network load balancer (which is recently brought up & working under progress)
3. Classic load balancer

## Application Load balancer (HTTP & HTTPS) for web application

- Application Load Balancer is best suited for load balancing of HTTP and HTTPS traffic and provides advanced request routing targeted at the delivery of modern application architectures, including microservices and containers. Operating at the individual request level (Layer 7), **Application Load Balancer routes traffic to targets within Amazon Virtual Private Cloud (Amazon VPC) based on the content of the request.**
- Incoming request will route based on HOSTNAME & PATHNAME (Ex:- [www.icicibank.com/creditcard/signup](http://www.icicibank.com/creditcard/signup)) So it will route only to credit-card which is path name
- It read URL

## Network Load balancer (PORT NUMBERS)

- It will route the traffic based on the port numbers
- Network Load Balancer is best suited for load balancing of Transmission Control Protocol (TCP), User Datagram Protocol (UDP) and Transport Layer Security (TLS) traffic where extreme performance is required. Operating at the connection level (Layer 4)
- Ex:- [www.icicibank.com://8080](http://www.icicibank.com://8080) -> so this will route only based on port numbers & Highly secured purpose
- Network Load Balancer routes traffic to targets within Amazon Virtual Private Cloud (Amazon VPC) and is capable of handling millions of requests per second while maintaining ultra-low latencies. Network Load Balancer is also optimized to handle sudden and volatile traffic patterns

## Classic Load balancer (Old load balancer)

- It will not read URL
- It will re-direct the request through IP address & port numbers. It operates at both the request level and connection level
- Classic Load Balancer is intended for applications that were built within the EC2-Classic network

- Currently we are only this classic load balancer in real-time

**Step by step process along with screenshot will show below.**

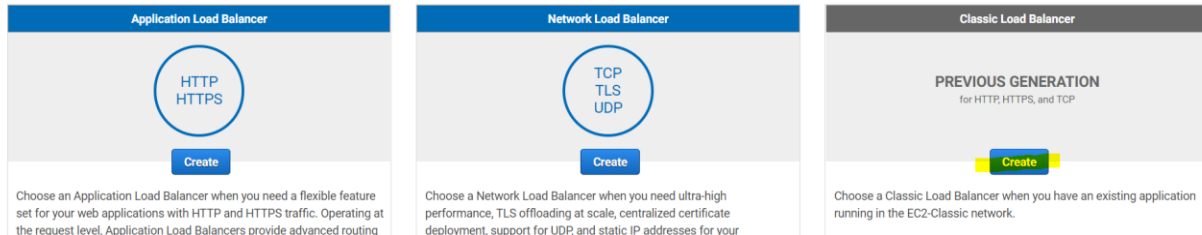
## Step 1:-

### First created one LOADBALANCER

#### ▼ LOAD BALANCING

##### Load Balancers

Here we going to create classic loadbalancer



## Creating loadbalancer name as myLB

### Step 1: Define Load Balancer

#### Basic Configuration

This wizard will walk you through setting up a new load balancer. Begin by giving your new load balancer a unique name so that you can identify it from other load balancers you might create. You will also need to configure ports and protocols for your load balancer. Traffic from your clients can be routed from any load balancer port to any port on your EC2 instances. By default, we've configured your load balancer with a standard web server on port 80.

Load Balancer name:

Create LB Inside:

Create an internal load balancer: ☐ (what's this?)

Enable advanced VPC configuration: ☐

Listener Configuration:

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port
HTTP	80	HTTP	80
HTTPS (Secure HTTP)	443	HTTP	80

## Assign security groups

### Create new group & named as SG

#### Step 2: Assign Security Groups

You have selected the option of having your Elastic Load Balancer inside of a VPC, which allows you to assign security groups to your load balancer. Please select the security groups to assign to this load balancer. This can be changed at any time.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name:

Description:

Type	Protocol	Port Range	Source
HTTP	TCP	80	Custom 0.0.0.0/0

Add Rule

## Health check on our ec2 instances

#### Step 4: Configure Health Check

Your load balancer will automatically perform health checks on your EC2 instances and only route traffic to instances that pass the health check. If an instance fails the health check, it is automatically removed from the load balancer. Customize the health check to meet your specific needs.

Ping Protocol:

Ping Port:

Ping Path:

Advanced Details

Response Timeout	5	seconds
Interval	30	seconds
Unhealthy threshold	2	
Healthy threshold	10	

## Review our loadbalancer before creating

Step 7: Review  
Please review the load balancer details before continuing

Define Load Balancer [Edit load balancer definition](#)

Load Balancer name: myLB  
Scheme: internet-facing  
Port Configuration: 80 (HTTP) forwarding to 80 (HTTP)

Configure Health Check [Edit health check](#)

Ping Target: HTTP:80/index.html  
Timeout: 5 seconds  
Interval: 30 seconds  
Unhealthy threshold: 2  
Healthy threshold: 10

Add EC2 Instances [Edit instances](#)

Cross-Zone Load Balancing: Enabled  
Connection Draining: Enabled, 300 seconds  
Instances:

VPC Information [Edit subnets](#)

VPC: vpc-c0727fa8

Then click create.

myLB	myLB-1141630165.ap-south...	vpc-c0727fa8	ap-south-1a, ap-south-...	classic
------	-----------------------------	--------------	---------------------------	---------

So load balancer created successfully.

Then go to Auto scaling configurations

Create Launch Configuration [Cancel and Exit](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Quick Start

My AMIs

AWS Marketplace

Community AMIs

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0217a85e28e625474 [Select](#)

Amazon Linux 2.29.1, and the latest software packages through extras. 64-bit

Root device type: ebs Virtualization type: hvm

setting up pre configured user data & named as ASG for autoscaling

## Create Launch Configuration

Name [i](#) myASG

Purchasing option [i](#) ☐ Request Spot Instances

IAM role [i](#) None

Monitoring [i](#) ☐ Enable CloudWatch detailed monitoring [Learn more](#)

Advanced Details

Kernel ID [i](#) Use default

RAM Disk ID [i](#) Use default

User data [i](#) ☒ As text ☐ As file ☐ Input is already base64 encoded

```
#!/bin/bash
yum install httpd
service httpd start
echo " this is my LB" >/var/www/html/index.html
```

IP Address Type [i](#) ☒ Only assign a public IP address to instances launched in the default VPC and subnet. (default)

Once done, Go to auto-scaling group and enter the details below

1. Configure Auto Scaling group details 2. Configure scaling policies 3. Configure Notifications 4. Configure Tags 5. Review

Create Auto Scaling Group Cancel and Exit

Group size ⓘ Start with 2 instances

Network ⓘ vpc-c0727fa8 (172.31.0.0/16) (default) Create new VPC

Subnet ⓘ

- subnet-d26349ba(172.31.32.0/20) | Default in ap-south-1a
- subnet-9464c6ef(172.31.16.0/20) | Default in ap-south-1c
- subnet-681b8324(172.31.0.0/20) | Default in ap-south-1b

Create new subnet

Each instance in this Auto Scaling group will be assigned a public IP address. ⓘ

▼ Advanced Details

Load Balancing ⓘ ☒ Receive traffic from one or more load balancers Learn about Elastic Load Balancing

Classic Load Balancers ⓘ myLB ×

Target Groups ⓘ

Health Check Type ⓘ ☒ ELB ☐ EC2

Once done, select use scaling policies to adjust the capacity of this group

In that select scale from 2 to instances, which means if once instance goes down, automatically another instance is to get created

### Create Auto Scaling Group

You can optionally add scaling policies if you want to adjust the size (number of instances) of your group automatically. A scaling policy is a set of instructions for making such adjustments in response to an Amazon CloudWatch alarm that you assign to it. In each policy, you can choose to add or remove a specific number of instances or a percentage of the existing group size, or you can set the group to an exact size. When the alarm triggers, it will execute the policy and adjust the size of your group accordingly. [Learn more](#) about scaling policies.

- ☐ Keep this group at its initial size
- ☒ Use scaling policies to adjust the capacity of this group

Scale between 2 and 3 instances. These will be the minimum and maximum size of your group.

Increase Group Size ×

Name: Increase Group Size

Execute policy when: No alarm selected + Add new alarm

Take the action: Add 0 capacity units + Add step ⓘ

Instances need: 300 seconds to warm up after each step

[Create a simple scaling policy](#) ⓘ

Now we going to set threshold, here shown below is just sample to get fast output for increments gng to 3 percent for utilization and for decrease 70% of utilizations

**Go to add new alarm for increment** and set consecutive period as 1 minute, I have set 5minutes, but make sure set 1 minute

Create Alarm ×

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.  
To edit an alarm, first choose whom to notify and then define when the notification should be sent.

☒ Send a notification to: dynamodb

Whenever: Average of CPU Utilization

Is: >= 3 Percent

For at least: 1 consecutive period(s) of 5 Minutes

Name of alarm: increase

CPU Utilization Percent

**Go to add new alarm for decrement**

Create Alarm

×

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

☐ Send a notification to: dynamodb

Whenever: Average of CPU Utilization

Is: >= 70 Percent

For at least: 1 consecutive period(s) of 1 Minute

Name of alarm: decrease

CPU Utilization Percent

myasg

Cancel

Create Alarm

Review:-

## Create Auto Scaling Group

Please review your Auto Scaling group details. You can go back to edit changes for each section. Click **Create Auto Scaling group** to complete the creation of an Auto Scaling group.

### Auto Scaling Group Details

Group name	myasg
Group size	2
Minimum Group Size	2
Maximum Group Size	3
Subnet(s)	subnet-d26349ba,subnet-9464c6ef,subnet-681b8324
Load Balancers	myLB
Target Groups	
Health Check Type	ELB
Health Check Grace Period	300
Detailed Monitoring	No
Instance Protection	None
Service-Linked Role	AWSServiceRoleForAutoScaling

### Scaling Policies

Increase Group Size	With alarm = increase; Add 0 capacity units and 300 seconds for instances to warm up
Decrease Group Size	With alarm = decrement; Remove 0 capacity units

So now auto scaling group also done successfully and now we could see 2 instance running in EC2 dashbaord

	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP
<input type="checkbox"/>		i-006f9da810bede312	t2.micro	ap-south-1b	running	Initializing	None	ec2-13-232-221-208.ap...	13.232.221.208
<input type="checkbox"/>		i-02b522fb324e9dbd	t2.micro	ap-south-1a	running	Initializing	None	ec2-13-232-224-124.ap...	13.232.224.124

So far we have done is created **LOAD BALANCER & CREATED AUTO-SCALING GROUP** and set criteria as well

After clicking our website multi times, obviously thresold level wil get increase

Then we get alarm in CLOUDWATCH

CloudWatch > Alarms

Alarms (3) ☐ Hide Auto Scaling alarms Refresh Add to dashboard Act

Insufficient d...

<input type="checkbox"/>	Name	State	Conditions	Act
<input type="checkbox"/>	increase	⌚ Insufficient data	CPUUtilization >= 3 for 1 datapoints within 1 minute	-

This is how it notified alert in cloud watch.

Now we going to terminate the instance, assume that one instance got accidentally got terminated, lets how auto scaling works

Im terminating one instance below

<input type="checkbox"/>	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
		i-019eb4e7186bae7...	t2.micro	ap-south-1a	shutting-do...		None
<input type="checkbox"/>		i-053ceab110fe0d820	t2.micro	ap-south-1b	running	2/2 checks ...	None

After termination received alarm in cloud watch as well, then

Alarms (3) ☐ Hide Auto Scaling alarms Refresh Add to dashboard Action

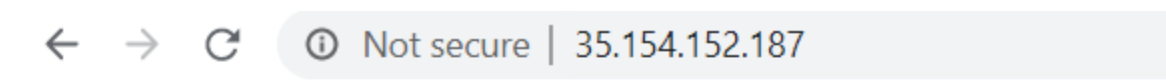
In alarm

<input type="checkbox"/>	Name	State	Conditions	Actions
<input type="checkbox"/>	increase	⚠ In alarm	CPUUtilization >= 3 for 1 datapoints within 1 minute	-

could see automatically instance is creating in ec2, if one instance got automatically terminated

<input type="checkbox"/>	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP
		i-01f464fd35be4b389	t2.micro	ap-south-1a	pending	Initializing	None	ec2-35-154-152-187.ap...	35.154.152.187
<input type="checkbox"/>		i-053ceab110fe0d820	t2.micro	ap-south-1b	running	2/2 checks ...	None	ec2-15-206-148-250.ap...	15.206.148.250

Now I am hitting newly launched instance IP



this is my LB

YAAAH!!!! I can able to access my web server with out losing my connectivity

## Note:-

In this notes, we covered,

- Created loadbalancer as my LB
- Then autoscaling configurations, we created 2 instances and added our Loadbalancer name as well, then configure security group set as http & ssh.
- Then we created auto scaling groups, in that we modified that 2 instance to 3 instance as per we have set threshold too,
- Then multi-time we hit our application browser, then we checked in cloud watch, it crossed threshold and we got alert as well,
- Then accidentally deleted 1 EC2 instance, then as per autoscaling scenario, immediately one EC2 instances got created with out losing productivity of web server application

Diagram

